

Claudia Eckert
IT-Sicherheit
De Gruyter Studium

Weitere empfehlenswerte Titel



Softwareentwicklung

Albin Meyer, 2018

ISBN 978-3-11-057580-4, e-ISBN (PDF) 978-3-11-057837-9,
e-ISBN (ePUB) 978-3-11-057588-0



Smart Data Analytics

Andreas Wierse, Till Riedel, 2017

ISBN 978-3-11-046184-8, e-ISBN (PDF) 978-3-11-046395-8,
e-ISBN (ePUB) 978-3-11-046191-6



IT-Sicherheitsanalysen

Daniela Simic, 2017

ISBN 978-3-11-051492-6, e-ISBN (PDF) 978-3-11-051698-2,
e-ISBN (EPUB) 978-3-11-051498-8



Trusted Computing

Dengguo Feng, 2017

ISBN 978-3-11-047604-0, e-ISBN (PDF) 978-3-11-047759-7,
e-ISBN (EPUB) 978-3-11-047609-5

Claudia Eckert

IT-Sicherheit

Konzepte - Verfahren – Protokolle

10. Auflage

DE GRUYTER
OLDENBOURG

Autorin

Prof. Dr. Claudia Eckert

Technische Universität München
Lehrstuhl für Sicherheit in der Informatik
Fakultät für Informatik
Boltzmannstr. 3
85748 Garching bei München

Fraunhofer-Institut für Angewandte und Integrierte Sicherheit (AISEC)
Institutsleitung
Parkring 4
85748 Garching bei München
claudia.eckert@aisec.fraunhofer.de

ISBN 978-3-11-055158-7
e-ISBN (PDF) 978-3-11-056390-0
e-ISBN (ePUB) 978-3-11-058468-4

Library of Congress Control Number: 2018944105

Bibliografische Information der Deutschen Nationalbibliothek

Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über <http://dnb.dnb.de> abrufbar.

© 2018 Walter de Gruyter GmbH, Berlin/Boston
Einbandabbildung: KTSDESIGN / SCIENCE PHOTO LIBRARY / Getty Images
Druck und Bindung: CPI books GmbH, Leck

www.degruyter.com

Vorwort

Informationstechnologie (IT) ist heute in nahezu allen Bereichen von zentraler Bedeutung: Gesundheit, Mobilität, Bildung, Unterhaltung, Produktion, Logistik, aber auch Handel, Finanzen, und öffentliche Verwaltung. IT ist eine Schlüsseltechnologie, die neue Anwendungen und auch neue Geschäftsmodelle ermöglicht. Durch die Einbettung von IT in Alltagsgegenstände und durch die zunehmende Dienste-Orientierung entsteht das Internet der Dinge und der Dienste. Der IT-Sicherheit kommt bei dieser Entwicklung eine Schlüsselrolle zu.

Lösungen der IT-Sicherheit haben zum einen eine Wegbereiter-Funktion, da neue Anwendungen häufig nur eingesetzt und akzeptiert werden, wenn die Sicherheit der Daten gewährleistet wird. Zum anderen hat die IT-Sicherheit natürlich die bekannte Schutzfunktion. Gezielt und korrekt eingesetzte Maßnahmen der IT-Sicherheit reduzieren die Risiken wirtschaftlicher Schäden, die zum Beispiel durch eine unautorisierte Weitergabe von Daten oder durch kriminelle Aktivitäten wie Wirtschaftsspionage entstehen können. Maßnahmen der IT-Sicherheit sind aber auch notwendig, um vor Schäden an Leib und Leben zu schützen, die zum Beispiel durch manipulierte Gesundheitsdaten oder durch manipulierte Fahrzeugsensorik entstehen können.

Das vorliegende Buch hat zum Ziel, fundierte Kenntnisse über wirksame Maßnahmen zu vermitteln, die zur Erhöhung der Sicherheit heutiger Systeme beitragen können. Die Qualität eines sicheren IT-Systems hängt wesentlich davon ab, dass seine Konstruktion methodisch und systematisch erfolgt. Das Buch führt die hierfür notwendigen Techniken und Vorgehensweisen ein, wie Bedrohungsanalyse, Modellierung und Bewertung. Die zur Umsetzung der Sicherheitsanforderungen benötigten Mechanismen, Verfahren und Protokolle werden eingehend erklärt sowie anhand von Fallbeispielen erläutert. Ziel ist es, die Ursachen für Problembereiche heutiger IT-Systeme zu verdeutlichen und die grundlegenden Sicherheitskonzepte mit ihren jeweiligen Vor- und Nachteilen vorzustellen. Der Leser soll ein Verständnis für die vielschichtigen Probleme sicherer Systeme erlangen sowie ein breites und grundlegendes Wissen zu deren Behebung erwerben.

Das Buch beschäftigt sich mit Fragestellungen der Sicherheit technischer Systeme im Sinne des englischen Begriffs Security. Zur Gewährleistung

von Security-Eigenschaften setzt man Konzepte und Maßnahmen ein, um Bedrohungen abzuwehren, die im Wesentlichen durch absichtliche oder unabsichtliche Angriffe von außen auf das IT-System entstehen. Sicherheitsaspekte, die organisatorische Fragen betreffen, und solche, die durch den Begriff Safety charakterisiert werden, liegen außerhalb des behandelten Themenrahmens. Safety beschreibt die Funktionssicherheit eines Systems. Zu deren Gewährleistung benötigt man Konzepte und Verfahren, um solche Bedrohungen abzuwehren, die im Wesentlichen durch das Fehlverhalten des IT-Systems selber entstehen. Es handelt sich hierbei um Fragestellungen der Fehlervermeidung und Fehlertoleranz sowie der Steigerung der Zuverlässigkeit und Verfügbarkeit von IT-Systemen.

Das Buch richtet sich an Studierende der Informatik, Mathematik und Elektrotechnik sowie an interessierte Leser mit Informatik-Hintergrundwissen, die grundlegende Kenntnisse auf dem Gebiet der IT-Sicherheit erwerben bzw. bereits vorhandene Kenntnisse vertiefen wollen. Das Buch ist für die eigenständige Beschäftigung mit dem Stoff geeignet.

Vorwort zur zehnten Auflage

Das Gebiet der IT-Sicherheit ist einem steten Wandel unterworfen, so dass eine regelmäßige Aktualisierung des Buches notwendig ist. Die vorliegende Auflage enthält deshalb eine korrigierte und aktualisierte Überarbeitung und Erweiterung. Neben einigen kleineren Korrekturen und Aktualisierungen wurden auch einige neue Themen aufgenommen. Kapitel 2 wurde um die Darstellung der Angriffsklassen Meltdown und Spectre erweitert (vgl. Abschnitt 2.8), die im Januar 2018 bekannt geworden sind. Es handelt sich dabei um gravierende Sicherheitsprobleme, die nahezu alle gängigen Mikroprozessor-Architekturen betreffen. Kapitel 14 wurde um zwei größere Abschnitte erweitert. Zum einen wurde das Signal-Protokoll (vgl. Abschnitt 14.7) aufgenommen, das in bekannten Messengern wie WhatsApp oder Facebook Messenger genutzt wird, um eine Ende-zu-Ende-Verschlüsselung in Messaging-Diensten zu ermöglichen. Zum anderen wurde das Kapitel um die Darstellung der Blockchain-Technologie und von Bitcoin in Abschnitt 14.8 erweitert. Das Internet of Things (IoT) nimmt zunehmend Gestalt an und erfordert geeignete Sicherheitstechnologien. ZigBee ist ein standardisiertes Protokoll für die energiesparende Funkkommunikation zwischen IoT-Geräten und wird in Kapitel 15 in Abschnitt 15.6 ausführlich erläutert.

Mit Kapitel 13 wurde ein neues Kapitel aufgenommen, das zwei ausführliche Fallstudien enthält. Diese sind die Sicherheitsarchitektur von Apple iOS einschließlich ApplePay und HomeKit sowie die Sicherheitsarchitektur von Windows 10.

Um den Umfang des Buches durch die zusätzlich aufgenommenen Aspekte nicht zu vergrößern, wurden im Gegenzug einige Abschnitte gekürzt bzw. überholte Abschnitte gestrichen. Dazu zählen beispielsweise der Abschnitt zur OSI-Sicherheitsarchitektur, die Beschreibung des PEM-Protokoll (eMail), die Ausführungen zum sprachbasierten Schutz oder auch die Beschreibung des veralteten Hashverfahrens MD5.

Bedanken möchte ich mich ganz herzlich bei allen Lesern, die mir Hinweise auf Fehler und Unklarheiten zukommen ließen, sowie bei Frau Sperlich vom DeGruyter-Verlag für ihre Unterstützung.

Für Hinweise auf Fehler und für Verbesserungsvorschläge bin ich jederzeit dankbar. Bitte senden Sie diese an claudia.eckert@aisec.fraunhofer.de.

München, im April 2018

Claudia Eckert

Inhaltsverzeichnis

1	Einführung	1
1.1	Grundlegende Begriffe	3
1.2	Schutzziele	7
1.3	Schwachstellen, Bedrohungen, Angriffe	15
1.3.1	Bedrohungen	17
1.3.2	Angriffs- und Angreifer-Typen	18
1.3.3	Rechtliche Rahmenbedingungen	25
1.4	Computer Forensik	30
1.5	Sicherheitsrichtlinie	31
1.6	Sicherheitsinfrastruktur	34
2	Spezielle Bedrohungen	43
2.1	Einführung	43
2.2	Buffer-Overflow	45
2.2.1	Einführung	46
2.2.2	Angriffe	48
2.2.3	Gegenmaßnahmen	51
2.3	Computerviren	54
2.3.1	Eigenschaften	54
2.3.2	Viren-Typen	56
2.3.3	Gegenmaßnahmen	63
2.4	Würmer	65
2.5	Trojanisches Pferd	71
2.5.1	Eigenschaften	71
2.5.2	Gegenmaßnahmen	73
2.6	Bot-Netze und Spam	75
2.6.1	Bot-Netze	75
2.6.2	Spam	77
2.7	Mobile Apps	79
2.7.1	Sicherheitsbedrohungen	80
2.7.2	Gegenmaßnahmen	82
2.8	Meltdown- und Spectre-Angriffsklassen	83
2.8.1	Einführung	83

2.8.2	Background	85
2.8.3	Angriffsklassen	88
3	Internet-(Un)Sicherheit	97
3.1	Einführung	97
3.2	Internet-Protokollfamilie	99
3.2.1	ISO/OSI-Referenzmodell	99
3.2.2	Das TCP/IP-Referenzmodell	105
3.2.3	Das Internet-Protokoll IP	107
3.2.4	Das Transmission Control Protokoll TCP	113
3.2.5	Das User Datagram Protocol UDP	115
3.2.6	DHCP und NAT	116
3.3	Sicherheitsprobleme	119
3.3.1	Sicherheitsprobleme von IP	119
3.3.2	Sicherheitsprobleme von ICMP	125
3.3.3	Sicherheitsprobleme von ARP	127
3.3.4	Sicherheitsprobleme mit IPv6	128
3.3.5	Sicherheitsprobleme von UDP und TCP	130
3.4	Sicherheitsprobleme von Netzdiensten	134
3.4.1	Domain Name Service (DNS)	135
3.4.2	Network File System (NFS)	140
3.4.3	Weitere Dienste	146
3.5	Web-Anwendungen	150
3.5.1	World Wide Web (WWW)	151
3.5.2	Sicherheitsprobleme	157
3.5.3	OWASP Top-Ten Sicherheitsprobleme	163
4	Security Engineering	171
4.1	Entwicklungsprozess	172
4.1.1	Allgemeine Konstruktionsprinzipien	172
4.1.2	Phasen	173
4.1.3	BSI-Sicherheitsprozess	174
4.2	Strukturanalyse	177
4.3	Schutzbedarfsermittlung	179
4.3.1	Schadenszenarien	179
4.3.2	Schutzbedarf	182
4.4	Bedrohungsanalyse	184
4.4.1	Bedrohungsmatrix	184
4.4.2	Bedrohungsbaum	186
4.5	Risikoanalyse	191
4.5.1	Attributierung	192
4.5.2	Penetrationstests	197

4.6	Sicherheitsarchitektur und Betrieb	199
4.6.1	Sicherheitsstrategie und Sicherheitsmodell	199
4.6.2	Systemarchitektur und Validierung	200
4.6.3	Aufrechterhaltung im laufenden Betrieb	200
4.7	Sicherheitsgrundfunktionen	201
4.8	Realisierung der Grundfunktionen	205
4.9	Security Development Lifecycle (SDL)	207
4.9.1	Die Entwicklungsphasen	208
4.9.2	Bedrohungs- und Risikoanalyse	209
5	Bewertungskriterien	213
5.1	TCSEC-Kriterien	213
5.1.1	Sicherheitsstufen	214
5.1.2	Kritik am Orange Book	215
5.2	IT-Kriterien	216
5.2.1	Mechanismen	217
5.2.2	Funktionsklassen	218
5.2.3	Qualität	218
5.3	ITSEC-Kriterien	219
5.3.1	Evaluationsstufen	220
5.3.2	Qualität und Bewertung	221
5.4	Common Criteria	222
5.4.1	Überblick über die CC	223
5.4.2	CC-Funktionsklassen	227
5.4.3	Schutzprofile	229
5.4.4	Vertrauenswürdigkeitsklassen	232
5.5	Zertifizierung	237
6	Sicherheitsmodelle	239
6.1	Modell-Klassifikation	239
6.1.1	Objekte und Subjekte	240
6.1.2	Zugriffsrechte	241
6.1.3	Zugriffsbeschränkungen	242
6.1.4	Sicherheitsstrategien	242
6.2	Zugriffskontrollmodelle	244
6.2.1	Zugriffsmatrix-Modell	244
6.2.2	Rollenbasierte Modelle	252
6.2.3	Chinese-Wall Modell	260
6.2.4	Bell-LaPadula Modell	265
6.3	Informationsflussmodelle	272
6.3.1	Verbands-Modell	272
6.4	Fazit und Ausblick	275

7	Kryptografische Verfahren	279
7.1	Einführung	279
7.2	Steganografie	281
7.2.1	Linguistische Steganografie	282
7.2.2	Technische Steganografie	283
7.3	Grundlagen kryptografischer Verfahren	285
7.3.1	Kryptografische Systeme	285
7.3.2	Anforderungen	290
7.4	Informationstheorie	291
7.4.1	Stochastische und kryptografische Kanäle	291
7.4.2	Entropie und Redundanz	293
7.4.3	Sicherheit kryptografischer Systeme	295
7.5	Symmetrische Verfahren	300
7.5.1	Permutation und Substitution	300
7.5.2	Block- und Stromchiffren	302
7.5.3	Betriebsmodi von Blockchiffren	308
7.5.4	Data Encryption Standard	317
7.5.5	AES	326
7.6	Asymmetrische Verfahren	331
7.6.1	Eigenschaften	331
7.6.2	Das RSA-Verfahren	335
7.7	Elliptische Kurven Kryptografie (ECC)	347
7.7.1	Grundlagen	348
7.7.2	Einsatz elliptischer Kurven	353
7.8	Kryptoanalyse	358
7.8.1	Klassen kryptografischer Angriffe	358
7.8.2	Substitutionschiffren	360
7.8.3	Differentielle Kryptoanalyse	362
7.8.4	Lineare Kryptoanalyse	363
8	Hashfunktionen und elektronische Signaturen	365
8.1	Hashfunktionen	365
8.1.1	Grundlagen	366
8.1.2	Blockchiffren-basierte Hashfunktionen	372
8.1.3	Dedizierte Hashfunktionen	373
8.1.4	Message Authentication Code	376
8.2	Elektronische Signaturen	380
8.2.1	Anforderungen	381
8.2.2	Erstellung elektronischer Signaturen	382
8.2.3	Digitaler Signaturstandard (DSS)	386
8.2.4	Rechtliche Rahmen	390

9	Schlüsselmanagement	397
9.1	Zertifizierung	397
9.1.1	Zertifikate	398
9.1.2	Zertifizierungsstelle	399
9.1.3	Public-Key Infrastruktur	403
9.2	Schlüsselerzeugung und -aufbewahrung	410
9.2.1	Schlüsselerzeugung	410
9.2.2	Schlüsselspeicherung und -vernichtung	412
9.3	Schlüsselaustausch	415
9.3.1	Schlüsselhierarchie	416
9.3.2	Naives Austauschprotokoll	418
9.3.3	Protokoll mit symmetrischen Verfahren	420
9.3.4	Protokoll mit asymmetrischen Verfahren	423
9.3.5	Leitlinien für die Protokollentwicklung	425
9.3.6	Diffie-Hellman Verfahren	428
9.4	Schlüsselrückgewinnung	435
9.4.1	Systemmodell	436
9.4.2	Grenzen und Risiken	439
10	Authentifikation	443
10.1	Einführung	443
10.2	Authentifikation durch Wissen	445
10.2.1	Passwortverfahren	446
10.2.2	Authentifikation in Unix	459
10.2.3	Challenge-Response-Verfahren	465
10.2.4	Zero-Knowledge-Verfahren	469
10.3	Biometrie	472
10.3.1	Einführung	472
10.3.2	Biometrische Techniken	474
10.3.3	Biometrische Authentifikation	477
10.3.4	Fallbeispiel: Fingerabdruckerkennung	480
10.3.5	Sicherheit biometrischer Techniken	482
10.4	Authentifikation in verteilten Systemen	486
10.4.1	RADIUS	486
10.4.2	Kerberos-Authentifikationssystem	491
11	Digitale Identität	503
11.1	Smartcards	503
11.1.1	Smartcard-Architektur	504
11.1.2	Betriebssystem und Sicherheitsmechanismen	507
11.1.3	Smartcard-Sicherheit	510

11.2	Elektronische Identifikationsausweise	515
11.2.1	Elektronischer Reisepass (ePass)	515
11.2.2	Personalausweis	535
11.3	Universal Second Factor Authentication	554
11.3.1	Registrierung eines U2F-Devices	556
11.3.2	Login beim Web-Dienst	559
11.3.3	Sicherheitsbetrachtungen	563
11.3.4	U2F-Protokoll versus eID-Funktion	570
11.4	Trusted Computing	573
11.4.1	Trusted Computing Platform Alliance	574
11.4.2	TCG-Architektur	575
11.4.3	TPM 1.2	580
11.4.4	Sicheres Booten	594
11.5	Physically Unclonable Functions (PUF)	604
11.5.1	Einführung	605
11.5.2	Einsatz von PUFs in Sicherheitsprotokollen	610
11.5.3	Sicherheitsuntersuchungen von PUFs	613
12	Zugriffskontrolle	615
12.1	Einleitung	615
12.2	Speicherschutz	616
12.2.1	Betriebsmodi und Adressräume	616
12.2.2	Virtueller Speicher	618
12.3	Objektschutz	622
12.3.1	Zugriffskontrolllisten	623
12.3.2	Zugriffsausweise	627
12.4	Zugriffskontrolle in Unix	632
12.4.1	Identifikation	632
12.4.2	Rechtevergabe	633
12.4.3	Zugriffskontrolle	638
12.5	Systembestimmte Zugriffskontrolle	642
12.6	Service-orientierte Architektur	644
12.6.1	Konzepte und Sicherheitsanforderungen	644
12.6.2	Web-Services	647
12.6.3	Web-Service Sicherheitsstandards	649
12.6.4	SAML	656
13	Fallstudien: iOS-Ecosystem und Windows10	663
13.1	iOS-Ecosystem	663
13.1.1	iOS-Sicherheitsarchitektur im Überblick	664
13.1.2	Sichere Enklave	666
13.1.3	Touch ID	667

13.1.4	Systemsicherheit	669
13.1.5	Passcode	671
13.1.6	Dateischutz	671
13.1.7	Keybags	680
13.1.8	Keychain	682
13.1.9	App-Sicherheit	683
13.1.10	Apple Pay	686
13.1.11	HomeKit-Framework	691
13.2	Windows 10	695
13.2.1	Architektur-Überblick	695
13.2.2	Sicherheits-Subsystem	699
13.2.3	Datenstrukturen zur Zugriffskontrolle	702
13.2.4	Zugriffskontrolle	707
13.2.5	Encrypting File System (EFS)	709
14	Sicherheit in Netzen	715
14.1	Firewall-Technologie	716
14.1.1	Einführung	716
14.1.2	Paketfilter	719
14.1.3	Proxy-Firewall	728
14.1.4	Applikationsfilter	731
14.1.5	Architekturen	734
14.2	Sichere Kommunikation	740
14.2.1	Verschlüsselungs-Layer	741
14.2.2	Virtual Private Network (VPN)	747
14.3	IPSec	751
14.3.1	Überblick	753
14.3.2	Security Association und Policy-Datenbank	755
14.3.3	AH-Protokoll	760
14.3.4	ESP-Protokoll	763
14.3.5	Schlüsselaustauschprotokoll IKE	767
14.3.6	Sicherheit von IPSec	772
14.4	TLS/SSL	778
14.4.1	Überblick	779
14.4.2	Handshake-Protokoll	781
14.4.3	Record-Protokoll	784
14.4.4	Sicherheit von TLS	787
14.5	DNSSEC	796
14.5.1	DNS-Schlüssel und -Schlüsselmanagement	796
14.5.2	DNS-Anfrage unter DNSSEC	799
14.6	Elektronische Mail	801
14.6.1	S/MIME	801

14.6.2	Pretty Good Privacy (PGP)	806
14.7	Signal-Protokoll für Messaging-Dienste	814
14.7.1	Extended Triple Diffie-Hellman (X3DH)	815
14.7.2	Double Ratchet-Protokoll	819
14.8	Blockchain	826
14.8.1	Technische Grundlagen	828
14.8.2	Smart Contracts	836
14.8.3	Sicherheit von Blockchains	838
14.8.4	Fallbeispiel: Bitcoin	841
14.8.5	Fazit und kritische Einordnung	846
15	Sichere mobile und drahtlose Kommunikation	851
15.1	GSM	852
15.1.1	Grundlagen	852
15.1.2	GSM-Grobarchitektur	853
15.1.3	Identifikation und Authentifikation	854
15.1.4	Gesprächsverschlüsselung	858
15.1.5	Sicherheitsprobleme	861
15.1.6	GPRS	865
15.2	UMTS	867
15.2.1	UMTS-Sicherheitsarchitektur	868
15.2.2	Authentifikation und Schlüsselvereinbarung	870
15.2.3	Vertraulichkeit und Integrität	874
15.3	Long Term Evolution (LTE) und SAE	876
15.3.1	EPC und LTE	878
15.3.2	Interworking	881
15.3.3	Sicherheitsarchitektur und Sicherheitsdienste	882
15.3.4	Sicheres Interworking	888
15.4	Funk-LAN (WLAN)	891
15.4.1	Einführung	891
15.4.2	Technische Grundlagen	893
15.4.3	WLAN-Sicherheitsprobleme	897
15.4.4	WEP und WPA	899
15.4.5	802.11i Sicherheitsdienste (WPA2)	903
15.4.6	802.1X-Framework und EAP	914
15.5	Bluetooth	920
15.5.1	Einordnung und Abgrenzung	921
15.5.2	Technische Grundlagen	922
15.5.3	Sicherheitsarchitektur	927
15.5.4	Schlüsselmanagement	932
15.5.5	Authentifikation	937
15.5.6	Bluetooth-Sicherheitsprobleme	940
15.5.7	Secure Simple Pairing	943

15.6	ZigBee	949
15.6.1	Überblick	949
15.6.2	Sicherheitsarchitektur	952
15.6.3	Schlüsseltypen	953
15.6.4	Netzzutritt und Schlüsselmanagement	956
15.6.5	ZigBee 3.0	958
15.6.6	Sicherheitsbetrachtungen	963
Literaturverzeichnis		969
Abkürzungsverzeichnis		983
Index		993

1 Einführung

Informations- und Kommunikationstechnologie (IKT) ist heute in nahezu allen Bereichen von zentraler Bedeutung. Eingebettete Systeme, Machine-to-Machine (M2M) Kommunikation, Vernetzung, aber auch on-demand beziehbare Mehrwertdienste aus der Cloud sind zentrale Wachstumstreiber einer immer stärker digitalisierten Wirtschaft. Der IT-Sicherheit kommt hierbei eine Schlüsselrolle zu.

IT-Sicherheit hat die Aufgabe, Unternehmen und deren Werte (Know-How, Kundendaten, Personaldaten) zu schützen und wirtschaftliche Schäden, die durch Vertraulichkeitsverletzungen, Manipulationen oder auch Störungen der Verfügbarkeit von Diensten des Unternehmens entstehen können, zu verhindern. Da eine vollständige Vermeidung oder Verhinderung von Angriffen in der Praxis nicht möglich ist, umfasst das Gebiet der IT-Sicherheit insbesondere auch Maßnahmen und Konzepte, um das Ausmaß potentieller Schäden, die durch Sicherheitsvorfälle entstehen können, zu reduzieren und damit die Risiken beim Einsatz von IKT-Systemen zu verringern. Schwachstellen und konkrete Angriffe oder Angriffsversuche auf IKT-Systeme müssen frühzeitig und mit möglichst hoher Präzision erkannt werden können und auf eingetretene Schadensfälle muss mit geeigneten technischen Maßnahmen reagiert werden. Techniken der Angriffserkennung und -Reaktion gehören deshalb ebenso zur IT-Sicherheit, wie methodische Grundlagen, um IKT-Systeme so zu entwickeln, dass sie qua Design ein hohes Maß an Sicherheit bieten. Man spricht in diesem Zusammenhang auch oft von *Secure by Design*. Gleichzeitig dient IT-Sicherheit als Enabling-Technologie. Sie ermöglicht die Entwicklung neuer, vertrauenswürdiger Anwendungen und Dienstleistungen verbunden mit innovativen Geschäftsmodellen beispielsweise im Gesundheitsbereich, bei Automotive-Anwendungen oder aber auch in zukünftigen, intelligenten Umgebungen, wie Smart Grid, Smart Factory, Smart Health oder auch Smart Cities.

Zudem dienen die Technologien und Verfahren der IT-Sicherheit dazu, die Sicherheit in einem allgemeineren Sinn (u.a. im Sinne öffentlicher Sicherheit) zu erhöhen. Zur Überwachung von beispielsweise kritischen Infrastrukturen, wie Kraftwerken, Produktionsanlagen, Transportleitsysteme etc. wird bereits heute eine Vielzahl von Sensoren eingesetzt, die

kontinuierlich Umgebungsdaten erfassen, Daten austauschen und die das Ablaufverhalten bzw. den operativen Betrieb komplexer Anlagen permanent überwachen. Probleme und auffälliges Verhalten werden an zumeist zentral organisierte Leitstellen gemeldet, so dass auf der Basis dieser Steuerungsdaten kontrollierend in kritische Abläufe (zum Teil vollständig automatisiert) eingegriffen wird. Es ist unmittelbar klar, dass die zur Steuerung und Kontrolle verwendeten Daten vor Manipulationen zu schützen sind. Sie müssen zudem rechtzeitig und vollständig vorliegen. Vielfach ist es zudem aus Gründen des Datenschutzes auch notwendig, die Daten vertraulich zu verarbeiten, man denke nur an die aktuelle Diskussion zu den digital erfassten und per Datenkommunikation übermittelten Stromverbrauchsdaten privater Haushalte. IT-Sicherheitskonzepte sind die wichtige Basis, damit die zur Überwachung und Steuerung der Anlagen eingesetzten IKT-Komponenten auch verlässlich für sicherheitskritische Anwendungen eingesetzt werden können.

Das vorliegende Buch hat zum Ziel, fundierte Kenntnisse über Funktionsweise und Wirksamkeit existierender Maßnahmen zur Absicherung von IT Systemen zu vermitteln. Das vorliegende Kapitel führt hierzu die notwendigen generellen Grundlagen und Begrifflichkeiten ein.

Kapitelüberblick

In Abschnitt 1.1 werden die grundlegenden Begriffe und Definitionen eingeführt, die für den Bereich sicherer IT-Systeme von Bedeutung sind. Datenobjekte und Informationen gehören zu den zu schützenden Gütern eines technischen Systems und bilden die Grundlage zur Festlegung der Schutzziele. Das sind Sicherheitsanforderungen, die an ein System gestellt werden und die durch die Sicherheitseigenschaften des Systems schlussendlich zu gewährleisten sind. Die wichtigsten dieser Eigenschaften werden in Abschnitt 1.2 vorgestellt. Aktionen und Ereignisse, die die Sicherheitseigenschaften eines Systems in Frage stellen, sind Angriffe auf das System. Angriffe nutzen Schwachstellen des Systems aus. Abschnitt 7.8.1 erklärt die in diesem Kontext interessierenden Begriffe und erläutert deren Zusammenhänge. Mit der Zunahme der Computerkriminalität gewinnen Maßnahmen an Bedeutung, durch die elektronische Spuren eines digitalen Einbruchs gesichert werden können, um sie in einem Rechtsstreit oder zur Strafverfolgung verwerten zu können. Mit diesem relativ jungen Zweig der IT-Sicherheit, der Computer Forensik, beschäftigt sich Abschnitt 1.4. Die einzuhaltenden Sicherheitseigenschaften eines IT-Systems werden formal oder informell in einer Sicherheitsstrategie festgelegt. Abschnitt 1.5 gibt einen ersten Einblick in Sicherheitsstrategien. Abschnitt 1.6 fasst abschließend die wichtigsten Komponenten eines sicheren IT-Systems zusammen.

1.1 Grundlegende Begriffe

Dieser Abschnitt dient dazu, die für das Folgende wesentlichen Begriffe und Termini einzuführen. Da in der Literatur keine einheitliche Begriffsbildung verwendet wird, ist eine entsprechende Festlegung und Einführung notwendig. In diesem Buch beschäftigen wir uns mit der Sicherheit informationstechnischer Systeme, kurz IT-Systeme, so dass wir zunächst den Begriff des IT-Systems präzisieren.

Definition 1.1 (IT-System)

Ein IT-System ist ein geschlossenes oder offenes, dynamisches technisches System mit der Fähigkeit zur Speicherung und Verarbeitung von Informationen.

IT-System

□

Unter einem geschlossenen System verstehen wir ein System, das auf der Technologie eines Herstellers aufbaut, zu Konkurrenzprodukten nicht kompatibel ist und dessen Ausdehnung sich auf einen bestimmten Teilnehmerkreis und ein bestimmtes räumliches Gebiet beschränkt. Geschlossene Systeme sind in der Regel homogen und werden zentral verwaltet. Unter offenen Systemen verstehen wir vernetzte, physisch verteilte Systeme, die sich an Standards zum Informationsaustausch mit anderen Systemen orientieren. Jedes Teilsystem ist offen für die Kommunikation mit anderen Systemen. Offene Systeme sind meist durch heterogene Hardwarekomponenten und Betriebssysteme gekennzeichnet, für die keine zentrale Administration und Verwaltung zur Verfügung steht.

geschlossen

offen

Die im vorliegenden Buch behandelten Sicherheitseigenschaften von IT-Systemen gelten sowohl für geschlossene als auch für offene Systeme. Die Offenheit eines Systems erfordert jedoch erweiterte und andere Realisierungsmaßnahmen zur Gewährleistung der Sicherheitseigenschaften als sie in einer geschlossenen Umgebung benötigt werden. Der von uns verwendete Systembegriff bezieht sich auf den weiten Bereich der Systeme der Informations- und Kommunikationstechnik (IuK), wozu Betriebssysteme und Kommunikationssysteme ebenso zählen wie Datenverarbeitungssysteme für spezifische Anwendungsbereiche (u.a. ein Krankenhausinformationsystem, eine Homebanking-Anwendung oder ein Telearbeitsplatz).

IT-Systeme sind Bestandteile soziotechnischer Systeme. Das heißt, sie sind eingebettet in gesellschaftliche, unternehmerische und auch politische Strukturen und werden von Benutzern mit sehr unterschiedlichem technischen Know-how und für sehr unterschiedliche Zwecke genutzt. Bei der Betrachtung der technischen Aspekte der IT-Sicherheit muss man

soziotechnisches System

sich dieser Einbettung bewusst sein. Fragen der Nutzbarkeit und Akzeptanz von Sicherheitstechnologien sind hier ebenso zu betrachten, wie die Einhaltung von gesetzlichen und organisatorischen Regelungen und Vorschriften (u.a. EU-Datenschutzgrundverordnung, unternehmensspezifische Geheimhaltungsvorschriften, Betriebsverfassungsgesetz). Das vorliegende Buch beschränkt sich jedoch in erster Linie auf die Behandlung der technischen Aspekte der IT-Sicherheit.

Information und Datenobjekte

Objekt IT-Systeme speichern und verarbeiten Informationen. Die Information ist aber ein Abstraktum, das in Form von Daten bzw. Datenobjekten repräsentiert wird. Wir unterscheiden zwischen passiven Objekten (z.B. Datei, Datenbankeintrag) mit der Fähigkeit, Informationen zu speichern, und aktiven Objekten (z.B. Prozesse) mit der Fähigkeit, sowohl Informationen zu speichern als auch zu verarbeiten. Informationen und die Objekte, die sie repräsentieren, sind schützenswerte Güter (engl. *asset*) eines Systems.

Information Die Information, die durch ein Datum repräsentiert wird, ergibt sich aus einer festgelegten Interpretationsvorschrift. So kann beispielsweise ein Integer-Datenobjekt einen numerischen Wert besitzen, der, abhängig von der Interpretationsvorschrift, zum Beispiel den Kontostand eines Bankkontos, eine Speicheradresse oder aber auch eine Entfernungswert repräsentiert. Ferner ist es möglich, dass ein und dieselbe Information in unterschiedlichen Datenobjekten repräsentiert ist. Ein geheimes Passwort kann zum Beispiel als eine Folge von Binärzeichen auf der Festplatte in einem Plattenblock (Datenobjekt) gespeichert sein, oder nach der Eingabe durch den Benutzer im Tastatureingabepuffer (Datenobjekt) stehen oder als Nutzdaten in einem IP-Nachrichtenpaket (Datenobjekt) enthalten sein. Will man also die Information als abstraktes Gut schützen, muss man sich darüber klar werden, in welchen verschiedenen Datenobjekten diese Information im System repräsentiert wird. Eine Nichtbeachtung dieser Zusammenhänge hat in der Vergangenheit wiederholt zu Sicherheitsproblemen geführt. Ein Beispiel dafür ist das verschlüsselnde Dateisystem EFS unter Windows (siehe auch Abschnitt 13.2.5). EFS stellt Funktionen zur Verfügung, über die sensible Inhalte von Dateien nach der Bearbeitung verschlüsselt auf der Festplatte abgelegt werden können, damit sie nicht mehr im Klartext im System verfügbar sind. In den ersten Versionen dieser Software wurde jedoch vernachlässigt, dass das Speichermanagement des Betriebssystems aus Effizienzgründen häufig genutzte Daten in einem Schnellzugriffspeicher (engl. *cache*) ablegt. Die sensible Information war somit auch in Cache-Blöcken repräsentiert und aufgrund fehlender Bereinigungsoperationen auch nach dem Schließen der Datei noch für einige Zeit im Klartext im System vorhanden. Ein aktuelles Beispiel sind die in 2018 bekannt gewordenen Probleme bei Prozessoren,

die durch die Meltdown und Spectre-Angriffe (vgl. Abschnitt 2.8.1) ausgenutzt werden können. Hintergrund sind performanzsteigernde Maßnahmen, die der Prozessor durchführt, die aber mit Seitenkanalangriffen für Angriffe ausgenutzt werden können.

Datenobjekte sind über wohl definierte Schnittstellen in Form von Operationen bzw. Methoden von anderen Objekten des Systems oder von der Umwelt des Systems benutzbar. Zur Umwelt gehören insbesondere seine Benutzer. Die Benutzer eines Systems und alle Objekte, die im Auftrag von Benutzern im System aktiv sein können, wie z.B. Prozesse, Server und Prozeduren, werden als die Subjekte des Systems bezeichnet.

Subjekt

Eine Interaktion zwischen einem Subjekt und einem Objekt in einem System, durch die ein Informationsfluss zwischen Subjekt und Objekt auftritt, nennen wir einen Zugriff auf die Information. Jeder Zugriff auf ein Datenobjekt ist gleichzeitig auch ein Zugriff auf die dadurch repräsentierte Information. Für den Zugriff auf die zu schützende Information bzw. auf die sie repräsentierenden Daten eines IT-Systems sind Zugriffsrechte festzulegen und an die Subjekte zu vergeben. Besitzt ein Subjekt die Berechtigung zum Zugriff auf eine Information bzw. auf ein Datenobjekt, so sagen wir, dass das Subjekt zu diesem Zugriff autorisiert ist.

Zugriffsrecht

Informationskanäle

Die in einem System potentiell auftretenden Kanäle, über die Informationen fließen können, werden nach Speicherkanälen, legitimen und verdeckten Kanälen klassifiziert. Die legitimen Kanäle sind diejenigen, die ein Subjekt in der Regel für den Informationsaustausch nutzt (u.a. Nachrichten und Parameter bei Operationsaufrufen). Unter Speicherkanälen versteht man Objekte, die von Subjekten gemeinsam benutzt werden können (u.a. Dateien). Verdeckte Kanäle (engl. *covert channel*) sind solche, die nicht für einen Informationstransfer vorgesehen sind, aber dazu missbraucht werden können. Zum Beispiel kann durch die Veränderung der Ausführungszeit eines Programms eine verdeckte Information an den Beobachter der Ausführungszeiten übermittelt werden. So könnte beispielsweise die Erhöhung der Ausführungszeit signalisieren, dass ein Konkurrent ein preiswerteres Angebot erstellt hat als der Empfänger dieses Signals. Aufgrund der Komplexität heutiger Software-Systeme mit ihrer Vielzahl von direkten und indirekten Wechselwirkungen zwischen verschiedenen Systemkomponenten sind verdeckte Kanäle nicht vollständig zu verhindern. Durch eine methodische Systemkonstruktion kann jedoch dafür gesorgt werden, dass die Bandbreite verdeckter Kanäle, also die Menge der Informationen, die über diese Kanäle transferiert werden kann, beschränkt ist. So ist der Informationsgehalt eines 1-Bit Signals (Ja/Nein) selbstverständlich deutlich geringer, als der einer

legitimer Kanal

verdeckter Kanal

Nachricht von mehreren Byte Länge oder von einer Datei mit einer Größe von mehreren KByte.

Auf der Basis der getroffenen Festlegungen können wir nun die wichtigsten Sicherheitsbegriffe einführen.

Definition 1.2 (Sicherheit)

funktionssicher

Unter Funktionssicherheit (engl. *safety*) eines Systems verstehen wir die Eigenschaft, dass die realisierte Ist-Funktionalität der Komponenten mit der spezifizierten Soll-Funktionalität übereinstimmt. Ein funktionssicheres System nimmt keine funktional unzulässigen Zustände an. Anders formuliert verstehen wir unter der Funktionssicherheit eines Systems, dass es unter allen (normalen) Betriebsbedingungen funktioniert.

informationssicher

Die Informationssicherheit (engl. *security*) ist die Eigenschaft eines funktionssicheren Systems, nur solche Systemzustände anzunehmen, die zu keiner unautorisierten Informationsveränderung oder -gewinnung führen.

Datensicherheit

Die Datensicherheit (engl. *protection*) ist die Eigenschaft eines funktionssicheren Systems, nur solche Systemzustände anzunehmen, die zu keinem unautorisierten Zugriff auf Systemressourcen und insbesondere auf Daten führen. Damit umfasst die so beschriebene Sicherheit der Daten insbesondere auch Maßnahmen zur Datensicherung (engl. *backup*), also den Schutz vor Datenverlusten durch Erstellung von Sicherungskopien.

Datenschutz

Unter dem Begriff Datenschutz im engeren Sinn (engl. *privacy*), wie er unter anderem durch das deutsche Bundesdatenschutzgesetz (BDSG) [45] bzw. die seit Mai 2018 geltende EU-Datenschutzgrundverordnung, bzw. General Data Protection Regulation¹ (GDPR) festgelegt wird, versteht man die Fähigkeit einer natürlichen Person, die Weitergabe von Informationen, die sie persönlich betreffen, zu kontrollieren. In diesen Bereich fallen insbesondere Sicherheitsanforderungen, die der deutsche Gesetzgeber durch das informationelle Selbstbestimmungsrecht geregelt hat.

□

Mit den getroffenen Festlegungen können wir zwischen Systemen, die vordringlich auf den Schutz der verarbeiteten Informationen (Informationssicherheit) und solchen, die auf den Schutz der Daten als deren Repräsentanten (Datensicherheit) abzielen, unterscheiden. Damit ist eine Präzisierung vorgenommen, die in den meisten Abhandlungen zu dieser Thematik fehlt. Herkömmlicherweise wird unter der Sicherheit eines Systems stets dessen Datensicherheit verstanden. Die eingeführte Definition ist auch eine

¹ <https://www.eugdpr.org/>

Präzisierung der Sicherheits-Begriffe im Vergleich zu dem allgemeinen, durch die ISO (International Standards Organization) geprägten Sicherheitsbegriff [91]. Dort wird die Sicherheit (*security*) als „die Minimierung der Verwundbarkeit von Werten und Ressourcen“ definiert.

Aus den Festlegungen von Definition 1.2 ist ferner die wesentliche Erkenntnis abzuleiten, dass die Sicherstellung der Informations- und Datensicherheit eines IT-Systems ein Prozess ist (vgl. auch Kapitel 4.1), der einer ständigen Überprüfung und Revision unterliegt, da sich Systemeigenschaften über die Zeit dynamisch ändern können. Die Definition 1.2 verdeutlicht zudem, dass die Funktionssicherheit die Grundlage für die Informations- bzw. Datensicherheit eines Systems ist. Die Funktionssicherheit ist eng verwandt mit den Begriffen der Zuverlässigkeit bzw. der Verlässlichkeit.

Definition 1.3 (Verlässlichkeit)

Unter der Verlässlichkeit (engl. *dependability*) eines Systems verstehen wir die Eigenschaft, keine unzulässigen Zustände anzunehmen (Funktionssicherheit) und zu gewährleisten, dass die spezifizierte Funktion zuverlässig (engl. *reliability*) erbracht wird.

Verlässlichkeit

□

Zur Gewährleistung von Funktionssicherheit (Safety) setzt man Konzepte und Verfahren ein, die darauf abzielen, die Verlässlichkeit von IT-Systemen zu gewährleisten. Es handelt sich dabei im Wesentlichen um Maßnahmen zur Abwehr von solchen Bedrohungen, die durch das technische Fehlverhalten des IT-Systems selber (von innen) entstehen. Derartige Bedrohungen ergeben sich insbesondere durch Programmierfehler, die mit Techniken der Programmvalidierung oder -verifikation aufzudecken sind.

Safety

Im vorliegenden Buch beschäftigen wir uns mit der Sicherheit technischer Systeme im Sinne der englischen Begriffe *Security* und *Protection*. Behandelt werden Konzepte und Maßnahmen zur Abwehr von Bedrohungen, die durch unberechtigte Zugriffe auf die zu schützenden Güter des IT-Systems entstehen und im Wesentlichen von außen erfolgen. Anzumerken ist, dass die Grenzen zwischen Safety- und Security-Fragestellungen fließend sind und es durchaus Überlappungen gibt.

1.2 Schutzziele

Informationen bzw. Daten sind zu schützende Güter informationssicherer bzw. datensicherer Systeme. Der Zugriff auf diese ist zu beschränken und zu kontrollieren, so dass nur dazu autorisierten Subjekten ein Zugriff gewährt wird. Die Schutzziele, die diese Anforderungen präzisieren, sind

die Datenintegrität und Informationsvertraulichkeit. Zugreifende Subjekte müssen eindeutig identifiziert und ihre Identität muss verifiziert sein. Die entsprechende Eigenschaft nennt man die Authentizität von Subjekten. Ist ein Subjekt authentifiziert und berechtigt, also autorisiert, einen Zugriff auf ein Objekt bzw. eine Information durchzuführen, dann sollte das System gewährleisten, dass dieser Zugriff auch möglich ist; man spricht von der Eigenschaft der Verfügbarkeit. Hat ein Subjekt einen Zugriff bzw. eine Aktion durchgeführt, so ist es vielfach notwendig, dass auch noch im Nachhinein die Urheberschaft des Zugriffs bzw. der Aktion eindeutig dem entsprechenden Subjekt zuordenbar ist. Man spricht hier von der Verbindlichkeit oder Zuordenbarkeit des Systems. Die angesprochenen Schutzziele werden nun im Folgenden präzisiert.

Definition 1.4 (Authentizität)

Authentizität

Unter der Authentizität eines Objekts bzw. Subjekts (engl. *authenticity*) verstehen wir die Echtheit und Glaubwürdigkeit des Objekts bzw. Subjekts, die anhand einer eindeutigen Identität und charakteristischen Eigenschaften überprüfbar ist.

□

Authentifikation

Die Authentizität eines Subjekts bzw. Objekts wird durch Maßnahmen zur Authentifikation (engl. *authentication*) überprüft. Dazu muss nachgewiesen werden, dass eine behauptete Identität eines Objekts oder Subjekts mit dessen charakterisierenden Eigenschaften übereinstimmt.

Subjekt-Auth.

In herkömmlichen Systemen wird eine Authentifikation meist nur für Benutzer als Subjekte durchgeführt. Die Identifikation basiert auf der Vergabe von eindeutigen Benutzerkennungen (engl. *account*) oder Benutzernamen. Charakterisierende Eigenschaften zum Nachweis der Identität sind beispielsweise Passwörter, deren Kenntnis der Benutzer beim Systemzugang nachweisen muss, oder biometrische Merkmale wie Fingerabdrücke. Identitätsnachweise werden häufig allgemein als Credentials bezeichnet, womit man von dem spezifischen, zum Identitätsnachweis tatsächlich verwendeten Verfahren abstrahiert.

Objekt-Authentizität

Mit dem Übergang zu offenen Systemen werden auch zunehmend Maßnahmen erforderlich, die die Authentizität von Objekten, wie beispielsweise Web-Server, Access Points (bei 802.11 WLAN) oder Code nachweisen. Hierbei beschränkt man sich jedoch i.d.R. auf einfache Mechanismen, nämlich kryptografische Verfahren (vgl. Kapitel 7), um die Echtheit von Daten zu überprüfen, die über ein unsicheres Transportmedium wie dem Internet übertragen werden. Diese Echtheitsprüfung beschränkt sich auf einen Ursprungs- bzw. Urheber nachweis, ohne Aussagen über die Funktionalität

des Objekts zu treffen. Die Authentizität eines Objekts im engeren Sinn würde den Nachweis erfordern, dass seine spezifizierte Funktionalität mit seiner tatsächlich erbrachten übereinstimmt. Entsprechende Nachweise sind sehr schwierig zu führen und werden in der Praxis noch nicht eingesetzt. Hier sind noch weitere, verstärkte Forschungsaktivitäten notwendig, da gerade mit der ansteigenden Mobilität von Apps (vgl. Kapitel 2.7) ein Authentizitätsnachweis, der neben einem Ursprungsnachweis auch eine Aussage über eine korrekte Funktionalität beinhaltet, für die Sicherheit offener Systeme dringend erforderlich ist.

Definition 1.5 (Datenintegrität)

Wir sagen, dass das System die Datenintegrität (engl. *integrity*) gewährleistet, wenn es Subjekten nicht möglich ist, die zu schützenden Daten unautorisiert und unbemerkt zu manipulieren.

Integrität

□

Die Eigenschaft der Datenintegrität erfordert zum einen die Festlegung von Rechten zur Nutzung von Daten. Beispiele hierfür sind Lese- oder Schreibberechtigungen für Dateien oder das Recht, von einem bestimmten Konto einen Betrag bis zu einer festgelegten Obergrenze abheben zu dürfen. Zum anderen sind Rechte an Subjekte zu vergeben, so dass diese autorisiert sind, die entsprechenden Zugriffsrechte wahrzunehmen. Abhängig von den damit getroffenen Festlegungen können Integritätsaussagen unterschiedlicher Qualität gemacht werden. So wird beispielsweise durch die Vergabe von Schreibberechtigungen an Dateien die Möglichkeit zur Modifikation des Datei-Objekts nicht weiter beschränkt, so dass Subjekte zu beliebigen Manipulationen berechtigt sind. Auf dieser Basis sind nur eingeschränkte Aussagen über die Integrität im Sinne einer authentischen, korrekten Funktionalität des Daten-Objekts möglich. Legt man demgegenüber die Berechtigungen zur Nutzung von Objekten durch wohl definierte Methoden des Objekts fest, so werden die Nutzungsmöglichkeiten und damit die Manipulationsmöglichkeiten auf die Funktionalität dieser Zugriffsoperationen eingeschränkt. Auf formale Techniken zur Festlegung und Vergabe von Zugriffsrechten wird in Kapitel 6 eingegangen. Die benötigten Mechanismen und Verfahren zur Gewährleistung des Schutzzieles der Datenintegrität gehören zum Bereich der Zugriffskontrolle.

Rechtefestlegung

Definition 1.5 fordert, dass unautorisierte Manipulationen nicht unbemerkt bleiben dürfen. Das bedeutet, dass in Umgebungen, in denen eine solche Manipulation nicht a priori verhindert werden kann (z.B. in Netzen), Techniken erforderlich sind, mit deren Hilfe unautorisierte Manipulationen a posteriori erkennbar sind. Auf diese Weise kann verhindert werden, dass unautorisierte

Manipulations-
Erkennung

siert manipulierte Daten weiterverarbeitet werden und der mögliche Schaden begrenzt wird. Zur Erkennung von durchgeführten Datenveränderungen werden kryptografisch sichere Hashfunktionen (vgl. Kapitel 8.1) eingesetzt.

Definition 1.6 (Informationsvertraulichkeit)

Vertraulichkeit

Wir sagen, dass das System die Informationsvertraulichkeit (engl. *confidentiality*) gewährleistet, wenn es keine unautorisierte Informationsgewinnung ermöglicht.

□

Die Gewährleistung der Eigenschaft der Informationsvertraulichkeit erfordert in datensicheren Systemen die Festlegung von Berechtigungen und Kontrollen der Art, dass sichergestellt ist, dass Subjekte nicht unautorisiert Kenntnis von Informationen erlangen. In informationssicheren Systemen sind Maßnahmen zur Festlegung sowie zur Kontrolle zulässiger Informationsflüsse zwischen den Subjekten des Systems erforderlich, so dass ausgeschlossen werden kann, dass Information zu unautorisierten Subjekten „durchsickert“. Das entsprechende Problem wird Confinement-Problem [108] genannt. Mit den Festlegungen zur Kontrolle der Informationsflüsse ist spezifiziert, welche Subjekte Kenntnis von welchen Informationen erlangen dürfen. Formale Techniken zur Formulierung zulässiger bzw. verbotener Informationskanäle werden mit den Informationsflussmodellen in Kapitel 6 eingeführt.

Confinement

Im Zusammenhang mit Datenbanksystemen ergeben sich spezifische Vertraulichkeitsprobleme, auf die wir im Rahmen dieses Buches jedoch nicht weiter eingehen werden. Die entsprechenden Probleme sind unter dem Stichwort Interferenz-Kontrolle (engl. *interference control*) bekannt (u.a. [51]). Hierbei geht es darum, dass aus der Kenntnis von Einzelinformationen weitere Informationen abgeleitet werden können, wobei diese abgeleitete Information jedoch dem betreffenden Subjekt eigentlich nicht zugänglich sein darf. Man denke hier beispielsweise an Datenbankanfragen, die Auskünfte über Einzelpersonen ermöglichen, wobei aber die Anonymität der Personen gewahrt sein muss. Falls es durch geschicktes Formulieren entsprechender Anfragen gelingt, sehr kleine Antwortmengen zu erhalten, sind Rückschlüsse auf spezifische Personen unter Umständen möglich.

Interferenz

Anforderungen an die Informationsvertraulichkeit im weiteren Sinn werden durch Verschlüsselungstechniken (vgl. Kapitel 7) erfüllt. Hierbei besteht das Ziel darin, die Daten geeignet zu transformieren, so dass unautorisierte Dritte nicht in der Lage sind, ohne den korrekten Entschlüsselungsschlüssel die Daten sinnvoll zu interpretieren.

Beispiel 1.1 (Unzulässiger Informationsfluss)

Die unterschiedlichen Anforderungen für datensichere und informationssichere Systeme ergeben sich aus dem beschriebenen Zusammenhang zwischen Information als Abstraktum und den Daten als deren Repräsentation. Mit der Kontrolle der Datenzugriffe werden nämlich nicht notwendigerweise auch alle Informationszugriffe im erforderlichen Umfang kontrolliert. Man betrachte dazu beispielsweise ein Szenario mit zwei Benutzern Bill und Joe, zwei Datei-Objekten Datei_1 und Datei_2 und der Vertraulichkeitsanforderung, dass das Subjekt Joe keine Kenntnis über die durch die Datei_1 repräsentierten Informationen erlangen darf.

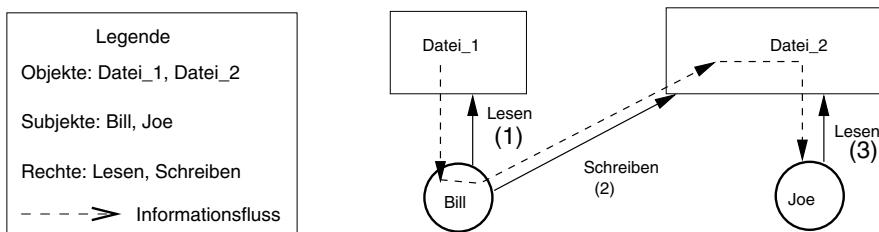


Abbildung 1.1: Unzulässiger Informationsfluss

Der Benutzer Bill sei berechtigt, schreibend auf Datei_2 sowie lesend auf Datei_1 zuzugreifen, während der Benutzer Joe nur lesend auf Datei_2 zugreifen darf. Erfolgen nur autorisierte Zugriffe gemäß dieser Rechtevergabe, so kann dennoch ein unzulässiger Informationsfluss auftreten. Abbildung 1.1 skizziert die Situation, die eintritt, wenn der Benutzer Bill zunächst Informationen aus Datei_1 ausliest (1) und diese anschließend in Datei_2 schreibt (2). Daraufhin erhält Benutzer Joe auf diese Informationen lesenden (3) Zugriff.



Zur Gewährleistung von Vertraulichkeitsanforderungen sind somit spezielle Maßnahmen erforderlich, die über eine reine Kontrolle der Zugriffe auf die Datenobjekte hinausgehen. Verwendet werden hierbei, neben kryptografischen Verfahren, insbesondere auch Labeling-Techniken, wodurch Datenobjekte eine spezielle Sicherheitseinstufung erhalten. Durch spezielle Kontrollen kann sichergestellt werden, dass Information, die z.B. als sensitiv eingestuft ist, nicht in unautorisierte Hände gelangt.

Für autorisierte und authentifizierte Subjekte ist zu gewährleisten, dass sie die ihnen zugebilligten Zugriffsrechte auch wahrnehmen können, d.h. dass das System die entsprechenden Leistungen (z.B. Zugriff auf einen Service, für dessen Nutzung der Kunde bezahlt hat) zur Verfügung stellt.

Definition 1.7 (Verfügbarkeit)

Verfügbarkeit

Wir sagen, dass das System die Verfügbarkeit (engl. *availability*) gewährleistet, wenn authentifizierte und autorisierte Subjekte in der Wahrnehmung ihrer Berechtigungen nicht unautorisiert beeinträchtigt werden können.

□

Da in einem IT-System in der Regel zu einem Zeitpunkt unterschiedliche Prozesse eines Benutzers oder auch verschiedener Benutzer um gemeinsame Ressourcen wie CPU-Zeit oder Ein- und Ausgabegeräte konkurrieren, kann es für die einzelnen Prozesse zu Ausführungsverzögerungen kommen. Verzögerungen, die aus „normalen“ Verwaltungsmaßnahmen resultieren (u.a. Prozess-Scheduling), werden als autorisierte Beeinträchtigungen betrachtet, die *a priori* noch keine Verletzung der Verfügbarkeit darstellen. Es ist klar, dass hier im Gegensatz zu den zuvor eingeführten Eigenschaften die Trennlinie zwischen unautorisierten und autorisierten Aktionen unscharf ist, da u.U. ein Angreifer durch autorisierte Zugriffe die Verwaltungsmaßnahmen des Systems beeinflussen und damit absichtlich einen Angriff auf die Verfügbarkeit hervorrufen kann. Dies kann beispielsweise durch das gezielte Monopolisieren der CPU mit einem hoch prioren Prozess geschehen. Entsprechend schwierig ist deshalb auch das Erkennen einer unautorisierten Beeinträchtigung. So kann ein hohes Datenaufkommen, das zu Stausituationen in einem Netz führt, sowohl auf normalen Datenverkehr als auch auf einen Angriff durch synthetisch erzeugte Nachrichten zurückzuführen sein.

Maßnahmen

Zur Gewährleistung der Verfügbarkeit sind Maßnahmen, wie die Einführung von Quoten, von Interesse, durch die die Nutzung von Systemressourcen wie beispielsweise CPU-Zeit oder Speicher reglementiert wird. Auf Maßnahmen zur Erfüllung von Verfügbarkeitsanforderungen gehen wir in diesem Buch jedoch nicht näher ein.

Schließlich bleibt noch die Eigenschaft der Verbindlichkeit zu präzisieren.

Definition 1.8 (Verbindlichkeit)

Verbindlichkeit

Wir sagen, dass das System die Verbindlichkeit bzw. Zuordenbarkeit (engl. *non repudiation*) einer Menge von Aktionen gewährleistet, wenn es nicht möglich ist, dass ein Subjekt im Nachhinein die Durchführung einer solchen Aktion abstreiten kann.

□

Verbindlichkeitseigenschaften sind besonders in dem rasant wachsenden Bereich des elektronischen Handels (engl. *Electronic Commerce*) und der elektronischen Geschäfte (engl. *Electronic Business*) von großer Bedeutung, um die Rechtsverbindlichkeit durchgeföhrter geschäftlicher Transaktionen

(Käufe, Verträge etc.) zu garantieren. Diese Anforderungen lassen sich durch den Einsatz digitaler Signaturen (vgl. Kapitel 8.2) erfüllen. Die Verbindlichkeit ist aber auch allgemein bei der Nutzung von Systemressourcen in Mehrbenutzersystemen oder auch zunehmend bei der Nutzung von Cloud-Plattformen von Interesse. Beispiele relevanter Aktionen sind hier der Verbrauch von Rechenzeit oder die Nutzung teurer Ein/Ausgabe-Geräte. Mit der Verbindlichkeitseigenschaft ist die Forderung nach Abrechenbarkeit (engl. *accountability*) unmittelbar verbunden. Dies erfordert Maßnahmen zur Überwachung (engl. *audit*) sowie zur Protokollierung einzelner Benutzeraktivitäten.

Accountability

Zusätzlich zu den betrachteten Sicherheitseigenschaften gewinnt im Kontext vernetzter und insbesondere mobiler Systeme zunehmend die Forderung, die Anonymität von Subjekten zu gewährleisten und deren Privatsphäre zu schützen, an Bedeutung.

Anonymität

Definition 1.9 (Anonymisierung und Pseudomisierung)

Unter der Anonymisierung versteht man das Verändern personenbezogener Daten der Art, dass die Einzelangaben über persönliche oder sachliche Verhältnisse nicht mehr oder nur mit einem unverhältnismäßig großen Aufwand an Zeit, Kosten und Arbeitskraft einer bestimmten oder bestimmbaren natürlichen Person zugeordnet werden können.

Anonymisierung

Eine schwächere Form der Anonymisierung stellt die Pseudomisierung dar. Dabei handelt es sich um das Verändern personenbezogener Daten durch eine Zuordnungsvorschrift (z.B. die Verwendung von Pseudonymen) derart, dass die Einzelangaben über persönliche oder sachliche Verhältnisse ohne Kenntnis oder Nutzung der Zuordnungsvorschrift nicht mehr einer natürlichen Person zugeordnet werden können.

Pseudomisierung

□

Bei jedem Internetzugriff (z.B. beim Surfen) fallen eine Vielzahl von Daten über persönliche und sachliche Verhältnisse des Nutzers an, obwohl diese häufig für die Abwicklung der Dienstprotokolle nicht erforderlich sind. Beispiele dafür sind die IP-Adresse des Nutzers, seine Herkunftsadresse, die URL der zuvor geladenen Web-Seite oder auch Datum und Uhrzeit des Zugriffs. Diese Daten können sowohl von Angreifern (Dritten), die die Datenleitungen abhören, als auch von Dienst-Anbietern dazu verwendet werden, unautorisiert Profile über Nutzer zu erstellen. Anonymisierungsdienste versuchen, durch Kombinationen aus Vermeidungs- (z.B. Daten unterdrücken) und Verschleierungstechniken (z.B. Ersetzen durch Standardmuster, Verschlüsseln) eine Anonymisierung oder Pseudomisierung zu erzielen. Pseudonymität erzielt man durch die Einführung von Pseudony-

Anonymisierungsdienste

men, so dass die Identität eines Subjekts zwar einem vertrauenswürdigen Dritten bekannt ist, nicht jedoch jedem Kommunikationspartner. Im engeren Sinn zielen Anonymisierungstechniken darauf ab, Aufenthaltsorte und Kommunikationsbeziehungen so zu verschleiern, dass keine Bewegungs-, Kommunikations- oder Zugriffsprofile einzelner Benutzer durch unautorisierte Dritte erstellt werden können.

Privacy

Die oben angesprochenen Möglichkeiten, anhand von Verkehrs- und Aufenthaltsdaten Profile zu erstellen, führen zu einer Bedrohung der Privatsphäre (engl. *privacy*) des Nutzers. Problematisch ist, dass der Nutzer i.d.R. keinen Einfluss darauf hat, ob und in welchem Umfang derartige sensible Daten erhoben und gespeichert werden. So ist es beispielsweise beim Versenden einer EMail durchaus wünschenswert, die eigene EMail-Adresse anzugeben, während man aber dem Empfänger oder unbeteiligten Dritten keine weitere Informationen, wie z.B. die URL der zuvor genutzten WWW-Seite zukommen lassen möchte. Das Recht auf die informationelle Selbstbestimmung wurde bereits 1983 vom Bundesverfassungsgericht im Volkszählungsurteil verankert. Das Grundrecht gewährleistet die Befugnis des Einzelnen, grundsätzlich selbst über die Preisgabe und Verwendung seiner persönlichen Daten zu bestimmen. Einschränkungen dieses Rechts auf informationelle Selbstbestimmung sind nur im überwiegenden Allgemeininteresse zulässig. Mit dem informationellen Selbstbestimmungsrecht hat das Bundesverfassungsgericht auch den Anspruch auf Anonymisierung anerkannt (BVerfGE 65, 1, 49). Mit der zunehmenden Vernetzung und den Entwicklungen im Bereich des so genannten Internet of Things (IoT) mit u.a. vernetzten Geräten in Haushalten (smart Home), vernetzten Geräten zur Überwachung von Gesundheitsdaten (smart Health) oder aber auch mit modernen Mobilitätsszenarien nehmen die Bedrohungen der Privatheit sehr stark zu.

Vertrauen

Vertrauen

In den letzten Jahren wird der Begriff des Vertrauens (engl. *trust*) sehr häufig verwendet. Im Gegensatz zu den eingeführten Schutzzügen, beschreibt der Vertrauensbegriff keine Sicherheitseigenschaft eines Systems. Laut der Definition des Dudens bedeutet Vertrauen, das feste Überzeugtsein von der Verlässlichkeit und Zuverlässigkeit einer Person oder Sache. Ob diese Person oder Sache dieses Vertrauens auch würdig ist, also vertrauenswürdig ist, wird mit dem Vertrauensbegriff nicht erfasst. Vertrauen gemäß der Duden Definition ist also eher ein Gefühl, denn eine Eigenschaft, die man auch beweisen und nachweisen kann. Überträgt man den Vertrauensbegriff auf IT-Systeme, so könnte man definieren, dass ein Trusted System ein System ist, dem man vertraut, dass es spezifizierte Dienste, bzw. eine spezifizierte Funktionalität auch beim Auftreten von Störungen oder Fehlern korrekt und verlässlich erbringt. Sicherheitstechnologien, wie sie in diesem Buch eingehend behan-

delt werden, können dazu verwendet werden, um Vertrauen in IT-Systeme aufzubauen. Beispiele hierfür sind die Verwendung von Identitätsnachweisen, um das Vertrauen zu erhöhen, dass Dienste nur von authentischen und autorisierten Akteuren genutzt werden.

Eine Präzisierung des Begriffs *vertrauenswürdig* bzw. der *Vertrauenswürdigkeit* ist schwierig. In dem Buch *The Trusted Advisor* [46], das jedoch nicht für den IT-Bereich, sondern allgemein für Marketing-Interessierte geschrieben ist, findet man hierfür eine interessante Formel. Die Formel lautet wie folgt:

$$\text{Vertrauenswürdigkeit} = \frac{\text{Glaubwürdigkeit} + \text{Zuverlässigkeit} + \text{Vertrautheit}}{\text{Selbstorientierung}}$$

vertrauenswürdig

Wendet man eine solche Formel auf den IT Bereich an, um die Vertrauenswürdigkeit von Diensteanbietern in einer offenen Dienst-orientierten Umgebung, oder in einer offenen Cloud zu bewerten, so könnte man zur Bewertung der Glaubwürdigkeit und Zuverlässigkeit eines Dienstes bzw. des Diensteanbieters Reputationssysteme verwenden, die Auskunft geben, ob der Dienst entsprechend seiner Funktionalität zuverlässig arbeitet. Vertrautheit ergibt sich durch die eigene Erfahrung mit dem Dienst oder dem Provider, der den Dienst anbietet, während die Selbstorientierung schwierig zu bewerten sein wird. Würde man die Attribute der Formel quantifizieren, so ist klar, dass ein hoher Wert bei der Selbstorientierung den Grad der Vertrauenswürdigkeit verringert. Das erscheint sehr einleuchtend, da eine hohe Selbstorientierung eines Diensteanbieters darauf schließen lässt, dass der Diensteanbieter sehr stark nur seinen eigenen Vorteil im Auge hat.

Obwohl in den heutigen IT-Systemen sehr häufig von deren Vertrauenswürdigkeit die Rede ist, werden wir im Folgenden diese Begrifflichkeiten aufgrund der mangelnden Präzision der Begriffsbildung nicht verwenden. Wir werden jedoch in Kapitel 11 im Zusammenhang mit den Techniken des Trusted Computing auf das Thema noch einmal eingehen.

1.3 Schwachstellen, Bedrohungen, Angriffe

Ein IT-System kann unterschiedliche Schwachstellen besitzen. Wird eine dieser Schwachstellen ausgenutzt, so kann es zu Beeinträchtigungen der Datenintegrität, Informationsvertraulichkeit oder auch Verfügbarkeit kommen.

Definition 1.10 (Schwachstelle und Verwundbarkeit)

Unter einer Schwachstelle (engl. *weakness*) verstehen wir eine Schwäche eines Systems oder einen Punkt, an dem das System verwundbar werden

Schwachstelle

Verwundbarkeit

kann. Eine Verwundbarkeit (engl. *vulnerability*) ist eine Schwachstelle, über die die Sicherheitsdienste des Systems umgangen, getäuscht oder unautorisiert modifiziert werden können.



Beispielsweise zählt die Diebstahlgefahr zu den physischen Schwachstellen mobiler Geräte, während Naturkatastrophen wie Feuer, Hochwasser, Erdbeben, Blitzschlag oder auch Stromausfall den natürlichen Schwachstellen zuzurechnen sind. Weitere Schwachstellen können durch die unsachgemäße Nutzung des Systems, durch Softwarefehler oder auch durch unsichere Kommunikationsverbindungen entstehen. Typische softwarebedingte Schwachstellen, die häufig zu einer Verwundbarkeit werden, sind Programme, in denen aufgrund einer fehlenden Eingabeüberprüfung Pufferbereichsüberläufe (vgl. Abschnitt 2.2) nicht korrekt abgefangen werden.

Um den Schutzbedarf eines IT-Systems zu ermitteln, ist es notwendig, eine detaillierte Schwachstellenanalyse durchzuführen (vgl. Abschnitt 4.4). Zur Ermittlung der Gefährdungslage und möglicher Schwachstellen sollte man sich zunächst einen Überblick über die vorhandenen Gefährdungsfaktoren verschaffen. Eine mögliche Klassifikation solcher Faktoren, wie sie beispielsweise in den BSI-Lehrbriefen zur IT-Sicherheit vorgeschlagen wird [35], ist der Abbildung 1.2 zu entnehmen.

Gefährdungsfaktor

Gefährdungsfaktoren

höhere Gewalt	Fahrlässigkeit	technisches Versagen
Blitzschlag Feuer Überschwemmung Erdbeben Demonstration Streik	Irrtum Fehlbedienung unsachgemäße Behandlung	Stromausfall Hardware–Ausfall Fehlfunktionen
Vorsatz		organisatorische Mängel
	Manipulation Einbruch Hacking Vandalismus Spionage Sabotage	unberechtigter Zugriff Raubkopie ungeschultes Personal

Abbildung 1.2: Klassifikation von Gefährdungsfaktoren

1.3.1 Bedrohungen

Ein IT-System ist vielfältigen Bedrohungen ausgesetzt, die mögliche Gefährdungen für das System darstellen.

Definition 1.11 (Bedrohung)

Eine Bedrohung (engl. *threat*) des Systems zielt darauf ab, eine oder mehrere Schwachstellen oder Verwundbarkeiten auszunutzen, um einen Verlust der Datenintegrität, der Informationsvertraulichkeit oder der Verfügbarkeit zu erreichen, oder um die Authentizität von Subjekten zu gefährden.

Bedrohung



Abhängig von der Funktionalität und Einsatzumgebung des Systems besitzen Bedrohungen ein unterschiedliches Gewicht. Eine unerlaubte Informationsgewinnung stellt zum Beispiel für öffentliche Datenbanken keine schwerwiegende Bedrohung dar, so dass auf aufwändige Maßnahmen zu deren Abwehr verzichtet werden kann. Im Gegensatz dazu stellt die Gewinnung von Informationen über Kundendaten oder Daten aus Forschungslabors für Unternehmen eine ernsthafte Bedrohung dar, die abzuwehren ist. Zur Bestimmung der tatsächlichen Gefährdungslage muss deshalb zunächst das Risiko bestimmt werden, das mit den potentiellen Bedrohungen verknüpft ist. Dazu sind die zu schützenden Güter, die *assets* zu bewerten und es ist das Schadenspotential, das durch das Eintreten eines Schadensereignisses auftreten kann, zu bestimmen. Ferner ist abzuschätzen, wie hoch die Eintrittswahrscheinlichkeit für Bedrohungen anzusiedeln ist. Damit ergibt sich nachfolgende Festlegung.

Gewichtung

Definition 1.12 (Risiko)

Unter dem Risiko (engl. *risk*) einer Bedrohung verstehen wir die Wahrscheinlichkeit (oder relative Häufigkeit) des Eintritts eines Schadensereignisses und die Höhe des potentiellen Schadens, der dadurch hervorgerufen werden kann.

Risiko



Das jeweilige Bedrohungsrisiko hängt in hohem Maß von den zu Grunde liegenden Angreifermodellen ab. In diesen Modellen versucht man, die potentiellen Angreifer nach ihren Fähigkeiten und Zielen zu klassifizieren (z.B. Kenntnisse, Ressourcen, Budget), um dann in Abhängigkeit von der Höhe der zu schützenden Werte zu einer aussagekräftigen Aussage hinsichtlich des Schutzbedarfs zu gelangen. Es ist die Aufgabe der Bedrohungs- und Risikoanalyse (vgl. Abschnitte 4.4 und 4.5) zu klären, welche tatsächlichen Bedrohungen für das System zu beachten sind und welche Bedeutung sie für das System besitzen, d.h. welches Risiko besteht. Dazu wird zum

Analyse

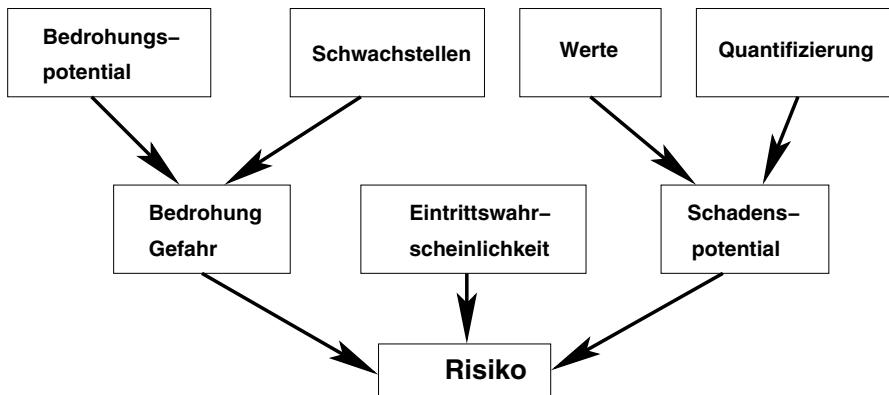


Abbildung 1.3: Zusammenhang zwischen Schwachstellen, Bedrohungen und Risiken

einen die Eintrittswahrscheinlichkeit für die Bedrohungen abgeschätzt und zum anderen wird das Schadenspotential ermittelt, das ein erfolgreicher Angriff birgt. Um das Schadenspotential zu bestimmen, ist es notwendig, die sicherheitsrelevanten Objekte, also die zu schützenden Werte zu erfassen und deren Bedeutung innerhalb des Systems, des Unternehmens oder auch der Gesellschaft zu quantifizieren. Dies ist sehr schwierig und erfordert hohe Kompetenz und großes Fachwissen sowohl über organisatorische Strukturen als auch über betriebliche Abläufe (Prozesse) und natürlich auch detaillierte Kenntnisse über die technischen Komponenten und deren Sicherheitsprobleme. Abbildung 1.3 visualisiert den Zusammenhang zwischen Schwachstellen, Bedrohungen und Risiken.

1.3.2 Angriffs- und Angreifer-Typen

Bedrohungen ergeben sich aus passiven und aktiven Angriffen auf das System.

Definition 1.13 (Angriff)

Angriff

Unter einem Angriff (engl. *attack*) verstehen wir einen nicht autorisierten Zugriff bzw. einen nicht autorisierten Zugriffsversuch auf das System. Wir unterscheiden passive und aktive Angriffe. Passive Angriffe betreffen die unautorisierte Informationsgewinnung und zielen auf den Verlust der Vertraulichkeit ab. Aktive Angriffe betreffen die unautorisierte Modifikation von Datenobjekten und richten sich somit gegen die Datenintegrität oder Verfügbarkeit eines IT-Systems.

□

passiver Angriff

Beispiele für passive Angriffe sind das Abhören von Datenleitungen in vernetzten Systemen (engl. *eavesdropping*) oder das unautorisierte Lesen

von Daten aus Dateien. Angriffe, durch die Passwörter ausgespäht werden, die so genannten Sniffer-Angriffe, gehören noch immer zu den häufigsten passiven Angriffen im Internet.

Sniffer

Beispiele für aktive Angriffe sind das Verändern oder Entfernen von Daten-Paketen aus einem Strom von übertragenen Nachrichten, das Wiederein-spielen von Nachrichten oder das unautorisierte Schreiben in Dateien. Zu den aktiven Angriffen zählen auch zwei spezielle Klassen von Attacken, die besonders häufig durchgeführt werden. Die erste Klasse umfasst Maskie-rungsangriffe, auch als Spoofing-Angriffe bekannt, die zum Ziel haben, eine falsche Identität vorzuspiegeln. Mittels solcher Maskierungsangriffe wird zum Beispiel versucht, eine falsche Absenderadresse in einer E-Mail anzugeben (E-Mail Address Spoofing). Ein mögliches Angriffsziel könnte hierbei darin bestehen, den Empfänger der Mail zur Preisgabe sensibler Informa-tionen zu veranlassen. Eine weitere Variante solcher Spoofing-Angriffe ist der Versuch, die Identität eines Serverrechners anzunehmen (DNS-Name Spoofing oder IP-Address Spoofing (siehe auch Seite 160), um Anfragen von Clients selber z.B. mit gefälschten Daten zu beantworten bzw. zu bearbeiten. Die zweite spezifische Klasse umfasst Angriffe, die darauf abzielen, die Verfügbarkeit von Systemkomponenten oder -diensten infrage zu stellen. Man spricht hier von so genannten Denial-of-Service oder auch Resource Clogging Angriffen. Beispiele dafür sind das Überschwemmen von Rech-nernetzen mit Nachrichten, so dass der Netzverkehr erheblich beeinträchtigt wird, oder das Blockieren eines Service-Ports eines Web-Service Anbieters (z.B. amazon.com) durch eine synthetisch erzeugte Überlast an Anfra-genachrichten.

aktiver Angriff

Spoofing

Denial-of-Service

Abwehr von Angriffen

Passive Angriffe können nur schwer verhindert werden, da auch die bereits angesprochenen verdeckten Kanäle Ansatzpunkte für passive Angriffe bie-ten und diese Kanäle nicht vollständig auszuschließen sind. Durch geeignete Maßnahmen kann man höchstens dafür sorgen, dass die Bandbreite die-ser Kanäle minimiert wird. Mit kryptografischen Verfahren (vgl. Kapitel 7) stehen jedoch wirksame Mechanismen zur Verfügung, um solche passi-ven Angriffe wirkungslos zu machen, durch die Datenleitungen abgehört oder Speicherbereiche ausgelesen werden. Diese Mechanismen bieten auch einen gewissen Schutz gegen die angesprochenen Sniffer-Angriffe. Werden sensible Daten, wie beispielsweise Passwörter, verschlüsselt übertragen, so können sie mittels Sniffer-Programmen nicht mehr ausgespäht werden. Ver-schlüsselungstechniken bieten aber keinen Schutz dagegen, verschlüsselte Passwörter zu kopieren und diese dann in der verschlüsselten Form als Au-thentifikationsmerkmal vorzuweisen. Zur Abwehr solcher Angriffe, die zur

Verschlüsseln

Klasse der Maskierungsangriffe gehören, sind weitere Maßnahmen (vgl. Kapitel 10) notwendig.

minimale Rechte

Aktive Angriffe auf die Datenintegrität können durch eine Beschränkung von Rechten zur Durchführung von Schreibzugriffen verhindert oder zumindest begrenzt werden. Hierbei hilft der Einsatz von Sicherheitsmodellen (vgl. Kapitel 6), die es erlauben, von Realisierungsdetails zu abstrahieren und sich auf das Wesentliche, hier die Modellierung und Vergabe von Schreibberechtigungen, zu konzentrieren. Zusätzlich zu einer *a priori* Eingrenzung sind Mechanismen und Verfahren notwendig, um aktive Angriffe zu erkennen und wirkungslos zu machen. Das Erkennen aktiver Angriffe ist allein durch den Einsatz kryptografischer Verfahren nicht immer möglich, sondern erfordert zusätzliche Mechanismen, wie die Einführung von Sequenznummern. Damit ist man in der Lage zu kontrollieren, ob zusätzliche Daten in einen Datenstrom eingeschleust oder Daten entfernt wurden.

Monitoring

Die Abwehr von Angriffen auf die Verfügbarkeit ist schwierig. Eine mögliche Abwehrmaßnahme ist die oben bereits angesprochene Vergabe von Quoten zur Ressourcennutzung. Eine weitere Maßnahme besteht im intensiven Beobachten (engl. *monitoring*) der gefährdeten Ressourcen, um eine verdächtige Überlastsituation frühzeitig zu erkennen und durch gezielte Eingriffe zu beheben. Im Bereich der Rechnernetze stehen entsprechende Werkzeuge zur Analyse des Netzverkehrs zur Verfügung.

Grauzone

Die Vermeidung oder Verhinderung von Angriffen wird in vielen Fällen dadurch erschwert, dass sie auch aus unabsichtlichem Fehlverhalten von Benutzern resultieren, d.h. insbesondere durch fehlerhafte Programmierung hervorgerufen werden können. Vielfach ist es kaum nachweisbar, ob ein Fehlverhalten auf Unachtsamkeit oder auf Absicht zurückzuführen ist. Beispiel 1.2 verdeutlicht diese Grauzone. Die Behandlung von Programmierfehlern und Techniken zum sicheren Programmieren fallen in den Bereich der Software Sicherheit bzw. in den Bereich Safety und werden hier nicht näher behandelt.

Beispiel 1.2 (Bug oder Angriff?)

Ein, wenn auch noch immer etwas umstrittenes Beispiel² dafür, dass es vielfach kaum möglich ist, zwischen absichtlichen Angriffen und unabsichtlichen, durch Programmierfehler entstandenen Attacken zu unterscheiden, war ein Programmierfehler, der in einem FORTRAN-Programm auftrat und zum Verlust einer amerikanischen Mariner 1 Venus Sonde führte. Auslöser soll eine absichtlich oder unabsichtlich fehlerhaft programmierte DO-Schleife gewesen sein. Anstatt des korrekten Codes

² siehe u. a. <http://catless.ncl.ac.uk/Risks/5.66.html>

Mariner 1 Sonde

DO 20 I = 1,100

soll das Programm die Codezeile

DO 20 I = 1.100

enthalten haben. Die derart abgeänderte Codezeile hätte zur Folge, dass DO20I als eine implizit deklarierte Variable³ interpretiert und eine Zuweisung an diese Variable durchgeführt würde, anstatt eine DO-Schleife zu durchlaufen. Es blieb bis heute unklar, ob es sich tatsächlich um einen Angriff zur Zerstörung der Sonde gehandelt hat, oder ob lediglich ein Programmierfehler bzw. ein schlichter Schreibfehler (Punkt statt Komma) vorlag.



Auch wenn in dem Mariner 1 Fall die genaue Ursache des Problems nach wie vor debattiert wird, so ist dennoch unbestritten, dass ein einfacher Zeichenfehler in der Kontrollsoftware die Ursache für ein schwerwiegendes Problem war. Das Beispiel lehrt aber, dass Funktions- und Qualitätstests von Software unerlässlich sind, um Sicherheitsverletzungen, die aus fehlerhafter Programmierung resultieren können, frühzeitig zu erkennen. Das bedeutet, dass zur methodischen Konstruktion von sicheren Software-Systemen sowohl ingenieurmäßige Methoden und Vorgehensmodelle als auch geeignete Test-Werkzeuge notwendig sind, um komplexe Software von hoher Qualität zu erstellen und diese Qualität nachhaltig, also auch im laufenden Betrieb, garantieren zu können.

Angreifer-Typen

Unterschiedliche Studien (u.a. [72]) verdeutlichen, dass nach wie vor ein großer Teil aller in Unternehmen, Firmen und Behörden bekannt gewordenen Angriffe durch interne Mitarbeiter, den Innenräumen, erfolgen. Durch die zunehmende Vernetzung und Öffnung von Systemen nimmt jedoch die Zahl der externen Angriffe ganz erheblich zu und stellt insbesondere durch die zunehmende Kriminalisierung der Angriffe eine sehr große Bedrohung dar.

Angriffsursachen

Unter einem Hacker versteht man einen in der Regel technisch sehr versierten Angreifer, dessen Ziel es ist, Schwachstellen und Verwundbarkeiten in IT-Systemen aufzudecken und Angriffe, so genannte Exploits, zu entwickeln, um damit diese Schwachstellen auszunutzen. Hacker wenden sich mit diesen Exploits in der Regel an die Öffentlichkeit, um auf Schwachstellen aufmerksam zu machen. Sie verfolgen meist nicht das Ziel, aus der Ausbeutung von Schwachstellen persönliche Vorteile, wie finanzielle Gewinne oder bewusste wirtschaftliche Schäden Dritter, zu ziehen. Man spricht hier auch

Hacker

³ FORTRAN ignoriert Zwischenräume und erlaubt implizit deklarierte Variablen.

gerne von der Hacker-Ethik. Man sollte sich jedoch klar darüber sein, dass Hacker zur Durchführung ihrer Angriffe häufig illegale Wege einschlagen, sich also außerhalb der gesetzlichen Regelungen bewegen.

Skript Kiddie

Den zuvor genannten, technisch versierten Angreifern steht mit den Skript Kiddies eine stetig wachsende Zahl von Internet-Nutzern gegenüber, die zwar über viel Zeit, aber nicht notwendigerweise über vertieftes technisches Know-how verfügen. Diese Gruppe nutzt die frei verfügbaren Exploits, um ihrerseits Angriffe durchzuführen. Obwohl diese Gruppe meist eher von Motiven wie dem Spieltrieb und der Neugierde denn durch die Absicht, Dritte vorsätzlich zu schädigen, getrieben ist, bedeutet die Leichtigkeit, mit der Skript Kiddies vorgefertigte Exploits (s.u.) durchführen können, eine erhebliche Bedrohung für heutige und erst recht zukünftige IT-Systeme. Zu beachten ist aber, dass zu den veröffentlichten Exploits meist sehr schnell Patches zum Schließen der ausgebeuteten Schwachstellen zur Verfügung gestellt werden (z.B. unter <http://www.securityfocus.com>). Somit lassen sich Nachahmungstäter effektiv abwehren, falls die zuständigen Sicherheitsadministratoren die einschlägigen Mailinglisten und Diskussionsforen überwachen und Patches möglichst zeitnah einspielen.

Wirtschaftsspionage NSA

In jüngster Zeit mehren sich auch Lauschangriffe zum Zwecke der Wirtschaftsspionage oder als Folge geheimdienstlicher Tätigkeiten. Unter dem Codenamen Echelon betreibt die NSA (National Security Agency) in den USA schon seit Anfang der 80er Jahre eine flächendeckende Überwachung aller Telefonate, Faxe, E-Mails und Telexe, die über internationale Telekommunikationssatelliten, regionale Satelliten sowie über Kabel und Funktürme gesendet werden. Die breitflächigen Überwachungstätigkeiten der amerikanischen und britischen Geheimdienste NSA und GCHQ (Government Communications Headquarters) wurden im Sommer 2013 durch die Veröffentlichungen des NSA-Mitarbeiters Edward Snowden der breiten Öffentlichkeit bekannt. Mit Überwachungsprogrammen wie *PRISM* und *TEMPORA* wurde flächendeckend der Datenverkehr u.a. über die Transatlantischen Glasfaserverbindungen ausgespäht und große IT-Unternehmen, wie Google und Facebook, mussten auf richterliche Veranlassung den Diensten den Zugriff auf ihre Server gewähren. Die geheimdienstlichen Tätigkeiten verdeutlichten den großen Bedarf an vertrauenswürdigen, kontrollierbaren Sicherheitsmaßnahmen, um unternehmerische Werte vor dem Ausspionieren zu schützen (Wirtschaftsspionage), Manipulationen an Daten wirksam zu unterbinden und die Vertraulichkeit von Daten sowie die Privatsphäre der Menschen zu schützen.

Kriminelle

Neben der Wirtschaftskriminalität durch Industrie-Spionage ist aber auch ein Anstieg an anderen kriminellen Delikten unter Nutzung der vernetzten IT-Systeme zu beobachten. Nach den heutigen Erkenntnissen greifen

Kriminelle zunehmend bestimmte Zielgruppen an, wie Regierungen oder Firmen aus dem Finanzsektor, und verursachen einen hohen individuellen Schaden. Neben unterschiedlichen Formen der Erpressung steht zurzeit die profitorientierte Web-Kriminalität an der Spitze der Skala. Die E-Mail bzw. auch Browser-basierten Phishing-Angriffe⁴, bei denen der Benutzer durch gezielte Täuschungsversuche dazu verleitet wird, sein Passwort preiszugeben, haben hieran nach wie vor einen erheblichen Anteil. Phishing-Angriffe sind eine spezielle Ausprägung von Identitätsdiebstählen.

Kriminelle bedienen sich für ihre Aktivitäten zunehmend des Know-hows von technisch versierten Angreifern. Zu nennen ist hier insbesondere der Aufbau von so genannten Bot-Netzen durch solche Angreifer. Hierfür werden viele (in der Regel mehrere tausend) PCs so präpariert, dass von diesen präparierten PCs gezielt Angriffe auf ein Opfersystem gefahren werden können (vgl. Seite 75). Die dafür erforderliche Schadsoftware wird in der Regel über Trojanische Pferde (vgl. Kapitel 2) unbemerkt vom PC-Besitzer auf dessen PC eingeschleust. Die Ausführung der Schadsoftware wird entfernt durch den Angreifer kontrolliert und initiiert, um z.B. einen Denial-of-Service (DoS) Angriff auf ein bestimmtes Opfersystem durchzuführen. Das Bot-Netz wird von den Angreifern gerne an Kriminelle vermietet, die das vorbereitete Netz u.a. für Erpressungen gegen den Betreiber des Opfersystems nutzen. In einem solchen Bot-Netz-Szenario sind dann natürlich die Besitzer der missbrauchten und präparierten PCs nicht nur Opfer einer Schadsoftware-Attacke sondern unwissentlich auch selber Angreifer, wenn ihr PC an einer entsprechenden DoS Attacke beteiligt ist.

Die steigende Kriminalisierung der Internet-basierten Angriffe spiegelt sich auch in der Zunahme an erpresserischer Malware wider, die unter dem Kunstnamen Ransomware bekannt geworden sind. Hierbei handelt es sich um Schadsoftware, die auf fremden Rechnern eindringt und Daten auf der lokalen Festplatte des fremden Rechners verschlüsselt, so dass diese Daten für den Nutzer nicht mehr zugreifbar sind. Für die Entschlüsselung der Daten fordert der Angreifer üblicherweise einen Geldbetrag, der über ein Online-Zahlungssystem wie PayPal zu entrichten ist. Die Bezeichnung der Ransomware ergibt sich aus der Kombination aus dem Begriff Malware sowie dem englischen Begriff für Lösegeld (ransom). Eine bekannte Variante dieser Schadsoftware war der Virus Gpcode, der sich im Sommer 2006 verbreitete, bzw. seine neuen Varianten aus dem Jahr 2007. Gpcode wurde über SPAM-Mails verbreitet. Der Virus verschlüsselt Daten der lokalen Festplatte, die eine spezifische Endung haben, mittels RSA. Die neueren Varianten des Gpcode-Virus verwenden häufig keine RSA-Verschlüsselung mehr, sondern legen Daten von der lokalen Festplatte in einem zip-Archiv

Bot-Netz

Erpressung

⁴ Phishing ist aus den Begriffen Password und Fishing entstanden.

ab, das mit einem starken Passwort gesichert wird. In analoger Weise wird dann ein Geldbetrag dafür gefordert, dass der Angreifer dem Nutzer dieses Passwort nennt, um ihm seine eigenen Daten wieder zugänglich zu machen.

BSI-Lageberichte

Die Lageberichte⁵ zur IT-Sicherheit in Deutschland des Bundesamts für Sicherheit in der Informationstechnik (BSI) bestätigt diese Trends zur Kommerzialisierung und Professionalisierung der Internetkriminalität. Die Berichte verzeichnen einen nach wie vor deutlichen Anstieg von Angriffen, die Schadsoftware (engl. *malware*) mittels Viren, Würmer und Trojanische Pferde verbreiten. Auffallend ist, dass klassische Viren-Angriffe (vgl. Kapitel 2) zunehmend durch die gezielte Verbreitung Trojanischer Pferd-Programme ersetzt werden.

Spyware

Ein Hauptziel derartiger Trojaner besteht darin, unbemerkt Spyware Programme auf Rechnern zu installieren. Hierbei handelt es sich um Software, die gezielt auf fremden Rechnern Informationen sammelt und dem Angreifer zugänglich macht. Beispiele für derart ausgespähte Informationen sind auf der Festplatte gespeicherte Passwörter und Zugangscodes, oder aber Passwörter und Codes, die über die Tastatur eingegebene werden. Sensitive Dokumente, E-Mails oder aber auch Cookies, die Zugangsinformationen für Server im Web beinhalten, sind weitere Beispiele von unberechtigt ausgespähten Daten. Da ein großer Teil dieser ausgespähten Daten zur Identifizierung von Nutzern im System oder im Internet (u.a. Zugangscode zu Banken und Portalen) dienen, sind sie eine der Hauptursachen für die noch immer wachsende Zahl der Identitätsdiebstähle.

DoS

Weiterhin verzeichnen die Lageberichte, dass Schadprogramme immer effektiver arbeiten, die Zahl der Denial-of-Service-Angriffen zunimmt und die Sicherheitslücken in Betriebssystemen und Anwendungen immer schneller ausgenutzt werden. Zusätzlich wird darauf verwiesen, dass im Bereich der Cyber-Kriminalität eine zunehmende Kommerzialisierung zu verzeichnen ist. So werden große Botnetze von z.B. 10.000 PCs zum Anmieten für 200 US Dollar pro Tag auf dem Markt angeboten. Mit einem solchen Botnetz können z.B. SPAM-Angriffe durchgeführt werden, die sehr rentabel für die Betreiber des Botnetzes sein können. Erhebliche Zuwachsrate an Sicherheitsproblemen wird im Bereich der Smartphones mit den mobilen Apps (u.a. Android und iPhone) verzeichnet. In Zukunft werden insbesondere im Cloud-Computing ganz erhebliche Gefährdungen gesehen.

Positiv hat sich den Berichten zu Folge die Bewusstseinsbildung entwickelt. Das Sicherheitsbewusstsein der Anwender ist gestiegen, Betriebssystem-Updates werden häufiger durchgeführt und IT-Sicherheitstechniken deutliche häufiger angewendet.

5

vgl. <https://www.bsi.bund.de/ContentBSI/Publikationen/Lageberichte/bsi-lageberichte.html>

Die meisten der durch interne Mitarbeiter verursachten Angriffe gehen auf mangelhafte Kenntnisse sowohl der Systemgegebenheiten als auch der zur Verfügung stehenden Sicherheitsmechanismen und deren Bedeutung zurück. Weitere Hauptquellen für Angriffe sind Nachlässigkeiten im Umgang mit dem System und den zu verwaltenden sensiblen Informationen. Auch ein mangelhaftes Problembewusstsein, sowohl bei den Mitarbeitern als auch im mittleren und im Top-Management, wird in den entsprechenden Studien wiederholt angeprangert. Mit Maßnahmen des ganzheitlichen Sicherheitsmanagements könnten derartige organisatorische sowie technische Sicherheitsmängel behoben werden. Bei einem ganzheitlichen Ansatz werden sowohl organisatorische Strukturen als auch die Geschäftsprozesse und die technischen Komponenten zusammen betrachtet, um Brüche zwischen diesen verschiedenen Ebenen zu vermeiden.

Mitarbeiter

Schließlich ist auch noch auf Angriffe hinzuweisen, die nicht technischer Natur sind. Man spricht vom Social Engineering oder Social Hacking. Hierbei versucht ein Angreifer sein Opfer dazu zu bringen, dass es unabsichtlich oder absichtlich im guten Glauben sensitive Informationen an den Angreifer preisgibt. Ein sehr beliebter Ansatz ist es, sich gegenüber Mitarbeitern als Systemadministrator auszugeben (z.B. über einen Telefonanruf) und um die Angabe des Benutzer-Passworts zu bitten, um angeblich dringend notwendige administrative Arbeiten durchführen zu können.

Social Engineering

Für die Zukunft wird ein weiteres Ansteigen der Identitätsdiebstähle sowie eine Zunahme der Bedrohungen durch Schadsoftware und Spyware, insbesondere durch Trojanische Pferde erwartet. Ferner wird mit einem weiteren Anstieg von solchen Angriffen gerechnet, die durch Unkenntnis und Nachlässigkeit hervorgerufen werden, so dass der Bedarf an wirksamen Sicherheitsmechanismen und -diensten wächst und die Schulung von Mitarbeitern bzw. die fundierte Ausbildung auf dem Gebiet der Systemsicherheit unerlässlich ist. Ein weiterer Anstieg der Angriffe im Bereich der Computerkriminalität bis hin zu Cyber-Terrorismus ist ebenfalls zu erwarten.

Entwicklung

Cyber-Crime

1.3.3 Rechtliche Rahmenbedingungen

Die Frage, inwieweit Angriffe von Hackern und anderen Angreifertypen nach deutschem Recht strafbar sind, wurde 2001 neu geordnet. §202a des Strafgesetzbuches verbietet nur dann das Ausspähen von fremden Daten, wenn diese „gegen unberechtigten Zugang besonders gesichert sind.“ Am 1.2.2001 hat der Deutsche Bundestag das so genannte Zugangskontroll-diensteschutzgesetz (ZKDSG) verabschiedet. Das Ziel dieses Gesetzes ist es, die unberechtigte Nutzung kostenpflichtiger Angebote von Rundfunk- und Fernsehsendern sowie Tele- und Mediendiensten durch das Umgehen von technischen Vorsperr- und Verschlüsselungsverfahren zu unterbinden.

Gesetzeslage

Als Zugangskontrolldienste gelten sowohl Hardware-Lösungen wie Decoder und Smartcards als auch in Software implementierte Verschlüsselungsmechanismen.

Auswirkungen

Nach diesem Gesetz kann jeder mit einer Freiheitsstrafe bis zu einem Jahr belegt werden, der Einrichtungen für das Umgehen solcher Zugangskontrolldienste zu gewerbemäßigen Zwecken einführt, herstellt oder verbreitet. Ferner kann dann der gewerbliche Besitz, die Einrichtung, die Wartung und der Austausch der Hacker-Werkzeuge mit Geldbußen bis zu 50.000 Euro geahndet werden. Das Gesetz deckt insbesondere den Bereich des Pay-TV und der entgeltlichen Video-on-Demand Angebote ab. Nicht betroffen von der Regelung sind Techniken wie die elektronische Signatur, bei denen Daten aus Sicherheitsgründen verschlüsselt übertragen werden, oder auch der Kopierschutz bei Dateien. Mit diesem Gesetz werden aber nur gewerbsmäßig handelnde Personen erfasst, während Privatpersonen sich nicht strafbar machen, wenn sie gefälschte Smartcards oder gehackte Software besitzen. Das ist durchaus eine sinnvolle Regelung, da ein gutgläubiger Nutzer relativ leicht auf ein illegales Betrugsangebot hereinfallen kann. Durch die gesetzliche Regelung soll der eindeutig Schuldige zur Verantwortung gezogen werden. Die private Nutzung einer illegalen Umgehungsseinrichtung für Zugangskontrolldienste fällt in Deutschland unter den §265a des Strafgesetzbuches, der die „Erschleichung von Leistungen“ behandelt.

Phishing

Auch Phishing-Angriffe sind nach geltendem Recht strafbar, da über einen solchen Angriff Daten (hier Zugangsdaten) ausgespäht werden, mit dem Ziel, diese Daten zu missbrauchen, um beispielsweise beim Online-Banking unter der Identität des Opfers Geldtransaktionen zu Lasten des Opfers zu tätigen. Phishing-Angriffe können deshalb Straftatbestände des Ausspähens von Daten (Paragraph 202a StGB), des Betrugs/Computerbetrugs (§263/§263a StGB), der Fälschung beweiserheblicher Daten (§269 StGB) und der unbefugten Datenerhebung und -verarbeitung (§44, 43 BDSG) betreffen.

Hacker-Paragraph

Im September 2006 hat das Bundeskabinett einen Regierungsentwurf eines Strafrechtsänderungsgesetzes zur Bekämpfung der Computerkriminalität⁶ beschlossen, das den EU-Rahmenbeschluss über Angriffe auf Informationssysteme sowie das Europarat-Übereinkommen über Computerkriminalität in nationales Recht umsetzen sollte. Am 24.05.2007 hat der Bundestag diesen Entwurf trotz erheblicher Bedenken von Seiten von Sicherheitsexperten verabschiedet. Die Änderungen umfassen eine Ergänzung des Paragraphen 202a Absatz 1, der nun wie folgt lautet: „(1) Wer unbefugt sich oder anderen Zugang zu Daten, die nicht für ihn bestimmt und die gegen unberechtigten Zugang besonders gesichert sind, unter Überwindung der Zugangssicherung

⁶ siehe <http://www.buzer.de/gesetz/7846/index.htm>

verschafft, wird mit Freiheitsstrafen bis zu drei Jahren oder mit Geldstrafe bestraft.“

Erweitert wurde der Gesetzestext insbesondere um die Paragraphen 202b und 202c. Paragraph 202b behandelt den Straftatbestand des Abfangens von Daten: „Wer unbefugt sich oder einem anderen unter Anwendung von technischen Mitteln nicht für ihn bestimmte Daten (§202a Abs.2) aus einer nichtöffentlichen Datenübermittlung oder aus der elektromagnetischen Abstrahlung einer Datenverarbeitungsanlage verschafft, wird mit Freiheitsstrafe bis zu zwei Jahren oder mit Geldstrafe bestraft, wenn die Tat nicht in anderen Vorschriften mit schwererer Strafe bedroht ist.“

Der Paragraph 202c StGB betrifft das Vorbereiten des Ausspähens und Abfangens von Daten: „(1) Wer eine Straftat nach Paragraph 202a oder Paragraph 202b vorbereitet, indem er

1. Passworte oder sonstige Sicherungscodes, die den Zugang zu Daten (§202a Abs. 2) ermöglichen, oder
2. Computerprogramme, deren Zweck die Begehung einer solchen Tat ist, herstellt, sich oder einem anderen verschafft, verkauft, einem anderen überlässt, verbreitet oder sonst zugänglich macht, wird mit Freiheitsstrafe bis zu einem Jahr oder mit Geldstrafe bestraft.“

Der Gesetzesentwurf wurde im Vorfeld heftig kritisiert. Hauptkritikpunkte betrafen den Paragraphen 202c des Strafgesetzbuchs(StGB), der den Einsatz und die Verbreitung von so genannten Hacker-Tools unter Strafe stellt. In einer Stellungnahme der Gesellschaft für Informatik (GI)⁷ heißt es: „Problematisch ist die Einfügung des Paragraph 202c StGB, weil Programme und Tools nicht nach ihrer Einsatzart, sondern vielmehr nach ihrem Aufbau definiert werden. Eine Unterscheidung in Anwendungen, die zur Begehung von Straftaten und solche, die ausschließlich für legale Zwecke hergestellt werden, ist aber nicht möglich. Der gewählte Wortlaut führt zu einer Kriminalisierung der heute in allen Unternehmen, Behörden und von Privaten verwendeten Programme zur Aufdeckung von Sicherheitslücken in IT-Systemen. Derartige Programme und Tools sind zur Absicherung gegen Angriffe jedoch unverzichtbar (Penetration Testing).“

Kritik

Prof. A. Rossnagel von der Universität Kassel führt in der GI-Stellungnahme hierzu weiter aus: „Es besteht die Gefahr, den bloßen Besitz und die informationstechnische Entwicklung von Tools zu bestrafen, die der Identifizierung von Sicherheitslücken dienen, zugleich aber wesensnotwendig auch zum Eindringen in Systeme verwendet werden können, [...]. Das angeblich einschränkende objektive Tatbestandsmerkmal der *Zweckbestimmung für eine*

Zweckbestimmung

⁷ vgl. <http://www.gi-ev.de/aktuelles/meldungsdetails/meldung/159/>

Straftat (§202c Abs. 1 Nr. 2 StGB-E) ist keines, weil Computerprogramme keinen Zweck haben. Selbst wenn der Entwickler einen bestimmten Zweck intendiert, können sie immer missbraucht werden. Die Begründung erkennt dies implizit an, wenn sie angibt, es reiche aus, wenn die objektive Zweckbestimmung *auch* die Begehung einer Straftat sei. Noch problematischer wird dies dadurch, dass der bedingte Vorsatz erfasst ist. Es ist eine Vielzahl von Fällen vorstellbar, in denen ein Betroffener bei einer an sich legitimen Handlung in Kauf nehmen wird, dass ein Passwort oder ein Computerprogramm auch anderweitig verwendet wird [...]. Bereits die Gefahr, wegen einer solchen Tätigkeit belangt zu werden, kann die Entwicklung und Verbesserung von Sicherheitstechnik behindern, Industrie und Bürgern wichtiger Selbstanalysemöglichkeiten berauben und so die IT-Sicherheit gefährden.“

Kommentare

In der Drucksache 16/3656 des Bundestags⁸ wird auf einige Kritikpunkte Bezug genommen. Dort ist unter anderem zu lesen: „Durch die objektive Beschränkung auf Computerprogramme, deren Zweck die Begehung einer Computerstraftat ist, wird bereits auf Tatbestandsebene sichergestellt, dass keine Computerprogramme erfasst werden, die beispielsweise der Überprüfung der Sicherheit oder Forschung in diesem Bereich dienen. Unter Strafe gestellt werden lediglich das Herstellen, Verschaffen, Verbreiten usw. solcher Programme, denen die illegale Verwendung immanent ist, die also nach Art und Weise des Aufbaus oder ihrer Beschaffenheit auf die Begehung von Computerstraftaten angelegt sind. Bei Programmen, deren funktionaler Zweck nicht eindeutig ein krimineller ist und die erst durch ihre Anwendung zu einem Tatwerkzeug eines Kriminellen oder zu einem legitimen Werkzeug (z. B. bei Sicherheitsüberprüfungen oder im Forschungsbereich) werden (sog. dual use tools), ist der objektive Tatbestand des Paragraph 202c StGB-E nicht erfüllt. Die bloße Eignung von Software zur Begehung von Computerstraftaten ist daher nicht ausreichend, so dass auch solche Programme aus dem Tatbestand herausfallen, die lediglich zur Begehung von Computerstraftaten missbraucht werden können.“

Vorbereitung einer Straftat

Weiterhin heißt es dort: „Zudem muss die Tathandlung zur Vorbereitung einer Computerstraftat (§202a, 202b, 303a, 303b StGB) erfolgen. Entscheidend für die Tatbestandserfüllung des Paragraph 202c StGB-E ist, dass der Täter eine eigene oder fremde Computerstraftat in Aussicht genommen hat. Das ist nicht der Fall, wenn das Computerprogramm beispielsweise zum Zwecke der Sicherheitsüberprüfung, zur Entwicklung von Sicherheitssoftware oder zu Ausbildungszwecken in der IT-Sicherheitsbranche hergestellt, erworben oder einem anderen überlassen wurde, da die Sicherheitsüberprüfung, die Entwicklung von Sicherheitssoftware oder die Ausbildung im Bereich der IT-Sicherheit keine Computerstraftat darstellen.“

⁸ siehe <http://dip.bundestag.de/btd/16/036/1603656.pdf>

Auf den zulässigen Einsatz von Analysetools zum Aufdecken von Sicherheitsproblemen zum Beispiel im Zusammenhang mit der Durchführung von Penetrationstestings wird dort wie folgt hingewiesen: „Auch nach der Neufassung des Tatbestandes ist die Verschaffung des Zugangs zu Daten unter Verletzung von Sicherheitsmaßnahmen nur strafbewehrt, wenn der Täter unbefugt handelt. Nicht strafbar ist daher z. B. das Aufspüren von Sicherheitslücken im EDV-System eines Unternehmens, soweit der „Hacker“ vom Inhaber des Unternehmens mit dieser Aufgabe betraut wurde.“

Sicherheitsanalysen

Trotz der Vorbehalte und starken Kritik hat der Bundestag am 25. Mai 2007 das Gesetz unverändert verabschiedet. Bislang sind nach Kenntnis der Autorin die befürchteten negativen Auswirkungen in der Praxis nicht eingetreten.

Für vergleichsweise großes Aufsehen hat das so genannte Online-Durchsuchungsurteil des Bundesverfassungsgerichts vom 27. Februar 2008 (1BvR 370/07; 1 BvR 595/07) gesorgt. In diesem Urteil hat das Gericht ein neues Grundrecht auf *Gewährleistung der Vertraulichkeit und Integrität informationstechnischer Systeme* etabliert. Bei der Urteilsverkündung in Karlsruhe erläuterte der Präsident des Bundesverfassungsgerichts, dass dieses neue Grundrecht zu den anderen Freiheitsrechten wie insbesondere dem Recht auf Schutz des Telekommunikationsgeheimnisses, dem Recht auf Unverletzlichkeit der Wohnung und dem Recht auf informationelle Selbstbestimmung hinzu kommt. Dem Urteil ging eine Verfassungsbeschwerde voraus, die sich gegen die Vorschriften im Verfassungsschutzgesetz von Nordrhein-Westfalen zur Online-Durchsuchung richtete. In der Urteilsbegründung haben die Richter unterstrichen, dass eine heimliche Online-Durchsuchung von Rechnern nur unter strengen Auflagen möglich sei, da nach Maßgabe des Bundesverfassungsgerichts das allgemeine Persönlichkeitsrecht auch ein Grundrecht auf Gewährleistung der Vertraulichkeit und Integrität informationstechnischer Systeme umfasst. Das neue Grundrecht ist subsidiär, das heißt, es tritt gegenüber bestehenden Grundrechten zurück (u.a. dem Recht auf informationelle Selbstbestimmung) und soll zur Anwendung kommen, um Schutzlücken abzudecken. Die Konsequenzen und die Bedeutung dieses Urteils werden unter Juristen sehr stark diskutiert. Es ist noch offen, ob das Gesetz ggf. zivilrechtliche und auch haftungsrechtliche Konsequenzen für den Missbrauch von IT-Systemen nach sich ziehen kann und welche Auswirkungen eine derartige Rechtsprechung dann auf die IT-Sicherheitsqualität von IT-Komponenten und Software haben wird.

Neues Grundrecht

Dieser kurze Blick auf die Gesetzeslage ist keineswegs erschöpfend, sondern hatte lediglich zum Ziel, das Bewusstsein zu schärfen, dass neben technischen Randbedingungen bei der Konstruktion und Administration sicherer IT-Systeme stets auch eine Vielzahl rechtlicher Regelungen zu beachten und

zu befolgen sind. Mit der zunehmenden Globalisierung der Geschäftsprozesse nehmen auch die grenzüberschreitenden, Internet-basierten Aktivitäten von Unternehmen, aber auch von Einzelpersonen stark zu. Hierfür rechtliche Rahmenbedingungen zu erarbeiten ist Gegenstand von Forschungsaktivitäten an juristischen Fakultäten, die häufig unter dem Namen Cyber Law zusammengefasst werden.

1.4 Computer Forensik

Mit der Zunahme der Schadensfälle, die auf absichtliche Angriffe auf IT-Systeme zurückzuführen sind, hat sich in den letzten Jahren unter dem Begriff Computer-Forensik (engl. *computer forensics*) oder auch digitale Forensik eine spezielle Disziplin entwickelt (vgl. u.a. [61]). Methoden der Computer Forensik beschäftigen sich mit dem Nachweis und der Aufklärung von ggf. strafbaren Handlungen durch Auswertung von digitalen Spuren. Für einen gerichtsverwertbaren Nachweis (so er denn wirklich geführt werden kann) ist es notwendig, dass die Analysen nicht auf dem Originaldatenbestand, sondern nur auf Kopien durchgeführt werden (*never touch original*) und dass die Ergebnisse neutral, überprüfbar und nachvollziehbar präsentierbar sind. Forensische Untersuchungen werden nach Möglichkeit nur auf den erstellten forensischen Kopien (*forensic sound imaging*) und im read-only-Modus durchgeführt. Die Originalsysteme sollten – falls möglich – sicher verwahrt werden. Dadurch wird eine spätere, unabhängige Überprüfung der Analysen möglich.

Allgemeines Vorgehen

Da bei möglichen gerichtlichen Auseinandersetzungen die Beweisketten von Dritten jederzeit überprüfbar nachvollziehbar sein müssen, sind bei der forensischen Analyse Open Source Tools interessant, da deren Code offen gelegt und von Jedermann nutzbar ist. Im Folgenden benennen wir einige derartige Tools. Diese beispielhafte Benennung hat lediglich das Ziel, einen groben Einblick in die Funktionsweise derartiger Tools zu vermitteln.

Einer der wichtigsten Schritte bei der forensischen Analyse der Daten ist die Erstellung eines korrekten Abbildes (*image*) des Datenmediums. Das einfache *dd*-Programm⁹, das Bestandteil von Linux-Distributionen ist, erfüllt bereits alle Anforderungen, die z.B. das Computer Forensic Tool Testing Programm der US-Regierung¹⁰ an Imaging-Tools stellt. Es erstellt eine exakte 1:1 Kopie eines Datenträgers. Die *dd*-Werkzeuge sind zwar leistungsstark,

⁹ vgl. <http://www.gnu.org/software/coreutils/>

¹⁰ vgl. <http://www.cftt.nist.gov>

aber nicht sehr komfortabel in ihrer Bedienung. Diesen Mangel behebt das *AIR-Tool* (Automated Image & Restore¹¹).

Um mit der erstellten Kopie aussagekräftige Analysen vornehmen zu können, ist es notwendig, die Kopie in einem read-only Modus zu mounten, um zu verhindern, dass die Untersuchungsergebnisse durch versehentliche Veränderungen an den Datenbeständen unbrauchbar werden. Unter Linux steht mit dem *Enhanced Linux Loopback-Tool*¹² ein Programm zur Verfügung, das ein read-only Mounten des *dd*-Images ganzer Datenträger ermöglicht.

Read-only

Hat man ein exaktes Abbild erstellt, benötigt man geeignete Analysetools, um den Datenbestand zu untersuchen. Hierzu bietet beispielsweise Unix/Linux einige Werkzeuge und Programme wie *grep*, *strings*, *find*, *file*, *hexedit* standardmäßig an, die aber für die Analyse großer Datenbestände nicht sehr geeignet sind. Unter dem Namen *Sleuth Kit*¹³ werden Unix-basierte Kommandozeilen-Programme zusammengefasst, mit denen Dateisysteme wie NTFS, FAT, FFS, ext2fs sowie ext3fs analysiert und insbesondere nach manipulierten Datenbeständen gesucht werden kann.

Analyse

Die Computer Forensik gewinnt angesichts der Zunahme von Straftatdelikten im Bereich der IT zunehmend an Bedeutung. Obwohl bereits eine Vielzahl von Tools zur Verfügung steht, um effektiv elektronische Spuren zu sichern, ist die Thematik nicht unproblematisch. Bei der forensischen Analyse sind natürlich auch datenschutzrechtliche Aspekte zu beachten und die Persönlichkeitsrechte des Einzelnen sind zu wahren. Inwiefern forensische Analysen elektronischer Datenspuren, die ja beliebig manipuliert werden können, wobei auch der Manipulationsvorgang selber auch unkenntlich gemacht werden kann, tatsächlich zu einer lückenlosen Beweiskette zusammengeführt werden können, die einer rechtsstaatlichen Überprüfung standhalten, ist jedoch noch offen. Sicherlich ist es für interne Revisionszwecke hilfreich, Anscheinsbeweise anhand sorgfältiger forensischer Analysen vorlegen zu können. Die Durchführung einer solchen Analyse mit seriös nachvollziehbaren Ergebnissen ist jedoch eine schwierige Aufgabe, die ein tiefes Verständnis der Systeme voraussetzt.

Fazit

1.5 Sicherheitsrichtlinie

Das Spektrum der Sicherheitseigenschaften, die ein System oder eine einzelne Anwendung erfüllen muss, ist breit und abhängig von der Funktionalität, dem Einsatzgebiet, ggf. der unternehmerischen Geschäftsstrategie sowie sei-

¹¹ vgl. <http://air-imager.sourceforge.net/>

¹² vgl. ftp://ftp.hq.nasa.gov/pub/ig/ccd/enhanced_loopback/

¹³ vgl. <http://www.sleuthkit.org/sleuthkit/>

ner Nutzung. Die zu realisierenden Sicherheitseigenschaften definieren die Schutzziele eines Systems; diese werden in den Sicherheitsregeln¹⁴ [53] bzw. in Sicherheitsrichtlinien zusammengefasst.

Definition 1.14 (Sicherheitsrichtlinie)

Richtlinie

Die Sicherheitsrichtlinie (engl. *security policy*) eines Systems oder einer organisatorischen Einheit legt die Menge von technischen und organisatorischen Regeln, Verhaltensrichtlinien, Verantwortlichkeiten und Rollen sowie Maßnahmen fest, um die angestrebten Schutzziele zu erreichen.

□

Die Sicherheitsrichtlinie einer organisatorischen Einheit wie zum Beispiel eines Unternehmens, einer Abteilung oder eines Projekts besteht in der Regel aus einer Kombination von systembestimmten und benutzerbestimmten Anteilen. Systembestimmte Regeln sind globale Regeln, die von einer dafür zuständigen Einheit (z.B. dem Management eines Unternehmens) festgelegt werden. Wir unterscheiden Zugriffskontroll- und Informationsfluss-Richtlinien. In der Praxis werden hauptsächlich Varianten von Zugriffskontroll-Regeln verwendet, wobei zwischen benutzerbestimmten, rollenbasierten und systembestimmten Regeln unterschieden wird.

benutzer-bestimmbar

Die Klasse der benutzerbestimmten Richtlinien (engl. *discretionary policy*) erlaubt Benutzern, die Rechtevergabe für Objekte, die sie erzeugt haben, individuell zu kontrollieren.

rollenbasiert

Rollenbasierte Richtlinien reglementieren den Zugriff auf Objekte durch die Festlegung von Rollen, in denen Benutzer bzw. die in seinem Auftrag tätigen Prozesse und Maschinen in dem System aktiv sein dürfen. Eine Rolle benennt eine Aufgabe, die durchzuführen ist, und beschreibt die Berechtigungen und Verantwortlichkeiten, die mit der Rolle verbunden sind. Das bedeutet, dass Zugriffsrechte nicht an individuelle Benutzer, sondern an Rollen vergeben werden und dass Benutzer von einem Systemadministrator als Mitglieder von Rollen einzutragen sind. Die Rollen sollten die vom Benutzer durchzuführenden Arbeiten charakterisieren.

systembestimmt

Für die Klasse der systembestimmten, d.h. mandatorischen Richtlinien (engl. *mandatory policy*) gilt, dass die Vergabe von Zugriffsrechten systemglobal auf der Basis von a priori Festlegungen kontrolliert wird. Ein Beispiel für eine entsprechende a priori Festlegung ist die Klassifikation der in einem System zu verarbeitenden Informationen nach Sicherheitsklassen (z.B.

¹⁴ Im deutschen Sprachgebrauch auch häufig als Sicherheitspolitik bezeichnet. Da die Begriffe der Richtlinien oder Regelwerke den Sachverhalt am Besten beschreiben, werden wir im Folgenden diese Begriffe verwenden.

öffentlich, geheim, streng geheim) und die Einteilung von Objekten und Subjekten in diese Sicherheitsklassen. Die systembestimmte Richtlinie legt für jedes klassifizierte Subjekt fest, welche Informationsklassen diesem Subjekt maximal zugänglich sein können, unabhängig von den durch das Subjekt tatsächlich durchgeführten Aktionen. Richtlinien, die den Datenzugriff auf der Basis klassifizierter Subjekte und Objekte beschränken, nennt man auch Multi-Level-Sicherheitsrichtlinien. Ein weiteres Beispiel für mandatorische Regeln ist die Festlegung einer unternehmensweit geltenden Richtlinie, die bestimmt, dass E-Mails von Mitarbeitern nur verschlüsselt ausgetauscht werden dürfen und alle Mails auf einem zentralen E-Mail-Server zunächst entschlüsselt und auf Viren überprüft werden, bevor sie an die Adressaten weitergeleitet werden.

Eine Sicherheitsrichtlinie legt also die gewünschten Sicherheitseigenschaften eines Systems fest, so dass wir die in Definition 1.2 angegebenen Sicherheitsbegriffe entsprechend interpretieren müssen. Die Sicherheit eines Systems ist keine absolute, sondern immer eine relative Eigenschaft, bezogen auf die spezifizierten Sicherheitsregeln. In der Praxis werden Sicherheitsrichtlinien häufig textuell und damit informell festgelegt, so dass eine der Schwierigkeiten darin besteht, derartige Festlegungen auch kontrollierbar umzusetzen. Ein typisches Beispiel ist eine Richtlinie, mit der in Unternehmen der Umgang mit Passwörtern geregelt wird. Nachfolgend geben wir zur Illustration einen Auszug aus einer typischen Passwort-Policy an.

informelle
Richtlinie

Beispiel: Passwort-Richtlinie (Auszug)

- All system-level passwords (e.g., root, enable, NT admin) must be changed on at least a quarterly basis.
- All user-level passwords (e.g., email, web, desktop computer, etc.) must be changed at least every six months. The recommended change interval is every four months.
- User accounts that have system-level privileges granted through group memberships or programs such as *sudo* must have a unique password from all other accounts held by that user.
- Passwords must not be inserted into email messages or other forms of electronic communication.

Die Thematik des methodischen Erstellens von Regelwerken, des Policy-Engineerings, und systematischen Umsetzens (engl. *enforcement*) von Sicherheitsregeln ist noch Gegenstand aktiver Forschung. Wesentliche Fragestellungen hier betreffen die werkzeugunterstützte Erstellung von Regelwerken, die automatische Generierung von maschinell ausführbaren

Regelwerken, oder aber auch die Entwicklung von Systemarchitekturen zur kontrollierbaren, verteilten Umsetzung von Sicherheitsregelwerken.

Mehrseitige Sicherheit

In heutigen vernetzten IT-Systemen kommunizieren eine Vielzahl von Partnern miteinander, die ganz unterschiedliche Schutzziele verfolgen können. Betrachten wir zum Beispiel einen Content-Provider, der Web-Dienste anbietet. Der Provider ist sicherlich daran interessiert, möglichst viele und präzise Informationen über seine Kunden zu sammeln, um damit Kundenprofile für zugeschnittene Angebote zu erstellen. Der Kunde hat demgegenüber ein berechtigtes Interesse an Privatheit, also an der Preisgabe von möglichst wenigen personenbezogenen Informationen. Dieses Schutzbedürfnis von Benutzern wird in Deutschland auch durch das informationelle Selbstbestimmungsrecht unterstützt, das das Sammeln, Speichern und Verarbeiten personenbezogener Daten durch Dritte gesetzlich regelt (vgl. BVerfGE 65, 1, 49). Durch die im Mai 2018 in Kraft getretene EU-Datenschutzgrundverordnung wurden die Regeln für die Verarbeitung personenbezogener Daten europaweit neu geregelt und verschärft.

Der Interessenskonflikt zwischen dem Provider und seinen Kunden ist typisch für Szenarien, in denen mehrseitige (siehe u.a. [125]), einander durchaus auch widersprechende Sicherheitsinteressen zu berücksichtigen sind. Häufig stehen sich hierbei die Sicherheitsinteressen von Herstellern, Betreibern und Nutzern gegenüber. Mit der Zunahme von Telekooperationsanwendungen als Formen der entfernten Zusammenarbeit steigt der Bedarf nach Systemen und Anwendungen, die eine mehrseitige Sicherheit unterstützen. Es werden Systeme benötigt, die es erlauben, dass Kommunikations- und Geschäftspartner ihre beidseitigen gemeinsamen Sicherheitsinteressen aushandeln und sich auf ein gemeinsames Sicherheitsniveau einigen können. Mehrseitige Sicherheit und Wahrung der Privatsphäre spielt insbesondere bei E-Commerce-Transaktionen eine wichtige Rolle.

Abbildung 1.4 stellt die eingeführten Begriffe noch einmal miteinander in einen Zusammenhang und zeigt deren Abhängigkeiten auf.

1.6 Sicherheitsinfrastruktur

Die festgelegte Sicherheitsrichtlinie ist unter Nutzung von Sicherheitskonzepten, -mechanismen und -diensten einer Sicherheitsinfrastruktur zu realisieren und zu gewährleisten.

Definition 1.15 (Sicherheitsinfrastruktur)

Unter einer Sicherheitsinfrastruktur bzw. einer Sicherheitsarchitektur verstehen wir den Bestandteil einer System-Architektur, der die festgelegten

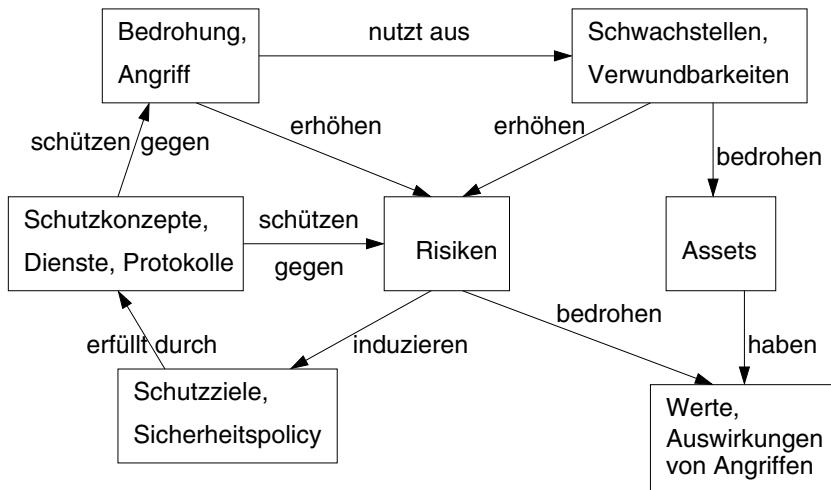


Abbildung 1.4: Zusammenhänge und Abhängigkeiten

Sicherheitseigenschaften durchsetzt und die für die Verwaltung der sicherheitsrelevanten Informationen und Konzepte erforderlichen Realisierungsmaßnahmen zur Verfügung stellt.

□

Abbildung 1.5 gibt einen stark vergröberten Überblick über wesentliche Komponenten (grau unterlegt) einer Sicherheitsarchitektur.

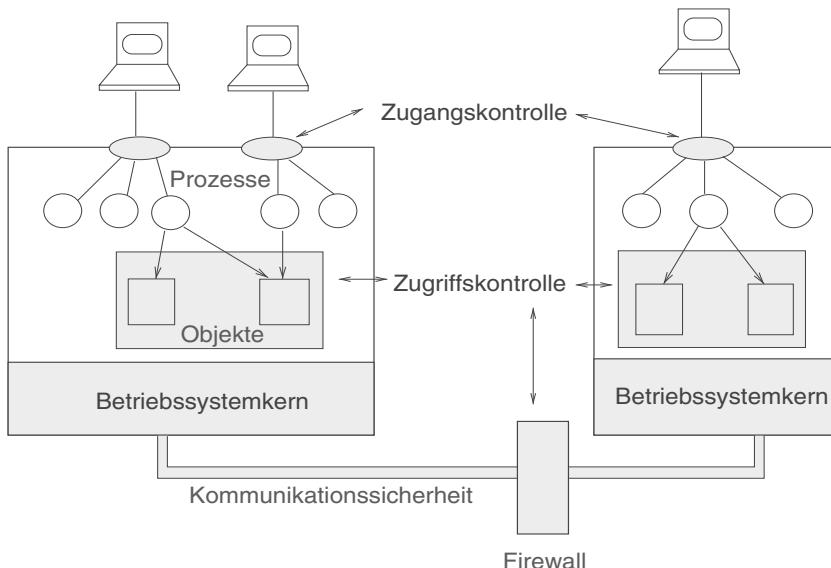


Abbildung 1.5: Komponenten einer Sicherheitsarchitektur (stark vergröbert)

Zugangskontrolle

Der kontrollierte Zugang eines Subjekts zum System erfolgt über die Zugangskontrolle, durch die das Subjekt identifiziert und authentifiziert wird. Auf Mechanismen und Verfahren zur Gewährleistung der Authentizität von Subjekten wird ausführlich in Kapitel 10 eingegangen.

Zugriffskontrolle

Die Zugriffskontrolle hat sicherzustellen, dass ein Zugriff auf ein zu schützendes Objekt nur den dazu berechtigten Subjekten gestattet wird. Der Zugriffskontrolle obliegt sowohl die Gewährleistung der gewünschten Integritätseigenschaften als auch die Durchführung von Informationsflusskontrollen, falls durch die Sicherheitsrichtlinie Vertraulichkeitseigenschaften gefordert sind. Der Betriebssystemkern stellt zusammen mit der Prozessor-Hardware Basismechanismen für diese Kontrollen bereit. Eine spezielle Ausprägung solcher Zugriffskontrollen realisieren Firewall-Rechner. Kapitel 12 widmet sich ausführlich den Techniken zur Durchsetzung der Zugriffskontrolle, während auf die Firewall-Technologie in Kapitel 14.1 eingegangen wird.

Kommunikation

Die Kommunikation in einem vernetzten IT-System erfolgt über Rechnernetze unter Nutzung von Kommunikationsprotokollen. Wesentliche Basismechanismen zur sicheren Kommunikation werden ausführlich in den Kapiteln 7 und 8 behandelt. Deren Einsatz zur Realisierung sicherer Kommunikationsprotokolle ist Gegenstand von Kapitel 14.

Ganzheitliche Sicherheit

Der Bereich der organisatorischen, baulichen und administrativen Maßnahmen zum Schutz eines Systems ist in Abbildung 1.5 nur implizit enthalten. Hierzu gehört der Diebstahlschutz ebenso wie speziell abgeschirmte Räume für den Systemzugang oder die Isolierung von Systemkomponenten in besonders geschützten Gebäudeteilen. Auf die organisatorischen und baulichen Maßnahmen gehen wir im Folgenden nicht ein, da das vorliegende Buch den Schwerpunkt auf die Behandlung der technischen Aspekte der Systemsicherheit legt. Wichtige administrative Aufgaben sowie organisatorische Regelungen, die das technische System selbst betreffen, werden an den entsprechenden Stellen aufgegriffen. Wobei auch hierfür gilt, dass die erschöpfende Behandlung aller Fragen eines ganzheitlichen Sicherheitsmanagements den Rahmen des Buches bei Weitem sprengen würde.

Business Continuity

Zu organisatorischen Maßnahmen gehört beispielsweise auch die Etablierung eines Notfallkonzeptes mit einer Planung sowie mit Vorgehensmodellen, wie in der jeweiligen Organisation systematisch und methodisch auf Sicherheitsvorfälle reagiert werden muss. Dies wird häufig unter dem Begriff Incident Handling zusammengefasst. Ferner gehören zu diesem Bereich auch Konzepte und Maßnahmen, um beim Eintreten eines Schadensfalls dafür zu sorgen, dass wichtige, die Interessen der Organisation betreffende Geschäftsprozesse möglichst aufrecht erhalten bleiben bzw. möglichst schnell wieder durchführbar sind. Diese Problematik wird un-

ter dem Stichwort Business Continuity Planning zusammengefasst. Um derartige Planungen und Strategien in Unternehmen zu etablieren, sind vertiefte Kenntnisse der Geschäftsprozesse des Unternehmens sowie betriebswirtschaftliche Analysen über Kosten und Risiken erforderlich. Auch die Behandlung dieser Thematik, obwohl sie ohne Zweifel sehr wichtig und interessant ist, geht über den Rahmen des vorliegenden Buches hinaus. Aussagekräftige Hinweise für den Bereich der organisatorischen Maßnahmen liefert unter anderem die IT-Grundschutz-Kataloge¹⁵ des Bundesamts für Sicherheit in der Informationstechnik (BSI). Die IT-Grundschutz-Kataloge umfassen rund 4500 Seiten und sind als jährlich aktualisierte Loseblattsammlung erhältlich¹⁶.

Sicherheitskette

Die folgenden Kapitel beschäftigen sich ausführlich mit den wichtigsten Teilbereichen der IT-Sicherheit. Es werden die heute gängigen Maßnahmen, Konzepte und Protokolle beschrieben, die man zur Lösung der jeweiligen Probleme einsetzen kann. Es muss jedoch stets klar sein, dass ein Absichern eines Teilbereichs eines Systems unzureichend ist, um tatsächliche IT-Sicherheit zu gewährleisten. Das Bild der Sicherheitskette verdeutlicht dies sehr gut, da eine Kette nur so stark ist, wie ihr schwächstes Glied.

Sicherheitskette

Abbildung 1.6 veranschaulicht diese Vorstellung. Sie zeigt sicherheitsrelevante Komponenten eines IT-Systems, sowie exemplarische Bedrohungen, die darauf abzielen, Schwachstellen der Komponenten auszunutzen und die Sicherheitskette zu brechen.

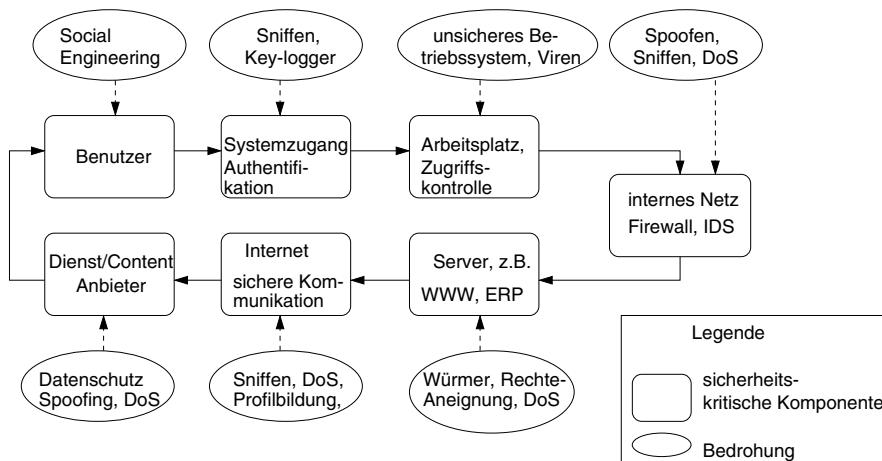


Abbildung 1.6: Sicherheitskette und ausgewählte Bedrohungen

¹⁵ Bis 2005 unter dem Namen Grundschutzhandbuch bekannt.

¹⁶ https://www.bsi.bund.de/DE/Themen/ITGrundschutz/ITGrundschutzKataloge/Inhalt/_content/kataloge.html

Natürlich kann man diese Kette auch verkürzen, falls in dem real existierenden IT-System Komponenten fehlen (z.B. der Zugriff auf externe Content-Anbieter). Das Bild dient dazu, auf eine sehr vergrößerte Weise die Abhängigkeiten zu veranschaulichen, die zwischen den Systemkomponenten bestehen.

Kettenglieder

Die Kette fängt beim Benutzer an, der natürlich in vielerlei Hinsicht eine Schwachstelle darstellt, was durch die möglichen Social Engineering-Angriffe angedeutet wird. Bevor ein Benutzer Dienste nutzen kann, muss er sich bei einem Rechner bzw. dem System anmelden, also authentifizieren. Hier ergeben sich weitere Angriffspunkte, falls der Authentifizierungsvorgang ungenügend abgesichert ist, sei es, dass zum Beispiel Passwörter im Klartext über das Netz übertragen werden, oder dass durch Key-logger Programme die Tastatureingaben abgefangen werden. Hat sich der Benutzer auf einem Arbeitsplatzrechner (u.a. PC, Laptop, Smartphone, Tablet) angemeldet, so muss dieser Zugriffscontrollen durchführen, um sicher zu stellen, dass nur autorisierte Daten- bzw. Informationszugriffe möglich sind. Mögliche Bedrohungen der Datenintegrität und Vertraulichkeit ergeben sich u.a. durch Viren oder auch durch unsichere Betriebssystemdienste wie beispielsweise manipulierte Gerätetreiber. Dies setzt sich fort, wenn innerhalb eines Unternehmensnetzes auf Daten zugegriffen werden soll, die sich auf unternehmenslokalen Servern (z.B. Mail-Server, WWW-Server, Enterprise Resource Planning (ERP) Programme) befinden. Dazu wird ein lokales Netz zur Datenkommunikation verwendet, das Bedrohungen u.a. durch Lauschangriffe (Sniffen) ausgesetzt ist. Handelt es sich bei dem lokalen Netz um ein drahtloses Zugangsnetz (WLAN), so sind die Bedrohungen, die durch das Abhören der Funkverbindungen auftreten, offensichtlich. Ist auch von außen ein Zugang über das lokale Netz möglich, so sollten die Daten über geeignete Firewall-Architekturen oder mittels Intrusion Detection Systeme kontrolliert und gefiltert werden. Soll darüber hinaus über ein öffentliches Netz wie dem Internet auf externe Content-Anbieter (u.a. amazon.com, e-commerce Portale) zugegriffen werden, so ist zum einen die Kommunikation dorthin abzusichern und zum anderen muss man darauf vertrauen können, dass die Daten auch auf den externen Servern sicher und u.a. datenschutzkonform verwaltet werden. Beispiele für Bedrohungen sind das Erstellen von datenschutzgefährdenden Benutzerprofilen.

Ganzheitliche Sicherheit

Fordert man ein angemessenes Sicherheitsniveau für ein System, so erfordert das eine ganzheitliche Absicherung der Systeme; eine Fokussierung auf Einzelaspekte (u.a. Beschränkung der Maßnahmen zur IT-Sicherheit auf die Konfigurierung einer Firewall und eines Virenscanners), wie es heute noch immer an der Tagesordnung ist, erweist sich in der Praxis immer wieder als unzureichend. Die Sicherheitskette verdeutlicht, dass z.B. eine sichere

Kommunikation nicht ausreicht, um Daten während des gesamten Verarbeitungsprozesses zu schützen. Erlangt ein Benutzer unautorisiert Zugriff auf diese Daten, wenn sie auf einem der beteiligten Rechner verarbeitet oder gespeichert werden, so bleibt der alleinige Schutz der Kommunikationsverbindung wirkungslos; die getroffenen Sicherheitsmaßnahmen sind nicht ausreichend.

Neue Herausforderungen

Das Bild der Sicherheitskette veranschaulicht jedoch auch die Problematik, die sich angesichts der zunehmenden Durchdringung unseres beruflichen und privaten Alltags mit IT-Technologie ergibt. Diese Entwicklung ist unter dem Begriff des Internet of Things (IoT) bekannt. Charakteristisch ist die Einbettung von Sensoren und Aktoren in Alltagsdinge und deren drahtlose Vernetzung. Eine bekannte Ausprägung dieser Entwicklung ist die weite Verbreitung von RFID-Chips, die zur Identifikation von Objekten und zum Überwachen von Objekten (u.a. Container, Lebensmittel, Medikamente, Chemikalien) insbesondere in Logistik-Prozessen verwendet werden. Bereits heute werden 99% aller neu produzierten Halbleiter in eingebetteten Systemen eingesetzt. Nach Schätzung von Roland-Berger¹⁷ liegt in einer modernen Industriegesellschaft der Anteil von Produkten, die eingebettete Systeme enthalten, bei ca. 80% der gesamten Wertschöpfung der verarbeitenden Industrie. Hand in Hand mit der Zunahme eingebetteter Hardware-Komponenten wächst auch das Volumen der darauf eingesetzten Software drastisch. Ein Beispiel mag diese Entwicklung verdeutlichen: Die in einem einzelnen Pkw eingesetzten Softwarekomponenten umfassten 2014 ca. 10 Millionen Codezeilen, für den Ford Pickup F150 wurden 2016 bereits über 150 Millionen Lines of Code angegeben. Angesichts der Ausrichtung der Automobilindustrie auf das autonome Fahren ist mit einem weiteren Anstieg zu rechnen. Zum Vergleich: Das Betriebssystem Windows 10 umfasst ca. 60-70 Millionen Codezeilen.

Eingebettete Systeme

Eingebettete Systeme verfügen häufig über sehr beschränkte Speicher, Rechen- und Energieressourcen, so dass klassische Sicherheitstechniken, wie sie im vorliegenden Buch behandelt werden, in der Regel nicht direkt übertragbar sind. Auch aufwändige Protokolle, zum Beispiel zum Austausch von Kommunikationsschlüssel, können nicht übernommen werden, so dass neue Verfahren für ein effizientes und vor allem auch skalierendes Schlüsselmanagement zu entwickeln sind. Eingebettete Systeme können oder dürfen häufig nach ihrem Ausrollen nicht mehr verändert werden, da sie durchgehend und häufig unter Realzeit-Anforderungen ihre Funktionen (z.B. Überwachungsfunktionen) erfüllen müssen. Die Entwicklung

¹⁷ Vgl. http://www.bitkom.org/files/documents/Zukunft_digitale_Wirtschaft_BITKOM-Roland_Berger_Studie.pdf

sicherer Change- und Update-Mechanismen ohne dass es dadurch zu einer sicherheitskritischen Beeinträchtigung des Systems und der umgebenden Infrastruktur kommt, sind ebenfalls noch weitestgehend offene Fragestellungen. Teilweise lassen sich existierende Sicherheitslösungen adaptieren, teilweise müssen auch existierende IT-Sicherheitsinfrastrukturen verändert werden, um den zusätzlichen Gefährdungen, die sich aus den ubiquitären Rechnern ergeben, zu begegnen.

CPS

Mit dem Vordringen der IKT Technologien in alle Lebens- und Arbeitsbereiche ist eine zunehmende Konvergenz von IT-basierten und physischen Systemen verbunden. Es entstehen so genannte Cyber Physical Systems [180], die häufig in sicherheitskritischen Anwendungen wie in Transportsystemen, der Energieversorgung oder auch der Logistik eingesetzt werden.

CPS sind charakterisiert durch eine große Heterogenität, starke Vernetzung, und eine zunehmende Integration von steuernden Systemen (eingebettete Systeme) in physikalische Umgebungen und betriebliche Prozesse (Physical System). Diese Systeme ermöglichen eine verstärkte Automatisierung und eine Steuerung ganzer Wertschöpfungsnetze in nahezu Echtzeit. CPS eröffnen ein sehr großes Potential an technologischen Innovationen, neuen Anwendungen und Geschäftsmodellen, wie in dem acatech-Roadmap Projekt *agendaCPS*¹⁸ ausführlich dargestellt. Die Voraussetzung dafür ist, dass die Daten rechtzeitig, vollständig, nicht manipuliert und ggf. auch vertraulich an die berechtigten Empfänger gelangen. Die Frage der Sicherheit ist deshalb für Cyber-Physical Systems von sehr großer Bedeutung.

Heterogene Komponenten und heterogene Vernetzungstechnologien werden im Internet der Zukunft unter dem gemeinsamen Dach des IP Protokolls zusammengeführt (All-IP) und sind damit den bekannten Gefährdungen ausgesetzt. Derartige Systeme müssen also hohe Sicherheits- und Zuverlässigkeitssanforderungen erfüllen. Angriffe wie der Stuxnet-Wurm im Jahr 2010 haben jedoch verdeutlicht, dass Cyber-Physical Systeme heute noch in hohem Maße verwundbar sind.

Forschungsfragen

Offene Forschungsfragen betreffen beispielsweise die Entwicklung vertrauenswürdiger Hardware-naher, eingebetteter Sicherheitslösungen, die als robuste und angriffsresistente Bausteine in physische Komponenten integriert werden können. Die Entwicklung von neuen Konzepten, Protokollen und Lösungen, zur Absicherung vernetzter Cyber-Physical Systeme und die frühzeitige Erkennung von möglichen Störungen und Angriffen stellen weitere zentrale Forschungsfragen von hoher wirtschaftlicher Relevanz dar. Angriffsversuchen müssen mit hoher Treffergenauigkeit frühzeitig erkannt

¹⁸ vgl. www.acatech.de/cps

werden, um eine kaskadierende Ausbreitung mit den entsprechenden Schäden zu verhindern.

Unter dem Begriff Cyber-Sicherheit oder auch Cyber Security fasst man die Herausforderungen in Bezug auf Sicherheit zusammen, die sich aus der Vernetzung von IKT Systemen und den zunehmenden Abhängigkeiten von vernetzten, sicherheitskritischen Infrastrukturen ergeben. Die Gesamtheit der IT-Infrastrukturen in Unternehmen, Behörden, oder aber auch in Produktionsanlagen, im Gesundheitswesen oder im Finanzbereich und der Logistik, die über das Internet oder vergleichbare Vernetzungstechnologien zugreifbar sind, werden als Cyber-Raum bezeichnet. Unter Cyber-Sicherheit versteht man deshalb auch die Ausweitung der Aufgaben der klassischen IT-Sicherheit auf den ganzen Cyber-Raum.

Cyber Security

In der öffentlichen Diskussion wird derzeit im Kontext Cyber-Sicherheit noch vordringlich die Frage diskutiert, mit welchen organisatorischen, technischen und rechtlichen Maßnahmen die Kriminalität, Wirtschaftsspionage und Sabotage aus dem Netz eingedämmt und eine höhere Transparenz in Bezug auf Sicherheitsvorfälle erzielt werden kann. Im Vordergrund der auf der Bundesebene geführten Debatte steht dabei die Verbesserung des Lagebildes über die Gefährdungslage in Deutschland, wofür im Jahr 2012 die Allianz für Cyber-Sicherheit¹⁹ als eine gemeinsame Initiative des BSI und des BITKOM²⁰ gestartet wurde.

Eine verbesserte Informationslage, um mögliche Gefährdungen frühzeitig zu erkennen und Abwehrmaßnahmen ergreifen zu können, ist sicherlich sehr wichtig. Langfristig tragfähige Lösungen erfordern jedoch die Entwicklung neuer Sicherheits-Technologien, die den Anforderungen hochgradig, u.U. spontan vernetzter und eingebetteter Systeme, z.B. in Bezug auf Energie-Effizienz oder Realzeit-Anforderungen, erfüllen. Neue Sicherheitskontroll- und Schutzmaßnahmen müssen bereits frühzeitig in den Entwurf der Systeme integriert werden, um zukünftige IT-basierte Produkte und Systeme robuster und resistenter gegen insbesondere auch Internet-basierte Angriffe zu gestalten.

Lagebild

Zur Lösung aller dieser offenen Fragen sind Kenntnisse über die klassischen Probleme in der IT-Sicherheit sowie der Konzepte, Protokolle und Verfahren zu deren Lösung, wie sie in den nachfolgenden Kapiteln erläutert werden, unumgänglich.

¹⁹ <http://www.allianz-fuer-cybersicherheit.de>

²⁰ Bundesverband der Informationswirtschaft, Telekommunikation und neue Medien e.V.

2 Spezielle Bedrohungen

Das Kapitel widmet sich Klassen von Bedrohungen, die Ursache für die in heutigen Systemen am häufigsten auftretenden Sicherheitsprobleme sind. Dazu zählen Bedrohungen durch Pufferüberlauf-Angriffe, Viren, Würmer und Trojanische Pferde sowie mobile Applikationen (Apps). Nicht abgefangene Pufferüberläufe (engl. *Buffer Overflow*) sind noch immer sehr weit verbreitete Schwachstellen, die durch nachlässige Programmierung sowohl in Betriebssystemen als auch in Anwendungsdiensten auftreten. Abschnitt 2.2 erläutert die Buffer-Overflow-Problematik, erklärt die prinzipielle Funktionsweise von Exploits, die solche Schwachstellen ausnutzen, und diskutiert Abwehrmaßnahmen. Abschnitt 2.3 beschäftigt sich dann mit dem allgemeinen Aufbau und der Funktionsweise von Computerviren und gibt Hinweise auf wirksame Maßnahmen zur Virenabwehr. Die charakteristischen Eigenschaften von Würmern zusammen mit möglichen Abwehrmaßnahmen werden in Abschnitt 2.4 behandelt. Abschnitt 2.5 widmet sich in gleicher Weise der Thematik der Trojanischen Pferde. In den letzten Jahren wurden zunehmend Angriffe über sogenannte Bot-Netze durchgeführt. Abschnitt 2.6 erläutert die prinzipielle Arbeitsweise eines Bot-Netzes. Abschnitt 2.7 beleuchtet die Probleme, die mit Apps einhergehen können. Abschließend gehen wir in Abschnitt 2.8 auf die Anfang 2018 bekannt gewordenen speziellen Sicherheitsprobleme Meltdown und Spectre ein, von denen nahezu alle heute im Einsatz befindlichen Prozessoren betroffen sind.

2.1 Einführung

Spezielle Klassen von Bedrohungen eines IT-Systems gehen von Viren, Würmern und Trojanischen Pferden aus, die wir als Schadsoftware (engl. *malware* oder *malicious code*) bezeichnen. Es handelt sich hierbei um Programme, die festgelegte, dem Benutzer jedoch verborgene Funktionen ausführen. Die Festlegung der zusätzlichen und beabsichtigten (i.d.R. böswilligen) Funktionalität unterscheidet diese Klasse von Programmen von den so genannten Wanzen (bugs), deren Fehlverhalten meist von nicht beabsichtigten Programmierfehlern verursacht wird. Der Befall von IT-Systemen durch Viren, Würmer und Trojanische Pferde führt jedes Jahr zu erheblichen

wirtschaftlichen Schäden. So schätzte die US-Beratungsfirma Computer Economics¹ den Schaden, den Schadsoftware bereits 2006 weltweit angerichtet hat, auf über 13 Milliarden Dollar. Im Jahr 2008 hat die Firma Symantec 1 656 227 neue Auftreten von Schadsoftware erkannt. Das bedeutet einen Anstieg um 265 Prozent im Vergleich zu 2007². Diese steigende Tendenz hat über die Jahre angehalten. So hat das Deutsche Bundeskriminalamt (BKA) für das Jahr 2016 fast 100.000 Fälle von Internetkriminalität registriert mit einem Schaden von über 50 Millionen Euro für Privathaushalte, wobei der Graubereich der nicht gemeldeten oder nicht bemerkten Vorfälle sehr hoch ist und in den Statistiken gar nicht auftritt. Im Mai 2017 verursachte allein der Trojaner WannaCry den Ausfall von über 200.000 nicht sorgfältig gepatchte Windows-Systemen weltweit mit Schäden in mehreren hundert Millionen Dollar.

Architektur-Eigenschaften

Basiskonzepte und -eigenschaften heutiger Rechnerarchitekturen erleichtern das erfolgreiche Durchführen von Buffer-Overflow-Angriffen bzw. ermöglichen die Existenz von Trojanischen Pferden und die Verbreitung von Viren und Würmern. Zu nennen ist dabei die gemeinschaftliche (aber nicht notwendigerweise gleichzeitige) Nutzung von Programmen und Daten (engl. *sharing*). Eine solche gemeinsame Nutzung wird u.a. über eine zum Teil sehr freizügige Netzwerkfreigabe oder über öffentliche Verzeichnisse ermöglicht. Die gemeinsame Nutzung von Ressourcen ist ebenfalls ein Charakteristikum von Peer-to-Peer Overlay-Netzen, die sich seit einiger Zeit etablieren. Weit verbreitet sind Musiktauschbörsen und File-Sharing Systeme wie Gnutella, KaZaA oder BitTorrent. Die Eigenschaft der gemeinsamen Ressourcennutzung wird insbesondere von Viren zur Verbreitung ausgenutzt. Eine weitere wesentliche Eigenschaft ist die universelle Interpretierbarkeit von Daten. Sie besagt, dass kein Unterschied zwischen Daten und Programmen besteht. Programme können demnach wie Daten leicht modifiziert werden, indem man sie beispielsweise um einen Schadensteil erweitert. Die universelle Interpretierbarkeit bedeutet aber auch, dass in Speicherbereiche, die eigentlich ausschließlich Daten beinhalten wie beispielsweise die Laufzeitkeller (engl. *stack*) von Prozessen, gezielt ausführbarer Code geladen werden kann. Werden diese binären Daten nun als ausführbarer Code interpretiert, so kann auf diese Weise Schadsoftware zur Ausführung gelangen. Dies wird von den Buffer-Overflow-Angriffen ausgenutzt.

Als dritte wesentliche Eigenschaft ist die Transitivität des Informationsflusses zu nennen. Falls eine Information vom Rechner A zum Rechner B fließt

¹ Quelle: Studie von Computer Economics http://www.welt.de/die-welt/article1037708/Computer_werden_zu_Zombies.html

² Quelle: Symantec Threat Report, <http://www.symantec.com/threatreport/>

und es Möglichkeiten gibt, Informationen von B zu einem Rechner C fließen zu lassen, so kann die Information auch von A zu C weitergeleitet werden. Die Transitivität wird insbesondere von Würmern ausgenutzt, um sich auf diese Weise zu verbreiten.

Techniken zur Bedrohungsabwehr

Zur Abwehr der Bedrohungen kann man drei Ansätze verfolgen, die in der Praxis in der Regel in verschiedenen Kombinationen eingesetzt werden. Zu nennen sind an erster Stelle Techniken zur Verhinderung (engl. *prevention*) von Angriffen wie beispielsweise die Verwendung stark typisierter Programmiersprachen, Code-Verifikation oder die Beschränkung von Zugriffen aufgrund von systembestimmten Regeln (z.B. kein Schreiben mit anschließendem Ausführen von Daten auf dem Stack-Segment). Im weitesten Sinn gehören in diese Klasse von Techniken auch Maßnahmen zur Sensibilisierung und Bewusstseinsbildung insbesondere auch von Entwicklern von Software-Systemen. Klar ist, dass Vermeidungstechniken nach dem Motto „Sicherheit per Design“ die konzeptuell befriedigendste Lösung darstellen, wir aber heute in der Praxis der System-Konstruktion hiervon noch sehr weit entfernt sind.

Verhinderung

Die zweite Klasse von Ansätzen beschäftigt sich deshalb mit Techniken zur Erkennung und frühzeitigen Abwehr möglicher Bedrohungen. Als Beispiele seien statische Analysen des Programmcodes genannt, um z.B. Code-Stücke, die eine Buffer-Overflow-Schwachstelle aufweisen (u.a. Kopieren von Zeichenfolgen ohne die Längenangaben zu prüfen), zu identifizieren und zu korrigieren. Aber auch die bekannten Viren-Scanner und Intrusion Detection Systeme gehören in diese Klasse von Techniken, die anhand charakteristischer Merkmale versuchen, potentielle Schadsoftware zu erkennen.

Erkennung und
Behebung

Zu der dritten Klasse von Ansätzen zählen Techniken, die darauf abzielen, die Auswirkungen von eingetretenen Bedrohungen zu begrenzen (engl. *mitigation*). Dazu zählen Isolierungsansätze wie das Sandboxing, die darauf abzielen, Anwendungsbereiche gegeneinander abzuschotten, so dass ein Schaden, der bei der Ausführung eines Anwendungsprogramms auftritt, sich nicht auf andere Anwendungsbereiche ausbreiten und womöglich das ganze System beeinträchtigen kann.

Schadens-
begrenzung

2.2 Buffer-Overflow

Buffer-Overflow-Angriffe nutzen Schwachstellen aus, die sich aus Implementierungsfehlern als Folge einer nachlässigen Programmierung ergeben. Der Ansatzpunkt für entsprechende Angriffe sind Programme, in denen Daten in einen Bereich einer Variablen fester Länge, z.B. in einen String oder ein Feld fester Länge, eingelesen bzw. kopiert werden, ohne dass das

Buffer-Overflow

Programm prüft, ob die kopierte Eingabe überhaupt in den bereitgestellten Bereich passt. Derartige Variablen, in die Daten kopiert werden, sind abstrakt gesehen Pufferbereiche, woraus sich der Name dieses Angriffs ableitet. Durch das Schreiben einer zu großen Datenmenge in den Bereich des Puffers wird dieser zum Überlauf (engl. *overflow*) gebracht.

2.2.1 Einführung

Wie auch bei den Viren, so kann man auch bei den Buffer-Overflow-Angriffen verschiedene zeitliche Entwicklungsphasen unterscheiden. Während die erste Generation der Angriffe vordringlich Operationen zum Kopieren von Zeichenketten für Angriff ausnutzten, werden heute zunehmend auch einzelne Integer-Zahlen zum Überlaufen gebracht oder es werden Anweisungen, wie Schleifen, die nicht korrekt terminieren, für Überlauf-Angriffe missbraucht.

C, C++

Schwachstellen, die für Buffer-Overflow Angriffe ausgenutzt werden können, finden sich häufig in Programmen, die in den Programmiersprachen C bzw. C++ geschrieben sind. Probleme ergeben sich beispielsweise durch die Verwendung von Funktionen zur Verarbeitung von Zeichenketten, wie `strcpy()`. Diese Funktion kopiert Zeichen für Zeichen des Eingabestrings auf den Zielstring, ohne dabei zu prüfen, ob der Bereich des Zielstrings groß genug ist, um die Eingabe zu speichern. Der Kopievorgang terminiert erst dann, wenn im Eingabestring der ASCII-Wert 0 gelesen wird. Auf diese Weise ist es möglich, gezielt vorhandene Werte von Variablen, die auf dem Speicherbereich abgelegt sind, der sich unmittelbar an den Bereich des Zielstrings anschließt, mit eingeschleusten Werten zu überschreiben.

Häufig sind auch Funktionen wie `sizeof()` Ausgangspunkt für eine Buffer-Overflow Verwundbarkeit. So wird beispielsweise bei der Transformation von Daten in Unicode ein Datenstring einer bestimmten Länge, die durch das Kommando `sizeof()` bestimmt wird, in einen anderen Datenstring kopiert. In der Unicode-Transformation wird die Größenangabe jedoch in Byte und nicht in der Anzahl der Characters berechnet, so dass bei einer falschen Verwendung viel mehr Speicher überschrieben wird als vorgesehen war. Ein weltweit bekanntes Beispiel eines Angriffs, der eine derartige Schwachstelle ausnutzte, war der Code-Red Wurm im Jahre 2001, der einen Pufferüberlauf in Microsofts IIS (Internet Information Service) ausnutzte, um das System zu befallen.

Wie bereits weiter oben kurz angedeutet, werden heutzutage auch zunehmend Schwachstellen ausgenutzt, die sich durch die unsichere Programmierung einer Loop-Schleife ergeben. Häufiges Beispiel ist hier das zeichenweise Kopieren eines Eingabestrings in dieser Loop-Schleife, z.B. das Kopieren einer URL, wobei als Abbruchkriterium der Schleife das Le-

sen eines bestimmten Zeichens festgelegt wird. Fehlt dieses Abbruchzeichen in der Zeichenkette oder taucht es gezielt erst nach einer Menge von anderen Daten auf, so werden alle diese Daten Zeichen für Zeichen kopiert. Der bekannte Blaster-Wurm (siehe Seite 68) hat beispielsweise eine solche Schwachstelle ausgenutzt.

Da unter anderem alle UNIX-Derivate (u.a. BSD-UNIX, Solaris, HP-UX, Linux) und die aktuellen Windows-Versionen in C bzw. C++ geschrieben sind, bilden immer wieder fehlerhaft programmierte Betriebssystemdienste, Serverroutinen oder auch Bibliotheksfunktionen Angriffspunkte für Buffer-Overflows.

In einem Programmfragment in der Programmiersprache C könnte ein möglicher Ansatzpunkt für Buffer-Overflow Angriffe wie folgt gegeben sein:

```
cmd =lese_aus_netz();
do_something(cmd);
int do_something(char *string) {
    char buffer[4]; /* Pufferbereich */
    strcpy(buffer,string); /* Ansatzpunkt: */
    /* Einlesen von Daten in Pufferbereich */
    ....
    return 0;
}
```

Ansatzpunkt

Adressraumverwaltung

Um die Vorgehensweise und Ziele eines Buffer-Overflow-Angriffs verstehen zu können, muss man sich zunächst einmal klarmachen, auf welche Weise in heutigen Betriebssystemen die Daten eines Prozesses, also eines in Ausführung befindlichen Programms, verwaltet werden. Alle gängigen Betriebssysteme unterstützen heute virtuellen Speicher. Jedem Prozess ist ein virtueller Adressraum zugeordnet, der – wie in Abbildung 2.1 skizziert – vom Betriebs- und Laufzeitsystem verwaltet wird.

virtueller
Adressraum

Typischerweise wird der virtuelle Adressraum in drei logische Bereiche, die Segmente genannt werden, aufgeteilt, wobei das Text- bzw. Codesegment die ausführbaren Befehle des Prozesses enthält. In der Regel wird dieses Segment vom Betriebssystem gegen ein Überschreiben geschützt. Auf dem Halden-Bereich (engl. *heap*) werden zur Programmalaufzeit dynamisch erzeugte Datenobjekte abgelegt, aber auch globale Variablen sowie globale Konstanten. Typische dynamische Objekte sind Zeigerstrukturen oder Objekte in objektorientierten Programmiersprachen. Der Heap-Bereich wächst normalerweise von niedrigen Adressen zu höheren Adressen. In der Regel ist die Trennlinie zwischen dem Heap und dem Stack-Segment fließend.

Segmente

Heap

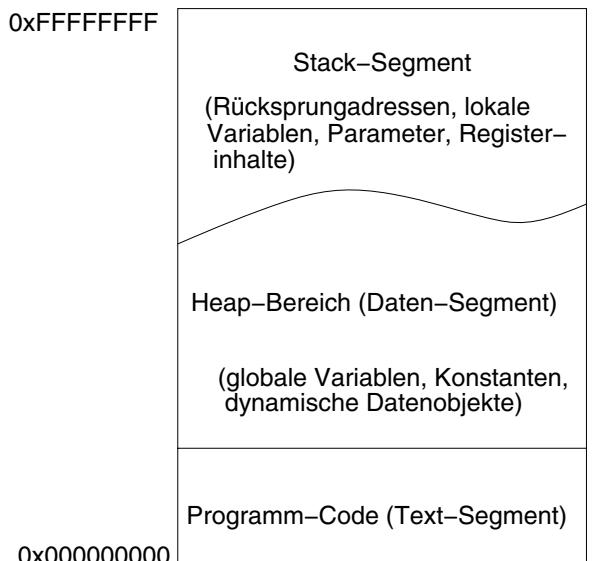


Abbildung 2.1: Segmentierung des virtuellen Prozessadressraums

Stack

Der Laufzeitkeller, das Stacksegment (engl. *stack*), wird LIFO³-artig verwaltet und dient zur Verwaltung von Prozeduraufrufen. Auf dem Stack werden u.a. lokale Variablen von Prozeduraufrufen, deren Eingabeparameter sowie die Rücksprungadresse des Aufrufs abgelegt. Der Stack-Bereich wächst von hohen zu niedrigen Adressen; er wächst also dem Heap entgegen. Nach Beendigung eines Prozeduraufrufs werden die lokalen Daten, Parameter und sonstigen Umgebungswerte des Aufrufs wieder vom Stack entfernt. Bei der Programmübersetzung erzeugt der jeweilige Compiler Maschinencode, um diese Daten auf Stack und Heap-Bereichen zu speichern bzw., wie im Falle des Stack-Bereichs, sie auch wieder zu entfernen.

Schutz?

Im Unterschied zum Code-Segment können weder das Stack- noch das Heap-Segment vom Betriebssystem gegen Überschreibungen geschützt werden, da ein schreibender Zugriff ja gerade beim Ablegen von dynamischen lokalen bzw. globalen Daten benötigt wird. Gravierender ist, dass die Daten in diesen Bereichen in der Regel nicht nur gelesen und geschrieben werden dürfen, sondern auch ausführbar sind. Wir werden bei der Diskussion der Gegenmaßnahmen hierauf zurückkommen.

2.2.2 Angriffe

Wie bereits einführend erwähnt, ist es das Ziel eines Buffer-Overflow-Angriffs den Bereich, der für eine Variable im Speicher reserviert ist, zu

³ Last in first out

überschreiben und in der Regel eben auch zum Überlaufen zu bringen. Derartige Daten können prinzipiell an unterschiedlichen Stellen im Speicher abgelegt werden. Dazu zählt in erster Linie das Stack-Segment, auf dem lokale Variablen sowie Eingabeparameter, die bei Prozedur- bzw. Methodenaufrufen verwendet werden, abgelegt werden. Es kann sich aber ebenso um den Heap-Bereich des Prozess-Adressraums handeln, auf dem die dynamisch erzeugten Datenobjekte liegen, oder aber auch um einen Bereich handeln, der von mehreren Prozessen gemeinsam genutzt und an jeweils unterschiedliche Stellen des virtuellen Adressraums des einzelnen Prozesses eingeblendet wird. Im Folgenden konzentrieren wir die Beschreibung von Angriffen auf die Stack-Bereichs-Überläufe, da sie sehr häufig auftreten. Aber natürlich muss man sich der anderen Angriffspunkte stets bewusst sein.

Buffer-Overflows auf Stack-Segmenten

Ein Buffer-Overflow-Angriff zielt nun darauf ab, über Eingabeparameter bei einem Prozederaufruf den Speicherbereich derjenigen lokalen Variable (d.h. den Puffer) auf dem Stack, die diese Eingabe aufnehmen soll, zum Überlauf zu bringen. Das Ziel besteht dann meist darin, durch einen gezielt konstruierten Eingabestring die auf dem Stack abgelegte Rücksprungadresse des Prozederaufrufs zu überschreiben (vgl. Abbildung 2.2). Hierbei wird wesentlich die Organisation des Stacks ausgenutzt, der wie gesagt von hohen zu niedrigen Adressen wächst. Bei einem Prozederaufruf werden stets zunächst die Verwaltungsdaten wie z.B. die Rücksprungadresse auf dem Stack abgelegt, bevor Speicherplatz für die lokalen Variablen, die zur Prozederausführung benötigt werden, angelegt wird. Bei der Speicherung von aktuellen Werten in diesen reservierten Speicherbereichen, also z.B. bei der Übernahme der Eingabewerte eines Eingabestrings, werden die Daten beginnend bei den niedrigen Anfangsadressen der lokalen Variablen auf dem Stack abgelegt.

überschreiben

An Bild 2.2 kann man sich unmittelbar klarmachen, dass durch eine zu lange Eingabe die gespeicherten Werte, die auf dem an den für die lokale Variable reservierten Adressbereich unmittelbar angrenzenden Bereich abgelegt sind, überschrieben werden.

Einschleusen von Fremd-Code

Nach dem Überschreiben der Rücksprungadresse kann der Fall vorliegen, dass die überschriebenen Speicherzellen keine sinnvolle Adresse mehr enthalten, so dass das Programm bei der Ausführung des Rücksprungbefehls aufgrund eines Speicherzugriffsfehlers (engl. *segmentation fault*) abstürzen wird. Damit hat der Angreifer zwar u.U. bereits großen Schaden verursacht, aber ggf. noch nicht sein eigentliches Ziel, ausführbaren Code auf dem Stack zu platzieren, erfolgreich erreicht. Der Angreifer wird also versuchen, seinen Eingabestring so zu konstruieren, dass über die Rücksprungadres-

Fremd-Code

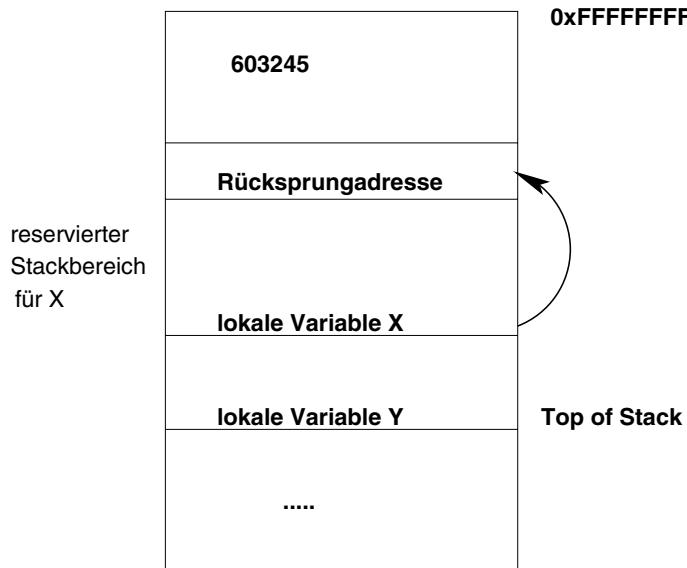


Abbildung 2.2: Buffer-Overflow mit Überschreiben der Rücksprungadresse

se die Anfangsadresse des eingeschleusten Schadcodes oder ein anderer, bereits geladener Code angesprungen und dann ausgeführt wird. Der eingeschleuste Code wird mit den Berechtigungen des Prozesses ausgeführt, dessen Stack durch den Buffer-Overflow manipuliert wurde. Da der Angreifer auf diese Weise seine Zugriffsrechte vermehrt und im Falle von befallenen Systemprozessen auch anhebt, spricht man auch häufig von Elevation of Privileges-Attacken. Ein erfolgreicher Buffer-Overflow-Angriff ist nur ein Beispiel dafür, wie man als Angreifer seine Rechte ausweiten kann.

Das Problem, das sich hierbei ergibt, besteht darin, dass der Angreifer die genaue, d.h. die absolute Adresse seines eingeschleusten Codes auf dem Stackbereich kennen muss, um gezielt dorthin zu springen. Die absolute Adresse kann jedoch nicht unbedingt vorab ermittelt werden, da sie unter anderem davon abhängig ist, wie viele lokale Variablen und Registerinhalte auf dem Stack abgelegt werden. Für dieses Problem gibt es jedoch eine einfache Lösung. In den zu präparierenden Eingabestring kann der Angreifer eine Anzahl von No-Operation Befehlen (NOP) einstreuen und die Rückprungadresse mit der Adresse eines dieser NOP-Befehle überschreiben. Da Prozessoren NOP-Befehle einfach ignorieren und überspringen, wird automatisch der nächste Befehl angesprungen bis der eingeschleuste Code erreicht ist (vgl. Abbildung 2.3)

NOPs

komplexe
Funktionalität

Das nächste Problem, das ein Angreifer nun haben könnte, ist der relativ begrenzte Speicherbereich, der ihm mit dem Stack als Speicherort für seinen eingeschleusten Code zur Verfügung steht. Dies ist aber nur ein kleines

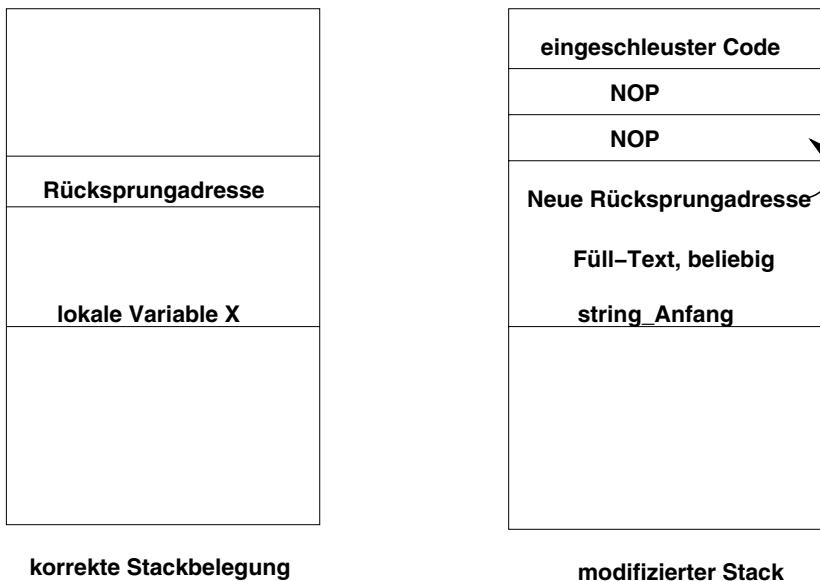


Abbildung 2.3: Einschleusen von Code mit NOPs-Befehlen

Hindernis, da zum einen mit hoch optimiertem Assemblercode komplexe Programme erstellt werden können, die nur einen geringen Speicherbedarf besitzen. Zum anderen stehen dem Angreifer viele Funktionen direkt zur Verfügung, ohne dass er sie selber einschleusen muss. So liegen in Windows-Systemen viele Bibliotheken (DLLs) bereits im Speicher. Ferner stehen dem Angreifer alle diejenigen Funktionen des Windows API zur Verfügung, die bereits zu dem attackierten Programm hinzu gebunden sind. Befindet sich unter diesen gebundenen Funktionen beispielsweise auch die Funktion `LoadLibrary`, so können durch den Exploit sogar beliebige Funktionen dynamisch nachgeladen werden. Da Bibliotheken häufig an gleichen Stellen im Speicher abgelegt werden, kann ein Angreifer diese direkt aufrufen, um beispielsweise TCP/IP-Verbindungen aufzubauen oder Befehle einer Shell auszuführen.

2.2.3 Gegenmaßnahmen

Aus dem Besprochenen wird klar, wie man sich als Software-Entwickler vor Buffer-Overflow-Angriffen schützen kann. Durch sorgfältige Programmierung (u.a. [175]), bei der insbesondere alle Eingabewerte geprüft und Bereichsgrenzen kontrolliert werden, kann man verhindern, Ziel von gezielten Buffer-Overflow-Angriffen zu werden. Wie bereits eingangs erwähnt, treten besonders in Programmen der Programmiersprachen C und C++ Pufferüberlaufprobleme auf. Durch den Übergang auf Programmiersprachen mit einer höheren Programmiersicherheit wie Java, könnten die entspre-

sichere Programmierung

Wrapper

Compiler-
unterstützung

chenden Probleme überwunden werden. So überprüft beispielsweise das Java-Laufzeitsystem die Einhaltung von Bereichsgrenzen automatisch. Aber auch im C bzw. C++ Umfeld gibt es Möglichkeiten, die Programmiersicherheit zu erhöhen. So kann man beispielsweise anstelle der bereits angesprochenen unsicheren Funktion `strcpy()` die Funktion `strncpy` verwenden, die es ermöglicht, die Zahl der zu schreibenden Zeichen zu begrenzen⁴. Da die Funktion `strcpy()` aber nur eine von vielen derartigen problematischen Bibliotheksfunktionen ist, sollte man eine spezielle Bibliothek verwenden, durch die sichere Bibliotheksfunktionen zur Verfügung gestellt werden. Unter Linux leistet dies beispielsweise die Bibliothek `Libsafe`⁵, mit der Standard-C Bibliotheksaufrufe gekapselt (engl. *to wrap*) und durch die gesicherte Version ersetzt werden.

Hat man den Quellcode von Programmen zur Verfügung, so kann man diesen unter Zuhilfenahme von speziellen Tools bearbeiten, um Überlaufangriffe zu erkennen. Ein Beispiel hierfür ist das `StackGuard Tool`⁶ für UNIX bzw. `StackShield` für Linux, das im Prinzip einen Patch für den GNU C-Compiler bereitstellt. Der derart modifizierte Compiler sichert bei jedem Funktionsaufruf die Returnadresse und korrigiert sie bei Bedarf. Der `Stack-Smashing-Protector`, der im GNU C-Compiler seit der Version 4.1 verfügbar ist, fügt ein spezielles Kontrollzeichen, das so genannte `Canary`⁷, direkt hinter die Rücksprungadresse auf den Stack ein. Ferner erzeugt er Code, der vor dem Rücksprung prüft, ob das Kontrollzeichen verändert wurde. Wird eine solche Änderung erkannt, so wird automatisch eine Warnmeldung in die Log-Datei des Systems geschrieben und das Programm abgebrochen. Ein analoges Konzept wurde mit den `Stack Cookies` von Microsoft u.a. in den Windows Server 2003 und Vista integriert.

Natürlich sind solche Maßnahmen unbefriedigend, da damit ja nicht die Ursache des Problems behoben, sondern nur die Auswirkungen begrenzt werden. Da die Absicherung von Systemen gegen Angriffe von außen ein ständiger Wechsel zwischen Aktionen seitens der Angreifer und Reaktionen seitens der Software-Entwickler und Systemadministratoren bedeutet, ist klar, dass die skizzierten Compilermodifikationen keinen nachhaltigen Effekt haben werden. Angreifer werden ihre Angriffe daraufhin verfeinern und dafür sorgen, dass das eingefügte Kontrollzeichen trotz Manipulation erhalten bleibt.

⁴ Microsoft bietet `strncpy_s` anstelle von `strncpy`.

⁵ <http://www.research.avayalabs.com/gcm/usa/en-us/initiatives/all/nsr.htm>

⁶ <http://immunix.org>

⁷ In Anlehnung an das frühere Vorgehen in Bergwerken, wo man einen Kanarienvogel unter Tage mitgenommen hat. Sobald dieser aufhörte zu singen bzw. gar starb war klar, dass ein bedrohlicher Sauerstoffmangel herrschte.

Der gerade beschriebene Ansatz setzt voraus, dass die abzusichernden Programme im Quellcode vorliegen, so dass sie mit den entsprechend modifizierten Compilern erneut übersetzt werden können. Eine Neu-Übersetzung der Software ist aber häufig gar nicht möglich, wie zum Beispiel bei proprietären Betriebssystemen oder bei kommerzieller Anwendungssoftware.

propriétärer Code

Mit den Techniken ASLR (Address Space Layout Randomization) und DEP (Data Execution Prevention) stehen in herkömmlichen Betriebssystemen Konzepte zur Verfügung, um den Schaden von erfolgreich durchgeföhrten Buffer-Overflows zu begrenzen bzw. Angriffe prinzipiell zu erschweren. Mit DEP werden Speicherbereiche als nicht-ausführbar gekennzeichnet, so dass kein Code, insbesondere kein Schadcode in einem solchen Datensegment ausgeführt werden kann. Ein entsprechender Versuch, Code auszuführen, löst eine Speicherschutzverletzung aus.

ASLR, DEP

Die ASLR-Technik erschwert Angriffe, die Kenntnisse über das Speicherlayout nutzen, um den Speicher zu manipulieren. Beispiele für solche Angriffe sind neben den beschriebenen Buffer-Overflows auch Heap Overflows bzw. auch Underflows, Format-String Verwundbarkeiten, oder auch Array Index Overflows. Die randomisierte Speicherung von ausführbarem Binärkode, von Dynamic Link Libraries (DLLs), Text-, Heap- und teilweise auch Stack-Segmenten mittels der ASLR-Technik (vgl. u.a. [162]) verschleiert deren Speicheradresse, die für die genannten Klassen von Angriffen oder auch für Angriffe unter Nutzung von ROP (Return-oriented Programming) notwendig ist, um die gespeicherten Module aufrufen zu können. Die ASLR Technik wurde als erstes im Betriebssystem OpenBSD verwendet. Sie wird seit Windows Vista auch mit Windows-Betriebssystemen angeboten und auch unter MAC-OS, Linux mit PaX ASLR und auch iOS oder auch Android sind ASLR-Varianten im Einsatz.

Beispielsweise kann unter Windows 7 für eine ausführbare Binärdatei (.exe) oder auch eine DLL (.dll) durch das Setzen eines speziellen Bits in dem PE Header des Executables⁸ die Verwendung der ASLR Technik angefordert werden. Für das zu ladende Executable wird in diesem Fall ein spezieller Offset-Wert berechnet, um dynamisch, aus Sicht eines Angreifers randomisiert, die Speicherposition zu bestimmen. Der Offset ist ein Wert aus dem Wertebereich von 1 bis 256 und ist an 64 KB ausgerichtet (aligned). Wird ein Programm ausgeführt, das ASLR Bit gesetzt hat, wird auch das Speicherlayout des Prozesses randomisiert, indem seine Stack- und Heap-Segmente randomisiert angelegt werden und auch der initiale Stack-Pointer durch einen randomisierten Wert aus dem Wertebereich 1 bis 16384 dekrementiert wird. Das Speicherlayout der Stack- und Heap-Segmente wird bei jeder Programmausführung erneut randomisiert festgelegt, während das

ASLR bei
Windows

⁸ PE (Portable Executable) beschreibt ein Binärformat ausführbarer Programme.

Layout des Code- und Datensegments sich nur bei einem Re-Boot ändert. Der Offset von gemeinsam genutzten DLLs wird über eine systemglobale Variable zum Bootzeitpunkt ermittelt und die DLL wird in den gemeinsam genutzten Speicherbereich in den Bereichen zwischen 0x50000000 und 0x78000000 abgelegt. Die DLLs werden zudem in randomisierter Reihenfolge geladen.

Trusted OS

Speziell abgesicherte Varianten von Standardbetriebssystemen, die als Trusted Operating Systems vertrieben werden (z.B. Trusted Solaris⁹, Trusted HP-UX), führen so genannte Compartments ein, die Systembereiche gegeneinander abschotten, so dass sich die Auswirkungen von Buffer-Overflow-Angriffen auf ein Compartment beschränken lassen.

Abschließend sei noch einmal darauf hingewiesen, dass die Buffer-Overflow-Verwundbarkeit sich nicht auf den Stack beschränkt, sondern Heap-Bereiche in gleicher Weise betroffen sind. Auch beschränken sich die heutigen Angriffe bei Weitem nicht nur auf die gängigen Operationen zum Kopieren von Zeichenketten, sondern nutzen eine Vielzahl weiterer Angriffsflächen aus. Wir haben bereits auf die Loop-Probleme, Integer-Overflows oder auch die Unicode Probleme hingewiesen. Alle diese Schwachstellen röhren von Programmierfehlern her, die man zum Teil frühzeitig durch statische Programmanalysen und durch eine sorgfältige Programmierung in den Griff bekommen könnte. Auf diesem Gebiet sind zurzeit eine Reihe von Tool-Entwicklungen in Arbeit, z.B. auch in den Forschungsbereichen von Microsoft, so dass die Hoffnung besteht, dass Buffer-Overflows in Zukunft nicht mehr länger die „Schwachstellen des Jahrzehnts“ sind, wie sie noch in den 90er Jahren von Bill Gates bezeichnet wurden.

2.3 Computerviren

Ein Virus (deutsch *giftiger Saft*) bezeichnet in der Biologie einen Mikroorganismus, der auf eine lebende Wirtszelle angewiesen ist, keinen eigenen Stoffwechsel besitzt und fähig ist, sich zu reproduzieren. Diese Eigenschaften sind direkt auf Computerviren übertragbar.

2.3.1 Eigenschaften

Der Begriff des Computervirus wurde zum ersten Mal 1985 durch Arbeiten von F. Cohen (u.a. [43]) der breiten Öffentlichkeit bekannt.

⁹ Trusted Solaris wurde 2006 als eigenständiges Produkt eingestellt.

Definition 2.1 (Computervirus)

Ein Computervirus ist eine Befehlsfolge, die ein Wortsprogramm zur Ausführung benötigt. Viren sind zur Reproduktion fähig. Dazu wird bei der Ausführung des Virus eine Kopie (Reproduktion) oder eine modifizierte Version des Virus in einen Speicherbereich, der diese Befehlssequenz noch nicht enthält, geschrieben (Infektion). Zusätzlich zur Fähigkeit zur Reproduktion enthalten Viren in der Regel einen Schadensteil. Dieser kann unbedingt oder bedingt durch einen Auslöser aktiviert werden.

Virus

□

Mit den Festlegungen von Definition 2.1 ist klar, dass ein Computervirus kein selbständig ablauffähiges Programm ist. Falls ein Virus sich nicht identisch reproduziert, also keine identische Kopie erzeugt, sondern die den Virus definierende Befehlssequenz modifiziert, so spricht man von einem mutierenden Virus.

mutierend

Reproduktionsfähigkeit

Zur Codierung der Befehlsfolge eines Virus verwendet man Maschinen-sprachen ebenso wie Kommandosprachen, Skriptsprachen oder auch Hochsprachen. Potentielle Speicherbereiche für Viren sind der Code ausführbarer Programme von Benutzern, Bereiche des Betriebssystems oder auch Sekto-ren eines Hintergrundspeichermediums. Der allgemeine Aufbau eines Virus ist in Abbildung 2.4 skizziert. Ein Virus besteht aus einer Viren-Kennung sowie optional aus einem Infektions-, einem Schadens- und einem Sprung-teil. Durch die Ausführung des Infektionsteils kopiert sich ein Virus in einen Speicherbereich. Handelt es sich dabei um den in einer Datei gespeicherten Code eines ausführbaren Programms, so werden i.d.R. die Strukturinfor-mationen des infizierten Programms durch die Angabe der neuen Dateilänge sowie durch die Veränderung der Einsprungadresse angeglichen. Die neue Einsprungadresse entspricht der Anfangsadresse des Virus-Codes. Soll nach der Ausführung des Virus doch noch das Programm mit der ursprünglichen Funktionalität ausgeführt werden, so enthält der Virus-Code im optionalen Sprungrteil eine Rücksprungadresse, die die Einsprungadresse in das ursprüngliche Programm ist.

Struktur

Anhand einer speziellen Viren-Kennung (z.B. des Wertes 4711 in der Ab-bildung 2.4), kann ein Virus in seiner Infektionsphase erkennen, ob ein Programm schon vom selben Virus befallen ist. Dies verhindert das wieder-holte Infizieren von Programmen. Diese Viren-Kennung (oder eine andere typische Zeichenfolge) dient im Gegenzug Virenerkennungsprogrammen, den Viren-Scannern, zur Aufdeckung von infizierten Programmen.

Kennung

Die Ausführung des Schadensteils kann von Randbedingungen abhängig gemacht werden. Im Beispiel von Abbildung 2.4 wird auf ein spezielles Da-

Bedingung

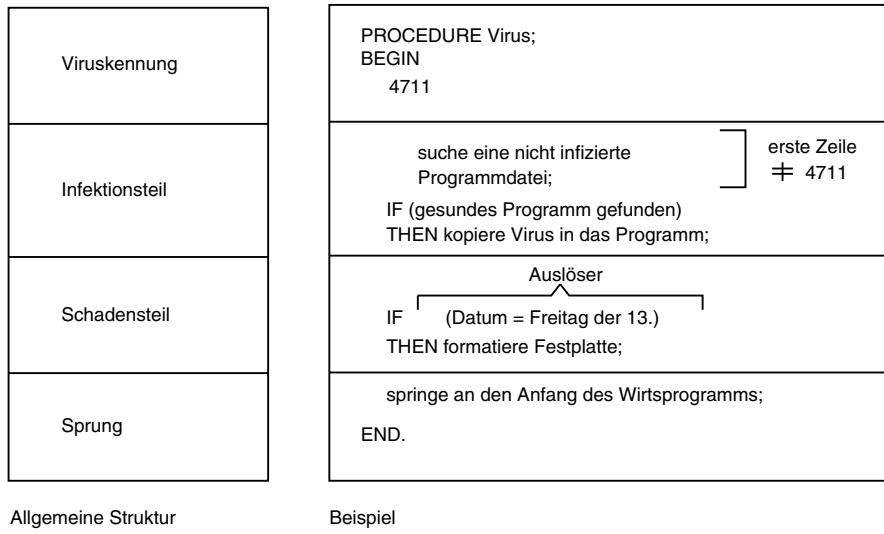


Abbildung 2.4: Allgemeiner Aufbau eines Virus und Beispiel

tum, nämlich Freitag den 13ten, gewartet. Ist zum Zeitpunkt der Ausführung des Wirtsprogramms die Bedingung erfüllt, so wird die angegebene Schadensfunktion wie die Neuformatierung der Festplatte ausgeführt. Soll nach der Abarbeitung der Befehlssequenz des Virus das ursprüngliche Programm ausgeführt werden, so wird ein Sprungbefehl eingefügt, mit dem an die Anfangsadresse des Programms gesprungen wird.

Durch Viren treten in erster Linie Bedrohungen der Integrität und Vertraulichkeit auf.

2.3.2 Viren-Typen

Erste Generation

Viren-Verbreitung

Isolierung

Mit dem Einzug des Personal Computers (PCs) in den 80er Jahren in Behörden, Firmen und in Privathaushalte erfolgte eine Abkehr von zentral, meist noch manuell administrierten Großrechenanlagen (engl. *mainframe*) hin zu einer dezentralen Verwaltung isoliert betriebener PCs durch deren Benutzer. Das heißt, die entsprechenden Benutzer übernahmen Aufgaben der Systemadministration und waren berechtigt, Software auf ihren PCs zu installieren. Zusammen mit der steigenden Anzahl gemeinsam genutzter Programme, insbesondere Spielprogrammen und Shareware, war damit der Boden zur Verbreitung von Viren gegeben. Die Viren der ersten Generation verbreiteten sich in erster Linie über das Kopieren von Daten von Diskette zu Diskette sowie über die meist noch manuelle Installation von Programmen auf einzelnen PCs.

Durch ein Nachbilden der ursprünglichen, zentralistischen Mainframe-Administration kann jedoch in solchen Umgebungen die Verbreitung von

Viren relativ einfach und wirkungsvoll eingegrenzt werden. Dazu ist ein zentraler Server-Rechner so zu konfigurieren, dass jedes neue Softwareprodukt zunächst auf diesem Rechner installiert und getestet wird. Der Server stellt somit eine Art „Quarantänestation“ für Software dar. Das systematische Testen und Kontrollieren von Software muss jedoch einem ausgewählten, vertrauenswürdigen Personenkreis vorbehalten bleiben, damit die Kontrolle nicht unterlaufen wird bzw. nicht durch eine unsachgemäße Behandlung Sicherheitslücken bestehen bleiben. Bemerkenswerter Weise gehört das Einrichten von Quarantäne-Netzen und Quarantäne-Rechnern heute wieder zu den häufig verwendeten Schutz-Maßnahmen in Unternehmen, um in einer dezentralen IT-Landschaft über zentral administrierte Komponenten neue Software zunächst in einem beherrschten und kontrollierbaren Umfeld zu evaluieren, bevor sie auf Rechner im Unternehmen verteilt werden.

Die häufigsten Viren, die in dieser Zeit auftraten, waren die so genannten Programm- und die Bootsektor-Viren.

Programm-Viren

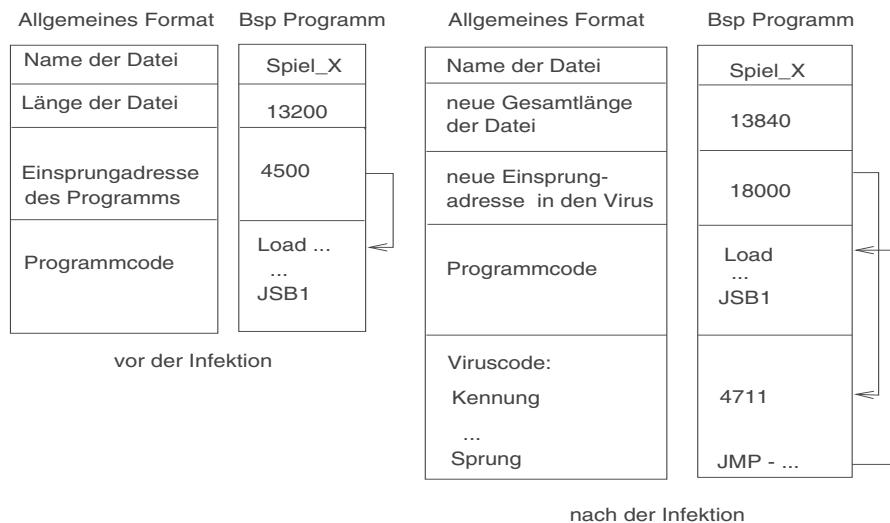
Ein Programm-Virus (auch bekannt als Link-Virus) kopiert sich in eine ausführbare Datei. Er wird durch den Aufruf des Programms verbreitet, da mit der Programmausführung auch der Virus-Code ausgeführt wird. In Abbildung 2.5 ist die Struktur eines infizierten Programms skizziert. Nach der Infektion liegt ein ausführfähiges Programm vor, das beim Aufruf zunächst den Virus und dann erst das eigentliche Programm ausführt. Zur Verschleierung des Virus wurden auch die Strukturdaten der infizierten Datei verändert.

Link-Virus

Boot-Viren

Boot- oder Bootsektor-Viren befallen die Bereiche einer Diskette oder einer Festplatte, deren Daten beim Hochfahren des Rechners, also beim Systemstart, gelesen und in den Hauptspeicher geladen werden. Beim Booten (vgl. auch Seite 594) eines Rechners initialisiert sich zunächst die CPU und startet einen Code zur Überprüfung der Hardware. Dieser Code befindet sich im ROM BIOS (Basic Input/Output System) auf dem Motherboard. Durch die Ausführung des BIOS-Codes werden im nächsten Schritt die Befehle geladen, die im Bootsektor der Diskette bzw. Festplatte abgelegt sind. Der Bootsektor enthält das Ladeprogramm, durch dessen Ausführung das Betriebssystem in den Hauptspeicher geladen und ausgeführt wird. Ein Boot-Virus wird meist vor dem Bootsektorprogramm in den Speicher geschrieben, so dass bei jedem Hochfahren des Systems zunächst der Virus ausgeführt wird. Bootsektor-Viren sind resident im Hauptspeicher geladen. Die residente Speicherung von Daten bedeutet, dass diese nicht auf den Hintergrundspeicher ausgelagert werden, sondern während der gesamten

Boot-Virus

**Abbildung 2.5:** Infiziertes Programm

Arbeitsphase des Systems im Hauptspeicher eingelagert sind. Abbildung 2.6 skizziert einen infizierten Bootsektor. Ein Boot-Virus kann beispielsweise die Interrupt-Handler-Tabelle verändern, die u.a. die Anfangsadresse der Treiberoutinen enthält, die bei einem Gerät-Interrupt auszuführen sind. Konnte beispielsweise anstelle der Anfangsadresse des Treiber-Codes für die Maus der Anfangscode des Virus platziert werden, würde das bedeuten, dass bei einem durch die Maus ausgelösten Gerät-Interrupt stets der Virus-Code anstelle des Handlers für die Maus gestartet würde. Obwohl sie großen Schaden verursachen können, spielen Boot-Viren heutzutage keine nennenswerte Rolle mehr, da sich die Verbreitungswege von Viren und deren Funktionsweise geändert haben.

Virus Code	Lade Virus (resident)
	Virus-Kennung ...
Bootprogramm	Lade Betriebssystem
	Lade Treiber
	Lade Konfigurationsdaten
	...

Abbildung 2.6: Infizierter Bootsektor

Viren der zweiten Generation

Durch Übergang zu vernetzten, offenen Systemen eröffnet sich eine Vielzahl von Kanälen, über die fremder Code auf den lokalen Rechner gelangen kann. Beispiele für solche in der Praxis häufig anzutreffende Kanäle sind E-Mails mit Anhang (engl. *attachment*), Java-Applets, elektronische Dokumente oder auch Bild-Dateien. Auch die Verbesserung heutiger Software-Werkzeuge trägt erheblich zur Verbreitung von Viren bei. Diese zunächst wenig einleuchtende These lässt sich jedoch einfach mit der zunehmenden Automatisierung von Vorgängen bei der Informationsverarbeitung begründen. Wesentliches Ziel der entwickelten Werkzeuge ist es nämlich, möglichst viele Aktivitäten automatisch und transparent für den Benutzer durchzuführen. Diese eigentlich höchst wünschenswerte Eigenschaft hat jedoch für den Virenbereich fatale Konsequenzen. Unter Nutzung von Werkzeugen wie Postscript- und PDF-Interpretern, Textverarbeitungsprogrammen (u.a. Word-Prozessor) oder MIME-Interpretern werden Befehle, die in fremden Dokumenten eingebettet sein können, automatisch ausgeführt. Auf diese Weise werden dann die in den Befehlen enthaltenen Viren transparent für den Benutzer, also ohne dessen explizites Eingreifen, in Umlauf gebracht.

Werkzeuge

Die Verbreitung von Viren über Rechnernetze hinweg hat darüber hinaus auch zu einer Veränderung der Funktionalität von Viren geführt. Während die Viren der ersten Generation vordringlich destruktiv ausgerichtet waren, findet man unter den neueren Viren zunehmend solche, die gezielt einen weiteren Angriff vorbereiten, indem sie sicherheitsrelevante Informationen des Zielrechners sammeln (u.a. abgelegte Passworte), oder indem sie gezielt Konfigurations- und Systemdateien modifizieren. Mit dem Wandel der Schadfunktionen von Viren und deren geänderten Verbreitungswegen verschwimmen auch zunehmend die konzeptuellen Unterschiede zwischen Viren und Würmern (vgl. Abschnitt 2.4). Häufig werden die Begriffe Virus und Wurm deshalb synonym verwendet.

Funktionalität

Makro- oder Daten-Viren

Makros sind in der Regel kleine Code-Teile, die in Daten-Dokumenten eingebettet werden. Solche Makros können beispielsweise bei der Erstellung von Word-Dokumenten oder Tabellenkalkulationen mittels Excel einem Daten-Dokument hinzugefügt werden. Damit erhalten diese Dateien einen ausführbaren Teil, so dass ein Dokument das Programm, durch das sie erstellt wurden, wie z.B. den Word-Prozessor, über die Ausführung eines solchen Makros kontrollieren und das Verhalten des Programms an die spezifischen Anforderungen des Dokuments anpassen kann. Beispielsweise kann man über ein Makro, das einem Word-Dokument hinzugefügt wird, dafür sorgen, dass bei jedem Abspeichern des Dokuments automatisch ein

Makros

Programm zur Prüfung der Rechtschreibung aufgerufen wird. Makros enthalten Steuerbefehle, die in einer Makrosprache wie zum Beispiel Microsofts Visual Basic for Applications (VBA), formuliert sind und interpretativ ausgeführt werden. Die Erweiterung von reinen Daten-Dateien zu Objekten mit ausführbaren Bestandteilen stellt einen gravierenden Schritt für die Virenproblematik dar. Nun sind auch Daten-Dateien, die bislang nur gelesen und nicht ausgeführt wurden, potentielle Worte für Viren. Dadurch ergibt sich ein sehr großes Potential neuer Bedrohungen. Vom Standpunkt eines Benutzers aus betrachtet reicht jetzt bereits ein lesender Zugriff auf ein Dokument aus, z.B. das Öffnen eines Word-Dokuments, um eine Virus-Verbreitung zu initiieren. Dass sich hinter dem Lese-Zugriff die Ausführung von Makro-Kommandos und damit das Starten eines Virus verbergen kann, ist für den Benutzer nicht unbedingt ersichtlich.

Makro-Virus

Seit 1995 sind die ersten Makro-Viren bekannt. Sie stellen bereits heute die häufigste Ursache für einen Virenangriff dar. Die ersten Vertreter dieser Viren-Klasse traten vorzugsweise in Dateien der Programme Word für Windows (WinWord) sowie in Excel-Programmen auf. Makro-Viren sind in der Regel unabhängig von der verwendeten Rechnerplattform, da sie interpretiert ausgeführt und nicht in Maschinencode übersetzt werden. Diese Plattformunabhängigkeit von Makro-Viren stellt eine weitere gravierende Verschärfung der Viren-Problematik dar, da auf diese Weise die „natürlichen“ Barrieren für die Verbreitung von Viren in einer heterogenen Umwelt entfallen. Die WinWord-Viren haben sich beispielsweise auf PCs mit unterschiedlichen Betriebssystemen u.a. Windows 3.xx, Windows NT oder Windows 95 verbreitet.

Attachment

Daten-Viren treten häufig auch in Postscript-Dateien und in Anhängen (engl. *attachment*) von MIME (Multipurpose Internet Mail Extension) Mails auf. MIME ist eine Erweiterung des SMTP-Protokolls zur Nachrichtenkommunikation, das neben der Übertragung einfacher ASCII-Dateien auch die Übertragung von Grafiken, Sprache etc. ermöglicht. Aus Sicherheitssicht problematisch ist, dass MIME-Nachrichten Befehle beinhalten können, die beim Empfänger der Nachricht ausgeführt werden. So ist es zum Beispiel möglich, beim Empfänger einen Datei-Transfer mittels FTP zu initiieren.

Postscript

Für Viren, die sich in Postscript-Dateien (ps-Dateien) verstecken, gilt ganz Analoges. Postscript ist eine Beschreibungssprache zur Darstellung von Grafikdaten und formatierten Texten. Postscript-Dateien (.ps) werden von einem Interpreter ausgeführt, der bei Postscript-fähigen Druckern direkt in den Drucker integriert ist oder als Programm zur Verfügung stehen muss, um die Darstellung von Postscript-Dateien auf dem Bildschirm zu ermöglichen (z.B. mittels *ghostview*). Die Postscript-Sprache enthält einige Befehle, die durch den Interpreter ausgeführt und für Angriffe missbraucht werden kön-

nen. Dazu gehören hauptsächlich Befehle zum Löschen oder Umbenennen von Dateien oder zur Erzeugung einer Kommunikationsverbindung, über die Daten ausgetauscht werden können.

Varianten der Makro-Viren sind Viren, die in Dateien eingebettet werden, die eigentlich keinen ausführbaren Code enthalten, wie beispielsweise .gif, .wmf oder .ani-Dateien¹⁰. Die entsprechenden Dateien werden in der Regel über Webseiten oder e-Mails verbreitet. Eine Schwachstelle in den Routinen zur Verarbeitung von Dateien zur Definition des Cursors (.cur), animierter Cursors (.ani) und Icons (.ico) kann von einem Angreifer dazu ausgenutzt werden, beliebigen Programmcode einzuschleusen und mit System-Privilegien auf dem Opfer-Rechner auszuführen. Die Schwachstelle besteht darin, dass die Routine zum Laden der Cursor keine Bereichsprüfung durchführt, sondern der Längenangabe, die die Größe des Headers angibt, vertraut. Die Routinen werden z.B. vom Internet Explorer oder von Outlook aufgerufen, wenn sie .ani-Dateien in einer Webseite oder einer HTML e-Mail empfangen. Um die Schwachstelle erfolgreich auszunutzen, ist es also erforderlich, dass das Opfer eine entsprechende Datei auf einer Webseite ansieht bzw. herunter lädt oder eine entsprechende E-Mail-Nachricht öffnet.

ani-Viren

Beispiel 2.1 (Sobig Virus)

An dem Sobig-Virus, der sich Anfang 2003 rasant ausbreitete, lässt sich die Funktionsweise klassischer Viren sehr gut verdeutlichen. Der Virus nutzte auf schon fast klassische Weise Social Engineering Vorgehen aus, um sich via E-Mail-Attachments zu verbreiten. In der Subjekt-Zeile der Mail traten unverfängliche, die Neugier des Empfängers anregende Bezeichnungen auf, wie *Re: Sample*, *Re: Document* oder *Re: Movies*. Die eigentliche Mail enthielt eine Nachricht der Art *Please see the attached file for Details*, die den Empfänger dazu animieren sollte, das beigelegte Attachment, das eine mit einem Virus infizierte pif-Datei (z.B. Sample.pif) enthielt, zu öffnen. Durch einen Doppelklick auf den Anhang wurde dessen Infizierungs- und Schadfunktion ausgelöst.

Sobig

Der aktivierte Virus führte seinen Infektionsteil aus, der zunächst eine Kopie von sich selbst unter dem Namen *Winmgm32.exe* im Windows-Verzeichnis ablegte und folgende zwei Einträge (Registry-Keys) in der Windows-Registry eintrug, so dass der Virus beim Booten automatisch gestartet wird:

Social Engineering

Infektionsteil

```
HKEY_CURRENT_USER\Software\Microsoft\Windows  
\CurrentVersion\Run  
"WindowsMGM"=C:\WINDOWS\winmgm32.exe
```

¹⁰ Animated Cursor Handling

```
HKEY_LOCAL_MACHINE\Software\Microsoft\Windows  
\CurrentVersion\Run  
"WindowsMGM" = C:\WINDOWS\winmgm32.exe
```

Ferner wurde anhand der Netzwerkfreigabe aktiv nach möglichen Opfersystemen gesucht, um den Virus auf diese Systeme zu kopieren. Durchsucht wurden folgende Verzeichnisse:

```
\WINDOWS\ALL USERS\START MENU\PROGRAMS\STARTUP oder  
\DOCUMENTS AND SETTINGS\ALL USERS\START MENU\PROGRAMS  
\STARTUP
```

Schadenstein

Im Schadenstein führte der Virus verschiedene Teilschritte durch. In einem ersten Schritt verschickte er sich selber in einer E-Mail an alle E-Mailadressen, die u.a. in Adressbüchern des befallenen Rechners zu finden waren. Des Weiteren öffnete er eine jpeg-Datei, die übrigens zu allem Überfluss auch noch ein Pornobild enthielt, und startete damit eine weitere Schadfunktion, die versuchte, ein Trojanisches Pferd (vgl. Abschnitt 2.5) aus dem Netz zu laden. War dies erfolgreich, so wurde das Trojaner-Programm gestartet, das u.a. einen Key-logger (*key-logger.dll*) auf dem befallenen Rechner installierte, um die Passwort-Eingaben von Benutzern abzufangen. Weiterhin spähte das Trojaner-Programm Registry-Einträge aus, um Informationen über die System-Konfiguration des befallenen Rechners zu sammeln. Über eine TCP-Verbindung, die der Trojaner zu der Seite *geocities.com* in periodischen Abständen aufbaute, erhielt er eine vom Angreifer vorab präparierte URL, an die die gesammelten Informationen (u.a. IP-Adresse des befallenen Rechners, Benutzerkennung(en) und Passwort(e)) übergeben wurden. Außerdem versuchte der Trojaner den Port 1180 auf dem befallenen Rechner zu öffnen, wodurch dem Angreifer ein entfernter Zugriff auf den Rechner ermöglicht wurde.

Trojaner

Zusammenfassend ist festzuhalten, dass der Sobig-Virus viele klassische Charakteristika aufweist. Seine schnelle und große Verbreitung erfolgte über E-Mail-Attachments, die von sorglosen Benutzern gutgläubig geöffnet wurden (Social Engineering), da die Mail angeblich von einer bekannten E-Mail-Adresse stammte, eine unverdächtige Subjekt-Zeile aufwies und der Mail-Körper auf Details im Anhang verwies und damit die Neugierde des Empfängers weckte. Das Ablegen von Einträgen in der Windows-Registry, um ein automatisches Starten des Virus beim Booten zu erreichen, wie auch die Installation von Trojaner-Programmen auf den befallenen Rechnern, zählen zu den üblichen Schadensroutinen heutiger Viren.



2.3.3 Gegenmaßnahmen

Wir können die Menge der möglichen Gegenmaßnahmen grob in zwei Klasse einteilen, nämlich der präventiven Maßnahmen zur Virenabwehr der Werkzeuge zur Erkennung eines Virenbefalls.

Präventive Maßnahmen

Zu dieser Klasse von Abwehrmaßnahmen zählen wir das Beschränken von Schreibberechtigungen, das Verschlüsseln von Daten, die Verwendung digitaler Fingerabdrücke sowie administrative Maßnahmen zur Abwehr von Makro-Viren. Auf diese Bereiche wird im Folgenden kurz eingegangen.

Da heutzutage nahezu alle im Einsatz befindlichen Rechner an das Internet angebunden sind, ist jedes dieser Geräte potentiell gefährdet und erfordert den Einsatz von Überwachungswerkzeugen und Viren-Scannern. Die beschränkte Vergabe von Schreibrechten an Programme ist eine wirksame Abwehrmaßnahme gegen Programmviren, da dadurch das Modifizieren bestehender Programme erschwert und damit die Verbreitung von Viren eingedämmt wird. So benötigen zum Beispiel viele Spielprogramme höchstens Schreibzugriffe auf temporär anzulegende Dateien, so dass auch nur für das TMP-Verzeichnis entsprechende Berechtigungen erteilt werden müssen.

Rechte-
beschränkung

Durch die Verschlüsselung von gespeicherten Programmen kann man verhindern, dass ein Virus gezielt in der in Abbildung 2.5 skizzierten Weise in ein Programm integriert wird. Ein einfaches Überschreiben des verschlüsselten Codes durch einen Virus führt nach der Entschlüsselung zu nicht ausführbarem Code. Das Überschreiben zerstört zwar die Quelldatei, aber diese Vorgehensweise unterbindet die Viren-Verbreitung sowie die Ausführung von Schadensteilen von Viren. Die Verschlüsselung von Dateien wird zurzeit bereits durch eine Reihe von Produkten wie dem Encrypting File System (EFS) als Bestandteil von Windows (vgl. Abschnitt 13.2.5) unterstützt.

Verschlüsseln

Eine weitere Möglichkeit, durch präventive Maßnahmen eine Vireninfektion vor Ausführung eines Programms erkennen zu können, besteht darin, einen digitalen Fingerabdruck des Programms (vgl. Abschnitt 8.1) zu berechnen. Ein digitaler Fingerabdruck (engl. *digest*) ist ein Bitstring fester Länge, der das Programm eindeutig beschreibt. Diese Bitstrings sind schreibgeschützt zu verwalten, so dass vor der Ausführung eines Programms dessen Fingerabdruck zunächst erneut berechnet und mit dem ursprünglichen Wert verglichen werden kann. Da der Fingerabdruck einen Programmcode eindeutig beschreibt, führt die Veränderung des Codes nach der Integration von Virus-Code zu einem neuen Bitstring als Fingerabdruck des modifizierten Programms. Diese Abweichung zwischen dem ursprünglichen und dem aktuellen Fingerabdruck ist vor einer Programmausführung erkennbar. Auf diese Weise kann die Verbreitung des Virus unterbunden, der erstmalige Befall jedoch nicht verhindert werden. Zur Behebung eines Virenbefalls

Hashfunktion

ist eine gesicherte Version von einem Backup-Medium wiedereinzuspielen, wobei wiederum die digitalen Fingerabdrücke überprüft werden müssen, um sicherzustellen, dass keine infizierte Version eingespielt wird.

Quarantäne

Zur Abwehr von Makro-Viren ist zu verhindern, dass diese beim Lesen eines Dokumentes durch die zum Lesen verwendeten Werkzeuge (u.a. Word-Prozessor, Postscript-Interpreter) automatisch ausgeführt werden. Da eine automatische Meldung, ob ein zu ladendes Dokument Makros enthält, nicht immer erfolgt, sollte in Umgebungen, die keine automatisch ablaufenden Makros erfordern, die Ausführung solcher Makros unterbunden werden. Die Kontrolle von erweiterten E-Mails oder anderer über das Netz ausgetauschter Dokumente ist nicht immer möglich. Mails bzw. Dokumente von unbekannten Absendern sollten zunächst unter Nutzung von Viren-Scannern überprüft und im Zweifelsfall ungelesen gelöscht oder nur in einer isolierten Ausführungsumgebung gelesen werden. So stehen beispielsweise unter Windows zur automatischen Überprüfung von Dokumenten Suchprogramme zur Verfügung, die periodisch im Hintergrund ablaufen und somit die laufende Anwendung kaum behindern.

Werkzeuge zum Antivirenmanagement

Scanner

Die am häufigsten eingesetzten Werkzeuge gegen Viren sind die Viren-Scanner. Bekannte Viren besitzen eine Viren-Kennung oder enthalten spezifische Bytemuster bzw. Codesequenzen, woran sie erkannt werden können. Viren-Scanner verwalten umfangreiche Datenbanken mit Kennungen bekannter Viren sowie mit Regeln, die u.a. festlegen, welche Dateien bzw. Dateitypen (z.B. alle ausführbaren Programme, u.a. .exe oder .com-Dateien) auf Viren zu überprüfen sind. Bei einem Viren-Scan werden die gespeicherten Dateien auf das Vorhandensein von Viren-Kennungen untersucht und es werden die infizierten Objekte sowie die gefundenen Viren angezeigt. Auf diese Weise können natürlich nur bereits bekannte Viren erkannt werden. Mutierende Viren oder neue Viren schlüpfen in der Regel durch das Netz dieser Scanner.

Heuristik

Aus diesem Grund versuchen einige Erkennungsprogramme anhand von Heuristiken auch unbekannte Viren bzw. Abwandlungen bekannter Viren zu entdecken. Die Heuristiken suchen nach verdächtigen Codesequenzen in Programmen, die auf Virenaktivitäten hindeuten, wie zum Beispiel Befehle, die dem Suchen nach neuen Opfern und dem Kopieren/Replizieren des Virus entsprechen. Die anhaltenden Probleme mit Viren-verseuchten Rechnern verdeutlichen jedoch, dass diese Heuristiken unzureichend sind, da die Anzahl der möglichen Veränderungen existierender Viren sehr groß ist und durch Heuristiken unmöglich alle Mutationen bekannter Viren abgedeckt werden können. In Anbetracht der rasanten Entwicklung und Verbreitung

neuer Viren ist deshalb auf jeden Fall eine regelmäßige Aktualisierung der Datenbanken mit Viren-Kennungen unerlässlich.

Ein anderer Ansatz zur Bekämpfung von Viren wird durch Werkzeuge verfolgt, die eine Aktivitätskontrolle durchführen. Dazu werden Programme während ihrer Ausführung überwacht und auf Verhaltensweisen untersucht, die für einen Virenbefall typisch sind. Verdächtige Aktionen umfassen den wiederholten, modifizierenden Zugriff auf ausführbare Dateien, das Suchen nach ausführbaren Dateien oder Versuche, direkt auf externe Speichermedien zuzugreifen. Derartige Tools können als sehr einfache Varianten von Einbruchserkennungs-Systemen (engl. *intrusion detection system IDS*) aufgefasst werden.

Aktivitätskontrolle

Als dritte Klasse von Werkzeugen sind die Monitoring-Tools zu nennen, die Dateien (im Wesentlichen handelt es sich dabei um wichtige Systemdateien) überwachen und durchgeführte Modifikationen anhand geänderter Strukturinformationen oder anhand von Abweichungen bei berechneten digitalen Fingerabdrücken erkennen.

Monitoring

Auf Maßnahmen zur Abwehr von Viren wird auch in Zukunft ein Augenmerk bei der Sicherheit von IT-Systemen liegen müssen. Solange jedoch Softwarehersteller und Benutzer nicht in die Pflicht dafür genommen werden können, dass sie ihre Software mit mangelhaften Sicherheitsniveaus ausstatten bzw. ihre Systeme unsicher vorkonfigurieren und fahrlässig mit potentiellen Schadprogrammen umgehen, werden weltweite Virenvorfälle immer wieder auftreten. Erste Ansätze, unachtsamen Anwendern, von deren PCs sich Viren verbreitet haben, mit Klagen auf Schadensersatz wegen der für die Beseitigung der Schäden verursachten Kosten zu drohen, haben aber zurzeit wohl noch keine sehr große Aussicht auf Erfolg. Trotzdem erscheint das Vorgehen, über Fahrlässigkeitsklagen Haftungsansprüche durchzusetzen und Schaden sowie Verantwortung zu teilen, sehr sinnvoll, um in Zukunft von Unternehmen und Einzelpersonen eine größere Verantwortung gegenüber Dritten erwarten zu können. Eine allgemeine rechtliche Verpflichtung zur Vorsorge mit entsprechenden Haftungsansprüchen erscheint angesichts der großen Verbreitung von Viren durchaus möglich. Beim heutigen Kenntnisstand über die Virenproblematik muss es deshalb schon fast als Fahrlässigkeit angesehen werden, wenn auf einen Virenschutz verzichtet wird.

Haftung

2.4 Würmer

Im Gegensatz zu einem Virus ist ein Wurm ein eigenständiges, ablauffähiges Programm. Das bedeutet, dass ein Wurm kein Wirtsprogramm zur Ausführung benötigt.

Definition 2.2 (Wurm)**Wurm**

Ein Wurm ist ein ablaufähiges Programm mit der Fähigkeit zur Reproduktion. Ein Wurm-Programm besteht in der Regel aus mehreren Programmteilen, den Wurm-Segmenten. Die Vervielfältigung erfolgt selbstständig meist unter Kommunikation mit anderen Wurm-Segmenten.

**Verbreitung**

Die Verbreitung von Würmern erfolgt insbesondere über ein Netzwerk, indem sich der Wurm auf andere Rechner innerhalb des Netzes kopiert. Ausgangspunkt für einen Wurmangriff sind häufig Systemprozesse, die ständig rechenbereit sind oder in regelmäßigen Abständen aktiviert werden. Würmer bedienen sich dieser Prozesse, um beispielsweise über eine Buffer-Overflow-Attacke in das Opfersystem zu gelangen. So nutzte im Jahr 2001 der Code Red Wurm einen Pufferüberlauf im Microsoft Internet Information Service (IIS) aus, um sich zu verbreiten. Die ausführbaren Programmteile eines Wurms können den Quellcode eines Programms beinhalten, der auf dem zu befallenden Rechner übersetzt werden muss, sie können aber auch direkt in ausführbarer Maschinensprache oder in interpretierbarer Sprache, z.B. Shell-Kommandos, geschrieben sein. Bei einem Pufferüberlauf-Angriff auf einen Systemprozess werden diese Befehle dann sogar im privilegierten Modus des Betriebssystemkerns ausgeführt, d.h. der Angreifer erlangt unbeschränkte Zugriffsberechtigungen.

Bedrohungen

Durch Würmer treten ebenso wie durch Viren Bedrohungen der Integrität und Vertraulichkeit, aber auch Bedrohungen der Verfügbarkeit auf, die so genannten Denial-of-Service Angriffe. Würmer beanspruchen in der Regel viele Ressourcen, da sie bei ihrer Verbreitung über das Netzwerk häufig eine sehr hohe Netzlast erzeugen und durch einen möglichen Mehrfachbefall von Rechnern deren Speicherressourcen ausschöpfen können.

Die ersten Wurm-Programme wurden 1979 im Xerox Palo Alto Research Center als sinnvolle und nicht bedrohliche Anwendungen entwickelt, um eine verteilte Berechnung wie zum Beispiel das Komprimieren von Daten durchzuführen. Einer der ersten und bekanntesten Würmer mit einer bedrohlichen Funktionalität war der so genannte Internet-Wurm.

Beispiel 2.2 (Internet-Wurm)**Internet-Wurm**

Am Abend des 2. November 1988 wurde der Internet-Wurm (u.a. RFC1135) gestartet. Der Wurm verbreitete sich im Verlauf der Nacht über das Internet, und als er am nächsten Morgen gestoppt wurde, hatte er bereits 6000 Rechner befallen.

Der Wurm nutzte einige bekannte Fehler und Eigenheiten von Unix aus, die mittlerweile in den meisten Systemen behoben sein sollten. Zu den allgemeinen, auch in heutigen Systemen noch vorhandenen Schwachstellen, die indirekt zur Verbreitung des Wurms beitrugen, zählt, dass Dienstprogramme und ihre Hintergrundprozesse keinem bestimmten Benutzer zugewiesen werden, so dass sich ein Angreifer ihrer bedienen kann.

genutzte
Schwachstellen

Angriff

Das Ziel des Internet-Wurms war es, auf einem fremden Ziel-Rechner eine Shell zu starten, um damit das Wurmprogramm auf diesem Rechner zu laden, dort zu übersetzen und auszuführen. Um eine solche Shell zu starten, wurden drei verschiedene Angriffsversuche durchgeführt: (1) Mit einem „Brute force“ Angriff wurde versucht, Passworte auf dem Ziel-Rechner zu knacken (Remote Shell Angriff).

remote shell

(2) Falls der Ziel-Rechner eine bekannte Sicherheitslücke im `sendmail` Programm enthielt, wurde diese ausgenutzt, um ein Programm zu senden, das auf dem Zielrechner ausgeführt wurde, um eine Shell zu starten. Die Entwickler des `sendmail` Kommandos hatten zum Debuggen einen speziellen Debug-Modus eingeführt, durch den eine Nachricht zu einem Programm und nicht zum Mailer gesendet wurde. Falls der Ziel-Rechner es ermöglichte, diesen Modus einzustellen, wurde er vom Wurm durch das Versenden einer dediziert konstruierten Befehlsfolge ausgenutzt. Die Ausführung dieser Befehlssequenz bewirkte, dass auf dem Ziel-Rechner eine Shell gestartet wurde, mit der das Wurmprogramm übersetzt und ausgeführt wurde. Da mit dem Debug-Modus Superuser-Rechte verknüpft waren, konnte sogar eine Shell mit Root-Rechten erzeugt werden.

`sendmail`

(3) Als Drittes wurde versucht, einen Implementierungsfehler im `fingered` Programm auszunutzen. Mit diesem Programm kann man Informationen über Benutzer eines Systems erfragen. Das `fingered` Programm nutzt die Systemroutine `gets()`, die eine Eingabe in einen String kopiert. In alten Versionen dieses Programms fehlte eine Bereichsprüfung, so dass mit einer Buffer-Overflow Attacke der Puffer der `gets`-Routine durch einen zu langen Eingabestring zum Überlaufen gebracht werden konnte. Der Wurm nutzte diese Lücke aus, indem er gezielt einen langen Eingabestring erzeugte und Maschinenbefehle auf dem Systemkeller ablegte, durch deren Ausführung eine Shell auf dem befallenen Rechner gestartet wurde. Damit konnte der Angreifer auf dem fremden Rechner Kommandos ausführen.

`fingered`



Als Reaktion auf den Wurm gründete die U.S. Defense Advanced Research Projects Agency (DARPA) das erste Computer Emergency Response Team

CERT

(CERT), das die Aufgabe hat, sich mit Sicherheitsfragen rund um das Internet zu beschäftigen.

ILOVEYOU

Im Mai 2000 sorgte der ILOVEYOU-Wurm für einiges Aufsehen, da er sich in einer bis dahin nicht bekannten Geschwindigkeit verbreitete und zum Zusammenbruch der elektronischen Datenverarbeitung in vielen europäischen und amerikanischen Firmen und Behörden führte. Man schätzt den verursachten Schaden durch Datenverluste und Produktivitätsausfall weltweit auf über 8.76 Millionen US Dollar.

Beispiel 2.3 (ILOVEYOU)

Bei dem ILOVEYOU-Wurm handelte es sich um einen Wurm, der als .vbs-Datei (Visual Basic Script) über E-Mail Attachments verbreitet wurde, in deren Betreff-Zeile ILOVEYOU stand. Die Verbreitung des Wurms setzte eine Microsoft Windows-Umgebung voraus, da der Wurm sich der Daten von Windows Outlook bediente, um eine Mail mit dem Wurm als Attachment an die in der Adressdatei gespeicherten Mailadressen zu senden. Der Wurm zerstörte gezielt Dateien des lokalen Dateisystems, die u.a. JPEG-Bilder, MP2- oder MP3 Musik-Daten oder Videodateien enthielten. Schließlich durchsuchte der Wurm auch die lokale Festplatte nach Passworten und versuchte, eine Verbindung aufzubauen, um diese Daten an den Programmierer des Wurms zu übermitteln.



Nahezu wöchentlich treten neue Wurm-Varianten in Erscheinung, die häufig nach einem ähnlichen Angriffsschema ablaufen. Als Beispiel hierzu wird abschließend kurz auf den Blaster- bzw. Lovesan-Wurm eingegangen.

Beispiel 2.4 (Lovesan- bzw. Blaster-Wurm)

Lovesan

Im Sommer 2003 verbreitete sich der so genannte Lovesan-Wurm¹¹ äußerst rasant und richtete beträchtliche Schäden an. Das Besondere an diesem Wurm war, dass er nicht nur Server-Rechner sondern insbesondere auch Heim-PCs zum Angriffsziel hatte. Sein Angriffsvorgehen basiert jedoch wiederum auf Standardangriffstechniken, die im Folgenden kurz erläutert werden.

Angriffsziel

Das eigentliche Angriffsziel war der Server *windowsupdate.com*, der Patches für Windows-Systeme bereitstellt, die von Windows-Benutzern via Windows-Update auf den eigenen Rechner herunter geladen werden können. Der Blaster-Wurm verursachte einen Distributed Denial of Service (DDoS) Angriff auf diese Web-Site. Mit diesem Angriff wurden systematisch Opfer-Systeme vorbereitet, die nach dem Stichtag 16.08.2003 SYN-Anfragen an

¹¹ auch unter dem Namen Blaster-Wurm bekannt

den Port 80 des Update-Servers senden sollten. Es handelte sich hierbei um eine SYN-Flood Attacke mit gespooften Absenderinformationen, wie sie auf der Seite 122 beschrieben wird. Die gespooften Absenderadressen waren hier zufällig gewählte IP-Adressen eines Klasse B Netzes. Durch den DDoS-Angriff sollte der Update-Server so überlastet werden, dass die Benutzer keine Patches mehr herunterladen konnten. Um dies zu vermeiden hatte Microsoft, nach bekanntwerden des drohenden DDoS-Angriffs vorsorglich den DNS-Eintrag *windowsupdate.com* entfernen lassen.

Zur Angriffsvorbereitung nutzte der Wurm eine bekannte Schwachstelle aus, nämlich eine Buffer-Overflow Verwundbarkeit (vgl. Abschnitt 2.2) im DCOM RPC-Dienst an TCP-Port 135 unter dem Windows-Betriebssystem. Um Systeme zu finden, die einen derartigen Angriffspunkt aufweisen, scannete der Wurm entweder das lokale Klasse C-Subnetz oder wählte einen beliebigen IP-Adressbereich aus, um in diesem Adressbereich Systeme mit geöffnetem Port 135 (oder auch Port 139, 445 oder 593) zu identifizieren. An den Port 135 eines derart als mögliches Opfer identifizierten Rechners sendete er sodann ein präpariertes TCP-Paket. Ausgenutzt wurde dabei, dass TCP-Pakete, die an Port 135 empfangen werden, ungeprüft an den DCOM-RPC Dienst weitergeleitet wurden. Das präparierte Datenpaket verursachte einen Buffer-Overflow, wodurch ausführbarer Code auf das Opfersystem gelangte und gleichzeitig den RPC-Dienst zum Absturz brachte sowie einen Reboot des Systems verursachte.

Der gezielte Buffer-Overflow Angriff gewährte dem Angreifer Superuser-Rechte bei der Ausführung des eingeschleusten Codes. Diese wurden genutzt, um eine entfernte Shell (remote Shell) an TCP-Port 444 des Opfer-Rechners zu starten. Über diese war es dann in weiteren Schritten möglich, eine TFTP-Verbindung über den UDP-Port 69 zum Ausgangsrechner der Wurm-Attacke aufzubauen sowie die Datei *msblast.exe* in das Windows-Systemverzeichnis des Opferrechners herunter zu laden und zu starten. Mit der Ausführung von *msblast.exe* wurde ein Windows-Registry-Eintrag beispielsweise der folgenden Art

```
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows  
\CurrentVersion\Run  
windows auto update= msblast.exe I just want to say  
                                LOVE YOU SAN!! bill
```

erzeugt¹². Der Blaster-Code prüfte das aktuelle Datum des Opfersystems und versendete beim Erreichen bzw. Überschreiten des Datums 16.08.03 die oben bereits angesprochenen 40 Byte SYN-Pakete mit einer gefälschten Absender-IP-Adresse an den Rechner *windowsupdate.com*.



Buffer-Overflow

remote Shell

¹² Daher röhrt der Name Lovesan-Wurm.

Fazit

Der Lovesan- bzw. Blaster-Wurm ist ein typisches Beispiel eines Wurmes, der einen Programmierfehler in einem Betriebssystemdienst ausnutzt (Buffer-Overflow), um mit Superuser-Privilegien einen Kommandointerpreter (Shell) auf dem Opfersystem zu starten, TCP- und/oder UDP-Verbindungen aufzubauen und darüber beliebigen weiteren Code auf das Opfersystem herunter zuladen. In diesem speziellen Fall wurden die Opferrechner „nur“ für einen DDoS-Angriff instrumentalisiert, aber natürlich könnten sich auf diesem Weg genauso auch Angriffe auf die Integrität und Vertraulichkeit der Daten auf dem Opferrechner selber ergeben. Einmal mehr zeigt das Beispiel, dass eine sorglose Konfiguration mit z.B. zu vielen geöffneten Ports leicht auszunutzende Angriffsflächen bietet.

Das erfolgreiche Wirken von Würmern wie dem Blaster-Wurm wird durch Design- und Implementierungsfehler in Systemdiensten ermöglicht. Davon sind Open Source Betriebssysteme ebenso betroffen wie proprietäre Betriebssysteme, wie die Systeme der Windows-Familie. Letztere stehen aber stärker im Visier, da sie viel größere Verbreitung besitzen, so dass die Betriebssystem-Monokultur auch die Ausbreitung von Würmern fördert. Mangelnde Sicherheitsmaßnahmen in diesen Systemen eröffnet Programmen nahezu unbeschränkte Zugriffe auf Daten der lokalen Festplatte. Daneben verdeutlichen die rasante Verbreitung von Würmern und die sehr hohe Wiederholungsrate (Auftreten von Wurmvarianten) erneut die mangelnden Kenntnisse und das mangelnde Sicherheitsbewusstsein heutiger Benutzer.

Gegenmaßnahmen

Patch

Im Gegensatz zu Viren, die sich meist auf „legalem“ Weg verbreiten, versuchen Würmer Bugs und Lücken in System- und Anwendungssoftware auszunutzen. Da auf den einschlägigen WWW-Seiten (siehe u.a. <http://www.securityfocus.com>) und Mailinglisten sehr schnell aktuelle Informationen über ausgenutzte Sicherheitsschwachstellen veröffentlicht und auch Patches zum Beheben der Schwachstellen angeboten werden, sollte sich jeder zuständige Systemadministrator hierüber informieren und seine Systeme absichern. Die Lücken in den sendmail bzw. fingered Programmen waren beispielsweise schon lange bekannt, bevor der Internet-Wurm sie ausnutzte. Aktuelle Informationen über Lücken und Problembereiche werden regelmäßig von den CERTs veröffentlicht.

Vielfach lassen sich mögliche Einfallstore für Würmer bereits durch eine geeignete Konfigurierung mit einem eingeschränkten Diensteangebot (z.B. Schließen von nicht benötigten Ports) verkleinern oder schließen.

minimale Rechte

Durch eine restriktive Vergabe von Zugriffsberechtigungen, insbesondere auch von Leseberechtigungen, lässt sich ein unerlaubtes Akquirieren von Informationen und das Einbringen von fremdem Code beschränken. Beson-

ders Passwortdaten sind so zu schützen, dass nicht jeder Benutzer Leserecht darauf erhält. Generell sind differenzierte und restriktive Rechtefestlegungen und Kontrollen notwendig, insbesondere bei Zugriffen von entfernten Rechnern. Die fehlende differenzierte Zugriffskontrolle in den Windows Betriebssystemen (Windows 95, 98) hat ursächlich dazu beigetragen, dass sich der *ILOVEYOU*-Wurm in einer bis dahin noch nicht bekannten Geschwindigkeit ausbreiten konnte.

2.5 Trojanisches Pferd

Der Begriff des Trojanischen Pferdes geht zurück auf die Sage vom Kampf um die Stadt Troja, in der die Griechen nach 10-jährigem Kampf als Zeichen des Rückzuges den Trojanern ein großes, hölzernes Pferd zum Geschenk vor die Tore der belagerten Stadt stellten. Die siegestrunkenen Trojaner zogen das Pferd in das Innere der Stadtmauern, um sich an ihrem Geschenk zu erfreuen und um ihren vermeintlichen Sieg zu feiern. Im Verlauf der Nacht offenbarte das Pferd sein verstecktes Geheimnis; es verbarg griechische Soldaten in seinem Inneren, die die Tore der Stadt öffneten, so dass sie von den Griechen eingenommen werden konnte.

Sage

2.5.1 Eigenschaften

Das Trojanische Pferd in der Sage beschreibt die Charakteristika von Programmen, die als Trojanische Pferde bezeichnet werden, sehr genau: Es wird eine Funktionalität vorgetäuscht, die Vertrauen erweckt, die aber durch eine verborgene Schadens-Funktionalität ergänzt wird.

Definition 2.3 (Trojanisches Pferd)

Ein Trojanisches Pferd (auch Trojaner genannt) ist ein Programm, dessen implementierte Ist-Funktionalität nicht mit der angegebenen Soll-Funktionalität übereinstimmt. Es erfüllt zwar diese Soll-Funktionalität, besitzt jedoch eine darüber hinausgehende, beabsichtigte zusätzliche, verborgene Funktionalität.

Trojanisches Pferd



Unter die Definition 2.3 fallen somit keine Programme, die infolge von Programmierfehlern ein von der Soll-Funktionalität abweichendes Verhalten zeigen, sondern nur solche, für die diese Erweiterungen absichtlich integriert wurden. Ein Trojanisches Pferd besitzt verborgene Eigenschaften, um z.B. in ein System einzudringen, um Daten aufzuzeichnen oder zu manipulieren. Auf Benutzerebene wird dabei korrektes Verhalten vorgetäuscht. Trojanische Pferde können mit dem Programmstart aktiviert oder durch spezielle Aus-

logische Bombe

löser, wie beispielsweise das Eintreten eines bestimmten Datums, gestartet werden. Man spricht hierbei auch häufig von einer logischen Bombe.

Trojanische Pferde können in ganz unterschiedlichen Bereichen eines Systems auftreten. Beispiele dafür sind Editoren oder Textverarbeitungsprogramme, die die Inhalte edierter Dateien unbemerkt und unautorisiert kopieren. Weitere Beispiele sind manipulierte Datenbanken, durch die sensible Informationen zum Angreifer durchsickern können, oder auch manipulierte Betriebssystemfunktionen, durch die der Angreifer beispielsweise zusätzliche Zugriffsrechte erlangt.

Beispiel 2.5 (Trojanische Pferde)

Zinsberechnung

Eines der ersten bekannt gewordenen Trojanischen Pferde trat in einer Bank auf. Ein Angestellter hatte den Auftrag, ein Programm zu schreiben, das Zinsabrechnungen auf drei Stellen genau durchführt. Der Programmierer erfüllte diese Aufgabe und zusätzlich sorgte er dafür, dass Restbeträge, die beim Abrunden anfielen, auf sein Konto gutgeschrieben wurden. Innerhalb kürzester Zeit sammelte er dadurch einen großen Geldbetrag an.

CAD-Demo

Ein weiteres bekanntes Trojanisches Pferd, das 1992 für einiges Aufsehen gesorgt hatte, stammte von der Firma CadSoft, einer Firma, die CAD-Software entwickelt. Diese Firma verteilte eine Demoversion ihres Produkts. Bei der Ausführung dieses Demoprogramms wurde auch ein Formular erzeugt, das man zur Bestellung eines Handbuches verwenden konnte. Die zusätzliche, verborgene Funktionalität des Demoprogramms bestand darin, dass die Festplatte des Benutzer-PCs nach Programmen der Firma CadSoft durchsucht wurde und auf dem Bestellformular Art und Anzahl der gefundenen Programme codiert wurden. Da sich Benutzer, die im Besitz von Raubkopien der CAD-Software waren, bevorzugt das Handbuch zuschicken ließen, gelang es, viele widerrechtliche Benutzer zu entlarven. Anzumerken bleibt jedoch, dass das Ausschnüffeln der Festplatte von Benutzern mittels Trojanern rechtlich nicht zulässig ist und im vorliegenden Fall dann auch entsprechend geahndet wurde.



Trend

Heutige Trojanische Pferd-Programme bieten den Angreifern eine Vielzahl von Möglichkeiten, den befallenen Rechner zu kontrollieren, gespeicherte Daten auszuspähen, oder aber auch Tastatureingaben, wie zum Beispiel Passwörter, sowie Bildschirmausgaben aufzuzeichnen und über eine Netzwerkverbindung zu versenden. Häufig laden sie automatisch Updates und weiteren Schadcode aus dem Internet auf den befallenen Rechner herunter.

Aus dem Lagebericht des BSI zur IT-Sicherheit in Deutschland¹³ ist zu entnehmen, dass bereits seit 2007 der Anteil der Trojanischen Pferde unter den Schadprogrammen den Anteil der Viren und Würmer deutlich überholt hat.

2.5.2 Gegenmaßnahmen

Sensible Daten, wie Passworte, PINs und TANs sollten wenn möglich auf sicheren externen Medien, wie einer Smartcard oder einem verschlüsselten USB-Stick abgelegt werden. Ist eine Speicherung auf der Festplatte unumgänglich, so sind die Daten unbedingt verschlüsselt abzulegen, da ein Zugriffsschutz über das Betriebssystem nur einen unzureichenden Schutz gewährt. Ist ein Angreifer im physischen Besitz einer Festplatte, z.B. durch Diebstahl eines Notebooks, so ist es sehr leicht, das installierte Betriebssystem durch Booten eines unsicheren Betriebssystems zu umgehen und direkt auf den Speicher zuzugreifen. Werden dennoch sensible Daten gespeichert, so sind starke kryptografische Verfahren zu deren Verschlüsselung einzusetzen und der Zugriff auf die verschlüsselten Daten ist zu beschränken. Zugriffsrechte für Quellcode-Daten sollten nur besonders privilegierte Benutzer erhalten, um das gezielte Einschleusen von Trojanischen Pferden zu beschränken.

minimale Rechte

Zur Abwehr der besprochenen Bedrohungen ist es wichtig, die Rechte von Benutzern zur Ausführung von Betriebssystemdiensten so zu beschränken, dass jeder Benutzer nur diejenigen Dienste nutzen darf, die er zur Erledigung seiner Aufgaben benötigt (Prinzip der minimalen Rechte). Die Basis dafür ist eine Charakterisierung und Klassifizierung von Betriebssystemdiensten, die für manipulatorische Zwecke missbraucht werden können. Beispiele für solche Betriebssystemfunktionen sind:

- Befehle zur Änderung von Schutzattributen an Dateien,
- Funktionen zum Lesen und Bearbeiten von Passwortdateien,
- Anweisungen, um Netzverbindungen zu anderen Rechnern zu öffnen, und
- Anweisungen, die einen direkten Zugriff auf Speicherbereiche zulassen.

kritische Dienste

Beschränkt man die Zugriffsrechte eines Benutzers, so sind auch seine Manipulationsmöglichkeiten entsprechend beschränkt. Um Zugriffsrechte nach dem Prinzip der minimalen Rechte systematisch vergeben zu können, sind genaue Kenntnisse über die Funktionalität der einzelnen Systemdienste erforderlich. Man muss die Aufgabenprofile der einzelnen Benutzer präzise

Modellierung

¹³ Den jeweils aktuellen Lagebericht des BSI findet man unter https://www.bsi.bund.de/DE/Publikationen/Lageberichte/lageberichte_node.html

erfassen können und die Zusammenhänge kennen, die zwischen den Systemkomponenten bestehen. Des Weiteren ist es notwendig, die Auswirkungen von Rechtewahrnehmungen zu kennen. Da die betrachteten Systeme, insbesondere Betriebssysteme, komplexe Programme sind, kann man einen entsprechenden Überblick nur dadurch erhalten, dass man das System auf einer geeigneten Abstraktionsebene durch ein Modell beschreibt und die Funktionalität der Systembausteine semi-formal oder formal spezifiziert und verifiziert. Das Modell muss die zu schützenden Objekte und die agierenden Subjekte geeignet erfassen, Festlegungen für die in dem System zu vergebenden Zugriffsrechte treffen sowie die Sicherheitsstrategie des Systems spezifizieren. Auf der Basis des Sicherheitsmodells können Sicherheitseigenschaften überprüft werden. Die bekanntesten und wichtigsten Klassen von Sicherheitsmodellen werden in Kapitel 6 vorgestellt.

Signieren

Weiterhin kann man Programme mit der digitalen Unterschrift des Erzeugers versehen und vor der Programmausführung die Korrektheit der Signatur überprüfen. Die digitale Unterschrift sollte für den eindeutigen, digitalen Fingerabdruck, also einen kryptografischen Hashwert des Programms erstellt werden, so dass auf dieser Basis auch die Unveränderlichkeit des Codes überprüft werden kann (vgl. Kapitel 8). Signierter Code bietet natürlich keine Gewähr vor Trojanischen Pferden. Die Vorgehensweise basiert darauf, dass der Empfänger des signierten Codes dem Signierer vertraut. Das heißt, dass der Empfänger darauf vertraut, dass der Signierer Code ohne Schadsoftware-Anteile erzeugt; eine Verifikation, ob dies tatsächlich der Fall ist, findet beim Validieren der Signatur aber nicht statt. Zur Überprüfung von Signaturen benötigt man ein Zertifikat des Signierers und eine Infrastruktur zur Verteilung der Zertifikate, eine so genannte Public-Key Infrastruktur (vgl. Kapitel 9.1). Die Aussagekraft einer digitalen Signatur hängt in hohem Maß von der Vertrauenswürdigkeit der Zertifikate ab.

Trojanische Pferde werden häufig über Viren, Würmer und Buffer-Overflows in ein System eingeschleust. Dabei wird beispielsweise versucht, ein bereits vorhandenes ausführbares Programm bzw. einen Systemdienst durch modifizierten ausführbaren Code zu ersetzen. Eine andere Möglichkeit des Einschleusens besteht darin, den Quellcode vorhandener Dienste bzw. Programme direkt zu manipulieren und durch eine erneute Übersetzung den ausführbaren Objektcode auf der Maschine zu erzeugen. Auf diese Weise vermeidet der Angreifer, dass sein Code aufgrund von Portierungsproblemen auf dem Zielsystem nicht ausführbar ist. Modifikationen des Quellcodes lassen sich durch Quellcodeüberprüfung bzw. Verifikation aufdecken. Dieser Ansatz ist jedoch bei der Komplexität der heutigen Programme und der mangelnden automatischen Unterstützung mit erheblichem Aufwand verbunden und in der Regel nicht praktikabel. Zur Verhinderung von Quellcodemodifi-

Code-Inspektion

kationen ist deshalb dafür zu sorgen, dass der Code nicht direkt zugreifbar im Speicher liegt, also durch Zugriffsrechte und Verschlüsselungen gesichert wird.

2.6 Bot-Netze und Spam

Bot-Netze und Spam sind in heutigen IT-Systemen nahezu allgegenwärtig. Bot-Netze nutzen gezielt verbreitete Schadsoftware, wie Viren, Würmer und Trojaner, um insbesondere verteilte Angriffe unter Mitwirkung einer großen Anzahl von Rechnern durchzuführen, wobei das Bedrohungspotential dieser Angriffe durch die gezielte Koordinierung tausender befallener Rechner erheblicher größer ist, als durch einen infizierten Rechner.

Spam zählt in der Regel nicht direkt zu Schadsoftware, da Spam-Mails meist lediglich unerwünschte Nachrichten enthalten. Da aber zunehmend Trojaner über Massen-Mails wie Spam verbreitet werden, widmet sich ein Unterabschnitt dem Spam-Problem.

2.6.1 Bot-Netze

Ein Bot-Netz ist ein Verbund von infizierten Rechnern, den so genannten Bot-Rechnern, die miteinander kommunizieren und meist durch einen zentralen Server kontrolliert und ferngesteuert werden. Angreifer missbrauchen hierzu sehr häufig den IRC (Internet Relay Chat, vgl. (RFC 2810)), das Protokoll zum Austausch von Text-Nachrichten in Realzeit, das Chatten, bereit stellt. Das Kunstwort *Bot* ist eine Ableitung von dem Wort Robot, das wiederum vom tschechischen Wort Robota (deutsch *Arbeit*) abstammt. Allgemein ist ein Bot ein Programm, das ohne menschliche Eingriffe Aktionen ausführt. Im Kontext der IT-Sicherheit verstehen wir unter einem Bot ein Programm, das von einem Angreifer gezielt ferngesteuert wird. Bots werden herkömmlicherweise über Schadsoftware wie Viren, Würmer und Trojaner verbreitet. Die in einem Bot-Netz verbündeten Rechner führen ihre Angriffe in der Regel völlig transparent für den Besitzer des jeweiligen Rechners aus.

Bot-Netze können aus mehreren tausend Rechnern bestehen. Hauptangriffsziele sind verteilte Denial-of-Service Angriffe (DDoS) auf Anbieter von Internetdiensten und das massenhafte, über verteilte Absender (die Bots) initiierte Versenden von Spam-Nachrichten. Durch die ferngesteuerte Kontrolle von tausenden manipulierter Rechner steht einem Angreifer eine Bandbreite zur Durchführung eines Angriffs zur Verfügung, die die der meisten herkömmlichen Internetzugänge um Größenordnungen übertrifft. Mit einem ausreichend großem Bot-Netz ist es somit möglich, über das Versenden von großen Datenmengen attackierte Serviceanbieter zu überlasten;

also einen klassischen Denial-of-Service herbei zu führen. Neben diesen Angriffen, für die die Bot-Rechner als Zombies instrumentalisiert werden, können Bot-Programme natürlich auch Angriffe auf die befallenen Bot-Rechner selber durchführen und analog zu klassischen Viren und Trojanern als Key-logger agieren, Netzwerkverbindungen zum Download von Malware aufzubauen oder aber auch Viren-Scanner auf dem befallenen Rechner deaktivieren.

IRC

Wie bereits erwähnt, wird nach wie vor zur zentral koordinierten Fernsteuerung eines Bot-Netzes zumeist das IRC verwendet. Abbildung 2.7 veranschaulicht die allgemeine Struktur eines derartigen Bot-Netzes. Nach der Infizierung werden die Bot-Programme aktiviert und verbinden sich mit einem IRC-Server. Der IRC-Server dient als Relaisstation (daher Internet Relay Chat - IRC), damit die Bot-Programme miteinander über einen gemeinsamen Kanal, dem sie explizit beitreten, kommunizieren können.

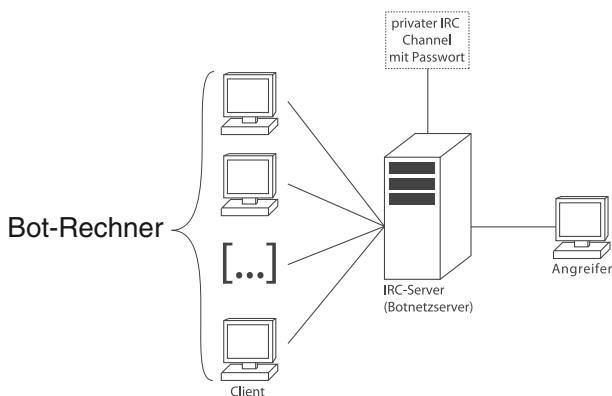


Abbildung 2.7: IRC-kontrolliertes Bot-Netz

Um die Mitgliedschaft in einem Bot-Netz zu kontrollieren, ist ein solcher Kanal zumeist Passwort-geschützt und das jeweilige Passwort wird nur den Bot-Programmen bekannt gegeben. Die Fernsteuerung der Bot-Rechner erfolgt dann durch einen Administrator des IRC-Servers. Beispiele für Kommandos, die ferngesteuert an die Bots weitergeleitet werden, sind die Aufforderung, nach weiteren Systemen, die infiziert werden können, zu scannen, einen DoS-Angriff auszuführen oder aber auch einen Download durchzuführen und das geladene Programm anschließend zu starten.

Abwehr

Da der IRC-Server bei diesen Angriffen die zentrale Kontrolleinheit ist, kann man das Bot-Netz deaktivieren, wenn es gelingt, diesen IRC-Server aus dem Netz zu entfernen. Um zu erkennen, ob ein IRC-Server ein Bot-Netz kontrolliert, wird in der Regel der Datenverkehr überwacht und auf verdächtige Nachrichten hin gescannt. Da zur Fernsteuerung von Bot-Rechnern der

IRC-Server, wie oben beschrieben, Steuerbefehle zur Initiierung von DDoS-Angriffen, zum Download von Malware bzw. zur Durchführung von Scans übermittelt, kann mittels eines Netzwerk Intrusion Detection Systems gezielt nach den Signaturen derartiger Steuerbefehle, also deren syntaktischer Struktur, gesucht werden. Da die Kommunikation in einem IRC-basierten Bot-Netz im Klartext erfolgt, ist eine derartige Überwachung möglich.

Die nächste Generation der Bot-Netze bedient sich jedoch zunehmend dezentraler Peer-to-Peer (P2P)-Strukturen und verwendet eine verschlüsselte Kommunikation. Dadurch wird sowohl das Erkennen eines Bot-Netzes als auch dessen Deaktivierung erheblich erschwert, da es ja gerade ein Charakteristikum eines dezentralen P2P-Netzes ist, dass es auch nach dem Ausfall eines Peers seine Funktionalität weiter aufrecht erhalten kann und sich selbst-organisierend rekonfiguriert.

Trends

Ein weiterer Trend, der auch vom BSI beobachtet wird, ist die Zunahme an Bot-Netzen, die vermehrt zu gezielten, kriminellen Aktivitäten eingesetzt werden. So können Bot-Netze von Kriminellen gemietet werden, um gezielt zum Beispiel einen Service-Provider mittels eines verteilten Denial-of-Service Angriffs zu blockieren und erst nach Zahlung eines Geldbetrages diese Blockade wieder aufzuheben.

2.6.2 Spam

Neben Viren, Würmern und Trojanern, die erheblichen finanziellen Schaden in heutigen Systemen bewirken können, zählt die zunehmende Flut der Spam-Mails zu einem großen Übel, das Unternehmen erhebliche Ressourcen kostet, um ihrer Herr zu werden. Unter Spam versteht man eine Massen-E-Mail, häufig Werbesendungen, die im Internet verbreitet wird. Diese Werbung wird unaufgefordert an Millionen von E-Mail-Adressen versendet. Die Adressen stammen von Adresshändlern, oder werden durch spezielle Programme im Internet gesucht. Ursprünglich handelt es sich bei Spam um Pressfleisch in Dosen, Spiced Porc And Meat. Der Begriff Spam-Mail geht angeblich auf einen Sketch von Monty Python zurück, in dem ein Restaurantgast nur Gerichte bestellen konnte, wenn er gleichzeitig auch Spam geordert hat. In Übertragung auf das Internet bedeutet das, dass ein Benutzer die Vorzüge des Internets und insbesondere des sehr kostengünstigen Versendens von E-Mails nur nutzen kann, wenn auch bereit ist, Spam-Müll in Kauf zu nehmen.

Spam

Untersuchungen z.B. von Nucleus Research (USA) gehen davon aus, dass jeder Mitarbeiter in einem Unternehmen im Durchschnitt täglich 13,3 Spam-Mails erhält, für die er 6,5 Minuten benötigt, um diese zu lesen und danach zu löschen. Das summiert sich jährlich auf etwa 25 Arbeitsstunden pro Kopf und macht etwa 1,4 Prozent der gesamten Arbeitszeit aus. Insgesamt

Schäden

ergeben sich durch die Spam-Mails erhebliche Verluste an Produktivität, die zu sehr hohen gesamtwirtschaftlichen Kosten führen. So kommt 2004 das US-Forschungsinstitut Ferris Research in einer Studie über Spam-Mails zu dem Schluss, dass allein in den USA Einbußen bis zu zehn Milliarden Dollar jährlich aufgrund spamverseuchter E-Mails anfallen. Neben den Kosten, die direkt durch diese Mails verursacht werden, müssen auch die Kosten betrachtet werden, die für die Entwicklung und Aufrechterhaltung von Antispamsystemen aufzubringen sind. Ferris Research schätzt, dass US-Unternehmen allein im Jahr 2004 dafür bis zu 120 Millionen Dollar aufbringen mussten.

Abwehr

Zur Abwehr der Spam-Flut werden Spam-Filterungen eingesetzt. Zur Filterung von Spam stehen im Internet frei verfügbare Filterprogramme zur Verfügung, wie beispielsweise SpamProbe (vgl. <http://spamprobe.sourceforge.net/>) für Unix-Derivate oder auch für Windows-Systeme unter der Cygwin-Umgebung. In Unternehmen erfolgt die Spam-Filterung häufig über zentral administrierte Mailrelays, die neben einer Spam-Filterung auch in der Regel eine Virenerkennung durchführen. Neben der Spam-Erkennung durch Spam-Filterungsprogramme muss über Regelwerke festgelegt werden, wie mit Mails, die unter Spam Verdacht stehen, umgegangen wird. Häufig beschränken sich die Filter darauf, in der Subject-Zeile der Mail diese als mögliche Spam zu markieren und ggf. nicht an den Empfänger weiterzuleiten, sondern sie in einer Spam-Quarantäne-Datei für den Empfänger zu sammeln und regelmäßig, z.B. einmal wöchentlich, dem Empfänger eine Mail zu schicken, in der die Subject-Zeilen und Absenderadressen aller als Spam eingestufter Mails zusammengefasst werden. Manche Unternehmen gehen jedoch auch soweit, Spam-Mails direkt zu löschen.

Rechtslage

Beide Vorgehen, also das Ändern der Subject-Zeile und das Löschen sind jedoch rechtlich bedenklich. Gemäß §303a des Strafgesetzbuches (StGB) macht sich jemand strafbar, der rechtswidrig Daten löscht, unterdrückt, unbrauchbar macht oder verändert. Ob die Subject-Zeile als geschützter Bestandteil der Mail anzusehen ist, wird in juristischen Kreisen noch stark diskutiert. Die DFN Forschungsstelle Recht in Münster rät davon ab, Subject-Zeilen zu manipulieren. Das Löschen von Spam ist laut dieser Forschungsstelle ebenfalls rechtlich nicht unbedenklich, falls in Unternehmen die private Nutzung der E-Mails zugelassen oder geduldet wird. Das Fernmeldegeheimnis ist zu beachten, und gemäß §206 Abs. 2 Nr. 2 des Strafgesetzbuches müssen grundsätzlich alle Mails ohne Verzögerung zugestellt werden. Ein Nutzer kann jedoch sein Einverständnis zum Löschen von Spam erklären.

Das Filtern von Spam-Mails ist aber auch problematisch, weil dadurch in das Fernmeldegeheimnis eingegriffen wird, das nicht nur den Inhalt, sondern

auch die Umstände der Telekommunikation schützt. Nach §206 StGB kann derjenige mit einer Freiheitsstrafe bis zu fünf Jahren bedroht werden, der „unbefugt einer anderen Person eine Mitteilung über Tatsachen macht, die dem Post- und Fernmeldegeheimnis unterliegen und die ihm als Inhaber oder Beschäftigtem eines Unternehmens bekannt geworden sind, das geschäftsmäßig Post- oder Telekommunikationsdienste erbringt.“ Auch hier muss also für das Filtern die Einwilligung des E-Mail Empfängers eingeholt werden.

Nach wie vor steigen die Angriffe durch Schadsoftware weiter an. Im Gegensatz zu früher ist hierbei jedoch der Anteil an Viren und Würmer erheblich zurückgegangen und der Anteil an Trojanischen Pferden liegt bei über 70 Prozent. E-Mails sind noch immer häufig genutzte Verbreitungswege für Schadsoftware. Als Ausgangspunkt für die Verbreitung dienen aber zunehmend URLs und gefälschte Web-Seiten, die vermeintlich interessante Downloads anbieten, so dass sich Benutzer sehr häufig auf diesem Weg einen Trojaner auf ihrem System installieren. Spam-Mails oder aber Massen-E-mails mit verseuchtem Anhang könnten durch die konsequente Nutzung von Signaturen eingedämmt werden. Es ist kritisch zu fragen, ob hier nicht auch die Politik und die Gesetzgebung gefordert sind, um über Regelungen zur Eindämmung der Flut nachzudenken.

Fazit

2.7 Mobile Apps

Im Zusammenhang mit Smartphones sind mobile Apps in den letzten Jahren äußerst populär geworden. Die Frage der Sicherheit tritt dabei verstärkt zu Tage, da vermehrt Apps auf mobilen Endgeräten in Umlauf gebracht werden, die Schadcode mit sich führen.

Mobile Apps¹⁴ sind bereits heute im Consumer-Bereich sehr weit verbreitet. Derzeit wird noch ein großer Teil der Apps als rein lokale Anwendung auf mobilen Geräten ausgeführt, jedoch ist die Entwicklung hin zu smarten mobilen Apps deutlich zu sehen. Kennzeichnend für diese fortgeschrittenen Apps ist ihre Eigenschaft, Dienste zu nutzen, die über das Internet oder ein privates Netzwerk bereitgestellt werden. Viele dieser Dienste werden in einer Cloud betrieben und ermöglichen es dem Nutzer somit, einen konsistenten Datenbestand auf unterschiedlichen mobilen und stationären Geräten zu führen. Eine App stellt dabei die Client-Seite eines Cloud-Dienstes dar, so dass es für den Nutzer keine Rolle mehr spielt, von wo und mit welchem Gerät der Zugriff auf seine Daten erfolgt. Mit diesen Eigenschaften werden smarte Ap-

Apps

¹⁴ Der Abschnitt basiert auf einem Teil eines Buchkapitels, das die Autorin zusammen mit Christian Schneider verfasst hat. Das Buch ist unter dem Titel *Smart Mobile Apps* im Springer Verlag erschienen.

ps zunehmend auch für den Geschäftsbereich attraktiv. Als Business-Apps werden sie ein integraler Bestandteil von Geschäftsprozessen werden.

2.7.1 Sicherheitsbedrohungen

Smartphones und Tablet-PCs werden von Nutzern als eine Art persönliche Informationszentralen verwendet. Die Geräte sind *always on* und werden von vielen Benutzern stets mitgeführt. Von zentraler Bedeutung für die Sicherheit der Anwendungen, und damit natürlich auch der Apps, ist die Systemsicherheit der mobilen Plattform, auf der die Anwendungen zur Ausführung gelangen. Der zweite sicherheitskritische Bereich wird durch die Anwendungsebene definiert. Sie bildet die direkte Ausführungsumgebung für die Apps.

Bedrohungen der Mobilen Plattformen

Plattform

Mobile Plattformen unterliegen einer höheren Gefährdung als stationäre. Sie werden an verschiedenen Orten und damit in unterschiedlich vertrauenswürdigen Umgebungen eingesetzt. Sie gehen schneller verloren oder werden gestohlen als etwa ein PC. Gleichzeitig kann sich ein Unbefugter Zugang zu einem mobilen Gerät verschaffen. Die kritischste Bedrohung für mobile Plattformen ist jedoch, dass solche personalisierten Geräte oftmals nicht oder nur unzureichend von einer unternehmensweiten Sicherheitsstrategie erfasst werden. Meistens werden sie von den Benutzern selbst administriert. Die Erfahrung aus der Welt der Desktop-Betriebssysteme lehrt jedoch, dass noch immer vielen Anwendern ein hinreichendes Sicherheitsbewusstsein fehlt: Zugriffssperren und Passwortschutz werden deaktiviert, da sie als störend empfunden werden, Sicherheitswarnungen werden ignoriert, Software-Updates werden, wenn überhaupt, sehr unregelmäßig eingespielt. In solchen Fällen vergrößert sich entsprechend das bereits vorhandene Bedrohungspotential mit einer unnötig großen Angriffsfläche der mobilen Plattform.

Update-Problematik

Viele mobile Betriebssysteme müssen für das Einspielen von Updates mit einem Rechner verbunden werden. Sogenannte *Over-the-Air* (OTA)-Updates ermöglichen Firmware-Updates ohne PC-Verbindung. Mit den schnellen Release-Zyklen von neuen Versionen mobiler Plattformen wie Android, können viele Hersteller von mobilen Geräten jedoch nicht mithalten. Sie stellen deshalb nur für einen kurzen Zeitraum Firmware-Updates für ihre Produkte zur Verfügung, und das auch nur mit deutlicher Verzögerung zur Android-Version. Erschwerend kommt hinzu, dass die Mobilfunkbetreiber viele Smartphones mit speziellen Anpassungen vertreiben. Diese benötigen jedoch ihrerseits speziell angepasst Firmware-Updates, die in der Regel wiederum noch später als die Updates der Hardware-Hersteller zur Verfügung gestellt werden. Die Konsequenz ist, dass auf den meisten Android-Geräten

veralteter System-Software installiert ist, die möglicherweise bekannte aber noch nicht geschlossene Sicherheitslücken enthält.

Bedrohungen der Anwendungsebene

Die meisten Apps werden über einen speziellen App-Marktplatz bekannt gemacht und vertrieben. Für die mobilen Endgeräte von Apple ist Apple's AppStore sogar der einzige Weg, um neue Anwendungen zu beziehen und auf den Geräten zu installieren. Der Nutzer muss also der Qualitätskontrolle der Marktplatzbetreiber vertrauen. D.h. der Nutzer muss darauf vertrauen, dass die über den Marktplatz angebotenen Apps keine Sicherheitslücken oder Schad-Funktionen aufweisen. Da die entsprechenden Kontrollverfahren nicht offengelegt sind und Apps keine standardisierte Zertifizierungsprozedur durchlaufen müssen, kann über die Qualität der durchgeföhrten Kontrollen keine verbindliche Aussage getroffen werden. So musste Apple in der Vergangenheit wiederholt Apps aus dem Store zurückrufen, da Sicherheitslücken entdeckt wurden. Für den offenen Android-Marktplatz sieht die Situation nicht besser aus.

Malware-App

Die Problematik des Vertrauens in Closed-Source-Software besteht zwar im Prinzip auch bei Software, die auf Desktop-Betriebssystemen installiert wird. Auf einer mobilen Plattform wird dieses Problem jedoch dadurch verschärft, dass beispielsweise Smartphones viele verschiedene Datenquellen aggregieren und diese über eine einheitliche Schnittstelle leicht für andere Anwendungen zugänglich machen. Darüber hinaus können schadhafte Apps ihrem Besitzer direkt beträchtliche Kosten verursachen, in dem etwa SMS-Nachrichten an Mehrwertdienste gesendet oder Sonderrufnummern angerufen werden. Wenn sich das Smartphone durch die NFC-Technologie auch noch als mobile Geldbörse etabliert, wird sich diese Problematik voraussichtlich weiter verschärfen.

Eine App kann aber umgekehrt auch nicht zuverlässig feststellen, ob sie in einer sicheren, nicht modifizierten Umgebung ausgeführt wird. Eine durch Schadcode manipulierte Ausführungsumgebung könnte die App missbrauchen, um gezielt Daten auszuspähen, zu manipulieren oder über die App Zugriffe auf die Daten anderen Benutzer, andere Geschäftsprozesse etc. zu gelangen. Es stellt sich die Frage, inwieweit die Daten einer Anwendung vor unberechtigten Zugriffen durch eine andere Anwendung geschützt sind. Hier ist zunächst die Zugriffskontrolle des mobilen Betriebssystems zu betrachten. Je nach Hersteller sind hier sehr starke Unterschiede zu verzeichnen. Die Größe der zu schützenden Einheit, also die Granularität der Kontrolle, spielt dabei eine wesentliche Rolle. Je geringer die verfügbare Granularität, desto größer sind die zu schützenden Einheiten und desto mehr Zugriffsrechte erhält eine App, auch wenn sie diese nicht im vollen Umfang benötigen würde. Da neue Software von unbekannten Autoren heutzutage auf einfachste Weise

Unsichere Umgebung

aus dem App-Marktplatz auf die mobilen Geräte gelangt, sollte die Isolation der Apps untereinander und die Beschränkung der Rechte auf das Nötigste oberstes Gebot sein.

Aber auch legitime Zugriffe können missbraucht werden und damit unerwünschte Folgen haben. Beispielsweise gleicht die offizielle Facebook-App, die für alle populären mobilen Plattformen verfügbar ist, die Kontaktliste auf Smartphones mit den Freunden auf Facebook ab, wozu die App natürlich Zugriff auf die Kontakte des Geräts benötigt. Allerdings lädt die App auch alle Kontakte in das Phonebook hoch, die nicht bei Facebook sind, ohne dass der Benutzer dies unterbinden kann. Auf diese Weise gelangt Facebook an viele Kontaktdaten von Nicht-Mitgliedern und kann diese über die Telefonnummern als Identifikatoren verknüpfen, um das soziale Netzwerk auch über die Mitglieder hinaus zu erweitern. Dieses Beispiel verdeutlicht, warum die Aggregation verschiedener Datenquellen auf einem Gerät und die Verfügbarkeit dieser Daten über eine systemweite Schnittstelle eine besondere Bedrohung darstellt.

2.7.2 Gegenmaßnahmen

Die Hersteller von mobilen Betriebssystemen, Endgeräten und klassischen Sicherheitslösungen sind sich der zuvor beschriebenen Bedrohungen natürlich bewusst und versuchen diesen auf unterschiedliche Weise zu begegnen. Im Folgenden wird auf einige Maßnahmen exemplarisch eingegangen.

Android

Google folgt auf seiner Android-Plattform der Philosophie, dass den Apps und ihren Autoren grundsätzlich nicht vertraut werden kann. Entsprechend restriktiv sind hier die Vorgaben für die Zugriffsrechte, so dass jede App zunächst nur ihre eigenen Daten lesen und schreiben darf. Um auf systemweite Daten, wie etwa die Kontakte oder den Kalender, zuzugreifen, muss eine App die dafür vorgesehene Schnittstelle zum Android-System verwenden. Diese Schnittstellen unterliegen einem speziellen Kontrollmechanismus. Damit der App diese Zugriffe vom System gewährt werden, muss sie schon bei der Installation angegeben haben, dass sie diese Berechtigung erwünscht. Die dem Vorgehen zugrundeliegenden allgemeinen Prinzipien des *default deny* und des *need-to-know* sind im Grundsatz sehr positiv. Allerdings definiert Android annähernd 200 verschiedene Rechte, die den Zugriff auf die persönlichen Daten regeln, oder aber auch auf Geräte wie Mikrofon, Kamera und GPS-Empfänger. Daneben vergibt Android Rechte zur Verwendung von Systemfunktionen, wie das Senden von SMS-Nachrichten und das Initiieren von Telefongesprächen. Diese Berechtigungen muss der Benutzer bestätigen, bevor die App installiert wird. Dieses Vorgehen schafft eine gewisse Transparenz, verlagert aber damit das Problem auf den Benutzer, der ein entsprechendes Problembewusstsein haben muss, um eine sinnvolle Entscheidung treffen zu können.

Apples iOS, siehe dazu auch Abschnitt 13.1, verfügt nicht über solche feingranularen Berechtigungen, sondern verfolgt eine andere Strategie zur Zugriffskontrolle. Anders als auf der Android-Plattform erlaubt Apple die Installation von Apps ausschließlich über den Apple AppStore. Dadurch gibt es für iOS-Geräte einen zentralen Punkt, an dem Apple regulierend eingreifen kann. Alle Programme, die in den AppStore eingestellt werden, müssen sich zunächst einem Review-Prozess unterziehen. Dabei analysiert Apple die verwendeten Systemfunktionen und testet das Programm rudimentär. Verwendet eine App Systemfunktionen, die sie zur Erbringung ihrer Funktionalität aber gar nicht benötigt, wird das Programm abgelehnt und nicht im AppStore aufgenommen. Hier übernimmt also der Hersteller in gewisser Weise die Verantwortung, dass eine App nur die Zugriffsrechte erhält, die sie benötigt.

Die skizzierten Maßnahmen wie Zugriffskontrollen und Transparenz verhindern jedoch nicht, dass Apps mit Schadfunktionen auf ein mobiles Endgerät gelangen können. Das haben auch die Hersteller von Sicherheits-Software erkannt und bieten mobile Varianten ihrer Security-Suiten an. Der Funktionsumfang dieser Produkte reicht vom obligatorischen Viren-Scanner über ein entferntes Löschen der Daten (Remote-Wipe) bei Verlust oder Diebstahl des Endgeräts bis hin zu Rufnummernsperrlisten und SMS-Spam-Filtern.

2.8 Meltdown- und Spectre-Angriffsklassen

2.8.1 Einführung

Am 3. Januar 2018 wurden erstmalig durch Google¹⁵ in der breiten Öffentlichkeit gravierende Sicherheitsprobleme bekannt, die nahezu alle gängigen Mikroprozessor-Architekturen betreffen. Die Probleme wurden bereits im Juli 2017 in Form eines Responsible Disclosure von Google an die Chiphersteller Intel, AMD und ARM gemeldet. Die Probleme können von Angreifern ausgenutzt werden, um die Speicherisolierung der meisten heutigen Prozessoren zu umgehen und unautorisiert Daten aus dem geschützten Bereich des Betriebssystems oder aus Speicherbereichen anderer Anwendungs-Prozesse auszulesen. Auf diese Weise können Angreifer in den Besitz von zum Beispiel Passworten, Schlüsselmaterialien oder auch anderen sensitiven Daten gelangen.

Die Sicherheitsprobleme stellen Bedrohungen für alle Rechner und Plattformen dar, auf denen es möglich ist, Programme (Code) von Dritten zur Ausführung zu bringen. Das betrifft in der Regel alle End-Kunden-Geräte,

iOS

Verwundbare
Systeme

¹⁵ <https://googleprojectzero.blogspot.de/2018/01/reading-privileged-memory-with-side.html>

auf denen ein Web-Browser läuft, in dem JavaScript aktiviert ist, so dass Fremdcode im Browserprozess ausgeführt werden kann (eine heute sehr übliche Web-Nutzung). Aber auch Systeme, auf denen unterschiedliche Anwendungen von Dritten ausgeführt werden (Hosting-Systeme), virtualisierte Umgebungen und Cloud-Plattformen, auf denen Code Dritter ausgeführt wird, sind durch die Lücke gefährdet. In den Anfang Januar 2018 veröffentlichten Papieren werden Proof-of-Concept (PoC) Angriffe vorgestellt, die die Ausnutzbarkeit der Verwundbarkeiten in drei Angriffsvarianten, dem Meltdown und den Spectre-Angriffen demonstrieren. Die Angriffe sind technisch zum Teil sehr anspruchsvoll und hinterlassen nach ihrer Durchführung keine Spuren auf den attackierten Systemen. Derzeit - Stand erstes Quartal 2018 - gibt es keine gesicherten Erkenntnisse, ob die PoC-Angriffe bereits aufgegriffen und Systeme systematisch gehackt wurden. Da jedoch die Meltdown-Angriffe mit leistbarem Aufwand durchführbar sind, ist nicht auszuschließen, dass zumindest Meltdown-artige Angriffe bereits in der realen Welt durchgeführt wurden.

Abwehr

Die Ursache der Problematik liegt in der Hardware der genutzten Prozessoren, so dass reine Software-Patches zur Lösung nicht ausreichen sondern Microcode-Patches zusammen mit Betriebssystem-Updates erforderlich sind, um die Probleme in den Griff zu bekommen. Diese Lösungen haben zum Teil erhebliche Leistungseinbußen zur Folge, da z.B. zusätzliche Kontextwechsel eingeführt oder die Möglichkeiten zur Parallelverarbeitung eingeschränkt werden. Das Problem wird langfristig nur durch Änderungen bei den Prozessorarchitekturen zu lösen sein.

Meltdown und Spectre im Überblick

Meltdown

Das Meltdown-Problem [109] wurde von Forscherteams vom Google Project Zero, Cyberus Technology sowie der TU Graz unabhängig voneinander aufgedeckt. Meltdown-Angriffe nutzen aus, dass insbesondere bei Intel-Prozessoren aus Performanzgründen der Speicherbereich des Betriebssystems (Kernelbereich) vollständig in den virtuellen Adressraum eines jeden Nutzerprozesses abgebildet ist. Das heißt, dass die Seitentabelle eines jeden Nutzerprozesses auch die Speicherseiten des Betriebssystems umfasst, so dass bei Zugriffen auf den Kernelbereich, zum Beispiel bei Systemcalls, keine aufwändigen Kontextwechsel mit Adressraumwechsel erforderlich sind. Diese Seiten sind als Kernel-Pages markiert, so dass normalerweise bei einem unberechtigten Zugriffsversuch aus dem User-Bereich auf den Kernel-Bereich eine Zugriffsverletzung erkannt und der Zugriff verweigert wird. Genau dieser Zugriffsschutz wird aber in gängigen Prozessoren aus Performanzgründen ganz bewusst temporär umgangen, wenn eine spekulative oder Out-of-Order Ausführung (siehe unten) durchgeführt wird. Dabei erfolgen Speicherzugriffe zunächst ohne Kontrolle. Eine Prüfung findet erst

statt, wenn klar ist, dass die betroffenen Befehle wirklich benötigt werden. Wird dabei eine Zugriffsverletzung erkannt, so werden die Ergebnisse der spekulativen Ausführung verworfen. Solche Ergebnisse sind beispielsweise Registerinhalte. Wie weiter unten genauer ausgeführt, werden durch das spekulative Vorgehen Daten unkontrolliert aus dem Adressraum des Kernels in den L1 Cache des Prozessors geladen und können über gezielte Cache-Seitenkanal-Angriffe (vgl. Seite 11.1.3) im User-Bereich sichtbar gemacht werden.

Die Spectre [104] Angriffe gefährden insbesondere auch virtualisierte Umgebungen und Cloud-Plattformen, die von unterschiedlichen Nutzern gemeinsam genutzt werden. Die nachfolgend geschilderten Familien von Angriffen ermöglichen den Zugriff auf Speicherbereiche anderer Nutzerprozesse, die durch die genutzten Virtualisierungstechniken eigentlich von einander abgeschottet sein sollten. Die Familie von Angriffen, die einen unerlaubten Zugriff auf Speicherbereiche im gleichen Prozess ermöglichen, stellen insbesondere eine Bedrohung für Web-Anwendungen dar, falls JavaScript auf dem Opfersystem aktiviert ist. Gelingt es einem Angreifer, schadhaften Javascript-Code auf dem Opferrechner einzuschleusen und zur Ausführung zu bringen, dann kann dieser Schadcode auf den Speicherbereich seines Host-Prozesses, das ist der Browser-Prozess, zugreifen. Da der Browser häufig beispielsweise Zugangspassworte zwischenspeichert, kann über einen Timing-Angriff versucht werden, diese Daten auszulesen.

Spectre

2.8.2 Background

Die Meltdown und Spectre-Angriffe nutzen aus, dass heutige und auch die meisten älteren Prozessoren zur Steigerung der Leistung der Prozessoren verschiedene Techniken einsetzen. Dazu gehören (implicit) Caching, Out-of-Order Execution, Speculative Execution und Branch Prediction. Durch eine Kombination verschiedener dieser Techniken entstehen Sicherheitsprobleme, auf die wir im Folgenden näher eingehen. Die eigentlichen Angriffe basieren auf wohlbekannten Techniken der Seitenkanalanalysen von Caches (u.a. [82, 135]).

Das Ziel der spekulativen Ausführung von Befehlen besteht darin, Wartezeiten, die sich bei der sequentiellen Bearbeitung von Befehlen ergeben können, zu nutzen, um Befehle auszuführen, für die der Prozessor spekuliert, dass sie als nächstes benötigt werden. Mögliche Wartezeiten bei der Befehlausführung ergeben sich dadurch, dass für die Ausführung je nach dem, auf welche Speicher zur Bearbeitung des Befehls zugegriffen wird, sehr viele Taktzyklen erforderlich sind, da ein elektronisches Signal pro Takt nur eine geringe Distanz zurücklegt. Je größer der physische Abstand zwischen der ALU und dem Speicher, aus dem Daten gelesen werden, ist,

Spekulative Ausführung

desto länger dauert der Zugriff. Durch den Aufbau von Speicherhierarchien mit verschiedenen Schnellzugriffsspeichern, den L1, L2 und L3 Caches, die deutlich geringere Zugriffslatenzen aufweisen als der Hauptspeicher, reduzieren Mikroprozessoren die erforderlichen Zugriffszeiten. Die Cachegrößen und Zugriffslatenzen beispielsweise eines Intel i7-4770 (Haswell) Prozessors¹⁶ sind in der Tabelle 2.1 zusammengestellt.

Typ	Größe	Latenz
L1 Data	32 KB, 64 B/line	4-5 cycles
L1 Instruction	32 KB, 64 B/line	(bei misprediction) 18-20 cycles
L2	256 KB, 64 B/line	12 cycles
L3	8 MB, 64 B/line	36 cycles
RAM	32 GB	36 cycles + 57 ns (0.29ns / cycle), d.h. ca 230 cycles

Tabelle 2.1: Cachegrößen und Zugriffslatenzen eines Intel i7-4770 Prozessors

implizites Cachen

Zur Erhöhung der Performanz lädt der Prozessor durch seine Prefetcher-Komponente Daten frühzeitig in Caches. Beispielsweise wählt der Prozessor bei einem bedingten oder indirekten Sprung einen Ausführungspfad spekulativ aus (Branch Prefetching). Die zur Ausführung der spekulativ ausgeführten Maschinenbefehle erforderlichen Daten werden in den L1 Cache geladen. Dies nennt man auch implizites Caching, da Daten geladen werden, noch bevor auf sie gemäß der sequentiellen von Neumann-Befehlsbearbeitung zugegriffen wird. Sollten in den folgenden Instruktionen die Daten benötigt werden, so kann der Speicherzugriff sehr schnell in wenigen Zyklen erfolgen, da die Daten bereits im L1 Cache vorhanden sind. Der Prozessor führt die Befehle auf diesem Pfad solange aus, bis die Sprungbedingung ausgewertet und das Ziel des Sprungs berechnet ist. Erst dann ist für den Prozessor klar, ob die bereits ausgeführten Befehle wirklich benötigt werden oder deren Ergebnisse zu verwerfen sind.

BTB

Um möglichst wenig fehlerhafte Vorhersagen (misprediction) bei einem Branch Prefetching zu erhalten, nutzen Prozessoren so genannte Branch Target Buffer (BTB). Dazu wird ein spezieller Cache, der Branch Target Buffer verwaltet, der die Zieladressen für Verzweigungen protokolliert.

Out-of-Order Execution

Ein weiteres Feature, um die Leistung heutiger Prozessoren zu erhöhen, ist die Out-of-Order Execution. Hierbei nutzt der Prozessor die Möglichkeit zur Parallelverarbeitung (superskalare Prozessoren), die durch die vorhandenen Subsysteme, wie z.B. Arithmetik-Einheit, Speicher, Vektor Logik gegeben ist. Das heißt, Instruktionen werden parallel ausgeführt, aber erst

¹⁶ <http://www.7-cpu.com/cpu/Haswell.html>

dann als gültig gesetzt, wenn alle vorherigen Instruktionen fehlerfrei abgearbeitet sind. Spekulative und Out-of-Order Ausführung führen zu sehr großen Leistungsverbesserungen. Werden jedoch im realen Programmablauf die vorausschauend bereitgestellten Daten und berechneten Ergebnisse doch nicht benötigt, beispielsweise weil das Programm an eine andere Stelle springt, so werden die Ergebnisse verworfen. Da die Berechnungen in den Wartezeiten des Prozessors durchgeführt wurden, ist das kein Performanznachteil. Wenn die fälschlich voraus berechneten Befehle völlig seiteneffektfrei zurück gerollt würden, würden keine Probleme auftreten. Diese ergeben sich jedoch gerade daraus, dass die Befehlausführungen bemerkbare Zustandsänderungen nach sich ziehen.

Basis für Angriffsmöglichkeiten

Dieses skizzierte prinzipielle Verhalten von Prozessoren birgt folgende Probleme. Zum einen werden Daten von spekulativ ausgeführten Befehlen ohne Prüfung der Zugriffsrechte in die L1 Caches geladen. Auf diese Weise können beispielsweise Daten des Betriebssystemkerns geladen werden und sind unter Umgehung der Rechtekontrolle im User-Bereich zugreifbar. Ein entsprechender Zugriff aus einem nicht privilegierten Bereich auf einen privilegierten, nämlich den Betriebssystembereich, würde normalerweise über eine Zugriffsverletzung erkannt und unterbunden.

Zudem werden die Daten nicht aus dem L1 Cache gelöscht, auch wenn die spekulativ ausgeführten Befehle nicht benötigt und die spekulativen Zwischenergebnisse aus den Registern verworfen werden. Das heißt, dass die spekulative Ausführung den Systemzustand verändert. Basierend auf gelesenen Daten können weitere Berechnungen durchgeführt und auch weitere Daten aus dem Hauptspeicher (RAM) gelesen werden. Alle diese Daten verbleiben auch dann im L1 Cache, wenn der Prozessor erkannt hat, dass die spekulative Berechnung zu verwerfen ist.

Beispiel 2.6 (Spekulative Ausführung)

Das nachfolgende Beispiel ist aus dem Google Project Zero Papier¹⁷ entnommen.

```
struct array {  
    unsigned long length;  
    unsigned char data[];  
};  
struct array *arr1 = ...;  
unsigned long untrusted_offset_from_caller = ...;
```

¹⁷ <https://googleprojectzero.blogspot.de/2018/01/reading-privileged-memory-with-side.html>

```

if (untrusted_offset_from_caller < arr1->length) {
    unsigned char value =
        arr1->data[untrusted_offset_from_caller];
    ...
}

```

Betrachten wir den Code im Beispiel. Der Prozessor weiß nicht, dass *untrusted_offset_from_caller* größer als *arr1->length* ist. Er wird die nächsten Instruktionen spekulativ ausführen. Das heißt, aus dem Hauptspeicher wird der Wert in *arr1->data[untrusted_offset_from_caller]* gelesen. Dieser Zugriff außerhalb der Array-Grenzen (out-of-bounds) führt aber nicht zu einer Fehlersituation, da der Prozessor im Laufe der Programmausführung feststellen wird, dass dieser Verzweigungspfad nicht beschritten wird und die Befehlsausführungen verwirft, d.h. die Registerwerte und Speicherwerte werden dem Programm nicht sichtbar gemacht. Dennoch liegen die gelesenen Daten im L1 Cache vor.



Angriffsklassen wie Meltdown und Spectre nutzen dieses Verhalten von leistungsfähigen Prozessoren gezielt aus, um unberechtigten Zugriff auf Daten zu erhalten. Wenn beispielsweise abhängig von einem spekulativ geladenen Datum weitere Befehle geladen werden, so kann man anhand der ausgeführten Befehle Rückschlüsse auf den Wert des geladenen Datums ziehen. Dazu werden wohlbekannte Techniken, die Cache-Timing-Angriffe, verwendet. Die Angriffe erfordern jeweils sehr exakte Zeitmessungen, was ein Ansatzpunkt sein könnte, um die Timing-Angriffe abzuwehren.

2.8.3 Angriffsklassen

Die im Januar 2018 veröffentlichten Angriffe beschreiben drei Angriffsfamilien:

- Variante 1: Bounds Check Bypass: CVE-2017-5753
- Variante 2: Branch Target Injection: CVE-2017-5715
- Variante 3: Rogue Data Cache Load: CVE-2017-5754

Der vergrößerte Ablauf ist bei allen drei Familien ähnlich. Um jeweils ein Bit an Information wieder herzustellen, sind folgende Schritte durchzuführen:

1. Trainieren des Branch Target Buffers (BTB).
2. Spekulative Ausführung des Codes, der den Speicher liest.
3. Messen der Cache-Zugriffszeiten, um die Daten aus Schritt 2 zu lesen.

Variante 1 Bounds Check Bypass

Das Angriffsziel ist es, solche Speicherinhalte des eigenen Prozess-Adressraums zu lesen, die im User-Bereich normalerweise nicht sichtbar sind. Der Angriff verläuft analog zu dem in Beispiel 2.6 skizzierten Vorgehen, indem der Angreifer dafür sorgt, dass die gewünschten Daten spekulativ in den L1 Cache geladen werden. Der Angreifer kann nun versuchen, den Cache-Speicher bitweise zu lesen. Nachfolgend wird das Beispiel aus dem Google-Papier¹⁸ zur Erläuterung aufgegriffen.

Beispiel 2.7 (Bounds Check Bypass)

Sei der unten angegebene Code gegeben. Im L1 Cache seien alle Daten außer arr1->length, arr2->data[0x200] und arr2->data[0x300] bereits vorhanden. Für die if-Anweisung wird die Vorhersage getroffen (branch prediction), dass die Bedingung zu *true* ausgewertet wird. Der Prozessor führt deshalb solange spekulativ die Befehle des True-Zweiges aus, bis arr1->length geladen und die spekulative Ausführung beendet wird. Die spekulative Befehlsausführung umfasst damit folgende Schritte:

- Lade value = arr1->data[untrusted_offset_from_caller].
- Lade index2-Wert.
- Lade, abhängig vom Wert von index2 den Wert aus arr2, also arr2->data[index2], in den L1 Cache.

```
struct array {
    unsigned long length;
    unsigned char data[];
};

struct array *arr1 = ...; /* small array */
struct array *arr2 = ...; /* array of size 0x400 */
                         /* >0x400 (OUT OF BOUNDS!) */
unsigned long untrusted_offset_from_caller = ...;
if (untrusted_offset_from_caller < arr1->length) {
    unsigned char value =
        arr1->data[untrusted_offset_from_caller];
    unsigned long index2 = ((value&1)*0x100)+0x200;
    if (index2 < arr2->length) {
        unsigned char value2 = arr2->data[index2];
    }
}
```

¹⁸ <https://googleprojectzero.blogspot.de/2018/01/reading-privileged-memory-with-side.html>

Timing-Attack

Nachdem der Prozessor erkannt hat, dass der Wert `untrusted_offset_from_caller` größer ist als der Wert `arr1->length`, wird der spekulative Pfad verworfen, jedoch bleibt der Wert `arr2->data[index2]` weiterhin im L1 Cache. Dies ermöglicht nun folgenden Seitenkanal. Ein Angreifer kann die Zeit messen, die erforderlich ist, um die Datenwerte `arr2->data[0x200]` und `arr2->data[0x300]` zu laden. Der Angreifer kann daraus ableiten, ob der Wert von `index2` während der spekulativen Ausführung den Wert `0x200` oder `0x300` hatte. Dies wiederum erlaubt den Rückschluss, ob `arr1->data[untrusted_offset_from_caller]&1` den Wert `0` oder `1` besitzt. Ein solcher Timing-Angriff enthüllt also einen 1-bit Wert.

Den Papieren von Google ist zu entnehmen, dass mit einer Datenrate von ca. 2000 Byte pro Sekunde Daten aus dem Speicher über solche Angriffe Bit für Bit erschlossen werden können. In den Papieren wird ein Proof-of-Concept erläutert, in dem unter einer Debian-Linux-Distribution aus dem nicht privilegierten User-Bereich beliebige Zugriffe auf 4GB Speicher des Betriebssystemkerns möglich waren.



Um dieses Verhalten des Prozessors gezielt für einen Angriff ausnutzen zu können, muss es einem Angreifer möglich sein, dafür zu sorgen, dass ein solcher verwundbarer Code im Kontext des Opferprozesses mit einem Indexwert außerhalb der Grenzen (out-of-bounds) ausgeführt wird. Der Code, der spekulativ ausgeführt wird, muss dazu bereits auf dem Rechner vorhanden sein. Darum sind für diese Angriffsszenarien JIT (just in Time) Compiler von besonderem Interesse, da sie genutzt werden können, um verwundbaren Code zu generieren. Auch JavaScript mit Exploits, die Daten aus der Browser Sandbox extrahieren, kann genutzt werden. Mit Techniken wie eBPF¹⁹ kann, wie der Proof-of-Concept von Google auch gezeigt hat, sogar Kernel-Speicher gelesen werden, da der eBPF Interpreter direkt im Kernel ausgeführt wird und aus dem User-Bereich nutzbar ist.

Abwehr

Derartige Timing-Angriffe sind zu verhindern, wenn keine Daten mehr spekulativ geladen werden, deren Adressen von anderen spekulativ geladenen Daten abhängen, oder wenn sichergestellt wird, dass solche Daten nicht im Cache verbleiben, wenn sie nicht vorher schon dort waren. Patches, wie der für die C/C++-Compiler von Microsoft, stellen eine Switch (Qspectre) zur Verfügung. Damit wird durch den Compiler automatisch eine so genannte spekulative Barriere in den übersetzten Code eingefügt, wenn er eine Instanz

¹⁹ Extended Berkeley Packet Filter. Paketfilter, der in der virtuellen Maschine im Kernel ausgeführt wird, um Netzwerkpakete frühzeitig zu filtern. Um BPF Bytecode in Assemblercode der Gastmaschine zu transformieren, kann der in den Kernel integrierte JIT Compiler genutzt werden.

der Variante 1 Problematik erkennt, so dass mit den noch nicht geprüften Werten keine vorausschauenden Ladeoperationen durchgeführt werden.

Variante 2 Branch Target Injection

Ausgangspunkt für diese Angriffsvariante ist der Branch Target Buffer (BTB), der über Prozessgrenzen hinweg gültig ist. Bislang waren nur Angriffe bekannt, durch die der Angreifer über diesen gemeinsamen Buffer Informationen darüber erhalten hat, an welcher Adresse Code im Opferprozess ausgeführt wird. In den neuen Angriffen versucht der Angreifer den Opferprozess direkt zu beeinflussen, indem er den BTB von anderen Prozessen trainiert. Dieses Training hat zur Konsequenz, dass für den Angriff nutzbarer Code spekulativ durch den Opferprozess ausgeführt wird. Konzeptuell besitzt der Angriff eine Ähnlichkeit mit ROP-Angriffen (Return-oriented Programming), wobei anders als bei ROP die genutzten Gadget(s) nicht sauber enden, dafür aber einen Seitenkanal aufbauen müssen. Laut dem zugrundeliegenden Papier von Google ist eine Datenrate von ca. 1500 Byte pro Sekunde möglich.

Die Basis für einen Angriff ist Code, der einen indirekten Sprung enthält²⁰. Der Angriff sorgt durch das Flushen von Caches dafür, dass das Sprungziel aus dem Speicher geladen werden muss. Wenn die CPU eine solche indirekte Verzweigung ausführt, kann das Sprungziel erst berechnet werden, nachdem die Cacheline wieder geladen ist. Dieser Ladevorgang ist zeitaufwändig und erfordert i.d.R. über hundert Taktzyklen. Die CPU kann diese Zeit nutzen, um spekulativ Befehle basierend auf der Branch Prediction auszuführen. Die Angriffsidee ist nun, diese Vorhersage zu beeinflussen und dafür zu sorgen, dass die CPU Befehle ausführt, die der Angreifer gezielt über dedizierte Sprungzieladressen in den BTB injiziert hat. Dafür ist es notwendig, das genaue Verhalten der Branch Prediction von Prozessoren zu verstehen. Dies ist jedoch nicht einfach, da die Hersteller keine vollständige Spezifikation offen legen.

Indirekter Sprung

vergrößerter Ablauf

In dem Google Papier ist ein Proof-of-Concept Angriff erläutert, bei dem u.a. ausgenutzt wird, dass der BTB nicht die vollständige Zieladresse sondern nur die unteren 32 Byte abspeichert. Weiterhin wird der Branch History Buffer (BHB) für den Angriff ausgenutzt. Der BHB speichert Informationen über die letzten Verzweigungen und deren Ziele und stellt damit einen Ausschnitt des Kontrollflusses dar. Diese Information ist für den Predictor hilfreich, wenn es für eine Verzweigung mehrere mögliche Ziele gibt. In dem

²⁰ Im Gegensatz zu einem direkten Sprung spezifiziert ein indirekter Sprung in seinem Argument nicht die Zieladresse des nächsten Befehls sondern eine Adresse, unter der die Adresse des nächsten Befehls zu finden ist (Indirektion). Die Zieladresse des nächsten Befehls kann also erst bestimmt werden, wenn der Sprungbefehl ausgeführt wird

KVM-Angriff

PoC Angriff aus dem Project Zero Papier wird durch gezieltes Herbeiführen von Mispredictions dafür gesorgt, dass aus den im BHB gespeicherten Informationen die Adresse eines interessierenden Speicherbereichs hergeleitet werden kann.

Um Daten aus dem Speicher zu extrahieren, wird wie oben bereits gesagt, der Cash gezielt geflushed, wozu die Operationen Flush+Reload genutzt werden. Dies erfordert jedoch die Kenntnis einer virtuellen Adresse des Kernelbereichs des Hosts im virtuellen Speicher des KVM-Gasts. Da der gesamte physikalische Speicher des Kernels des Hosts in einen bestimmten Bereich des Gastes (physmap) abgebildet wird, dieser Bereich aber den Gastprozessen zufällig zugewiesen wird, würde ein Bruteforce-Angriff mit Durchsuchen des möglichen Adressraums sehr aufwändig (aber möglich) sein. Durch einen BTB-Injection Angriff kann die Angriffsduer auf wenige Minuten verkürzt werden. Hierzu wird der BTB gezielt mit Daten injiziert, so dass Daten geladen werden, die vom Angreifer kontrolliert werden und der Angreifer kann damit testen, ob eine Seite des Gast-Adressraums geladen wird. Um schließlich die eigentlichen Daten aus dem Speicherbereich der Kernels des Hosts zu lesen, wird ein ROP-artiger Ansatz verfolgt. Dazu wird wiederum der eBPF Interpreter ausgenutzt, der im Kernelbereich des Hosts integriert ist. Normalerweise ist es jedoch nicht möglich, diesen Interpreter aus einer Gast-VM aufzurufen. Da jedoch der Code des Interpreters im Code-Segment des Kernels liegt, kann dieser Code als Gadget für einen ROP Angriff genutzt und ungeprüfter eBPF-Bytecode kann spekulativ ausgeführt werden,

BTP Injection

Zur Abwehr derartiger Angriffe wurden Maßnahmen vorgestellt, die teilweise jedoch zu kurz greifen oder sogar kontraproduktiv wirken können. Da die Ursache für Angriffe der Variante 2 indirekte Sprünge sind, wurde die Maßnahme namens Retpoline²¹ vorgeschlagen, um solche Sprünge und Verzweigungen zu vermeiden. Dazu wurden indirekte Sprünge durch Return-Befehle ersetzt, wobei die Return-Adresse beim Aufruf der Funktion auf den Stack geschrieben wird und der Stack beim Aufruf von Return im Cache vorliegt. Dadurch wird erreicht, dass der Prozessor keine Befehle spekulativ ausführt. Da jedoch in den letzten Jahren im Zuge der ROP-Angriffsklassen die Return-Befehle durch Sprung-Befehle ersetzt wurden, um ROP-Verwundbarkeiten zu mitigieren, wurde dieser Vorschlag bereits nach wenigen Tagen wieder verworfen.

Da die Angriffsvariante 2 den BTB gezielt durch Injektionen nutzt, ist das Leeren des Caches bei jedem Kontextwechsel eine wirksame Abwehrmaßnahme. Durch Microcode-Updates für die betroffenen Prozessoren wird

²¹ <https://support.google.com/faqs/answer/7625886>

hierfür eine Lösung geliefert, die von den jeweiligen Betriebssystemen genutzt werden kann.

Variante 3 Rogue Data Cache Load

Die dritte Variante der Angriffe ist unter dem Namen Meltdown bekannt.

Konzepte der Adressraum- und Speicherisolierung zählen zu den zentralen Schutzkonzepten heutiger Betriebssysteme. Damit wird sichergestellt, dass Prozesse im User-Space nicht wechselseitig auf ihre jeweiligen Speicherbereiche zugreifen können und dass aus der Anwendungsebene, dem User-Space, kein direkter Lese- oder Schreibzugriff auf Speicherbereiche des Betriebssystem-Kerns möglich ist. Wie bereits auf Seite 84 angesprochen, wird die Isolation zwischen Kernel- und User-Space-Speicherbereichen hardwareseitig durch das Setzen eines Supervisor-Bits für die Speicherseiten des Kernels realisiert. Dadurch können in heutigen Architekturen die Speicherbereiche des Kernels in den Adressraum eines jeden Prozesses abgebildet werden, da die Zulässigkeit von Zugriffen anhand des Bits kontrolliert wird. Dies ermöglicht eine effiziente Verwaltung, da beispielsweise bei einem Syscall, der einen Wechsel aus dem User-Space in den Kernel-Space nach sich zieht, kein aufwändiger Adressraumwechsel mit dem Laden der Seitentabelle des Kernel-Bereichs erforderlich ist. Die Seiten des Kernels sind in der Seitentabelle des Anwendungsprozesses vorhanden und können nach dem Wechsel in den Kernel-Modus direkt genutzt werden.

Adressraum-organisation

Bei einem Meltdown-Angriff wird dieses Isolations- und Kontrollkonzept umgangen. Mittels eines Meltdown-Angriffs kann aus dem User-Space heraus Information aus Speicher des Kernels ausgelesen, bzw. präziser ausgedrückt, diese Information kann mittels Seitenkanalanalysen rekonstruiert werden. Für den Angriff wird keine Software-Schwachstelle ausgenutzt, sondern analog zu den zuvor beschriebenen Vorgehensweisen wird das Verhalten der Out-of-Order-Execution gängiger Prozessoren genutzt, um über Seitenkanalangriffe die geschützte Information zu extrahieren. Der Angriff basiert damit auf der gleichen Idee wie die Angriffe der Variante 1. Mit einem Meltdown-Angriff ist es möglich, den gesamten Adressbereich des Kernels in einer Folge von Out-of-Order Ausführungen auszulesen und über verdeckte Kanäle der zugrundeliegenden Microarchitektur beispielsweise mittels Flush+Reload in Bereiche zur transferieren, in denen die Registerwerte dann mittels der Seitenkanalanalysen rekonstruiert werden können. Neben Zugriffen auf den Adressbereich des Kernels kann über Meltdown-Angriffe auch auf physischen Speicher direkt zugegriffen werden, der ggf. Daten anderer Prozesse enthält. Wird der Kernel-Bereich gemeinsam genutzt, wie dies durch Sandboxing mit Docker oder auch Xen mit Paravirtualisierung oder auch durch Hypervisor-Implementierungen der Fall

Meltdown-Angriff

ist, so kann aus dem User-Space auch auf Daten anderer Prozesse, z.B. in einer Cloud-Umgebung, zugegriffen werden. Meltdown stellt somit auch ein gravierendes Problem für Cloud-Anbieter dar, besonders wenn deren Guestsysteme nicht vollständig virtualisiert sind und u.a. aus Performanzgründen mit gemeinsamen Kernel-Bereichen arbeiten.

Da beim Meltdown-Angriff jedoch kein Adressraumwechsel erforderlich ist, kann der Speicher mit hoher Datenrate von ca. 500 kb/s gelesen werden und Angriffe sind damit ca. 300 mal schneller durchführbar, als die ähnlichen Angriffe der Spectre-Variante 1.

Gegenmaßnahmen

KAISER

Ein Schutz gegen die Meltdown-Angriffe bietet das unter dem Namen KAISER [75] bekannte Konzept, das ursprünglich eingeführt wurde, um einen Schutz gegen Seitenkanalangriffe auf KASLR (Kernel Address Space Layout Randomization) zu bieten. Die naheliegende Lösungsidee von KAISER besteht darin, die Adressbereiche des Kerns und des User-Space strikt zu trennen, so dass der Kernel-Adressbereich nicht mehr im virtuellen Adressbereich eines Nutzerprozesses vorhanden ist. Bis auf wenige Teile, die von der x86 Architektur benötigt werden, wie beispielsweise Interrupt Handler, kann durch die Aufteilung von Kernel- und User-Space-Adressraum sichergestellt werden, dass während der Ausführung im User-Mode in der Hardware keine Daten aus dem Kernel-Bereich verfügbar sind. Beim Wechsel zwischen Kernel- und User-Space wird der TLB geflushed, so dass eine strikte Isolierung umgesetzt wird. KAISER erfordert eine Integration der entsprechenden Maßnahmen in die zugrundeliegenden Betriebssysteme. Unter der Bezeichnung Kernel Page-Table Isolation (KPTI) wurde das KAISER-Konzept in den Linux-Kernel 4.15 integriert und auch frühere Linux-Versionen, wie der Linux-Kernel 4.4.110, 4.9.75 und 4.14.11 wurden gepatcht. Zu beachten ist, dass diese mit KPTI einhergehende Trennung der Seitentabellen und damit der Adressräume von Kernel und User-Space kein Schutz vor Angriffen der Spectre Varianten 1 und 2 darstellt. Die zusätzliche Isolierung bringt jedoch erhebliche Performanceinbußen mit sich; Messungen schwanken zwischen 5 und bis zu 30 Prozent, abhängig von der Anzahl der auszuführenden Systemaufrufe.

Abschließend bleibt festzuhalten, dass die Meltdown und Spectre-Lücken keine Software-Schwachstellen darstellen und deshalb nicht allein durch Software-Patches zu reparieren sind, sondern zumindest Microcode-Updates erfordern. Die auf den ersten Blick einfachste Lösung wäre natürlich, bei den Prozessoren auf die Out-of-order Execution oder auf die spekulative Ausführung zu verzichten. Da dies aber mit nicht tragbaren Performanzverminderungen verbunden wäre, ist dies keine akzeptable Lösung. Für das Meltdown-Problem könnte eine physische Aufteilung des User- und Kernel-

KPTI

physische
Aufteilung

Speicherbereichs eine Lösung sein. dies würde das Setzen eines neuen Hardwarebits in einem CPU Kontrollregister erfordern, wie beispielsweise CR4. Bei einer solche physischen Aufteilung könnte der Kernel-Bereich im oberen Adressbereich und der User-Space im unteren Adressbereich abgelegt werden. Eine mögliche Zugriffsverletzung könnte dann direkt aus der virtuellen Adresse abgeleitet werden, wodurch eine sehr effiziente Kontrolle möglich ist. Eine solche Maßnahme schützt jedoch nicht vor den Spectre-Angriffen.

Als Schutzmaßnahme gegen alle drei Angriffsvarianten stehen Microcode-Updates der Prozessor-Hersteller zur Verfügung und werden mittlerweile von fast allen gängigen Betriebssystem-Herstellern über Betriebssystem- oder BIOS-Updates den Nutzern zur Verfügung gestellt. So sorgt beispielsweise die neue Funktion *Indirect Branch Prediction Barriers (ibpb)* in den Intel-Prozessoren dafür, dass eine Vorhersage für einen indirekten Sprung nicht mehr von vorher ausgeführten Anweisungen abhängig sein kann. Es bleibt abzuwarten, wie in zukünftigen Prozessorarchitekturen Fragen der IT-Sicherheit und der Performanz mit neuen Architektur-Designs einheitlich abgedeckt werden; die Spectre-Schwachstellen haben den Bedarf für neue Architektur-Ansätze klar aufgezeigt.

Microcode-
Updates

3 Internet-(Un)Sicherheit

Nach einer knappen Einführung in Abschnitt 3.1 gibt der Rest des Kapitels einen Überblick über wesentliche Sicherheitsprobleme im Zusammenhang mit den bekannten Internet-Protokollen. Abschnitt 3.2 stellt die Internet-Protokollfamilie zusammen mit dem OSI- und dem TCP/IP-Referenzmodell vor. Die für das Verständnis der Sicherheitsprobleme wichtigsten Eigenschaften der Internet-Protokolle werden erklärt, bevor deren Sicherheitsprobleme in Abschnitt 3.3 diskutiert werden. Abschnitt 3.4 beschäftigt sich mit weit verbreiteten Diensten wie DNS und WWW und erläutert Sicherheitsprobleme, die bei deren Nutzung auftreten können.

3.1 Einführung

Informationstechnologie durchdringt heutzutage in gleichem Maße die privaten wie geschäftlichen Bereiche unseres Lebens. Mit der ständig zunehmenden Vernetzung von IT-Systemen und deren Anbindung an öffentliche Netze wie das Internet gewinnen Fragen der Informationssicherheit zunehmend an Bedeutung. Sicherheitsbedrohungen ergeben sich durch das unautorisierte Lesen elektronischer Nachrichten, die sensible, personenbezogene Informationen oder Geschäftsdaten beinhalten können, durch das unautorisierte Verändern von gesendeten Daten, so dass gefälschte Informationen beim Empfänger ankommen, oder auch durch das Maskieren und Vortäuschen einer falschen Absenderidentität, so dass der Kommunikationspartner im Vertrauen auf diese Identität vertrauliche Informationen preisgibt. Aufgrund der Bedeutung des Internets konzentriert sich dieses Kapitel auf Sicherheitsfragen im Zusammenhang mit den für das Internet festgelegten Kommunikationsprotokollen – der Internet-Protokollfamilie.

Die Ursprünge des Internets gehen auf die Entwicklung des ARPA-Netzes (Advanced Research Projects Agency) zurück, das in den frühen 70er Jahren durch das amerikanische Verteidigungsministerium initiiert wurde. Die für das ARPA-Netz entwickelten Protokolle (Internet Protocol (IP), Internet Control Message Protocol (ICMP), Transmission Control Protocol (TCP)) sind die Grundlage der heutigen Internetprotokolle. Der 1.1.1983 gilt als die Geburtsstunde des Internets, da zu diesem Zeitpunkt die Migration auf TCP/IP abgeschlossen war.

Dienstprotokolle

Bereits frühzeitig wurden Netzwerkdienste entwickelt, die relativ komfortable Möglichkeiten zur Nutzung der Internet-Protokolle zur Verfügung stellten. Die entsprechenden Arbeiten erfolgten im Rahmen von Forschungsaktivitäten, die an der University of California, Berkeley, im Kontext der Entwicklung des Betriebssystems BSD-Unix durchgeführt wurden. Zu diesen Netzwerkdiensten zählten der Telnet-Dienst für Terminalverbindungen zu entfernten Rechnern, FTP für den Dateitransfer und E-Mail-Dienste. Diese Dienste konnten jedoch zunächst nur über eine textuelle Benutzungsschnittstelle bedient werden. Eine benutzerfreundlichere, grafische Benutzerschnittstelle entstand erst durch den 1991 eingeführten Internetdienst Gopher. Gopher ermöglichte es, Daten unterschiedlichen Typs (u.a. Texte, Bilder, Ton) zu verknüpfen. Den Durchbruch zur Öffnung des Internets für einen Massenbenutzerkreis und zu dessen kommerzieller Nutzung ermöglichte 1993 die Entwicklung des plattformunabhängigen, grafischen Browsers Mosaic zusammen mit der Entwicklung des World Wide Web (WWW). Die Konzepte des Web gehen auf Arbeiten am Europäischen Kernforschungsinstitut CERN zurück. Mit der Einführung des WWWs ging ein rasanter Anstieg der Anzahl der ans Internet angeschlossenen Rechner einher. Man schätzt, dass sich diese Anzahl seit 1993 etwa alle 12 bis 15 Monate verdoppelt. Im Juli 1996 umfasste das Netz noch ca. 13 Millionen angeschlossene Rechner, während am 1.9. 1999 bereits die Marke von 43.230.000 Rechnern erreicht war. Laut aktueller Internet-Statistiken¹ wurden Ende 2017 weltweit 4,156,932,140 Nutzer² gezählt, wovon allein auf Asien über 2 Milliarden Nutzer und auf Europa 7004 Millionen Nutzer entfallen.

WWW**dezentrale Verwaltung**

Da die Wurzeln der Entwicklung des Internets vorwiegend im wissenschaftlichen Bereich lagen, stand der freie Austausch von Informationen im Vordergrund, und es entstand eine sehr flexible und von einer zentralen Verwaltung weitgehend unabhängige Struktur. Das Fehlen von zentralistischen Strukturen fördert einerseits die rasante Ausweitung des Netzes durch die einfachen, nahezu unbeschränkten Möglichkeiten, weitere Rechner an das Netz anzuschließen sowie neue Softwareprodukte zu integrieren. Andererseits fehlt damit jedoch auch eine Basis für Kontrollmöglichkeiten, die vor dem Hintergrund des zunehmenden Einsatzes des Internets in vielen Bereichen des beruflichen und privaten Alltags in verstärktem Maße erforderlich werden.

Offenheit

Mit der Öffnung des Internets für eine breite Anwenderschaft von Nicht-Spezialisten, die keine oder nur rudimentäre Kenntnisse über die mit dem Internet und seinen Diensten verbundenen Gefährdungen besitzen, steigt

¹ Vgl. u.a. <http://www.internetworldstats.com/stats.htm>

² Bei einer geschätzten Bevölkerungszahl für 2018 von 7,634,758,428 Menschen.

die Anzahl der möglichen Opfer von Computerkriminalität. Hand in Hand mit der zunehmenden Nutzung des Internets für kommerzielle Belange steigt auch die Höhe des durch erfolgreiche Angriffe verursachten Schadens. Da es im Internet keine zentrale Kontrollinstanz zur Verhinderung von Missbrauch gibt, sind Kenntnisse über Sicherheitsschwachstellen und über mögliche Bedrohungen für Benutzer unabdingbar, weil sich ein großer Teil der Gefährdungen bereits durch einfache administrative Maßnahmen sowie durch den Einsatz verfügbarer Softwareprodukte vermeiden oder zumindest begrenzen lässt.

3.2 Internet-Protokollfamilie

Im Folgenden werden die Internet-Protokolle nur so weit eingeführt, wie es für das Verständnis der damit verbundenen Sicherheitsprobleme notwendig erscheint. Für vertiefende Einblicke in die Netzwerkprotokolle sei auf die reichhaltige Fachliteratur hierzu verwiesen (vgl. u.a. [172]). Die zentralen Internetprotokolle sind TCP/UDP und IP. Sie bieten unterschiedliche Dienste an, deren jeweilige Ausprägung sich an dem ISO/OSI Referenzmodell orientiert, das eine Schichtenarchitektur für Kommunikationsprotokolle definiert. Sicherheitsprobleme treten in unterschiedlicher Weise auf diesen Ebenen auf. Da im Verlauf des Buches an verschiedenen Stellen auf die Schichten des Referenzmodells Bezug genommen wird, wird im Folgenden zunächst ein knapper Überblick über dieses Referenzmodell gegeben.

3.2.1 ISO/OSI-Referenzmodell

Um Nachrichten über ein Kommunikationsmedium zwischen Sender und Empfänger verschicken zu können, müssen diese nach einem festgelegten Protokoll transformiert, übertragen und wieder zurücktransformiert werden. Das ISO/OSI-Referenzmodell (vgl. [172]) legt dafür eine standardisierte Vorgehensweise fest. Das Modell besteht aus sieben Schichten, wobei jede Schicht die Aufgabe hat, Nachrichten der darüber liegenden Schicht entgegen zu nehmen, zu bearbeiten und sie unter Nutzung der Dienste der darunter liegenden Schicht weiter zu leiten. Jede Schicht fügt ihre Steuerinformationen in Form eines Headers an die eigentliche Nachricht an. Nach der Übertragung über das physikalische Medium empfängt die Bitübertragungsschicht des Rechners B die Daten. Vor der Weitergabe in die nächst höher liegende Schicht entfernt sie die Steuerdaten, die von ihrer Partnerschicht auf Rechner A angehängt worden sind. In Abbildung 3.1 sind die sieben Schichten des ISO/OSI-Modells angegeben.

Schichtenmodell

In einem Netzwerk sind Vermittlungsrechner (Router, Gateways etc.) erforderlich, um den Datenaustausch zweier nicht direkt miteinander ver-

Router

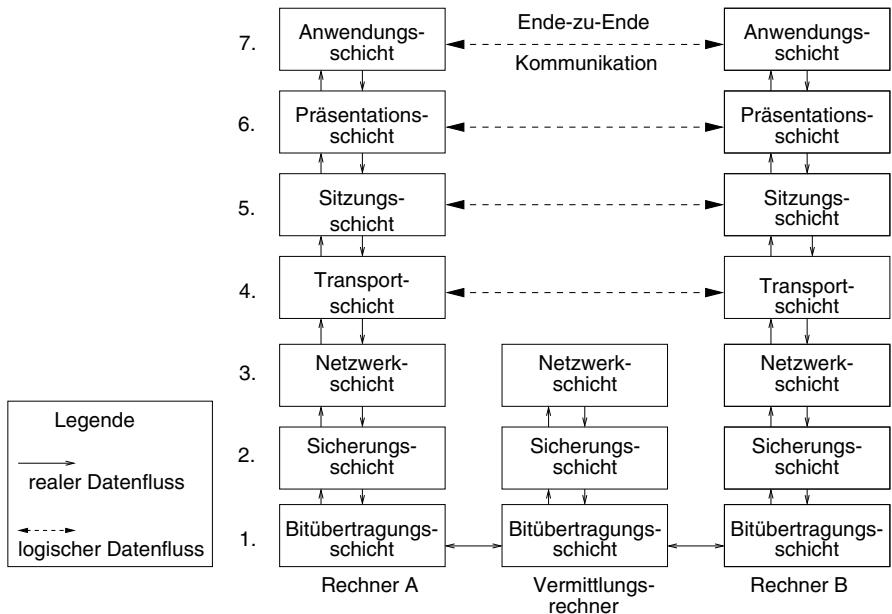


Abbildung 3.1: Die Schichten des ISO/OSI-Modells

bundenen Endsysteme zu ermöglichen. Um die Daten zwischen nichtbenachbarten Endsystemen korrekt weiterzuvermitteln, benötigen die Zwischenstationen nur die Protokolle aus den untersten drei Schichten (Schicht 1–3), dem so genannten Kommunikations-Subsystem.

Die sieben Schichten

Bitübertragung

Als unterste Schicht im OSI-Modell findet sich die physikalische Bitübertragungsschicht (engl. *physical layer*). Ihre Aufgabe besteht in der Herstellung einer physikalischen Verbindung zwischen zwei direkt verbundenen Kommunikationspunkten und in der Übertragung der Bitströme. Die physikalische Schicht legt die Charakteristika der physischen Verbindung fest wie die Spannung oder Voltzahl, die zur Repräsentation von 1 und 0 benötigt werden. Darüber hinaus wird für die Netzkomponenten die Anzahl der Pins und deren Funktion definiert, der Verbindungsaufl- und -abbau wird spezifiziert und der Umgang mit unterschiedlichen Transportmedien wie Koaxialkabel, Lichtwellenleiter oder Funkverbindungen wird geregelt.

Datensicherung

Die Sicherungsschicht (engl. *data link layer*) bündelt Bitströme der unteren Ebene zu Datenpaketen (engl. *frames*) bzw. teilt die Pakete der darüber liegenden Ebene auf Datenpakete auf und fügt Kontrollinformationen in Form von Prüfsummen hinzu, um Übertragungsfehler erkennen zu können. Die Datenframes werden sequentiell übertragen. Zu den Aufgaben dieser Schicht gehört der Einsatz fehlererkennender bzw. fehlerkorrigierender Codes (z.B.

Cyclic Redundancy Checks (CRC)) zur Erkennung und Behebung von Übertragungsfehlern sowie die Behandlung beschädigter, duplizierter oder verloren gegangener Frames. Darauf hinaus legt die Schicht Protokolle zur Regelung des Medienzugangs fest (u.a. CSMA/CD für das Ethernet) und reguliert den Nachrichtenfluss zwischen Sender und Empfänger (u.a. mittels Sliding Window-Verfahren).

Die Netzwerk- oder Vermittlungsschicht (engl. *network layer*) verknüpft und kontrolliert Teilnetze. Im Gegensatz zu den niedrigeren Ebenen, die nur eine Verbindung mit benachbarten Maschinen aufnehmen können, errichtet die Vermittlungsschicht eine Ende-zu-Ende Kommunikation zwischen Kommunikationspartnern, die nicht direkt benachbart sein müssen. Die Vermittlungsschicht bietet Protokolle an, die es ermöglichen, dass Teilnetze mit unterschiedlichen Adressierungsschemata, unterschiedlichen Paketgrößen oder auch mit heterogenen Protokollen miteinander kommunizieren können. Eine wesentliche Aufgabe besteht darin, für die zu transportierenden Pakete einen günstigen Weg auszuwählen (engl. *routing*). Mittels Routingprotokollen werden ankommende Pakete an Knotenrechner im Netz weitergeleitet. Solche Wege können über Tabellen statisch festgelegt sein, oder sie können beim Verbindungsauflauf dynamisch bestimmt bzw. für jedes übertragene Paket, abhängig von der aktuellen Netzbelauf, dynamisch neu ermittelt werden. Zu den Aufgaben dieser Schicht gehört auch, Stausituationen zu erkennen und zu beheben (engl. *congestion control*). Schließlich ist die Vermittlungsschicht auch für die Erstellung von Abrechnungen zuständig. Das heißt, dass Abrechnungsfunktionen definiert sind, die die übertragenen Paketzahlen, Bits oder Zeichen protokollieren und in Rechnung stellen.

Vermittlung

Die Transportschicht (engl. *transport layer*) übernimmt die Aufgabe, für jeden Verbindungsunsch der höheren Ebene eine Verbindung zwischen den Endsystemen zu etablieren. Abhängig vom Datenaufkommen und der Netzbelauf kann pro Verbindungsunsch eine eigene logische Verbindung aufgebaut oder zur Reduktion der Kosten können mehrere Verbindungen im Multiplexbetrieb auf einen Kanal gebündelt werden. Die Transportschicht ermöglicht eine Ende-zu-Ende Kommunikation zwischen einzelnen Prozessen bzw. Diensten auf den jeweiligen Endsystemen. Daneben realisiert sie in der Regel ein zuverlässiges, paketorientiertes Ende-zu-Ende Protokoll. Das bedeutet, dass das Protokoll die Daten der darüber liegenden Ebene auf Pakete aufteilt und sicherstellt, dass alle Pakete in der richtigen Reihenfolge beim Empfänger ankommen und jedes empfangene Paket quittiert wird. Des Weiteren hat diese Schicht die Aufgabe, eine Flusssteuerung für die Verbindungen durchzuführen. Dazu gehört die Verhinderung von Pufferüberläufen auf der Empfängerseite, indem die Senderate reguliert wird.

Transport

Sitzung

Die Sitzungsschicht (engl. *session layer*) koordiniert und synchronisiert die Kommunikation zwischen Anwendungsprozessen und trifft Vorehrungen für ein Wiederaufsetzen unterbrochener Sitzungen. Dazu integriert sie Informationen über Sicherungspunkte (engl. *checkpoint*) in den Nachrichtenstrom, so dass bei einer Verbindungsunterbrechung ein Dialog, z.B. ein komplexer Dateitransfer, bei einem dieser Sicherungspunkte wieder aufgesetzt werden kann.

Präsentation

Da in einem Rechnernetz unterschiedliche Datenformate existieren, übernimmt die Darstellungsschicht (engl. *presentation layer*) die Aufgabe, die Daten der Anwendungsschicht, also die maschinenabhängigen Repräsentationen, in ein netzeinheitliches Format umzuwandeln. Im Gegensatz zu den unteren Schichten, die vordringlich reine Transportaufgaben durchführen, ohne die Nachrichteninhalte zu interpretieren, beschäftigt sich diese Schicht mit Syntax und Semantik der übertragenen Informationen. Zur Beschreibung der hierzu eingesetzten Transfersyntax werden Sprachen definiert wie ASN.1 (Abstract Syntax Notation No 1). Weitere Aufgaben der Informationsrepräsentation betreffen die Datenkompression oder die Datenverschlüsselung, die häufig auf dieser Schicht angesiedelt werden.

Anwendung

Die Anwendungsschicht (engl. *application layer*) stellt schließlich eine Vielzahl von Protokollen zur Verfügung, die Dienste für Benutzer erbringen wie SMTP zum Versenden von Electronic Mails, FTP für den Dateitransfer bzw. Dateizugriff auf entfernte Rechner oder HTTP für den Zugriff auf Web-Objekte.

Netzwerkkomponenten

Im Folgenden werden die wesentlichen Komponenten einer Netzinfrastruktur knapp eingeführt, damit deren grundlegende Aufgaben geklärt sind. Wir weisen bereits an dieser Stelle auf einige Sicherheitsprobleme dieser Komponenten hin, auf die aber im Verlauf des Kapitels noch vertiefend eingegangen wird. Abbildung 3.2 verdeutlicht grob die Einbindung der Komponenten in heutige Netzinfrastrukturen.

Teil (a) zeigt zwei Ethernet-Segmente, an die über Transceiver-Komponenten, die den Bus (das Ethernet) abhören (Carrier Sense), Endgeräte wie PCs angeschlossen sind. Der Netzwerk-Controller der Geräte übernimmt die Aufgaben der Schichten 1 und 2 gemäß ISO/OSI.

Repeater

Die Aufgabe eines Repeaters besteht lediglich darin, in einem Netzsegment die Signalreichweite zu erhöhen, indem diese Signale verstärkt werden. Ein Repeater führt keinerlei Überprüfungen oder Isolierungen durch, sondern verbindet lediglich gleichartige Medien. Durch die Verwendung von Repeatern vergrößert sich somit auch die Kollisionsdomäne bei Broadcastmedien wie Ethernet. Gleichzeitig vergrößert sich natürlich auch das Netzsegment,

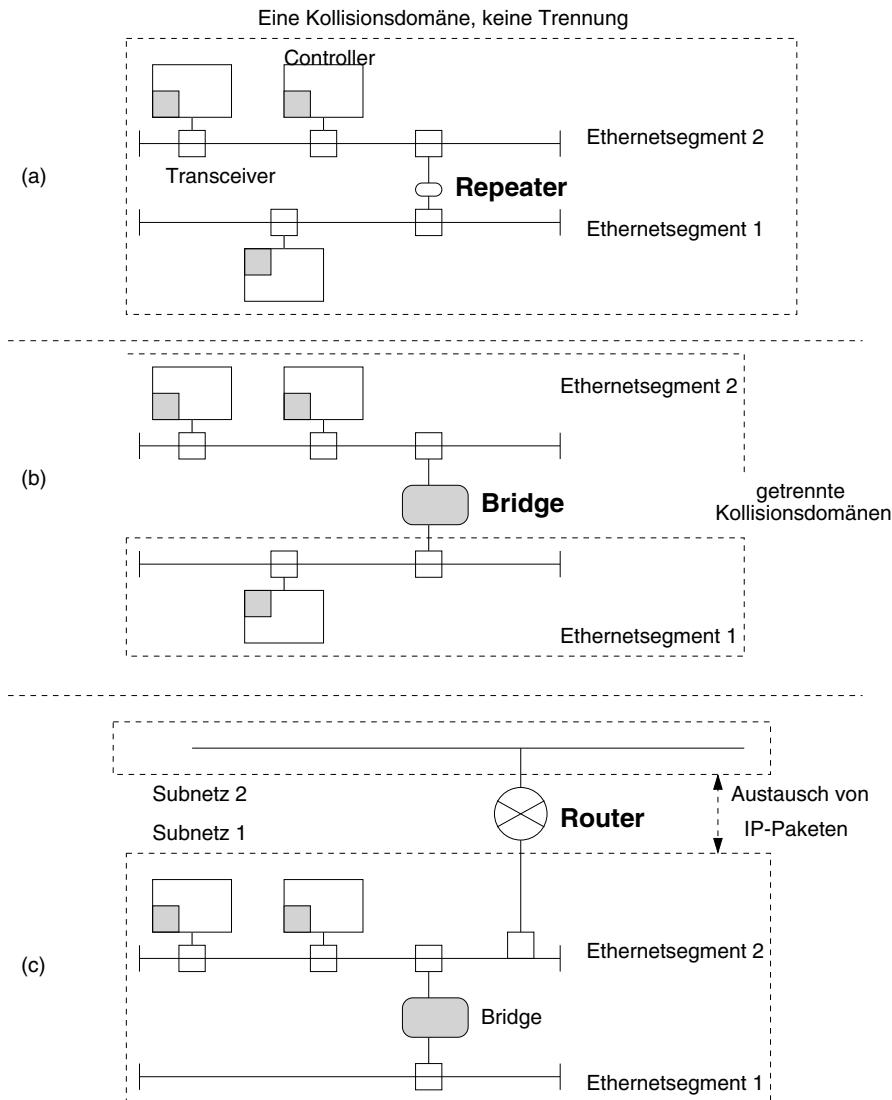


Abbildung 3.2: Netzwerkkomponenten

d.h. die Menge der potentiell an das Segment angeschlossenen Rechner, die alle in der Lage sind, die per Broadcast verteilten Datenpakete zu lesen. Ein Hub ist ein Multiportrepeater, der mehrere Netze verbindet. Er bereitet die an einem Port eingehenden Signale auf und leitet sie an alle ausgehenden aktiven Ports weiter (Broadcast). Da ein Hub lediglich eine Variante eines Repeaters ist, treffen die Repeater-Eigenschaften auch hier zu. Auch ein Hub übernimmt keinerlei Kontrollfunktionen. Das Abhören (Sniffen) von Datenverkehr in einem Netzsegment mit Repeatern bzw. Hubs ist

Hub

sehr einfach möglich. Wird beispielsweise die Ethernet-Karte eines an das Segment angeschlossenen Rechners im Promiscous-Mode konfiguriert, so kann dieser Rechner sämtliche Datenpakete mitlesen und mit frei verfügbaren Programmen wie *tcpdump* oder *ngrep* aufzeichnen und analysieren. Dieses Konfigurieren erfordert zwar Administrator-Berechtigungen, aber da Nutzer eines PCs darüber häufig verfügen, ist dies keine wirkliche Hürde.

Bridge

Teil (b) der Abbildung verdeutlicht die Wirkungsweise einer Bridge. Eine Bridge verbindet Netze, z.B. Ethernet-Segmente auf der Data-Link Layer. Zu ihren Aufgaben gehört die Weiterleitung von Datenframes an MAC-Adressen (Medium Access), wozu eine Bridge eine Tabelle mit Informationen über die ihr bekannten MAC-Adressen verwaltet. MAC-Adressen sind physikalische Adressen von Geräten (vgl. Seite 109). Im Gegensatz zu einem Hub oder Repeater ist eine Bridge in der Lage, unterschiedliche Medien zu verbinden, indem sie in Form eines einfachen Gateways die unterschiedlichen Protokolle aufeinander abbildet. Eine Bridge trennt zwar die Kollisionsdomänen der angeschlossenen Netzsegmente, aber ihre sonstigen Kontroll- und Isolierungsfunktionen sind sehr begrenzt. Insbesondere leitet sie auch Broadcast-Anfragen (z.B. die Anfrage nach einer MAC-Adresse zu einer IP-Adresse) in die angeschlossenen Netzsegmente weiter, so dass ein Angriff, der versucht, über einen Broadcast möglichst viele Rechner zu erreichen, von einer Bridge nicht wirkungsvoll gestoppt wird. Einen Access Point in einem drahtlosen lokalen Netz wie einem WLAN nach dem Standard IEEE 802.11 (vgl. Seite 891) kann man als eine spezielle Bridge auffassen.

Access Point

Switch

Ein Switch ist eine Multiport Bridge, d.h. analog zum Hub verbindet ein Switch verschiedene Netzsegmente auf der Data-Link Layer. Switches filtern den Datenverkehr nach MAC-Adressen, indem sie wie eine Bridge auch, eine Tabelle, den so genannten ARP Cache, verwalten, die eine Zuordnung von IP-Adresse und MAC-Adresse jedes Rechners enthält, der an das geswitchte Netz angeschlossen ist. In einem solchen Netz werden also Datenpakete nicht generell per Broadcast an alle Rechner verteilt, sondern der Switch leitet ein Datenpaket gezielt an einen Rechner weiter. Vordringliches Ziel bei der Einführung geswitchter Netze war die Erhöhung der Bandbreite durch die Reduktion der Anzahl der Broadcast-Nachrichten, nicht jedoch auch die Erhöhung der Sicherheit. Das Sniffen des Datenverkehrs in einem „geswitchten“ Netz ist deshalb zwar schwieriger als in einem Segment ohne Switches, aber keinesfalls unmöglich. Ein möglicher Ansatzpunkt hier ist es, durch Maskierungsangriffe auf den ARP-Cache des Switches (siehe Seite 127) dafür zu sorgen, dass alle Datenpakete, die für einen Rechner X (vorzugsweise ist das sogar ein Gateway-Rechner (siehe unten)) bestimmt sind, über den Angreiferrechner geleitet werden. Dazu muss der Angreifer nur eine gefälschte ARP-Nachricht an den Switch senden und ihm mitteilen, dass sich die MAC-Adresse des Rechners X geändert hat und dem Switch

als neue MAC-Adresse die des Angreifers mitteilen (bzw. bei einem ARP-Broadcast mit der Anfrage nach der MAC-Adresse zu der IP-Adresse des Rechners X mit der falschen MAC-Adresse antworten). Wenn der Angreifer die umgeleiteten Daten seinerseits mittels IP-Forwarding an den eigentlichen Empfänger weiterleitet, kann dieser Angriff unbemerkt durchgeführt werden. Frei verfügbare Programme wie *ettercap* für Linux und Windows (vgl. <http://ettercap.sourceforge.net/>) ermöglichen derartige Angriffe. Ein Angreifer kann aber auch einfach die MAC-Adresse eines Rechners fälschen, so dass der Switch nicht nur dem eigentlichen Empfänger, sondern auch dem Angreifer die Daten zusendet.

Eine Verbindung von mehreren Netzen auf der Schicht 3 erfolgt durch Router-Komponenten. Teil (c) der Abbildung veranschaulicht die Einordnung eines Routers in eine Netzinfrastruktur. Eine Hauptaufgabe eines Routers ist es, die Wegewahl durchzuführen und Pakete der Ebene 3, also im Internet-Kontext die IP-Pakete, zu den Empfängern weiterzuleiten. Im Gegensatz zu den zuvor besprochenen Komponenten können Router bereits einige Kontroll- und Filterfunktionen wahrnehmen, um den Verkehr zwischen den Netzen zu überwachen und zu reglementieren. Einfache Paketfilter-Firewalls (vgl. Abschnitt 14.1.2) können durch Router realisiert werden.

Um Netze unterschiedlicher Typen zu verbinden, benötigt man komplexere Komponenten, die eine vollständige Protokollangleichung vornehmen. Derartige Komponenten nennt man Gateways.

3.2.2 Das TCP/IP-Referenzmodell

Das Internet ist ein Weitverkehrsnetz, das Geräte verbindet, die über die Internet-Protokollfamilie miteinander kommunizieren. Die Geräte umfassen neben PCs, Notebooks oder Smartphones auch die oben skizzierten Vermittlungskomponenten wie Router und Gateways. Der Datenaustausch erfolgt paketvermittelt, das heißt, dass die Nachrichten in Datenpakete zerlegt, zu den Empfängergeräten weitergeleitet und dort wieder zu einer Nachricht zusammengesetzt werden. Die Weiterleitung von Datenpaketen ist die Aufgabe des Vermittlungsnetzwerkes, das aus Routern und Switches der Internet-Zugangsanbieter (engl. *Internet Access Provider* oder *Internet Service Provider, ISP*) besteht. Der Anschluss an das Vermittlungsnetz kann auf unterschiedliche Weise erfolgen, etwa durch ISDN-Leitungen, Standleitungen, herkömmliche Telefonleitungen oder auch über Funk-Netze (Wireless LANs).

Den Kern der Internet-Protokollfamilie bilden das Internet-Protokoll (IP) mit dem ICMP- und ARP-Protokoll sowie die Transportprotokolle TCP und

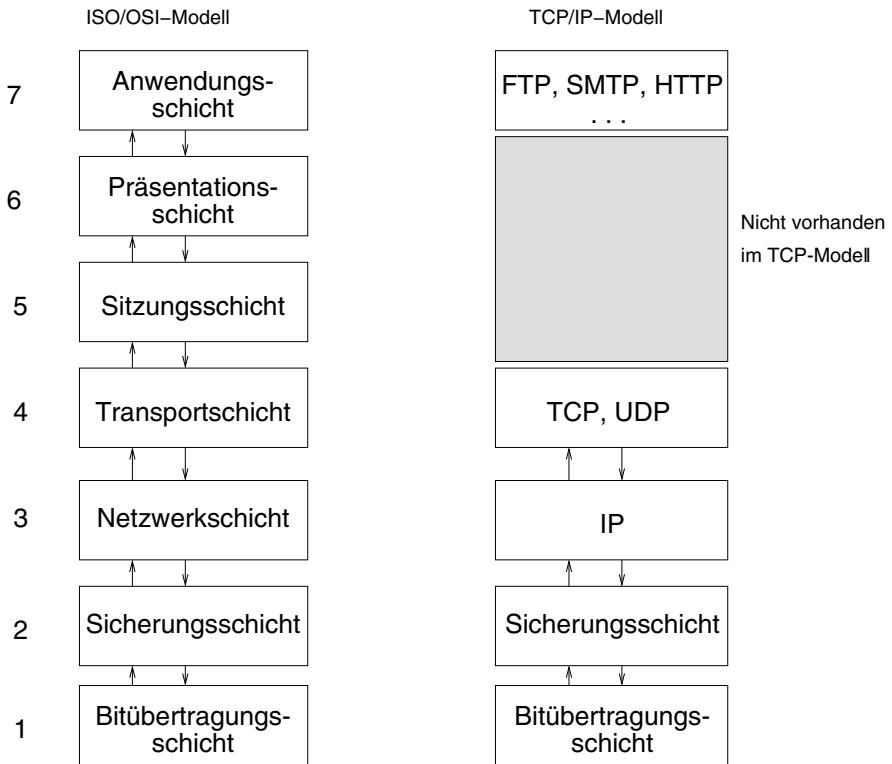


Abbildung 3.3: OSI- versus TCP/IP-Modell (vereinfachte Darstellung)

UDP. Die Internet-Protokolle übernehmen die Aufgaben der Netzwerk- und der Transportschicht des ISO/OSI-Modells. Der stark vergrößerte Zusammenhang zwischen dem TCP/IP-Modell und dem ISO/OSI-Referenzmodell ist in Abbildung 3.3 beschrieben. Die Abbildung verdeutlicht, dass die Anwendungsdienste im TCP/IP-Modell direkt auf der Transportebene, also dem TCP- bzw. UDP-Protokoll aufsetzen. Das heißt, dass Sicherheitsprobleme der TCP/IP-Protokolle eine unmittelbare Auswirkung auf diese Anwendungsprotokolle haben, da keine ausgleichende Zwischenschicht, die zum Beispiel Verschlüsselungsaufgaben durchführt, existiert.

Das TCP/IP-Modell definiert keine Protokolle für die Schichten 2 und 1. Die zwei am häufigsten im Internet-Umfeld eingesetzten Protokolle der Sicherungsschicht sind das SLIP- und das PPP-Protokoll. SLIP (Serial Line Internet Protocol) führt keine Fehlerkorrektur oder Authentifikation der Kommunikationspartner durch, und die Adressen der Partner müssen a priori bekannt sein. Aufgrund dieser Eigenschaften ist das SLIP-Protokoll für den Einsatz in heutigen und zukünftigen Internet-Umgebungen wenig geeignet. Die mangelnde Authentifikation ist zwar für die ursprünglichen Anwendungsbereiche von SLIP, nämlich die Kommunikation über gemietete

Standleitungen, akzeptabel. Dies gilt jedoch nicht mehr bei dynamisch vermittelten Verbindungen.

Das PPP (Point-to-Point Protocol) ist ein sehr häufig eingesetztes Protokoll, um entfernte Zugriffe abzuwickeln (z.B. eine Modemverbindung eines PCs zu einem Router eines Internet-Service Providers) oder ein lokales Netz an ein WAN (Wide Area Network), z.B. das Internet, anzubinden. PPP ermöglicht es, über eine Leitung die Pakete von verschiedenen Netzwerkprotokollen zu übertragen. Physikalische Verbindungen (link) werden mittels des LCP (Link Control Protocol) auf- und abgebaut sowie verwaltet. In der Verbindungsaufbauphase werden die später zu verwendenden Verfahren zur Authentifizierung der Benutzer, zur Verschlüsselung sowie zur Kompression ausgewählt. Die meisten Implementierungen von PPP unterstützen verschiedene Authentifikationsverfahren. Typische hierbei verwendete Protokolle sind das PAP (Password Authentication Protocol), bei dem Benutzerkennung und Passwort im Klartext übertragen werden, und das CHAP (Challenge Handshake Authentication Protocol), bei dem demgegenüber das Passwort nicht im Klartext, sondern nur gehasht übertragen wird, so dass dieses Verfahren eine deutlich höhere Sicherheit als PAP aufweist. Ferner bietet PPP Dienste, um die Parameter des zu verwendenden Protokolls der Schicht 3 auszuhandeln. Häufig handelt es sich hierbei um das Internet-Protokoll (IP), so dass beim Aufbau der PPP-Verbindung die Internet-Adressen (siehe Seite 108) der Partner ausgetauscht werden müssen. Wählt man sich beispielsweise mit dem heimischen PC über seinen Internet-Service-Provider (ISP) in das Internet ein, so weist der Provider dem PC eine IP-Adresse für die Dauer der Verbindung zu. Beendet der Benutzer die Verbindung, so wird seine temporäre IP-Adresse an den ISP zurückgegeben und dieser kann die Adresse für einen anderen Kunden wiederverwenden.

PPP

PAP, CHAP

3.2.3 Das Internet-Protokoll IP

Das Internet-Protokoll (siehe RFC791) (IP) übernimmt die Aufgaben der Netzwerkschicht. Es stellt einen paketvermittelnden, verbindungslosen Dienst zur Verfügung, der den Erhalt von Paketen nicht bestätigt. Man spricht von einem unzuverlässigen Dienst. Jedes Datenpaket wird zu seinem Zielsystem geleitet (routing), wobei ggf. mehrere Vermittlungsrechner durchlaufen werden. Die maximale IP-Paketlänge beim IPv4 beträgt 65.535 Bytes. Je nach Netzwerkinfrastruktur kann es allerdings notwendig sein, diese IP-Pakete nochmals in kleinere Fragmente zu unterteilen. Im klassischen Ethernet zum Beispiel können nur Pakete mit einer maximalen Größe von 1.518 Bytes übertragen werden. Jedes dieser Fragmente besitzt seinerseits einen IP-Kopf (header) und einen Teil der Daten des ursprünglichen Pakets. Fragmente können auf unterschiedlichen Wegen zum Empfänger-

unzuverlässiges
Protokoll

knoten geleitet werden. Das IP-Protokoll führt keine Maßnahmen durch, die gewährleisten, dass die gesendeten Pakete bzw. Fragmente vollständig, in der gesendeten Reihenfolge oder überhaupt beim Empfänger ankommen.

IPv4

Im Folgenden gehen wir auf die wichtigsten Aspekte des IPv4 Protokolls ein. In Abschnitt 3.2.3 diskutieren wir dann noch kurz die aus Sicherheits-Sicht wichtigsten Änderung beim Übergang zu IPv6.

Adressraum

Ein bereits seit langer Zeit bekanntes Problem bei IPv4 ist der kleine Adressraum der Internetadressen. Durch die rasante Entwicklung des Internets und der Internetfähigen Endgeräte, die alle eine eindeutige IP-Adresse benötigen, um zu kommunizieren, wurde sehr viel schneller als ursprünglich erwartet, die Adressraum ausgeschöpft. Die Vergabe der IP-Adressen ist weltweit wie folgt geregelt. Derzeit gibt es fünf aktive regionale Register, die Regional Internet Registries (RIR), die für die regionale Verwaltung und Zuteilung von IP-Adressen verantwortlich sind. Die 5 Regionen sind Europa, Amerika, Asia-Pazifik, Latein-Amerika und Karibik, sowie Afrika. Jede RIR erhält durch die Internet Assigned Numbers Authority (IANA) eindeutige IP-Addressbereiche zur Verwaltung zugeteilt. Die RIRs geben ihrerseits die ihnen zugewiesenen IP-Adressen blockweise an Local Internet Registries (LIR) weiter. Diese sind in der Regel Internet Service Provider (ISP), von denen dann schließlich die Endkunden ihrer IP-Adresse erhalten.

Am 3.2.2011 hat die IANA den letzten freien Block an IP Adressen dem RIG Asia Pacific Network Information Centre (APNIC) zugewiesen. Am 14. 9. 2012 folgte die letzte Zuteilung freier Adressen in der Region Europa/Naher Osten. Damit steht der seit langem erwartete Zeitpunkt, dass keine freien Internetadressen mehr verfügbar sind, unmittelbar bevor. Diese Situation macht nunmehr die Einführung von IPv6 unumgänglich.

IPv6 wird bereits seit 1998 entwickelt (vgl. RFC 2460), es ist also keineswegs eine ad-hoc Lösung, die nun zum Einsatz kommen wird. Die Frage der Sicherheit spielt beim Übergang von IPv4 zu IPv6 basierten Netzen eine sehr wichtige Rolle. Obwohl die Spezifikation von IPv6 schon seit vielen Jahren bekannt ist, wurden bislang nur sehr wenige Testimplementierungen durchgeführt; Erfahrungen mit der Sicherheit von IPv6 im operativen Betrieb gibt es, anders als mit IPv4, kaum. Wir gehen weiter unten (vgl. Seite 111) auf einige der mit IPv6 verbundenen Sicherheitsimplikationen genauer ein.

IP-Adressen

IP-Adresse

Da das IP-Protokoll paketvermittelnd und verbindungslos ist, wird jedes Datenpaket als eine eigenständige Einheit übertragen. Das bedeutet, dass jedes IP-Paket Informationen über den Sender und Empfänger enthalten muss, damit eine korrekte Vermittlung erfolgen kann. Auf der IP-Ebene handelt es sich bei Sendern und Empfängern um Rechner, die über IP-

Adressen identifiziert werden. Für jede Schnittstelle eines Rechners zum Netzwerk besitzt der Rechner eine eindeutige 32-Bit Internet-Adresse, die in der Regel durch vier Dezimalzahlen angegeben wird (z.B. 141.12.62.120).

Da 32-Bit Zahlen als Adressen auf der Anwendungsebene nur schwer handhabbar sind, besitzt jeder Rechner darüber hinaus einen Namen, der sich aus dem Rechnernamen gefolgt von dem so genannten Domänennamen zusammensetzt. Zur Abbildung von Rechnernamen auf IP-Adressen werden auf der Anwendungsebene der Domain Name Service (DNS) oder auch der Network Information Service (NIS) zur Verfügung gestellt. Auf Sicherheitsprobleme im Zusammenhang mit diesen bekannten Protokollen geht Abschnitt 3.4 ein.

Die IP-Adressen sind logische Adressen, auf deren Basis flexible Adressierungsstrukturen aufgebaut werden können. Da die auf den unteren Modellebenen (Schichten 1 und 2) eingesetzten Geräte und Treiber aber nicht in der Lage sind, logische Adressen zu interpretieren, ist jedem Rechner, genauer jedem Netzwerk-Interface, auch eine physikalische Adresse, die MAC-Adresse³, zugeordnet. Im Normalfall erhält jedes Netzwerk-Interface vom Hersteller eine eindeutige MAC-Adresse zugeordnet. Eine solche MAC-Adresse ist 48 Bit lang. Die ersten 24 Bit enthalten die eindeutige Kennung des Herstellers, während die zweiten 24 Bit eine fortlaufende Nummer sind. Da MAC-Adressen fälschbar sind, kann es sein, dass doch mehrere Geräte die gleiche MAC besitzen, was aber im Konzept eigentlich nicht vorgesehen war. Die Abbildung der IP-Adressen auf die physikalischen Adressen erfolgt durch das Address Resolution Protocol (ARP), das ein Bestandteil des Internet-Protokolls ist. Auf das ARP und seine Sicherheitsprobleme wird in Abschnitt 3.3.3 eingegangen.

Domänenname

physikalische
Adresse

Aufbau eines IP-Pakets

Ein IP-Paket besteht aus unterschiedlichen Feldern, wobei man grob zwischen dem Header und dem Datenteil, der die Nutzdaten (engl. *payload*) enthält, unterscheidet. Abbildung 3.4 zeigt den Aufbau eines IP-Pakets.

IP-Paket

Im Folgenden werden nur die grau unterlegten Felder kurz beschrieben, da diese die häufigsten Ansatzpunkte für IP-Angriffe darstellen.

- *Identifikation, Flags, Fragment-Offset:* Wird ein IP-Paket in kleinere IP-Fragmente zerlegt, so müssen die Informationen, die zum Zusammensetzen des ursprünglichen Pakets benötigt werden, in diesen drei Feldern gespeichert sein. Das fragmentierte Paket ist mit dem nicht fragmentierten Paket vom Aufbau der Felder her vollkommen identisch. Das Feld *Identifikation* ist für die Nummerierung der Frag-

Fragmentierung

³ Medium Access Control

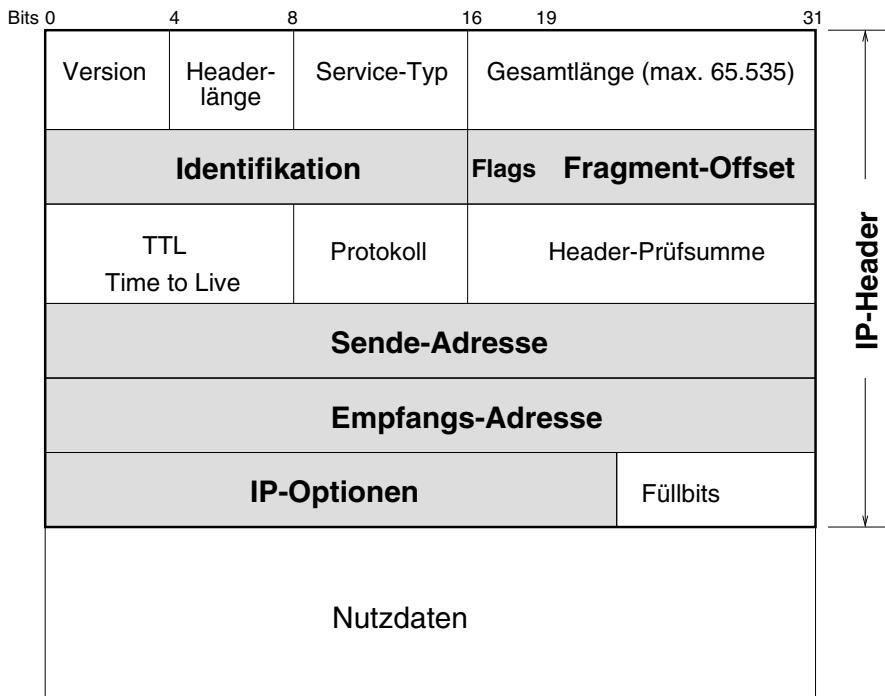


Abbildung 3.4: Format eines IP-Pakets

mente zuständig. Über das Feld *Flags* wird festgelegt, ob eine Fragmentierung erlaubt ist, und ob eventuell noch ein weiteres Fragment folgt. Der *Fragment Offset* liefert die für die korrekte Zusammensetzung der Paketfragmente erforderlichen Angaben über die Position der Daten des jeweiligen Datagramm-Fragments innerhalb des Original-Datagramms.

Adressen

- *Sende-Adresse, Empfangs-Adresse:* Das Feld *Sende-Adresse* gibt die IP-Adresse des sendenden Rechners und die *Empfangs-Adresse* gibt die des Empfängers an. Mithilfe der IP-Adressen ist es Vermittlungsrechnern möglich, IP-Pakete korrekt an die jeweiligen Endsysteme weiterzuleiten.

Optionen

- *IP-Optionen:* Durch die Angabe eines Optionsfeldes können IP-Pakete zusätzliche Funktionen erfüllen. Diese Möglichkeiten werden in der Regel von Systemadministratoren eingesetzt, um Verbindungen zu testen oder zu kontrollieren. Die für die spätere Sicherheitsdiskussion wichtigen Optionen sind das Loose Source Routing (Optionsnummer 3) und das Strict Source Routing (Optionsnummer 9). Über diese beiden Optionen kann der Weg, den das IP-Paket bis zu seinem Empfänger durchlaufen soll, bestimmt werden.

IPv6

Wie bereits weiter oben angesprochen ist der Adressraum von 32-Bit Adressen von IPv4 zu klein; die knapp über 4 Milliarden IP Adressen sind nahezu ausgeschöpft. IPv6 unterstützt deshalb 128 Bit lange IP Adressen. Mit einem Adressraum von 2^{128} unterschiedlichen Adressen vergrößert sich der verfügbare Adressraum um den Faktor 2^{96} ; es stehen damit auf absehbare Zeit genügend Adressen zur Verfügung, um auch im Internet der Dinge alle noch so winzigen smarten Objekte mit einer eigenen IP-Adresse auszustatten; wenn man das denn überhaupt möchte.

Die deutliche Verlängerung der IPv6 Adressen erfordert eine andere Notation (vgl. RFC 4291 für Details der Notation) als sie von IPv4 bekannt ist. Eine 128 Bit IPv6-Adresse wird in hexadezimaler Zahldarstellung notiert, während für IPv4 ja eine dezimale Darstellung verwendet wird. Eine IPv6 Adresse besteht aus acht Blöcken à 16 Bit, also 4 Hexadezimalstellen, wobei die Blöcke durch Doppelpunkte getrennt werden. Um das Zusammenspiel zwischen IPv4 und IPv6 zu erleichtern, dürfen die letzten 4 Byte der Adresse in dezimaler Notation angegeben werden.

Der Adressraum ist in der Regel so aufgeteilt, dass die ersten maximal 32 Bit den Internet Service Providern (ISP) von einer RIR zugewiesen werden. Die ISPs können die zugewiesenen Bereiche ihrerseits in Subnetze aufteilen. Die Größe der Subnetze ist vom Provider frei bestimbar, solange eine Mindestgröße nicht unterschritten wird. Diese Mindestgröße ist ein /64 Netz. Einem solchen Subnetz wird ein Präfix zugewiesen, der in der Regel 64 Bit lang ist. Die restlichen 64 Bit der Adresse definieren den sogenannten Interface Identifier. Falls kein DHCP Dienst vorhanden ist und der Stateless Autokonfiguration-Dienst verwendet wird, wird dieser Identifier aus der MAC Adresse der Netzwerkkarte erstellt (vgl. RFC 2462). Dieser Identifier charakterisiert eindeutig eine Netzwerkstelle. Da eine solche Schnittstelle über mehrere IP-Adressen erreichbar sein kann, ist es möglich, dass der Identifier in unterschiedlichen IP-Adressen auftritt, also mit unterschiedlichen Präfixen. Dies unterstützt beispielsweise ein Multihoming, da die Präfixe auch zu verschiedenen Providern gehören können.

Der Header besitzt in IPv6 eine feste Länge von 40 Byte und kann durch so genannte Extension Header erweitert werden. Diese befinden sich zwischen Header und Nutzdatenteil. Da die Erzeugung des Identifiers aus der global eindeutigen MAC Adresse der Netzwerkkarte direkt zu datenschutzrechtlichen Problemen führt, weil eine Profilbildung möglich ist, wurden die Privacy Extensions eingeführt. Mit der Privacy Extension (vgl. RFC 4941) wird sichergestellt, dass der Identifier zufällig generiert und regelmäßig gewechselt wird. Um eine Profilbildung auszuschließen ist zudem wünschenswert, wenn auch die Präfixe regelmäßig gewechselt würden, da

Notation

Interface Identifier

Privacy Extension

eine statische Zuteilung idR mit einem Eintrag in der öffentlichen *Whois* Datenbank verbunden ist.

ICMPv6

Unter IPv6 wird das aus IPv4 bekannte Address Resolution Protocol (ARP) durch das Neighbor Discovery Protocol (NDP) ersetzt, das wiederum stark auf Nachrichtenformate und den damit verbundenen Diensten des ICMPv6, also einer Weiterentwicklung des Internet Control Message Protokolls, zurückgreift. Während ICMP im IPv4 im Wesentlichen dafür verwendet wird, Status- und Fehlermeldungen zu versenden, bietet ICMPv6 eine ganze Reihe weiterer Funktionen, die von IPv6 auch genutzt werden. Der RFC 4890 schreibt die Nutzung von ICMPv6 in IPv6 zwingend vor, so dass ein Blokken von ICMPv6 durch Firewalls dazu führt, dass die Funktionalität von IPv6 stark eingeschränkt wird.

NDP

Das Neighbor Discovery Protocol kann man nun als eine Kombination aus ARP, ICMP Router Discovery und ICMP Redirect Protokollen und Nachrichten auffassen. Unter Nutzung des Nachrichtenformats Router Advertisement Message von ICMPv6 kann ein Router, wie üblich, seine Anwesenheit im Netzwerk bekannt geben, aber zusätzlich auch seine Nicht-Verfügbarkeit. Mit der ICMPv6-Redirect Nachricht kann ein Router einen anderen Rechner über eine bessere Verbindung zum Ziel informieren (besser first-hop node) oder anzeigen, dass ein bestimmter Rechner tatsächlich ein Nachbarknoten in einem lokalen Subnetz ist, so dass das Paket nicht unbedingt über den Router versandt werden muss. Um die Erreichbarkeit eines Knotens zu erfragen oder um die Link-Layer Adresse eines Knotens in Erfahrung zu bringen, kann die ICMPv6 Neighbor Solicitation Message genutzt werden. Als Antwort auf die Anfrage sendet der angefragte Rechner eine ICMPv6 Neighbor Advertisement Nachricht. Das ICMPv6 unterstützt einen Dienst, der als Stateless Autoconfiguration bezeichnet wird. Über diesen Dienst kann ein Host eine IPv6 Adresse beziehen, auch ohne dass ein DHCP Server im lokalen Netz verfügbar ist.

Wie bei IPv4 werden auch bei IPv6 mittels DNS (siehe Abschnitt 3.4.1) Namen auf IP-Adressen abgebildet bzw. mittels eines Reverse-Lookup die IP-Adresse zu einem DNS-Namen aufgelöst.

Caches

Aus Sicherheitssicht wichtig ist, dass das NDP Protokoll analog zum ARP auch Caches verwaltet. Diese sind der Neighbor Cache, der die Information zu den Nachbarknoten, mit denen kommuniziert wird, verwaltet und der Destination Cache, der im Wesentlichen die Ziel-IP-Adresse auf die Next-Hop-Adresse abbildet.

Mischbetrieb

IPv4 und IPv6 lassen sich parallel betreiben, so dass eine schrittweise Migration von IPv4 zu IPv6 möglich ist. Dazu muss beispielsweise auf den beteiligten Rechnern ein Dual-Stack Ansatz konfiguriert sein und das unterliegende Betriebssystem muss beide Protokollvarianten beherrschen. Beim

Dual-Stack Ansatz wird jeder Schnittstelle zusätzlich zur IPv4 Adresse noch eine IPv6 Adresse zugewiesen, so dass beide Protokolle unabhängig voneinander für die Kommunikation genutzt werden können. Eine Alternative besteht darin, IPv6 Pakete als Nutzlast in ein IPv4 Paket zu verpacken und durch ein IPv4 Netz zu einem IPv6 Netz weiterzuleiten, bzw. umgekehrt.

Der Sicherheitsstandard IPSec (siehe Kapitel 14.3) ist ursprünglich im Zusammenhang mit IPv6 entwickelt worden. In IPv6 ist die Unterstützung von IPSec zwar zwingend vorgeschrieben, dessen Nutzung bleibt jedoch den Netzadministratoren überlassen. IPSec bietet mit den beiden Protokollen AH (Authentication Header) und ESP Encapsulating Security Payload die Möglichkeit, IP-Pakete zu verschlüsseln (ESP) und die Absenderauthentizität des Pakets über einen MAC-Code (AH) zu prüfen. Zur Unterstützung von IPSec verwendet IPv6 IPSec entsprechende Extension Header, also einen Authentication Header und einen ESP Header.

IPSec

3.2.4 Das Transmission Control Protokoll TCP

Das Transmission Control Protocol (TCP) ist ein zuverlässiges, verbindungsorientiertes Protokoll der Transportebene, das auf dem IP-Protokoll aufsetzt. Es arbeitet aus der Sicht der darauf aufsetzenden Anwendungsschicht datenstromorientiert, wobei es den kontinuierlichen Datenstrom, den es von den Anwendungen erhält, in Datenpakete umwandelt. Eine Anwendung veranlasst das TCP-Protokoll zum Aufbau einer logischen Verbindung zwischen zwei Endsystemen und übergibt datenstromorientiert die Daten an die TCP-Ebene. Bei der Umwandlung des Datenstroms in TCP-Pakete fügt das TCP-Protokoll seinen Header an die Pakete an und leitet sie an die IP-Schicht weiter. Nach dem Transport der Daten zum Zielrechner wird geprüft, ob die Datenpakete in der korrekten Reihenfolge eingetroffen sind. Die Kontrolle der Reihenfolge der Pakete erfolgt über die Vergabe von Sequenznummern. Der Empfänger bestätigt dem Absender das Eintreffen von Paketen durch ein Acknowledgement. Im Fehlerfall werden Pakete erneut übertragen.

zuverlässiges
Protokoll

Acknowledgement

Port-Konzept

Das TCP-Protokoll kann Ende-zu-Ende-Verbindungen nicht nur zwischen zwei Endsystemen, sondern auch zwischen einzelnen Prozessen und Diensten herstellen. Um diese unterschiedlichen Dienste bzw. Kommunikationsendpunkte identifizieren zu können, wird das Port-Konzept verwendet. Ein Port ist eine 16-Bit Adresse. Dienste, zu denen eine Verbindung aufgebaut werden soll, werden an einen Port gebunden. Mit einem Dienst ist ein Dienstprozess (z.B. ein so genannter Dämonenprozess (*daemon*) unter Unix) verbunden, der die Aufgabe hat, an dem assoziierten Port auf eintreffende Nachrichten zu lauschen und diese zu bearbeiten. Auf der Anwendungsebe-

Port

ne stehen Dienstprogramme, wie der Portmapper unter Unix, zur Verfügung, die die Port-Adressen zuteilen.

well-known ports

Jeder allgemein bekannte Netzdienst besitzt eine eindeutige, ihm zugeordnete Port-Nummer, über die die Verbindung zwischen Endsystemen aufgebaut wird. In der Tabelle 3.1 sind einige der bekannten Port-Belegungen festgehalten. Die Port-Adressen ≤ 256 sind für Standarddienste reserviert; man bezeichnet sie auch als well-known Ports.

Port	Protokoll	Dienst
21	TCP	ftp
25	TCP	smtp
79	TCP	finger
80	TCP	http
88	UDP	kerberos
513	TCP	rlogin
514	TCP	shell
2049	UDP	NFS

Tabelle 3.1: Beispiele für Port-Belegungen

trusted ports

Im Kontext des Betriebssystems Unix werden die Ports in dem Bereich $\{0, \dots, 1023\}$ auch als vertrauenswürdige Ports (trusted ports) bezeichnet. Unter Unix dürfen nur Programme, die mit Superuser-Privilegien ausgeführt werden, Nachrichten von solchen Ports empfangen und Dienste an diese Ports binden. Auf diese Weise soll verhindert werden, dass ein beliebiger Benutzer Zugriff auf privilegierte Informationen erhält. So könnte zum Beispiel ein Angreifer am Port 513, an den der rlogin-Dienst⁴ gebunden wird, lauschen und sich Informationen über Passwörter verschaffen, die im Zuge des Aufbaus entfernter rlogin-Verbindungen an diesen Port übermittelt werden. Zu beachten ist, dass es keinen Internetstandard gibt, der die Vertrauenswürdigkeit der Ports festschreibt. Ein PC mit einer Ethernet-Karte kann somit beliebig TCP-Pakete von Ports mit niedrigen Nummern absenden und sich damit als vertrauenswürdigen Dienst ausgeben.

TCP-Verbindungsaufbau

Verbindung

Die TCP-Schicht unterstützt den Aufbau von logischen Verbindungen zwischen Ports auf den beteiligten Endsystemen. Jedes TCP-Paket enthält

⁴ remote Login

deshalb die 16-Bit Port-Adresse des Sende- und die des Empfangsports. Damit die korrekte Reihenfolge und zuverlässige Übertragung von Paketen kontrolliert werden kann, enthält jedes Paket darüber hinaus eine 32-Bit Sequenznummer, die gemäß der TCP-Spezifikation bei einem Verbindungsauftbau mit einem zufälligen, jedoch nicht zu großen, Wert initialisiert und dann fortlaufend inkrementiert wird. Der Empfänger bestätigt den Eingang eines Paketes durch das Zurücksenden eines eigenen Paketes, das eine Acknowledgementnummer enthält. Die Acknowledgementnummer ist die um 1 erhöhte Sequenznummer desjenigen Bytes, das der Empfänger als letztes erfolgreich empfangen hat.

3-Phasen Handshake

Der TCP-Verbindungsauftbau zwischen zwei Maschinen A und B wird über ein 3-Phasen Handshake Protokoll abgewickelt. In der ersten Phase erfolgt ein *Connection Request* von einem Prozess (Client) auf Rechner A. Dazu sendet der Client C ein Paket, in dem er unter anderem die Empfänger-IP-Adresse und den Empfänger-Port, mit dem er verbunden werden will, angibt. Weiterhin generiert er eine Sequenznummer Seq_C und löscht das Acknowledgement-Bit. Falls auf der Zielmaschine B ein Prozess (Server) an dem angegebenen Port lauscht, wird diesem das TCP-Paket zugestellt. In der zweiten Handshakephase wird der Empfang des Paketes mit dem wie oben beschriebenen Setzen des Acknowledge-Bits quittiert und das Antwortpaket wird mit einer neuen, vom Server S generierten Sequenznummer Seq_S versehen. Diese bestätigt der Client in der letzten Phase des Handshakes. Es werden also folgende Schritte durchlaufen:

Verbindungsauftbau

auf Rechner A	auf Rechner B	IP-Paket enthält
Client →	Server	Sequenznr Seq_C
Client ←	Server	Sequenznr Seq_S , Ack = $\text{Seq}_C + 1$
Client →	Server	Ack = $\text{Seq}_S + 1$
Client →	Server	Daten

3.2.5 Das User Datagram Protocol UDP

UDP ist ein verbindungsloses Transportprotokoll, das die Fähigkeiten von IP lediglich um die Möglichkeit zur Abwicklung einer Kommunikation zwischen Ports bzw. den damit verbundenen Prozessen erweitert. Im Gegensatz zum TCP-Protokoll werden im UDP eintreffende Pakete nicht quittiert und es werden keine Vorkehrungen getroffen, um Paketverluste zu erkennen oder die Reihenfolge von Paketen korrekt einzuhalten. Entsprechende Maßnah-

UDP

men sind durch die auf UDP aufsetzenden Protokolle der Anwendungsebene zu realisieren.

Bevor wir einige Sicherheitsprobleme genauer erläutern, soll mit einem Blick auf die Dienste DHCP und NAT der einführende Grundlagenteil abgeschlossen werden.

3.2.6 DHCP und NAT

Damit ein Rechner über das Internet Daten versenden und empfangen kann, muss er eine IP-Adresse besitzen (vgl. Seite 108). Da der Bereich der verfügbaren IP-Adressen in der Version IPv4 jedoch begrenzt ist (IP-Adressen haben nur eine Länge von 32-Bit), ist es wünschenswert, entweder erst beim Netzzugang dynamisch eine IP-Adresse zu erhalten, die dann nur eine gewisse Zeit gültig ist, oder den gleichen Adressbereich für verschiedene Netzdomänen verwenden zu können. Die dynamische Zuordnung von IP-Adressen ist die Aufgabe des Dynamic Host Configuration Protocols (DHCP), dessen Funktionsweise in den RFCs⁵ 2131 und 2132 spezifiziert ist. Beim Aufbau der Netzverbindung, das ist zum Beispiel beim Booten des Systems, fragt der DHCP-Client des Rechners in einer Broadcastnachricht nach einer eindeutigen IP-Adresse. Alle DHCP-Server, die diese Nachricht empfangen, antworten mit einem IP-Adressen-Angebot. Der Client nimmt das erste Angebot, das er empfängt, an und fordert über eine Request-Nachricht diese IP-Adresse vom Anbieter an. Der anbietende DHCP-Server vergibt die IP-Adresse und macht dies durch ein Acknowledgement bekannt. Jede solche dynamisch zugeteilte IP-Adresse ist jedoch nur für eine gewisse Zeitspanne gültig, sie ist sozusagen für eine gewisse Zeit gemietet (engl. *lease*). Falls der DHCP-Client nicht rechtzeitig eine Verlängerung der Gültigkeit beantragt, ist die IP-Adresse nach Ablauf der Gültigkeitsdauer für den Client nicht mehr verwendbar und der Server kann sie erneut vergeben.

NAT

NAT Die Wiederverwendung von IP-Adressen einer Netzwerk-Domäne wird vom Network Address Translation Protokoll (NAT) unterstützt. Bei der Netzwerkadressübersetzung wird auf einem Router oder Server eine NAT-Tabelle verwaltet. Der Router bzw. Server ist ein Gateway zum Internet, das eine global eindeutige IP-Adresse besitzt. Eine solche Komponente, die die Adressumsetzung durchführt, wird auch häufig als NAT-Box bezeichnet. Eine NAT-Box hat die Aufgabe, in den Datenpaketen, die von einem Rechner innerhalb der Netzdomäne an das Internet weitergeleitet werden sollen,

⁵ Request(s) for Comments, zentral gepflegte technisch-organisatorische Texte zu Internet-Protokollen.

die nur lokal gültige Absender-IP-Adresse durch eine global eindeutige IP-Adresse zu ersetzen. Die Zuordnung zwischen globaler Adresse und domänenlokaler Adresse wird in der NAT-Tabelle verwaltet, so dass die NAT-Box Antwort-Pakete von Rechnern aus dem Internet, die nur die globale Adresse kennen, korrekt an den lokalen Rechner weiterleiten kann. Dies ist zum Beispiel beim WWW-Zugriff des Domänen-Rechners notwendig.

NAT- Tabelle

Beispiel 3.1 (NAT-Beispiel)

Abbildung 3.5 veranschaulicht die Arbeitsweise einer NAT-Komponente. Auf der linken Seite der Abbildung sieht man ein nicht-öffentliches Unternehmensnetz mit IP-Adressen im Bereich 10.0.42.x sowie beispielhaft zwei Rechner in diesem Netz, nämlich den PC 1 mit der lokalen IP-Adresse 10.0.42.1 und den PC 2 mit der Adresse 10.0.42.2. Diese IP-Adressen sind lediglich lokal adressierbar und außerhalb des lokalen Netzes unbekannt. Das heißt, ein Rechner im Internet kann keinen dieser beiden PCs direkt adressieren. Da in diesem Beispiel nur die Adresse 130.3.18.39 nach außen bekannt ist, muss die NAT-Tabelle eine Abbildung zwischen dieser globalen und den lokalen IP-Adressen verwalten. Um unterschiedliche Verbindungen differenzieren zu können, werden zusätzlich auch die Port-Adressen betrachtet. Werden bei der Adressumsetzung auch die Port-Adressen transformiert, so spricht man vom PAT (Port Address Translation).

PAT

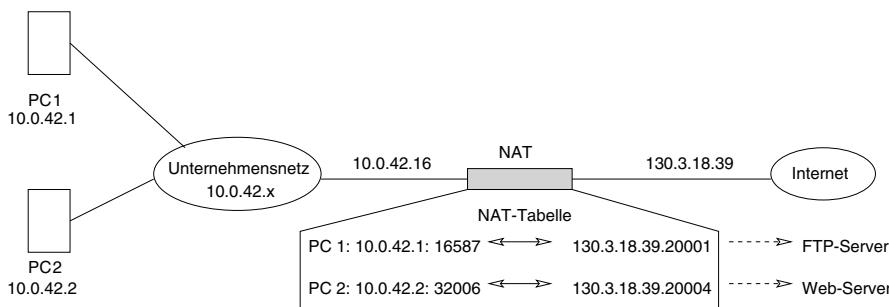


Abbildung 3.5: NAT-Beispiel

Das Beispiel skizziert ein Szenario, in dem der lokale PC 1 einen Dateitransfer über eine FTP-Verbindung von seinem lokalen Port 16587 zu einem FTP-Server außerhalb des Unternehmensnetzes aufbaut, während der PC 2 beispielsweise eine HTTP-Verbindung über seinen lokalen Port 32006 zu einem Web-Server im Internet aufgebaut hat. Die NAT-Komponente führt die Adress- und Portabbildung durch, indem sie ein Datenpaket mit der Absender-IP-Adresse 10.0.42.1 und dem Sende-Port 16587 auf die Absender-Adresse 130.3.18.39 und den Sende-Port 20001 abbildet. In

analoger Weise werden ankommende Pakete, die als Empfänger-Adresse 130.3.18.39 sowie den Empfänger-Port 20001 eingetragen haben, auf die lokale IP-Adresse und den lokalen Port des PC 1 abgebildet. Ankommende Datenpakete, die zu der HTTP-Verbindung des Rechners PC 2 gehören, werden anhand des Ports, der für diese Verbindung verwendet wird, erkannt und zum PC 2 weiter geleitet.



Fazit

Sowohl DHCP als auch NAT haben aus sicherheitstechnischer Sicht den Vorteil, dass sie die Adressen von Rechnern verschleiern und dass damit Angriffe auf diese Rechner, die auf der Kenntnis der korrekten IP-Adresse basieren, erschwert oder sogar unmöglich gemacht werden. Zu beachten ist, dass ein NAT-Server in keiner Weise die Aufgaben eines Filters im Sinne einer Firewall (vgl. Abschnitt 14.1) wahrnimmt. Das Gateway leitet alle Anfragen aus dem internen Netz an das öffentliche Netz weiter und setzt lediglich die IP-Adressen und ggf. auch die Portangaben um. Fortgeschrittenere Firewalls bedienen sich jedoch des NAT, um Filterungen und Kontrollen durchzuführen. NAT bietet eine gute Möglichkeit, die interne Struktur (wie viele und welche Rechner befinden sich im internen Netz, welche IP-Adressen besitzen sie etc.) eines lokalen Netzes nach außen zu verbergen. Je weniger ein potentieller Angreifer über die interne Struktur eines Netzes weiß, desto weniger Angriffspunkte hat er. Nachteilig am NAT ist, dass es eine Modifikation an den Daten eines Pakets vornimmt, so dass es im Zusammenspiel mit Protokollen wie beispielsweise IPSec (vgl. Abschnitt 14.3), die Integritätsprüfungen unter Einbeziehung dieser Daten durchführen, zu Problemen kommt. Problematisch ist NAT auch im Zusammenspiel mit dem Loggen von auffälligen Paketen. Erfolgt der Eintrag in die Logdatei erst nachdem die Adressumsetzung stattgefunden hat, so enthält der Logeintrag diese neue Adresse und es ist schwierig, einen Zusammenhang mit demjenigen internen Rechner herzustellen, der der tatsächliche Absender des Pakets war.

Auf der Basis der eingeführten Protokollgrundlagen können nun wesentliche Sicherheitsprobleme, die im Zusammenhang mit der Internet-Protokollfamilie auftreten, diskutiert werden. Die in Abschnitt 3.3 dargestellten Probleme beziehen sich auf IPv4 und seine genutzten Protokolle. Auf die Sicherheitsaspekte von IPv6 gehen wir auf Seite 128 noch einmal gesondert ein.

3.3 Sicherheitsprobleme

Im Folgenden untersuchen wir, in welchem Maß TCP bzw. IP-basierte Protokolle und Dienste Bedrohungen der Integrität, Vertraulichkeit, Verbindlichkeit, Verfügbarkeit sowie Authentizität ausgesetzt sind.

3.3.1 Sicherheitsprobleme von IP

Authentizität

Einer der häufigsten Angriffe im Internet ist das Address Spoofing, bei dem sich ein Angreifer maskiert und unter einer gefälschten Identität eine Kommunikation aufbaut. Ansatzpunkte für diese Klasse von Angriffen sind die in den IP-Paketen verwendeten IP-Adressen, hier insbesondere die des Absenders eines Paketes. Beim Address Spoofing erzeugt der Angreifer synthetisch hergestellte IP-Pakete mit gefälschter IP-Sendeadresse, um zum Beispiel in einem fremden Netzwerk Pakete einer internen Station oder die Identität eines spezifischen Rechners vorzutäuschen. Als Absenderadresse benutzt der Angreifer hierbei die IP-Adresse seines Opfers, das sich in diesem Netzwerk aufhält. Da die IP-Protokolle den Angaben in den Adresseinträgen der Pakete vertrauen, also die Absender und Empfänger nicht authentifiziert werden, können diese Adressen gefälscht werden. Ein Angreifer ist damit in der Lage, seine eigenen IP-Pakete unter einer fremden Absenderidentität einzuschleusen.

Spoofing

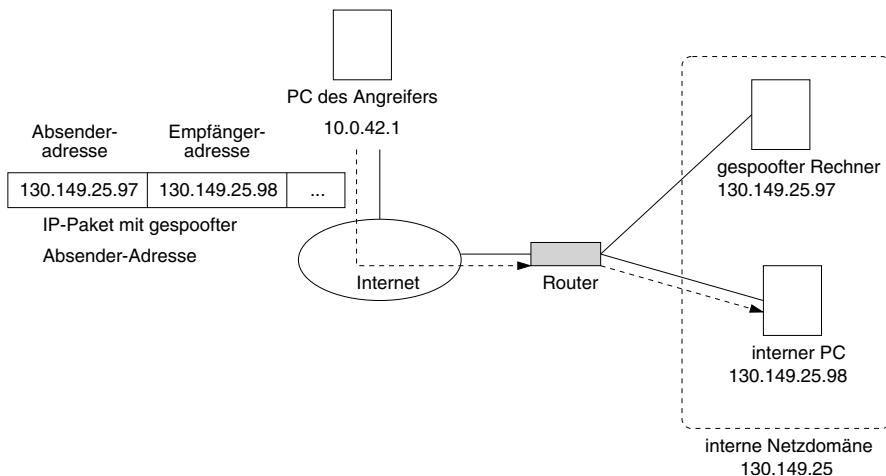


Abbildung 3.6: IP-Spoofing-Szenario

Abbildung 3.6 veranschaulicht ein einfaches Spoofing-Szenario. Der PC des Angreifers befindet sich außerhalb der Netzdomäne 130.149.25. Der Angreifer konstruiert Datenpakete, in denen er als Absender-IP-Adresse eine

IP-Spoofing

Adresse eines internen Opfer-PCs einträgt, im Beispiel sei dies die Adresse 130.149.25.97. Dadurch gibt er sich als ein zu dem lokalen Subnetz gehörender Rechner aus. Auf diese Weise können bereits schlecht konfigurierte Router überlistet werden, die Pakete mit internen IP-Adressen beliebig passieren lassen. Im Beispiel würde der Router das mit der gefälschten Absender-Adresse versehene Paket an den PC mit der IP-Adresse 130.149.25.98 weiterleiten.

Im Folgenden werden zur Verdeutlichung verschiedene Angriffe exemplarisch skizziert. Die Beispiele sollen das Bewusstsein für die Problemvielfalt schärfen. Sie sind aber keineswegs vollständig und decken nur Teilbereiche ab. Eine Vielzahl technischer Angriffsbeschreibungen sind u.a. in [115] zu finden.

Beispiel 3.2 (Spoofing-Angriff)

trusted host

.rhosts-Datei

Gegeben sei ein Unix-Rechner mit dem Namen `ziel.de`. In der Datei `/etc/hosts.equiv` kann ein Administrator eines Unix bzw. Linux-Rechners Namen derjenigen Maschinen eintragen, denen er vertraut. Vertrauen (engl. *trust*) bedeutet hier, dass Benutzer dieses vertrauenswürdigen Rechners sich ohne Angabe eines Passwortes auf demjenigen Rechner einloggen dürfen, der dieses Vertrauen mittels der angegebenen Datei ausgesprochen hat. Voraussetzung ist allein, dass diese Benutzer auf beiden Maschinen die gleiche Benutzerkennung besitzen. Daneben kann jeder Benutzer eines Unix-Rechners in seiner privaten `.rhosts`-Datei in seinem Home-Verzeichnis ebenfalls Namen von vertrauenswürdigen Rechnern eintragen, so dass von diesem Rechner aus ein entferntes Einloggen unter seiner Kennung ohne Passwortangabe zulässig ist. Mit der `.rhosts`-Datei können auch individuelle Benutzer mit einer anderen Kennung als vertrauenswürdig festgelegt werden, so dass diese unter der Kennung des Besitzers der `.rhosts`-Datei ohne Passwortangabe aktiv sein dürfen. Dies ist das Trusted Host Konzept von Unix.

In der Datei `/etc/hosts.equiv` des Rechners `ziel.de` sei der Name `mein_Freund` eines anderen Rechners eingetragen. Für das Beispiel gilt somit, dass sich ein Benutzer, der auf dem Rechner `mein_Freund.de` eingeloggt ist und die gleiche Benutzerkennung auf dem Rechner `ziel.de` besitzt, dort einloggen darf, ohne ein Passwort vorweisen zu müssen. Die `/etc/hosts.equiv`- bzw. `.rhosts`-Dateien werden bei der Ausführung eines `rlogin`- oder `rsh`-Aufrufs nach passenden Einträgen durchsucht. Somit erhält ein Angreifer, der es schafft, einem dieser Dienste des Zielrechners die Identität eines vertrauenswürdigen Rechners vorzutäuschen, einen direkten Zugang zum Zielrechner, ohne dass er sich au-

tentifizieren muss. Da sich der Aufrufer eines Netzdienstes über seine Absender-IP-Adresse in seinen IP-Paketen identifiziert und die Dienste auf die Unverfälschbarkeit dieser Adressen vertrauen, kann unser Angreifer nun versuchen, die IP-Adresse des Rechners `mein_Freund.de` in seine IP-Pakete zu platzieren. Mit Superuser-Privilegien und einigen Systemkenntnissen ist dies möglich, so dass ein erfolgreicher Spoofing-Angriff durchgeführt werden kann.



Beispiel 3.3 (Denial-of-Service Angriff)

Address Spoofing ist häufig ein Ausgangspunkt für Denial-of-Service Angriffe (DoS). Obwohl derartige Angriffe bereits seit langer Zeit bekannt sind, wurde die breite Öffentlichkeit erst Anfang Februar 2000 auf deren Gefährlichkeit aufmerksam, als nämlich so bekannte Web-Portale wie Yahoo, Amazon.com und eBay durch DoS-Angriffe lahm gelegt wurden. Die durchgeführten Angriffe verdeutlichen, in welchem Maß Diensteanbieter verwundbar sind, deren finanzielles Überleben in hohem Grad von einer ununterbrochenen Erreichbarkeit und Verfügbarkeit der Dienste abhängt.

Denial-of-Service

Zur Durchführung der oben erwähnten Denial-of-Service Angriffe wurden so genannte Zombie-Rechner verwendet. Dabei handelt es sich um Unix-, Linux- oder auch Windows-basierte PCs oder Servermaschinen innerhalb des Internets, die von einem Angreifer unterwandert und zur Durchführung der DoS-Angriffe vorbereitet wurden. Dazu nutzen Angreifer bekannte Sicherheitslücken oder Konfigurierungsfehler von Maschinen aus, um über ungeschützte Ports, unsichere Dienste oder andere Schwachstellen (u.a. Buffer-Overflow Verwundbarkeiten) Root-Zugriff auf das jeweilige Zielsystem zu erlangen. Ungeschützte Ports sind beispielsweise solche, die nicht durch Firewalls blockiert sind. Ein unsicherer Dienst ist unter anderem der Remote Procedure Call (RPC), mit dem man ein Kommando, das in eine RPC-Nachricht eingebettet wurde, durch den Kommandointerpreter des entfernten Systems ausführen lassen kann. Ausgestattet mit Root-Rechten können Angreifer dann auf den unterwanderten Maschinen Dämonenprozesse installieren, die den Netzverkehr beobachten und auf ein Startkommando des Angreifers warten, um den DoS-Angriff zu beginnen. Da bei diesem Angriff viele verteilte Rechner, hier die Zombies, instrumentalisiert werden, um den Angriff gemeinsam durchzuführen, spricht man in solchen Fällen von Distributed Denial of Service (DDoS) Angriffen (vgl. dazu auch die Vorgehensweise des Blaster-Wurms auf Seite 68, bzw. die Ausführungen zu Bot-Netzen auf Seite 75).

*Angriffs-
vorbereitung*

DDoS

Ist ein solcher Angriff erfolgreich angelaufen, kann man ihn nur noch dadurch abwehren, dass man die Adressen der Zombie-Rechner ermittelt und jeglichen Verkehr von diesen Rechnern abblockt. Es existieren bereits etliche frei im Internet verfügbare Softwarepakete, wie Trinoo, Tribe Flood Network (TFN) und Stacheldraht, die es einem Angreifer auf einfache Weise ermöglichen, Dämonenprozesse zur Durchführung von DoS-Angriffen zu konfigurieren.

UDP-flood Angriff

Ein möglicher solcher DoS-Angriff, der UDP-flood-Angriff, basiert auf UDP-Diensten. Beim UDP-flood sendet ein Angreifer ein UDP-Paket mit einer gefälschten Rücksendeadresse an sein Opfer. Mit dieser Nachricht verbindet der Angreifer den Dienst *chargen*, den das Opfer zur Erzeugung von Zeichen anbietet, mit dem Echo-Dienst derjenigen Zombie-Maschine, deren Adresse in der gefälschten Adresse angegeben ist. Der Echo-Dienst sendet die vom Opfer erzeugten Zeichen wieder zurück, so dass ein kontinuierlicher Nachrichtenstrom entsteht, der beide Maschinen lähmmt.

Damit der Angriff nicht an der geringen Bandbreite scheitert, die dem Angreifer ggf. nur zur Verfügung steht, kann dieser versuchen, die Anfrage an eine Broadcast-Adresse zu richten, so dass eine einzelne Anfrage des Angreifers, die nur eine geringe Bandbreite benötigt, automatisch per Broadcast vervielfacht wird.

SYN-flood-Angriff

Der so genannte SYN-flood-Angriff ist eine TCP-basierte Variante eines DoS-Angriffs. Hierbei werden die Protokollschrifte ausgenutzt, die beim Aufbau einer TCP-Verbindung durchgeführt werden (vgl. 3-Wege-Handshake). Ausgetauscht wird ein SYN-Paket, das vom Empfänger und dann wieder vom Sender bestätigt wird.. Mit dem SYN-flood-Angriff sendet ein Angreifer eine große Anzahl von SYN-Paketen an sein Opfer, wobei die Absenderadressen in diesen Paketen gefälscht sind und solche Rechner bezeichnen, die entweder nicht existieren oder nicht funktionsbereit (z.B. offline) sind. Nachdem das Opfersystem seine SYNC-ACK Pakete an diese gefälschten Adressen gesendet hat, wartet es vergeblich auf die Bestätigungen seiner SYNC-ACKs. Durch diese nicht bestätigten Pakete füllt sich auf dem Rechner des Opfers der Puffer, der vom Betriebssystem für die Verwaltung von offenen Verbindungen bereit gestellt wird. Das Opfer wird dadurch am weiteren Arbeiten gehindert und kann insbesondere keine legitimen Anfragen mehr bearbeiten.

Mit den SYN-Cookies (vgl. <http://cr.yp.to/syncookies.html>) steht ein geeignetes Konzept zur Abwehr von SYN-Flood-Angriffen zur Verfügung; es ist standardmäßig Linux und BSD-Unix-Derivaten integriert, wenn auch häufig defaultmäßig deaktiviert.

Smurf-Angriff

Schließlich ist unter diesen Angriffen auch noch die so genannte Smurf-Attacke zu nennen. Bei diesem Angriff verbreitet der Angreifer eine

ICMP-Echo-Anfrage (ping) an eine Vielzahl von Maschinen. Wiederum ist die Sendeadresse gefälscht, wobei diesmal als Sender und damit als Empfänger der angeforderten Echo-Pakete die Adresse des Opferrechners angegeben ist. Beim Erhalt der Echo-Anfrage werden also alle Maschinen eine Antwort an das Opfer senden und dessen System dadurch lahm legen.



Vertraulichkeit, Integrität und Verbindlichkeit

Weder die Nutzdaten noch die Verwaltungsdaten (Headerdaten) eines IP-Pakets werden vor der Übertragung verschlüsselt. Das bedeutet, dass auf allen Kommunikationsverbindungen sowie in allen Vermittlungsstellen diese Daten im Klartext vorliegen. Nutzdaten können z.B. vertrauliche Passwörter, die für ein entferntes Login benötigt werden⁶, vertrauliche Firmendaten oder personenbezogene, vertrauliche Daten beinhalten; sie werden vor unautorisiertem Lesen nicht geschützt. Die Offenlegung aller Verkehrsdaten kann von Angreifern für die Durchführung von Verkehrsflussanalysen ausgenutzt werden. Das bedeutet, dass anhand der Sende- und Empfangsadressen Zugriffs- und Kommunikationsprofile erstellbar sind. Anonymität wird somit ebenfalls nicht gewährleistet.

Vertraulichkeit

Das Internet-Protokoll stellt auch keine Mechanismen wie Hashfunktionen oder Message Authentication Codes zur Überprüfung der Integrität der übertragenen Nutzdaten zur Verfügung, so dass es dem Empfänger eines Nachrichtenpaketes nicht möglich ist, unautorisierte Modifikationen zu erkennen und Maßnahmen zur Abwehr der daraus resultierenden Angriffe zu ergreifen. Die in den Standardprotokollen verwendeten Prüfsummenverfahren wie CRC dienen zur Erkennung von Bitübertragungsfehlern, sind aber ungeeignet, gezielte Manipulationen zu erkennen, da ein Angreifer mit den manipulierten Daten auch deren korrekte Prüfsumme berechnen und in das Datenpaket integrieren kann.

Integrität

Die Identifikation der Kommunikationspartner erfolgt anhand der in den IP-Paketen eingetragenen, jedoch nicht fälschungssicheren IP-Adressen. Weitere Maßnahmen, wie digitale Signaturen, um transferierte Daten den absendenden Prozessen bzw. Benutzern zuzuordnen, werden nicht ergriffen. Das hat zur Konsequenz, dass IP-basierte Aktionen ohne zusätzliche Maßnahmen nicht verbindlich sind.

Verbindlichkeit

Bedrohungen der Vertraulichkeit, Integrität und Verfügbarkeit des IP-Protokolls gehen auch von den so genannten Routing-Angriffen aus, auf die wir im Folgenden noch kurz eingehen.

⁶ Zum Beispiel beim Zugriff auf das eigene Mail-Konto bei einem öffentlichen Mail-Provider.

Routing-Angriffe

Zur Wegewahl werden Routing-Protokolle benutzt, wobei der festgelegte Weg meistens ein symmetrischer ist, d.h. Antworten vom Zielrechner werden auf dem gleichen Weg an den Sender zurückgeleitet. Wie bereits erwähnt, lassen sich über das Optionsfeld eines IP-Pakets diverse Zusatzfunktionen aktivieren. Angriffspunkte bieten die Optionen *Loose Source Routing* oder das *Strict Source Routing*, da damit die Route festlegbar ist, die IP-Pakete durch das Netzwerk nehmen sollen. Beim *Strict Source Routing* muss der Sender aber die IP-Adressen, über die das Paket geschickt werden soll, in der richtigen Reihenfolge in das IP-Paket eintragen. Das heißt, dass zwei Rechner, die in der angegebenen Folge hintereinander stehen, auch direkt miteinander verbunden sein müssen. Dies erfordert vom Angreifer Kenntnisse über die Netzinfrastruktur.

Strict Source

Das *Loose Source Routing* ermöglicht dem Angreifer, explizit einen Pfad zum Zielrechner anzugeben, ohne dass die Rechner auf diesem Pfad direkt miteinander verbunden sein müssen. Der Angreifer kann also seinen eigenen Rechnernamen in diesen Vermittlungspfad aufnehmen lassen. Das hat zur Folge, dass die Antworten auf diesem Pfad zurückgesendet werden müssen und der Angreifer die Möglichkeit hat, alle gesendeten Pakete zu beobachten und zu analysieren (z.B. extrahieren von Klartext-Passwörtern). Um derartige Angriffe zu verhindern, weisen viele Router ankommende Pakete mit aktivierten *Source routing*-Optionen zurück, ohne sie zu bearbeiten.

RIP

Weiterhin sind auch Angriffe zu nennen, die das Routing Information Protocol (RIP) ausnutzen. Das RIP wird eingesetzt, um Routing-Informationen in einem Netzwerk zu propagieren. Typischerweise überprüfen die Empfänger diese Informationen nicht. Auf diesem Weg ist es einem Angreifer möglich, die Kommunikation zweier Maschinen an eine spezifische Zielmaschine umzuleiten. Dies geschieht, indem der Angreifer *X* einen Rechner *A* simuliert und manipulierte RIP Pakete an die anzugreifende Maschine *B* sowie an die zwischen *X* und *A* liegenden Gateways schickt. Über die RIP-Pakete werden sowohl *B* als auch die Gateways aufgefordert, jedes Paket von *B*, das eigentlich an *A* gerichtet ist, zum Rechner *X* zu senden. Nachdem der Angreifer die Pakete ausgewertet hat, sendet er sie weiter an den eigentlichen Empfänger *A*. Aktiviert der Angreifer zusätzlich noch die Option *Source Routing*, so kann er erreichen, dass auch die Antwortpakete von *A* zu *B* über den Angreifer *X* umgeleitet werden. Um RIP-Angriffen zu begegnen, müssen Router so konfiguriert werden, dass Änderungen der bestehenden Wegewahl nicht oder nur unter speziellen Randbedingungen möglich sind.

Das IP-Protokoll umfasst zwei Protokolle, die aus Sicherheitsicht von Bedeutung sind, da sie Ausgangspunkt von vielen Angriffen im Internet sind. Es handelt sich dabei zum einen um das ICMP (Internet Control Message Protocol) und zum anderen um das bereits angesprochene Address

Resolution Protocol (ARP). Auf Sicherheitsprobleme im Zusammenhang mit diesen beiden Protokollen wird im Folgenden kurz eingegangen.

3.3.2 Sicherheitsprobleme von ICMP

Das Internet Control Message Protocol (ICMP) ist für die Übermittlung von Fehler- und Statusmeldungen zuständig. ICMP-Fehlermeldungen informieren u.a. über die Nichterreichbarkeit von Empfängern (`destination unreachable`) oder fordern zum Fragmentieren von Paketen auf (`fragmentation needed`), falls das übermittelte Paket zu groß war. Statusmeldungen betreffen die Verfügbarkeit von Verbindungen oder von Maschinen. Erkennt die Netzwerkschicht eines Routers eine Stausituation in einem Teilnetz, so sendet sie eine `ICMP-redirect` Nachricht an die an das Teilnetz angeschlossenen Router. Damit werden diese aufgefordert, die überlasteten Verbindungen zu vermeiden und stattdessen die in der Nachricht angegebenen Wege zu wählen. Mittels einer `ICMP-source quench` Nachricht können aber auch direkt die Verursacher eines Staus, also die Sender, beeinflusst werden. Eine solche Nachricht fordert den Sender zum Drosseln seines Nachrichtenversands auf. Informationen über den Status einer spezifischen Maschine erhält man u.a. mittels der `ping`-Nachricht.

Fehlermeldung

Statusmeldung

Verbindungs-
abbruch

Angriffe zum Abbruch von Verbindungen

Kann ein Datenpaket nicht an die gewünschte Adresse vermittelt werden, so erhält der Absender eine ICMP-Nachricht `destination unreachable`. Das ICMP-Paket enthält den Paket-Header des nicht vermittelbaren Pakets mit den Sender- und Empfängerports. Daran kann die Sendestation erkennen, welche Verbindung aufgrund des angezeigten Fehlers abgebrochen werden muss. Es existieren jedoch einige ältere ICMP-Implementationen, die nicht fähig sind, Portinformationen von ICMP-Paketen zu analysieren. Wird vom ICMP ein entsprechender Fehler an den Absender von IP-Paketen signalisiert, muss dieser deshalb alle seine offenen Verbindungen zu dem betroffenen Empfänger-Rechner abbrechen. Dies bietet Angreifern eine leichte Möglichkeit, den Netzverkehr solcher Sender erheblich zu stören, da mit gefälschten ICMP-Nachrichten direkt der Abbruch einer großen Anzahl von Internet-Verbindungen bewirkt werden kann. Diese Angriffe können darüber hinaus in weiteren Schritten dazu führen, dass der Angreifer versucht, ein Opfer gezielt zu isolieren, um sich gegenüber Dritten unter dessen Identifikation zu maskieren (`spoofing`) und so die Verbindungen seines Opfers zu übernehmen.

Angriffe zur Erhöhung der Netzlast

Eine weitere Möglichkeit, den Netzbetrieb erheblich zu stören, bietet ein Angriff, der durch die ICMP-Nachricht `fragmentation needed and DF set` verursacht wird. Diese Nachricht wird versandt, wenn ein IP-Paket

Denial-of-Service

in seiner vollen Größe nicht weiter zu vermitteln und eine Fragmentierung des Pakets erforderlich ist, aber vom Absender eine Fragmentierung durch das gesetzte don't fragment Bit verboten wurde. Mit der Nachricht wird der Absender aufgefordert, Datenpakete zu fragmentieren, was zu einer Erhöhung der Netzlast führt, da jedes Fragment mit Headerinformationen auszustatten ist. Das Versenden synthetischer Aufforderungen zur Fragmentierung führt somit zu Denial-of-Service Angriffen. Gegenmaßnahmen gegen diese Angriffe sind schwierig, da das ICMP-Protokoll nicht zu deaktivieren ist. Allerdings existiert die Möglichkeit, Vermittlungsrechner so zu konfigurieren, dass sie nur eine maximale Anzahl von ICMP-Paketen pro Zeiteinheit weiterleiten. Im Normalbetrieb ist die Anzahl von ICMP-Nachrichten verhältnismäßig gering und die meisten Netzwerk-Managementsysteme erkennen einen anormalen Anstieg an eintreffenden ICMP-Paketen.

Dedizierte Denial-of-Service Angriffe

source quench

Eine weitere Angriffsmöglichkeit stellen die source quench-Nachrichten dar. Der Empfänger einer source quench-Nachricht reduziert seine Übertragungsrate so lange, bis er keine weitere solche Nachricht mehr empfängt. Auch hier besteht die Möglichkeit, mithilfe synthetisch erzeugter ICMP-Nachrichten den Datenverkehr erheblich zu stören. Durch das Beobachten des Netzverkehrs lässt sich ein vermehrtes Auftreten entsprechender Nachrichten erkennen und es können Gegenmaßnahmen eingeleitet werden.

Ping-to-death

Zu der Klasse der dedizierten Denial-of-Service Angriffe zählt auch die mittlerweile aber veraltete Ping-to-death Attacke. Dazu sendet der Angreifer ein ICMP-Ping Paket ab, dessen Nutzdaten mindestens die Größe von 65.510 Byte besitzen. Da dies die maximale IP-Paketgröße übersteigt, wird das Paket fragmentiert und zum Zielrechner gesandt. Dieser setzt die Fragmente wieder zusammen. Weil zu den Nutzdaten jetzt aber auch noch der Ping-Header hinzukommt, ergibt sich eine Länge von mehr als 65.510 Byte. Dies führte in der Vergangenheit bei Implementierungen von Netzwerktreibern, die einen Überlauf nicht abfangen, zu einem Systemabsturz.

Gezieltes Umleiten von Paketen

redirect

Die ICMP-Nachricht redirect wird von einem Router generiert, wenn er eine günstigere Route erkannt hat. Über die entsprechende ICMP-Nachricht wird dies propoliert. Falls ein Vermittlungsrechner nicht ausschließlich nach seinen Routing-Tabellen vermittelt, könnte ein Angreifer mithilfe solcher ICMP-Nachrichten eine Änderung der Vermittlungswege über beliebige Vermittlungsrechner erreichen. Gelingt es ihm z.B. Verbindungen über seinen eigenen Rechnerknoten umzuleiten, so kann er den Datenverkehr seiner Op-

fer filtern und analysieren. Im Gegensatz zum Angriff durch den Missbrauch der Source-Routing Option von IP-Paketen, der nur ein Datenpaket betrifft, kann durch einen erfolgreichen Redirect-Angriff eine Vielzahl von Paketen umgelenkt werden. Abwehrmöglichkeiten bestehen darin, Router so zu konfigurieren, dass ihre Routing-Tabellen nur unter spezifisch kontrollierten Randbedingungen dynamisch geändert werden können.

3.3.3 Sicherheitsprobleme von ARP

Wie bereits erklärt, unterscheidet man zwischen der physikalischen Adresse eines Rechners und seiner logischen Adresse. Die physikalische Adresse oder Maschinenadresse ist i.d.R. eine auf dem Netzwerk-Controller als Firmware permanent abgespeicherte Adresse, während logische Adressen IP-Adressen oder Domänennamen sind. IP-Adressen oder auch die Domänennamen sind für den Aufbau flexibler Adressierungstrukturen wesentlich besser geeignet als die starren physikalischen Adressen. Ein Netzwerktreiber ist jedoch nicht in der Lage, einen Rechner über dessen logische Adresse anzusprechen, sondern benötigt dafür stets die physikalische Adresse. Da in den IP-Paketen aber nur die Sender- bzw. Empfänger IP-Adressen enthalten sind, ist ein Protokoll erforderlich, das die notwendige Abbildung bewerkstelligt. Dies ist die Aufgabe des Address Resolution Protocols (ARP). Dazu verwaltet jeder Rechner eine ARP-Adresstabelle in Form eines Schnellzugriffspeichers (engl. *cache*). Ein Cache-Eintrag enthält eine IP-Adresse mit der ihr zugeordneten physikalischen MAC-Adresse. Falls bei der Adressabbildung die benötigte Information nicht im Cache zugreifbar ist, wird über einen Broadcast an alle Rechner des Netzsegments eine ARP-Request Nachricht verschickt. Erkennt ein Rechner in diesem Broadcast seine eigene IP-Adresse, so schickt er dem anfragenden Rechner seine MAC-Adresse in einer ARP-Reply Nachricht zurück und der Anfrager aktualisiert damit seine Cache-Einträge.

Aufgaben

ARP

ARP-Cache-Poisoning

Das ARP ist ein zustandsloses Protokoll, so dass von einem Rechner A auch Antwortpakete (ARP-Replies) angenommen werden, für die er gar keine Anfrage gestartet hat. Die eintreffenden Antworten werden von der Maschine A in deren Cache übernommen. Ein Angreifer X , der auf diese Weise auf der Maschine A ein Paket einschleust, das seine eigene Hardwareadresse mit der IP-Adresse eines Opfers B assoziiert, sorgt dafür, dass der Rechner A Nachrichten, die für die Maschine B bestimmt sind, an den Angreifer X weiterleitet. Da der ARP-Cache der Opfer-Rechner gezielt mit gefälschten ARP-Nachrichten-Inhalten versorgt werden, nennt man diese Art von Maskierungsangriff auch Cache-Poisoning-Angriff.

Maskierung

Denial-of-Service-Angriffe

Broadcast-Sturm

Werden künstlich generierte ARP-Pakete zur Suche nach nicht existierenden IP-Adressen über ein Broadcast versendet, wird zunächst das lokale Netzsegment mit diesem Broadcast belastet. Bleibt die Suche ergebnislos, so leiten Gateways die ARP-Anfragen per Broadcast an die übrigen geschlossenen Netzwerke weiter. Dies kann schnell zu einem so genannten Broadcast-Sturm führen. Dieser Broadcast-Sturm wird noch verstärkt, wenn der Angreifer synthetische ARP-Replies der nicht existierenden Adresse zurücksendet, die von den Gateways wiederum per Broadcast verbreitet werden. Derartige Broadcasts belegen sehr schnell einen großen Teil der zur Verfügung stehenden Übertragungsbandbreite, was die Funktionsfähigkeit des Netzwerks stark beeinträchtigen kann. Durch administrative Maßnahmen und Netzüberwachungen sind solche Überlastsituationen erkennbar.

3.3.4 Sicherheitsprobleme mit IPv6

Scannen

Obwohl IPv6 einige zusätzliche Sicherheitsfunktionen, wie die IPSec-Protokollfamilie, umfasst, und allgemein als sicherer als IPv4 angesehen wird, lassen sich einige Sicherheitsprobleme identifizieren, die beim Einsatz von IPv6 zu beachten sind.

Zunächst einmal scheint der große Adressbereich von IPv6 auch sehr attraktiv, um die aus IPv4 bekannten Scans nach angreifbaren Hosts deutlich zu erschweren. Unter IPv4 dauert es in der Regel nur wenige Minuten, um ein ganzes Subnetz zu durchsuchen, da die meisten IPv4 Netze eine 24Bit Subnetzmaske verwenden, so dass lediglich 8 Bit, also 254 IPv4 Adressen, zur Identifikation des Host übrig bleiben. Demgegenüber besitzt der Interface Identifier zur Host-Identifikation unter IPv6 in der Regel eine Länge von 64 Bit, so dass ein Adressraum von 2^{64} bei einem Scan durchsucht werden müsste, was praktisch nicht leistbar ist. Selbst bei einer Probing-Rate von 1 Million Hosts pro Sekunde und unter der Annahme eines LANs mit 10.000 Hosts würde das Scannen des Subnetzes und Entdecken eines potentiell angreifbaren Hosts in dem Subnetz mehr als 28 Jahre erfordern (vgl. [80]). Damit wäre also - in der Theorie - ein besserer Schutz gegen diese Art von passivem Angriff gegeben.

Suchraum-verkleinerung

Die obige Aussage ist aber nur dann korrekt, wenn tatsächlich der gesamte Identifier-Adressraum durchsucht werden muss, was aber in der Praxis nicht der Fall ist. So existieren beispielsweise spezielle, festgelegte IPv6-Multicast-Adressen, über die alle Router und DHCP Server in einem Subnetz identifizierbar sind. Zudem werden in der Praxis die Interface Identifier meist nach einem Muster generiert, so dass ein Angreifer das Muster lernen und einen Host anhand dieses Musters sehr viel schneller identifizieren kann, und nicht den gesamten Adressraum absuchen muss. Aber auch wenn die

Interface Identifier zufällig generiert werden, so sind die IPv6 Adressen von Servern über den Domain Name Service öffentlich bekannt, so dass diese Information ein guter Startpunkt für Angreifer darstellt. Hinzu kommt, dass davon auszugehen ist, dass unter IPv6 auch lokale Hosts aus administrativen Gründen bei lokalen DNS-Dienste registriert sein werden, so dass DNS-Server in Zukunft noch attraktivere Angriffsziele sein werden. Ist es einem Angreifer gelungen, einen Host zu kompromittieren, so kann auch die Information aus dem Neighbor Cache des Host dazu genutzt werden, neue angreifbare Hosts zu finden.

Durch das Konzept der Extension Header können weitere Sicherheitsprobleme auftreten. Da die Reihenfolge und Anzahl derartiger Header nicht genau festgelegt ist, kann das Parsen sehr aufwändig sein. Dies kann ein Angreifer für einen gezielten Denial-of-Service Angriffe (DoS) beispielsweise gegen Intrusion Detection Systeme nutzen, indem er lange Header-Listen generiert. Neben dieser generellen Problematik bietet jeder Extension Header zusätzliche Funktionalität, die zu weiteren Sicherheitsproblemen führen kann. Hierbei kommt dem *Routing Header* eine besondere Bedeutung zu, der zwar für Mobile IP sehr nützlich ist, aber gleichzeitig einiges Missbrauchspotential birgt. So kann ein Angreifer beispielsweise seinen kompromittierten Datenverkehr gezielt über einen, von Firewallrechnern als vertrauenswürdig eingestuften Host leiten, indem dessen IP-Adresse als Zwischenstation angegeben wird. Zusätzlich kann auch Schadsoftware wie ein Bot den Routing-Header nutzen, um legitimen Datenverkehr, der sensitive Information enthalten kann, über kompromittierte Rechner zum eigentlichen Ziel zu leiten.

Wie bereits bei der Diskussion von IPv6 erwähnt, ist das Protokoll auf die Unterstützung durch das ICMPv6 Protokoll angewiesen. Daher müssen die Filterregeln von Firewalls entsprechend angepasst werden, damit sie die notwendigen ICMPv6 passieren lassen, aber diese gleichzeitig auch analysieren, um zu erkennen, ob sie möglicherweise Schadcode enthalten. Denn analog zu den bekannten Smurf-Angriffen, können auch ICMPv6 Nachrichten missbraucht werden, um eine DoS Attacke zu starten. So könnte ein Angreifer durch das Senden eines speziell präparierten IP-Pakets an eine Multicast-Adresse einen Opferrechner lahm legen, wenn der Angreifer die Adresse des Opferrechners als Absendeadresse angibt. Entsprechend der ICMPv6 Funktionalität wird nämlich der Opferrechner in diesem Fall von allen Rechnern der Multicast-Gruppe eine Fehlermeldung als Antwort erhalten.

Da das Neighbor Discovery Protocol (NDP) keine Mechanismen zur Authentisierung von Nachrichten-Sendern enthält, können NDP-Nachrichten von Angreifern gezielt missbraucht werden (vgl. u.a. [38]). Es ermöglicht

Extension Header

ICMPv6

NDP

Angreifern beispielsweise, sich als Man-in-the-Middle in eine Kommunikation zweier Partner A und B im selben Subnetz einzuschleusen. Wenn der Partner A einen Pfad zu B etablieren möchte, wird er sich per NDP-Anfrage nach der MAC-Adresse von B erkundigen (Neighbor Solicitation Nachricht von A an die All-nodes Multicast-Adresse). Der Angreifer kann sich nun als B maskieren, indem er mittels einer nicht auf Echtheit geprüften Neighbor Advertisement Nachricht antwortet. Der Host A wird diese Antwort als korrekt akzeptieren und in seinen Cache eintragen. Sämtliche Kommunikation mit B wird dann über den Angreifer-Rechner geroutet. Analoge Angriffe können durch den Missbrauch von Router Advertisement Nachrichten gestartet werden, indem ein Angreifer sich ungeprüft als Router ausgibt oder die Adresse eines legitimen Routers spoofed und mit einer entsprechenden Nachricht seine Nicht-Verfügbarkeit im Netz bekannt gibt. Damit wird der legitime Router nicht mehr verwendet.

Um den skizzierten Problemen zu begegnen, wurde mit dem RFC 3971 das Secure Neighbor Discovery (SEND) Protokoll vorgeschlagen. SEND sieht die Verwendung von RSA zur Authentisierung der Absendeadressen vor. Dieses Protokoll wird aber derzeit in der Praxis kaum unterstützt.

Implementierung

Bislang wurden nur Sicherheitsprobleme angesprochen, die sich aus dem Design von IPv6 ergeben. Da das Protokoll jedoch sehr komplex ist und eine Vielzahl von Funktionen umfasst, die optional einsetzbar sind, ist zu erwarten, dass zukünftige IPv6 Implementierungen erhebliche Sicherheitsmängel aufweisen werden. Wird darüber hinaus das Dual-Stack Konzept verwendet, um einen Mischbetrieb von IPv4 und IPv6 zu unterstützen, müssen sowohl die verwendeten Firewallrechner als auch die eingesetzten Intrusion Detection Systeme sowohl IPv4- als auch IPv6 bezogenen Filterregeln umfassen und entsprechende Analysen durchführen. Dies kann sehr aufwändig sein.

Zusammenfassend kann man festhalten, dass die meisten der aus IPv4 und seinen Protokollen bekannten Sicherheitsprobleme auch für IPv6 relevant sind. Eine sichere Implementierung des komplexen IPv6 Protokolls einschließlich NDP und ICMPv6 ist eine komplexe Aufgabe und erfordert fundiertes Know-how, sorgfältige Planung und auch umfassende Tests, um das Zusammenspiel mit bestehenden Sicherheitskomponenten wie Firewallls und Intrusion Detection Systemen sicher zu gewährleisten.

3.3.5 Sicherheitsprobleme von UDP und TCP

UDP

Da das UDP ein verbindungsloses Protokoll ist, das den Empfang von Datenpaketen nicht quittiert und auch keine Maßnahmen zur Überprüfung der korrekten Reihenfolge der empfangenen Pakete vorsieht, ist es für einen Angreifer sehr leicht möglich, zusätzliche Datenpakete einzuschleusen oder

Pakete zu entfernen, ohne dass dies der Empfänger erkennen kann. Die Identifikation und Authentifikation der Kommunikationspartner erfolgt unter UDP allein auf der Basis fälschbarer IP-Adressen. Dagegen muss sich unter TCP der Kommunikationspartner zusätzlich noch über das Acknowledgement und die korrekte Sequenznummer ausweisen. UDP-basierte Dienste sind somit in weitaus höherem Maße Maskierungsangriffen ausgesetzt als TCP-basierte Dienste.

Durch das Fehlen von Quittierungsnachrichten kann darüber hinaus nicht zwischen der Initiierung einer Kommunikation (dies entspricht unter TCP einem Verbindungsaufbau) und der eigentlichen Abwicklung der Kommunikation unterschieden werden. Eine derartige Unterscheidungsmöglichkeit benötigen aber unter anderem Filterungs- und Kontrollrechner (z.B. ein Paketfilter Firewall vgl. Abschnitt 14.1), falls deren Aufgabe darin besteht, nur die eingehenden Initiiierungspakete zu kontrollieren und bei einer zulässigen Verbindung die nachfolgende Kommunikation ungehindert passieren zu lassen. Da eine differenzierte Filterung von UDP-Paketen durch einfache Paketfilter-Firewalls nicht möglich ist und wegen der oben erwähnten Angriffsmöglichkeiten, sollten UDP-Pakete externer Netze wie dem Internet von einem Vermittlungsrechner blockiert und nicht in das interne Netz (Intranet) weitergeleitet werden.

verbindungsloses
UDP

Sequenznummerangriff unter TCP

Obwohl mit der Sequenznummer und dem Acknowledgement unter TCP zusätzliche Maßnahmen zur Überprüfung der Identität der Kommunikationspartner zur Verfügung stehen, sind dennoch Maskierungsangriffe möglich. Eine Klasse solcher Angriffe ist unter dem Namen Sequenznummerangriff bekannt geworden [18]. Der Angriff nutzt Implementierungsschwächen von Unix-Implementierung von TCP aus und wird im Folgenden skizziert.

TCP

Unter TCP wird jedes empfangene Paket quittiert, indem im Acknowledgement die um 1 erhöhte Sequenznummer des zuletzt empfangenen Pakets zurückgesendet wird (vgl. Seite 115). Das bedeutet, dass ein Angreifer, der Pakete in eine aufgebaute Verbindung einschleusen oder unter der Identität eines anderen Absenders eine Kommunikationsverbindung aufbauen will, die richtige Sequenznummer kennen oder erfolgreich erraten müssen, um den korrekten Wert in seinem Acknowledgement eintragen zu können. Wie im 3-Phasen Handshake beschrieben, erzeugt ein Server nach dem Empfang eines Paketes, das einen Verbindungsaufbau initiiert, eine neue Sequenznummer und schickt sein Antwortpaket zurück. Obwohl in der Spezifikation des TCP-Protokolls gefordert ist, dass diese Sequenznummern zufällig generiert werden sollen, verwenden immer noch viele Implementierungen keinen Pseudozufallszahlengenerator, sondern einen einfachen

Sequenznummer-
angriff

Implementierungs-
problem

32-Bit Zähler. Dieser Zähler wird während der Dauer einer Verbindung jede Sekunde um den Wert 128 erhöht und jede Nachricht erhält den aktuellen Zählerstand als Sequenznummer. Bei der Einrichtung einer neuen Verbindung erzeugt der Zähler keinen zufälligen Wert, sondern erhöht nur den aktuellen Zählerstand um den Wert 64. Falls man Kenntnisse über den aktuellen Zählerstand besitzt, ist es möglich, die Sequenznummer, die ein System für den nächsten Verbindungsaufbau benutzen wird, mit hoher Wahrscheinlichkeit korrekt zu bestimmen.

Angriffsinitiierung

Damit ist klar, wie ein Maskierungsangriff vorzubereiten ist: Der Angreifer X initiiert eine TCP-Verbindung mit dem Server Z , indem er einen unkritischen Port auswählt. Ein Beispiel hierfür ist der Port 25, über den der Maildienst angesprochen wird.

$$(1) \quad X \rightarrow Z; \text{Port } 25, \text{ Sequenznummer } Seq_X.$$

Der Server antwortet und generiert dafür eine neue Sequenznummer.

$$(2) \quad Z \rightarrow X; \text{Acknowledgement: } Seq_X + 1, Seq_Z.$$

Mit der Kenntnis der Sequenznummer Seq_Z kann der Angreifer nun seinen Maskierungsangriff starten. Dazu täuscht er die Identität seines Opfers C vor und sendet an Z eine Nachricht zur Etablierung einer Verbindung, wobei er einen sicherheitskritischen Dienst anfordert wie zum Beispiel das entfernte Einloggen mittels rlogin über Port 513.

$$(3) \quad C \rightarrow Z; \text{Port } 513, Seq_{X'}.$$

Der Server antwortet wie oben angegeben, wobei die Antwort aber an C und nicht an X gesendet wird.

$$(4) \quad Z \rightarrow C; \text{Acknowledgement: } Seq_{X'} + 1, Seq_{Z'}.$$

Mit der Kenntnis der Sequenznummer Seq_Z aus seinem Vorbereitungsschritt ist der Angreifer nun in der Lage, die neue Sequenznummer $Seq_{Z'}$ des Servers zu berechnen und in einer Nachricht an Z diese Sequenznummer zu quittieren, wobei er sich wiederum als C maskiert.

$$(5) \quad C \rightarrow Z; \text{Acknowledgement: } Seq_{Z'} + 1.$$

Nach dieser angeblich von C stammenden Bestätigung geht der Server davon aus, dass er eine rlogin-Verbindung zu C etabliert hat. Der Angreifer kann diese Verbindung nun nutzen, um sich als C auf dem Server Z einzuloggen, falls er C 's Passwort kennt oder falls keine Authentifikation erforderlich ist (z.B. trusted host).

Angriffs-verschleierung

Ein Problem für den Angreifer besteht nun noch darin, dass die Antwortpakete des Servers Z an den Rechner C gesandt werden. Empfängt der Rechner C eine solche Antwort, ohne seinerseits eine zugehörige Anfrage zum Verbindungsaufbau gestellt zu haben, so sendet er an Z ein Reset-Paket zurück. Z bricht daraufhin die Verbindung ab. Der Angreifer muss also verhindern,

dass C ein solches Reset-Paket schickt. Dafür kann er ein Design- und Implementierungsproblem im Zusammenhang mit Ports ausnutzen. Das Problem besteht darin, dass die an einem Port ankommenden Nachrichten in einer Warteschlange gepuffert werden, aber beim Überschreiten der Pufferkapazität weitere ankommende Nachrichten einfach ignoriert werden und somit auch kein Reset-Paket generiert wird. Das heißt, dass ein Angreifer einen Port von C mit Anfragen überfluten muss und dann seine Maskerade starten kann, indem er diesen überfluteten Port als Empfängerport für das Reply von Z angibt. Die Schritte des Angriffs sind noch einmal in Abbildung 3.7 zusammengefasst.

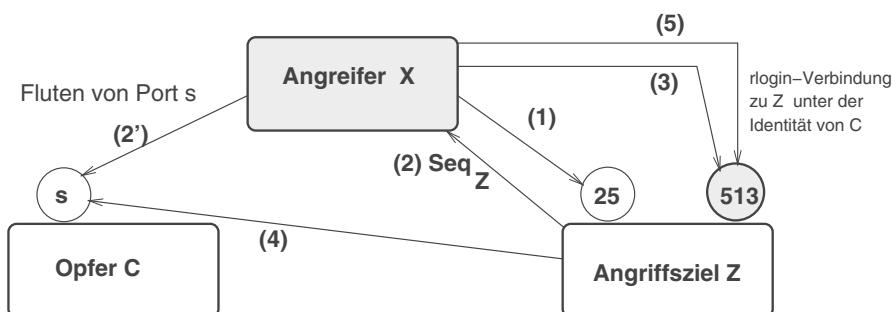


Abbildung 3.7: TCP-Sequenznummernangriff

Verbindungsübernahme

Dieser Angriff zielt darauf ab, eine bestehende Verbindung (z.B. rlogin, telnet) zwischen einem Client und einem Server zu übernehmen (engl. *session hijacking*). Das bedeutet, dass ein Angreifer durch gezielte Maßnahmen die ursprünglichen Kommunikationspartner desynchronisiert, unbemerkt an die Stelle einer der Partner tritt und in den Strom der übertragenen Nachrichten eigene Datenpakete einschleust. Ausgangspunkt ist wiederum die schwache Authentifikation von Kommunikationspartnern allein anhand der korrekten Acknowledgementdaten. Zur Durchführung des Angriffs können frei verfügbare Monitoring-Tools (u.a. snoop, ethload) verwendet werden, mit deren Hilfe alle übertragenen Datenpakete eines Netzbereiches beobachtbar sind. Aus den beobachteten Paketen lässt sich die Sequenznummer, die zum erfolgreichen Einschleusen eines Pakets benötigt wird, direkt ermitteln.

Session Hijacking

Durch das Einfügen von Datenpaketen werden die ursprünglichen Kommunikationspartner desynchronisiert. Das heißt, dass sie eine veraltete Sequenznummer verwenden. Der Empfänger einer veralteten Information schickt ein Paket mit einem Acknowledgementwert, der sich auf die Sequenznummer des Angriffspakets bezieht und somit vom Original-Kommunikationspartner nicht als gültige Antwort auf dessen Paket akzeptiert wird. Dieser wird

desynchronisieren

also seinerseits ein Paket schicken, um den Server auf den Fehler aufmerksam zu machen. In dieser desynchronen Situation tauschen Client und Server fortwährend Acknowledgementnachrichten aus, wodurch zum einen der Netzverkehr ansteigt und zum anderen die Verbindung des Client nicht mehr funktionsfähig ist. Entsprechende Situationen können aber vom Systemadministrator durch das Beobachten des Netzverkehrs erkannt und behoben werden.

Abwehr

Zur Abwehr „feindlicher Übernahmen“ sind Protokolle erforderlich, die den gesamten Nachrichtentransport absichern. In Kapitel 14.2 werden mit SSL und IPsec weit verbreitete Protokolle zur Kommunikationssicherheit vorgestellt. Beide Protokolle haben zum Ziel, die Integrität, Vertraulichkeit und, abhängig von der Konfigurierung, auch die Authentizität des Datenursprungs einzelner Datenpakete abzusichern. Eine Absicherung einer ganzen Transaktion wird damit nicht erreicht, so dass auch hier noch Session Hijacking Angriffe möglich sind.

Durch die korrekte Konfigurierung von Filterrechnern lässt sich das unbemerkte Einschleusen externer Nachrichten in ein internes Netz (z.B. ein Unternehmensnetz) unter der Maske eines internen Absenders verhindern. Dazu ist es erforderlich, dass Nachrichten bereits an den Eingängen eines Filters kontrolliert werden. Erfolgt diese Kontrolle dagegen erst beim Verlassen des Filters, so ist für diesen nicht mehr erkennbar, ob die Nachricht ursprünglich aus dem internen oder aus dem externen Netz, also z.B. dem Internet, stammte. Angriffe innerhalb eines internen Netzes können jedoch allein durch Filterungen nicht abgewehrt werden. Wie der Angriff gezeigt hat, ist eine Authentifikation der Kommunikationspartner anhand der IP-Adresse und der Sequenznummer unzureichend, um Spoofing-Angriffe wirksam abzuwehren. Die Protokolle müssen also um Authentifikationsmaßnahmen erweitert werden. Mit dem Kerberos-Protokoll wird in Abschnitt 10.4 ein Beispiel für ein solches Protokoll vorgestellt.

Fazit

Die Sicherheitsprobleme von TCP, UDP sowie IP verdeutlichen, dass die IP-Protokollfamilie ungeeignet ist, um darauf sicherheitskritische Anwendungsdienste, wie Internet-Homebanking oder E-Commerce Anwendungen, ohne zusätzliche Sicherheitsfunktionen aufzusetzen.

3.4 Sicherheitsprobleme von Netzdiensten

Die Anwendungsschicht der Internet-Protokollfamilie umfasst eine große Anzahl von Protokollen und Diensten, die hier nicht alle behandelt werden können. Im Folgenden beschränken wir uns auf einige bekannte Dienste, die kurz vorgestellt und deren Sicherheitsrisiken erläutert werden. Ausführlichere Abhandlungen finden sich unter anderem in [67]. Aufgrund der steigenden

Bedeutung von Sicherheitsproblemen Web-basierter Anwendungen, wird dieser Aspekt in Abschnitt 3.5 separat behandelt.

Zu den größten Gefährdungen bei der Nutzung der Internetdienste zählen der Verlust der Vertraulichkeit durch ein unautorisiertes Lesen der übertragenen Daten und der Verlust der Integrität durch die Manipulation der Daten. Viele der im Internet benutzten Dienste übertragen beispielsweise Passwörter unverschlüsselt, so dass unter anderem jeder, der einen privilegierten Zugang zu einem Vermittlungsrechner besitzt, diese Daten lesen oder verändern kann. Dariüber hinaus können über Sniffer-Programme gezielt sicherheitsrelevante Informationen aus IP-Paketen abgefangen werden. Sniffer-Programme werden häufig direkt mit dem Betriebssystem mitgeliefert oder stehen für unterschiedliche Plattformen als Freeware zur Verfügung (u.a. snoop, ethload, nfswatch), so dass es für Angreifer sehr einfach ist, Lauschangriffe zu starten. Das Mitlesen der versandten Daten ermöglicht aber auch Replay-Attacken, bei denen einmal zur Authentisierung benutzte Daten, wie z.B. verschlüsselte Passwörter, von einem Angreifer bei einem späteren Zugangsversuch wieder eingespielt werden.

Sniffer

Replay

3.4.1 Domain Name Service (DNS)

Der Domain Name Service (DNS) ist ein verteilter Namensdienst, der die Aufgabe hat, symbolische Namen (DNS-Namen) auf IP-Adressen abzubilden und umgekehrt, zu einer IP-Adresse den zugehörigen DNS-Namen zu ermitteln. Die logischen Teilnetze des Internet sind hierarchisch in so genannten Domänen strukturiert. Domänen der obersten Ebene sind zweistellige Ländercodes oder Kürzel, wie com, de, edu. Die logische Struktur kann sich von der physikalischen Adressstruktur unterscheiden, so dass man von einem logischen Rechnernamen nicht unbedingt auf dessen Standort schließen kann. Ein Rechner mit dem Namen server.sec.informatik.tu-muenchen.de kann sich beispielsweise an beliebigen Orten innerhalb oder außerhalb Münchens befinden. Durch diese Flexibilität des Adressschemas ist es möglich, Dienste transparent für Benutzer zu verlagern oder auch für einen Knoten mehrere dienstspezifische Namen, wie ftp.sec.informatik.tu-muenchen.de für den Datei-Transfer und www.sec.informatik.tu-muenchen.de für den WWW-Zugriff, zu vergeben.

Namensdienst

Jeder DNS-Server verwaltet zwei Datenbanken. Die Reverse-Zonen Datenbank enthält die Zuordnung zwischen IP-Adresse und Domänennamen und die Forward-Zonen Datenbank verwaltet die Zuordnung zwischen Domänennamen und IP-Adresse. Das Erfragen eines Domänenamens zu einer IP-Adresse wird als Reverse Lookup bezeichnet. Eine solche Anfrage wird unter anderem von den Berkeley r-Kommandos für den remote-Zugriff ver-

DNS Datenbanken

wendet, um bei einem Verbindungsaufbau zu überprüfen, ob IP-Adresse und Domänenname des zugreifenden Benutzers korrespondieren. Problematisch am Design von DNS ist, dass keine Maßnahmen zur Überprüfung der Konsistenz der beiden Datenbanken gefordert und realisiert werden. Die fehlende Konsistenzprüfung eröffnet einen Angriffspunkt, auf den wir weiter unten noch eingehen.

Replikation

Zur Erhöhung der Verfügbarkeit werden DNS-Namen repliziert auf unterschiedlichen DNS-Servern zur Verfügung gestellt und dezentral verwaltet. Für jede Hierarchiestufe gibt es jedoch einen primären DNS-Server, der die Originaldatenbanken der Stufe verwaltet. Eine DNS-Anfrage eines Clients wird an den Port 53 eines DNS-Servers gesandt, der sie entweder selber beantwortet oder zu einem anderen Server weiterleitet. Der für den DNS-Namen verantwortliche Server schickt an den Anfrager eine Antwort zurück, die dieser in seinem Cache ablegt, um bei einer erneuten Anfrage selber in der Lage zu sein, die Antwort direkt zu liefern. Da DNS aus Effizienzgründen auf dem unzuverlässigen UDP basiert, sorgen Client und Server dafür, dass unbeantwortete Anfragen wiederholt werden. Unerwartete Antwortpakete, dazu gehören auch Antworten auf Anfragen, für die bereits ein Antwortpaket angenommen wurde, werden einfach ignoriert. Da ein DNS-Server stets versucht, Anfragen direkt auf der Basis seiner lokalen Datenbanken und seines Caches zu beantworten, stellen diese lohnende Angriffsziele für Manipulationen dar.

Sicherheitsprobleme von DNS

DNS-Spoofing

Unter DNS-Spoofing versteht man Angriffe, bei denen es dem Angreifer gelingt, die Zuordnung zwischen IP-Adresse und Domänennamen zu fälschen. Mit einem solchen Spoofing-Angriff können auch weitere Attacken vorbereitet werden. Zur Erläuterung von DNS-Spoofing-Angriffen betrachten wir das in Abbildung 3.8 zusammengefasste Szenario.

Angriff

Wir gehen davon aus, dass es einem Angreifer X bereits gelungen ist, dem Domänenamen `mein_Freund.de` einer Maschine, die die IP-Adresse `aaa.bbb.ccc.ddd` besitzt, seine eigene IP-Adresse `xxx.yyy.zzz.www` zuzuordnen, indem er den entsprechenden Eintrag in der Reverse-Zonen-Datenbank des DNS-Servers platziert hat. Weiterhin gelte, dass der Rechner Z dem Rechner `mein_Freund.de` vertraut, diesen also in seiner Trusted Hosts Datei `/etc/hosts.equiv` (vgl. Seite 120) eingetragen hat. Nun kann der Angreifer unter der Maske von `mein_Freund.de` versuchen, mittels eines rlogin- oder rsh-Aufrufs Zugriff auf die Maschine Z zu erlangen.

Bei einem solchen Aufruf erfragt der entsprechende r-Dienst der Maschine Z beim DNS-Server den Domänennamen seines Auftraggebers. In unserem Szenario wird er also nach dem zu der ihm vorliegenden IP-Adresse

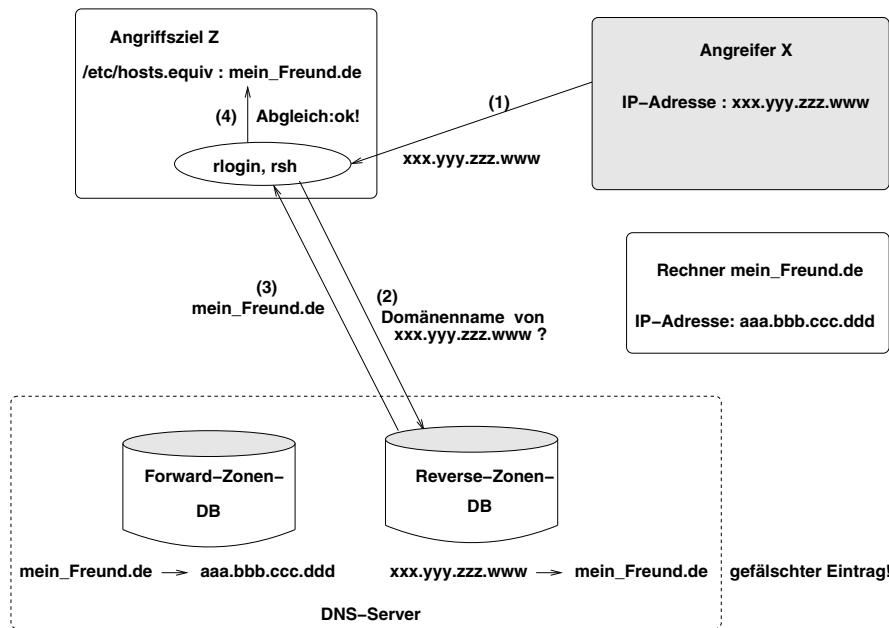


Abbildung 3.8: DNS-Spoofing-Angriff

xxx.yyy.zzz.www gehörenden Domänennamen fragen und den Namen mein_Freund.de als Antwort erhalten. Gibt sich der r-Dienst damit zufrieden und überprüft jetzt nur noch, ob er in seiner Trusted Hosts Datei einen Eintrag für mein_Freund.de vorliegen hat, so ist der Angreifer am Ziel. Der Rechner Z betrachtet den Angreifer als den vertrauenswürdigen Rechner mein_Freund.de und gewährt ihm, ohne eine weitere Authentifikation zu verlangen, Zugang.

Zur Abwehr dieser Angriffe wird in aktuellen Versionen der angesprochenen r-Dienste ein doppeltes Reverse Lookup durchgeführt. Das bedeutet, dass ausgehend von dem vom DNS-Server ermittelten Domänennamen die zugehörige IP-Adresse in der Forward-Zonen-Datenbank erfragt wird und auf diese Weise die Fälschung erkannt werden kann.

Angriffsabwehr

Cachingproblem

Wie erwähnt, verwenden DNS-Server Zwischenspeicher (Caches), um nachfolgende Anfragen schnell beantworten zu können. Ältere Versionen der DNS-Software akzeptieren auch Daten, die gar nicht von ihnen angefordert wurden, und tragen diese in ihren Cache ein. Auf diesem Weg ist es einem Angreifer möglich, einem DNS-Server einen falschen Namen unterzuschieben. So kann ein Angreifer zum Beispiel an Stelle des korrekten Namens eines Börsenrechners einen gefälschten angeben. Wird eine Anfrage nach dem Namen des Börsenrechners gestellt, so findet der DNS-Server den ge-

fälschten Eintrag in seinem Cache und gibt diese gefälschte Information als Antwort zurück. Baut der Client nun vertrauensvoll eine Verbindung mit dem vermeintlichen Börsenrechner auf, um sich die aktuellen Kurse anzeigen zu lassen, so kann dieser ihm beliebige, gefälschte Daten angeben. Das nachfolgende Beispiel verdeutlicht einen derartigen Angriff, der auch als Cache-Poisoning bezeichnet wird.

Beispiel 3.4 (DNS-Cache-Poisoning)

Poisoning-Angriff

Ausgangspunkt der Beschreibung eines Cache-Poisoning Angriffs sei das in Abbildung 3.9 dargestellte Szenario.

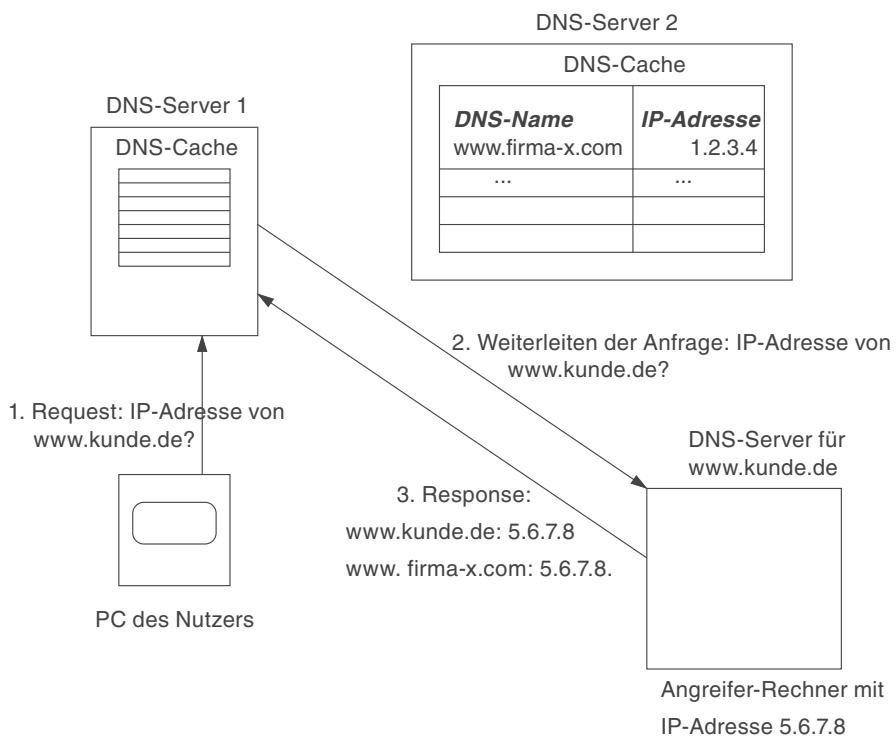


Abbildung 3.9: DNS-Cache-Poisoning Angriff (Teil 1)

In diesem Szenario wird vorausgesetzt, dass der Angreifer mit der IP-Adresse 5.6.7.8 (im Bild rechts unten) den DNS-Server der Domäne `www.kunde.de` kontrolliere. Weiterhin nehmen wir an, dass ein beliebiger Nutzer (links unten) auf den Rechner `www.kunde.de` zugreifen möchte und dazu den für den Nutzer zuständigen DNS-Server 1 nach der IP-Adresse des Rechners fragt (Schritt 1). Nehmen wir weiter an, dass im lokalen Cache dieses DNS-Servers diese Information nicht zu finden ist, so dass er die

Anfrage (Schritt 2) an den für den Rechner `www.kunde.de` zuständigen Server weiterleitet. Da der Angreifer den zuständigen DNS-Server kontrolliert, antwortet er auf diese Anfrage und schickt mit seiner Antwort gleichzeitig auch eine weitere gar nicht erfragte Abbildungszuordnung zwischen einem DNS-Namen und einer IP-Adresse, z.B. für den Server `www.firma-x.com` zum Anfrager zurück (Schritt 3). Abbildung 3.10 veranschaulicht das weitere Vorgehen.

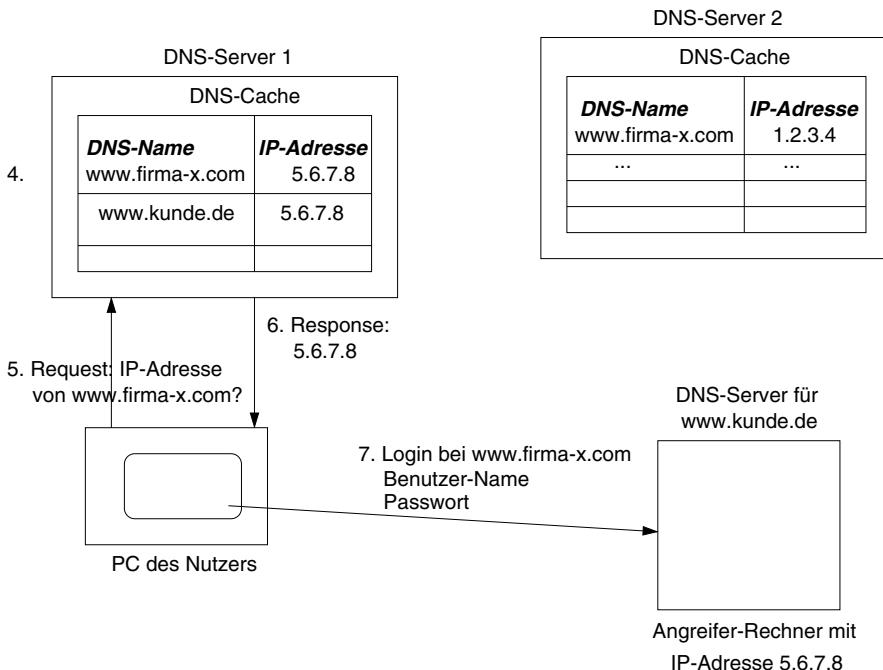


Abbildung 3.10: DNS-Cache-Poisoning Angriff (Teil 2)

Der DNS-Server 1 übernimmt die ankommenden Antwortdaten ungeprüft in seinen Cache (Schritt 4). Falls nun der Nutzer eine Anfrage zur Auflösung der IP-Adresse von `www.firma-x.com` stellt (Schritt 5), wird sein zuständiger DNS-Server 1 die in seinem Cache vorliegende Information an den anfragenden Nutzer zurücksenden. D.h. anstelle der korrekten IP-Adresse 1.2.3.4, die der DNS-Server 2 bei einer solchen Anfrage liefern würde, erhält der Nutzer die falsche Adresse 5.6.7.8, also die IP-Adresse des Angreifer-Rechners (Schritt 6). Da Clients den Antworten von DNS-Servern vertrauen, wird der Anfrager mit dieser IP-Adresse arbeiten und beispielsweise seine Zugangsdaten zu diesem Server senden (Schritt 7.). In einer solchen Situation hilft natürlich auch die Etablierung eines sicheren Kommunikationskanals zwischen Nutzer und Angreifer-Rechner (z.B. mittels SSL) nichts, da die Daten

auf jeden Fall beim Endpunkt des Kanals, das ist hier ja gerade der Angreifer-Rechner, im Klartext vorliegen. Der Angreifer könnte nun seinerseits diese Zugangsdaten verwenden, um sich bei dem richtigen Rechner `www.firma-x.com` unter der Maske des Nutzers anzumelden. Die dazu erforderlichen Credentials (z.B. Kennung und Passwort) hat er ja in Schritt 7 vom Nutzer liebenswürdigerweise erhalten.



Die Sicherheitsprobleme von DNS haben erneut verdeutlicht, dass eine Authentifikation allein auf der Basis von IP-Adressen und Domänennamen unzureichend ist.

3.4.2 Network File System (NFS)

verteiltes
Dateisystem

Das Network File System NFS von Sun Microsystems ist ein verteiltes Dateisystem, das den entfernten Zugriff auf Dateien, die auf einem Server verwaltet werden, ermöglicht. Die nachfolgenden Beschreibungen beziehen sich auf das NFS Version 2 und 3, die zwar weltweit noch immer im Einsatz sind, aber seit 2009 von NFS Version 4 abgelöst werden, da diese Version deutliche Verbesserung auch in Bezug auf die Sicherheit bedeutet. Da man jedoch an dem ursprünglichen Design von NFS einige grundlegende Probleme bei der Entwicklung verteilter Anwendungen verdeutlichen kann, dient die Beschreibung der älteren Version hier eher als didaktisches Mittel, denn zum Aufzeigen der Probleme im operativen Betrieb. Nachfolgend gehen wir auf die NFS-Versionen 2 und 3 ein und erläutern anschließend die Verbesserungen, die mit den wesentlichen Änderungen der Version 4 einhergehen.

Export

Dem NFS liegt eine Client-Server-Architektur zugrunde. NFS-Server exportieren Dateisysteme bzw. Bereiche ihres Dateibaumes und NFS-Clients fügen diese in ihr lokales Dateisystem ein. Man spricht hier vom Mounten eines Dateisystems. Die exportierten Verzeichnisse sind in der Datei `/etc/exports` des Servers aufgelistet. Für einen Eintrag in `/etc/exports` kann man Beschränkungen festlegen, die zum Beispiel angeben, welche Client-Maschinen das im Eintrag angegebene Dateisystem mounten dürfen. Falls keine Beschränkung eingetragen ist, darf ein beliebiger Rechner mounten. Eine solche Konfigurierung widerspricht ganz offensichtlich dem Prinzip der minimalen Rechte und auch dem Erlaubnisprinzip (vgl. Abschnitt 4.1.1). Systemadministratoren sollten unbedingt dafür sorgen, dass Dateisysteme stets nur beschränkt exportiert werden.

Mounten

Über das Mounten können Clients auf entfernte Dateien in gleicher Weise wie auf lokale Dateien zugreifen. Client und Server kommunizieren über einen RPC-Dienst (Remote Procedure Call), der aus Effizienzgründen UDP-

basiert realisiert ist. NFS-Server der Version 2 und 3 sind zustandslos, das heißt, dass sie keine Informationen darüber verwalten, welche Clients zu einem Zeitpunkt welche ihrer exportierten Dateisysteme gemountet haben. Ab NFS Version 4 sind die Server zustandsbehaftet und kommunizieren über TCP/IP.

Beispiel 3.5 (Mounten)

Der Server Z möchte das Verzeichnis /usr zum Client Rechner1 exportieren. Hierzu wird vom Systemadministrator beispielsweise die Zeile

```
/usr access = Rechner1
```

in die Datei /etc(exports des Servers Z eingefügt und mit dem Befehl exportfs -a exportiert.

Auf der Client Seite muss ein entferntes Verzeichnis gemountet werden, indem der Client dem Server einen Pfadnamen für die zu mountende Datei sowie einen Mountpunkt in seinem lokalen Dateisystem angibt, an den das entfernte Dateisystem angehängt werden soll. Der Befehl zum Mounten des Dateisystems /usr lautet mount Z:/usr /mnt, wobei /mnt der Mountpunkt ist. Abbildung 3.11 veranschaulicht diese Situation.

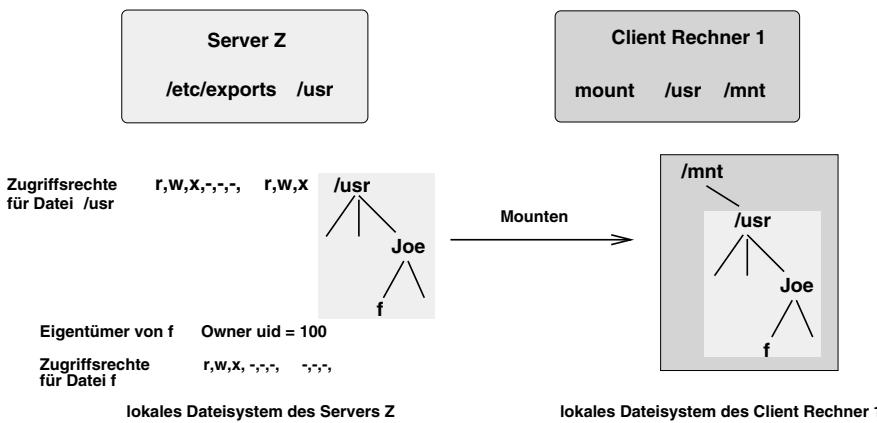


Abbildung 3.11: Mounten eines Dateisystems unter NFS



Zugriffskontrollen

NFS unterscheidet zwischen der Berechtigung, ein Dateisystem zu mounten, und der Berechtigung, auf einzelne Dateien innerhalb dieses Dateisystems lesend, schreibend oder ausführend zuzugreifen. Will ein Client ein Dateisystem importieren, so sendet er eine entsprechende Mountanforderung an

Zugriffs-
berechtigung

File Handle

den Server. Anhand der Identität des Clients sowie der festgelegten Mount-Beschränkungen überprüft der Server, ob der Client zum Mounten berechtigt ist. Ist dies der Fall, so erzeugt der Server ein so genanntes File Handle für das Wurzel-Verzeichnis des zu mountenden Dateisystems und sendet dieses an den Aufrufer. In obigem Beispiel würde also der Client Rechner 1 ein File Handle für das Verzeichnis `/usr` erhalten. File Handles sind Zugriffstickets, deren Besitz zum Zugriff berechtigt.

Kontrolle

Die Kontrolle der tatsächlichen Dateizugriffe basiert auf der Zugriffskontrolle von Unix (vgl. Kapitel 12.4). Das heißt, dass jede exportierte Datei auf dem Server durch eine Verwaltungsdatenstruktur, die so genannte `inode`, beschrieben ist. Sie enthält unter anderem die Eigentümer-, Gruppen- und Weltrechte der repräsentierten Datei. Eine ausführliche Beschreibung der Zugriffskontrolle unter Unix/Linux findet sich in Abschnitt 12.4 auf Seite 632ff. Will ein Client auf eine bestimmte Datei (z.B. auf Datei `f` in obigem Beispiel) zugreifen, so muss er diese zunächst öffnen. Dazu sendet er an den NFS-Server eine Anfrage, die im Standardfall seine eigene Benutzer- und Gruppen-Identifikation (engl. *unique identifier uid*) sowie den Namen der gewünschten Datei enthält. Dieser Standardfall wird Unix-artige Authentifikation genannt, da sich ein Zugreifer, wie auch in zentralen Unix Systemen üblich, nur über seine Identifikation authentifiziert. D.h. ein erneuter Nachweis der Identität, z.B. durch das Angeben eines Passwortes, ist bei Datei-Zugriffen nicht erforderlich. Anhand der `inode`-Informationen überprüft der NFS-Server die Zulässigkeit des Zugriffs. Falls der Client berechtigt ist, erzeugt der NFS-Server ein File Handle für die geöffnete Datei, so dass alle weiteren Zugriffe des Clients unter Vorlage dieses Handles direkt erfolgen.

Authentifikation

Sicherheitsprobleme von NFS

Im Folgenden werden einige der schwerwiegendsten Sicherheitsprobleme beim Umgang mit NFS aufgezeigt, wobei wir zwischen Design- und Implementierungsfehlern einerseits und administrativen Fehlern andererseits unterscheiden.

Mount-Berechtigung

Design- und Implementierungsfehler

Die erste Hürde, die ein Angreifer für einen unautorisierten Zugriff auf ein entferntes Dateisystem zu überwinden hat, ist das Mounten. Das Protokoll verfolgt die Politik, dass das Mounten erlaubt ist, so lange nicht explizite Einschränkungen festgelegt sind. Trifft ein Angreifer auf einen unsachgemäß administrierten NFS-Server, der keine Mount-Beschränkungen festlegt – was leider nur zu häufig vorkommt – stellt die Zugriffskontrolle beim Mounten also keine Hürde für ihn dar. Das aufgezeigte Sicherheitsproblem hat seine Wurzeln jedoch im fehlerhaften Protokolldesign, welches das Erlaubnisprinzip (vgl. Abschnitt 4.1.1) sicherer Systeme verletzt. Das Prinzip

fordert nämlich, dass Zugriffe solange verboten sind, bis sie explizit erlaubt werden.

Einige NFS-Versionen weisen darüber hinaus auch Implementierungsmängel in Bezug auf das Mount-Protokoll auf. Sie prüfen nur beim Mounten nach, ob der Client-Rechner die Erlaubnis hat, das entsprechende Dateisystem zu mounten und damit zu nutzen. Solange der Client das gemountete Dateisystem nicht wieder frei gibt, bleibt es für den Rechner zugreifbar. Dies trifft auch dann zu, wenn der Server in der Zwischenzeit seinen Eintrag in der `exports`-Datei geändert hat, d.h. eine Rechterücknahme wird dann nicht unmittelbar wirksam.

Mount-Protokoll

Wesentliche Sicherheitsprobleme resultieren aus der Designentscheidung, die Maßnahmen zur Zugriffskontrolle zentraler Unix-Systeme direkt auf die physisch verteilte Umgebung zu übertragen. Um auf eine Datei zugreifen zu können, muss der Client ein gültiges File Handle besitzen. Zur Ausstellung von File Handles führen NFS-Server standardmäßig, wie oben angegeben, die Unix-artige Authentifikation auf der Basis der Benutzer- und Gruppen-*uids* zur Überprüfung der Identität des aufrufenden Clients durch. Da diese *uids* fälschbar sind, eine Fälschung aber von NFS-Servern nicht erkennbar ist, lassen sich Maskierungsangriffe recht einfach durchführen.

Zugriffskontrolle

Beispiel 3.6 (NFS-Maskierungsangriff)

Als Beispiel betrachten wir erneut das in Abbildung 3.11 skizzierte Szenario. Die in den `inodes` der Dateien `/usr` und `f` spezifizierten Zugriffsbeschränkungen besagen, dass der Benutzer mit der `uid = 100` der Eigentümer (engl. *owner*) der Datei `f` ist und Lese-, Schreib- und Ausführungs-Rechte an `f` besitzt. Allen anderen Benutzern ist der Zugriff verboten. Will nun ein Benutzer *A*, der auf der Maschine Rechner 1 arbeitet, die entfernte, aber lokal auf Rechner Rechner 1 gemountete Datei `f` öffnen, so muss er seine `uid` angeben. Gelingt es ihm, sich die `uid = 100` auf der Maschine Rechner 1 zu verschaffen, so weist er diese bei einem entfernten Zugriff vor und der NFS-Server akzeptiert Benutzer *A* ohne weitere Kontrollen als den Eigentümer der Datei `f`. Alle Zugriffsrechte des Eigentümers der Datei stehen *A* nun offen.



NFS bietet als Alternativen zur Authentifikation von Benutzern mittels des Unix-artigen `auth_sys`-Protokolls eine Kerberos (siehe Seite 491) basierte `auth_kerb4`, eine Diffie-Hellman-basierte Authentifikation mit einem 192-Bit Modul `auth_DH192` oder eine Secure SunRPC-basierte Authentifikation. Diese Authentifikationsprotokolle stellen gegenüber der einfachen, Unix-artigen Kontrolle eine erhebliche Verbesserung dar.

zustandsloser Server

Die Designentscheidungen, einerseits die NFS-Server als zustandslose Server zu realisieren und andererseits Zugriffskontrollen nur beim Öffnen von Dateien durchzuführen, erfordert unverfälschbare, eindeutig einem Subjekt zuordenbare Zugriffsausweise zur Repräsentation der beim Dateiöffnen erteilten Zugriffsberechtigungen. Wegen der Zustandslosigkeit der NFS-Server, die keine Informationen über ihre Clients speichern, sind die Zugriffsausweise von den Clients zu verwalten. In NFS werden zu diesem Zweck die File Handles verwendet. Diese erfüllen aber die Anforderung nach Unverfälschbarkeit nicht. File Handles werden zwar unter Nutzung eines Pseudozufallszahlengenerators erzeugt (Operation `fsirand`), so dass die gezielte Konstruktion eines sinnvollen Handles durch den Angreifer erschwert ist. Da sie aber ungeschützt über das Netz übertragen werden, kann ein Angreifer Kenntnisse über Handles erlangen. Wegen seiner Zustandslosigkeit kennt ein NFS-Server keine Zuordnung zwischen File Handle und dem zur Nutzung berechtigten Subjekt. Ein NFS-Server ist deshalb nicht in der Lage, einen von einem regulären NFS-Dämonenprozess erzeugten File Handle von einem Handle, der von einem Angreifer konstruiert wurde, bzw. der vom Angreifer abgefangen und wieder eingespielt wird, zu unterscheiden. Jeder, der einen gültigen File Handle vorweist, erhält Zugriff auf die entsprechende Datei.

Administrative Fehler**Datei-Export**

Der unbeschränkte Export eines Dateisystems, das Home-Verzeichnisse von Benutzern enthält, kann zu erheblichen Sicherheitslücken führen. Falls dieses Dateisystem von einem Client-Rechner gemountet wird, der von einem nicht vertrauenswürdigen Administrator mit Superuser-Privilegien verwaltet wird, setzt sich der exportierende Server unter Umständen schwerwiegenden Angriffen aus. So kann ein Benutzer auf dem nicht vertrauenswürdigen Client-Rechner mit seinen Superuser-Berechtigungen `.rlogin`-Dateien der exportierten Home-Verzeichnisse modifizieren und sich anschließend ohne Angabe eines Passwortes auf dem Server einloggen.

Schreib-Rechte

Erhebliche Sicherheitsprobleme können auch aus dem Export von Dateisystemen resultieren, die Verzeichnisse beinhalten, auf die alle Benutzer eine Schreibberechtigung besitzen. Wird ein solches Dateisystem von einem nicht vertrauenswürdigen Client gemountet, so kann dieser gezielt vorab präparierte Dateien in ein solches Verzeichnis einbringen, da er ja die Schreib-Berechtigung besitzt. Das Platzieren von Trojanischen Pferden ist somit recht einfach möglich.

Version 4**Version 4**

Aufgrund der vielfältigen Sicherheitsprobleme von NFS Version 2 und 3 wurde unter der Leitung einer IETF Arbeitsgruppe eine Version 4 entwi-

ckelt (vgl. RFC 3530), die eine höhere Sicherheit bieten soll. Neben der Verbesserung der verwendeten Authentifikationsverfahren wird auch das Zusammenspiel zwischen NFS-Rechnern und Firewalls vereinfacht. Wie wir bei der Beschreibung von NFS Versionen 2 und 3 gesehen haben, setzt die Kommunikation zwischen einem NFS-Client und NFS-Server voraus, dass der Client über das Mount-Protokoll ein File-Handle erhalten hat. Da das Mount-Protokoll über RPC abgewickelt wird und keine feste Portadresse besitzt, sondern vom Portmapper bei jedem Start des Protokolls einen anderen Port zugewiesen erhält, ist es für statische Paketfilter-Firewalls (vgl. Abschnitt 14.1) sehr schwierig, Filterregeln festzulegen, um den Transfer von NFS-Paketen zu kontrollieren. In der Version 4 werden deshalb spezielle File-Handles verwendet, die der NFS-Client nutzen kann, ohne ein Mount durchgeführt zu haben. Der Datenverkehr zum NFS-Server kann dadurch über den für NFS reservierten Port 2049 abgewickelt werden, so dass diese Portadresse nunmehr auch von Firewalls kontrollierbar ist.

File handle

Zur Verbesserung der Authentifikation verwendet NFS Version 4 RPC-SEC_GSS (siehe RFC 2203), wodurch es dem RPC-Dienst möglich ist, das Generic Security Services Application Programming Interface (GSS-API (siehe RFC 2078)) zu nutzen. Das GSS-API bietet den darauf aufsetzenden Protokollen, wie zum Beispiel dem RPC-Protokoll, eine Reihe von Sicherheitsdiensten wie Integrität, Authentizität von Nachrichten sowie Vertraulichkeit. Das heißt, dass NFS Version 4 lediglich die Sicherheitsmechanismen und die gewünschten Sicherheitsdienste (z.B. Integrität und Authentizität von Nachrichten, Nachrichtenvertraulichkeit) spezifiziert, die beim RPC verwendet werden sollen. Diese Anforderungen werden dann über die RPCSEC_GSS Schicht zum GSS-API weitergereicht (vgl. Abbildung 3.12).

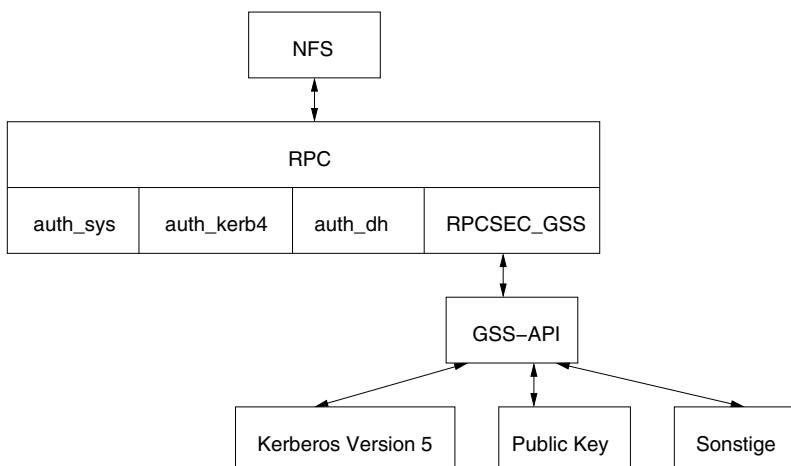
GSS-API

Abbildung 3.12: Einordnung von NFS Version 4

Flexibilität

Durch die beschriebene Einordnung von NFS Version 4 sind auch neue, in Zukunft vom GSS-API unterstützte Sicherheitsmechanismen und -dienste von NFS direkt nutzbar, ohne dass diese Dienste in das RPC-Protokoll integriert werden müssen. Das API unterstützt zumindest zwei Standardprotokolle für kryptografische Techniken. Es handelt sich dabei zum einen um das Kerberos-Protokoll und zum anderen um die RSA Public-Key Verschlüsselung.

3.4.3 Weitere Dienste

Electronic Mail

Der E-Maildienst zählt zu den am weitesten verbreiteten Diensten im Internet. Er erlaubt weltweit den Austausch von elektronischen Nachrichten zwischen verschiedenen Benutzern. Eine Nachricht besteht aus einem Kopfteil, der die E-Mailadressen des Senders und des Empfängers sowie Verwaltungsinformationen enthält, und dem Nachrichteninhalt. Die Übertragung der Nachricht zwischen dem Sender und dem Empfänger erfolgt mittels des Simple Mail Transfer Protokolls (SMTP) (siehe RFC0821). SMTP arbeitet nach dem Store-and-forward-Prinzip, so dass Nachrichten über Zwischenknoten, die die Nachrichten temporär speichern, weitergeleitet werden, bis sie schließlich in der Mailbox des Empfängers abgelegt werden. Der SMTP-Protokollteil auf dem Rechner des Empfängers kann entweder die Mailbox als Datei ansprechen und die Nachrichten auslesen oder das Post Office Protokoll (POP) verwenden, durch das von jedem Punkt aus dem Internet auf die Mailbox zugegriffen werden kann. SMTP erlaubt nur die Übertragung von 7-Bit ASCII-Texten. Mit dem Multipurpose Internet Mail Extension Protokoll (MIME) werden diese Fähigkeiten erweitert, so dass unter anderem internationale Zeichensätze, Multimediadaten oder Binärdaten übertragen werden können.

Sicherheitsprobleme**Vertraulichkeit**

SMTP verfügt über keine Verschlüsselungsfunktionen, so dass alle Mails unverschlüsselt übertragen und auf den Vermittlungsrechnern offen zwischengespeichert werden. Da Datenleitungen über frei verfügbare Sniffer-Programme einfach abgehört oder zwischengespeicherte Mails modifiziert werden können, sind eMails in hohem Maß Angriffen auf ihre Vertraulichkeit und Integrität ausgesetzt. SMTP bietet auch keine Mechanismen zur Überprüfung der Angaben des Absenders, so dass ein Angreifer beliebige Mailadressen als Absendeadressen angeben (E-Mail Spoofing) und sich erfolgreich maskieren kann.

Authentizität

Auch beim E-Mail Datenaustausch treten erhebliche Probleme in Bezug auf die Privatsphäre von Sender und Empfänger auf. Da man über den Sinn und Zweck anonymer E-Mails und anonymer News sicherlich streiten kann,

Privacy

beschränken wir uns im Folgenden nur auf die Sichtweise, dass sich ein Nutzer sehr wohl gegenüber seinem Kommunikationspartner identifizieren möchte, aber ansonsten so wenig wie möglich weitere Informationen über seine Arbeitsumgebung (z.B. welches Betriebssystem, welche Hardware) oder über den Sendekontext (z.B. auf welche Nachricht wird Bezug genommen) preisgeben möchte. Die Kontrollmöglichkeit entspricht dem Recht auf informationelle Selbstbestimmung.

Beim SMTP werden neben der beim Verbindungsaufbau zu übermittelnden IP-Adresse beim Aufruf von Mailing-Kommandos weitere kritische Daten übertragen. Das *Mail*-Kommando spezifiziert ein Argument, das den Pfad enthält, über den im Fehlerfall eine Nachricht zurückzusenden ist. Mit dem *Data*-Kommando werden die eigentlichen Mail-Daten übertragen, die ihrerseits aus einem Header-Teil und den Nutzdaten bestehen. Zu den hier relevanten Header-Feldern gehört der *Return-path*, der die Adresse des Absenders und die Route dorthin enthält, um im Fehlerfall dem Absender die Nachricht wieder zuzustellen. Jeder Server, der an der Zustellung der Nachricht beteiligt ist, fügt seine Informationen zusammen mit Datum und Uhrzeit dem Return-Pfad hinzu, so dass daran die genaue Route abzulesen ist. Zur Identifikation des Absenders dienen ferner auch die Felder *from* mit der Identität des absendenden Agenten (u.a. Maschine, Person), *sender*, das die Identität des Senders enthält, falls der Nachrichtenautor mit dem Sender nicht übereinstimmt, und *reply-to*, in dem die Mailbox(en) für Antworten angegeben werden können. Die Felder *In-Reply-To* und *References* enthalten sensible Kontextinformationen, nämlich die Identifikation der Nachricht, auf die geantwortet wird, und die Identifikation von Nachrichten, auf die verwiesen wird. Neben den im Standard festgelegten Feldern kann eine Mail auch benutzerdefinierte sowie erweiterte Felder (beginnend mit X-) enthalten. Da die meisten der in den Headern der Maildaten enthaltenen Informationen zur Abwicklung des SMTP nicht benötigt werden, können diese Informationen anonymisiert werden, so dass weder gegenüber Dritten noch gegenüber dem Kommunikationspartner Informationen preisgegeben werden, die über die direkte Identifikation des Absenders hinausgehen.

Da es sich bei E-Mail um einen asynchronen Dienst handelt, ist es nicht notwendig, dass die Nachrichten sofort beim Empfänger ankommen. Diese Eigenschaft machen sich die Remailer zunutze, die sich im Bereich der Anonymisierung von E-Mails und News durchgesetzt haben. Die ursprüngliche Konzeption von umkodierenden Mailern geht auf D. Chaum und seine umkodierenden Mixe [42] zurück. Ein Nutzer eines Remailer-Dienstes muss seine Daten explizit zu einem Remailer senden. Dieser entfernt die Header-Informationen und leitet danach die Nachrichten weiter. Zum Schutz vor Verkehrsflussanalysen verzögern die Remailer die Nachrichtenweiterleitung

Profilbildung

Remailer

und streuen zusätzlich noch Dummy-Nachrichten in den Datenstrom ein. Neben diesen Eigenschaften, die die Typ 1 Remailer oder Cypherpunk-Remailer charakterisieren, gibt es noch die Typ 2 Remailer, die so genannten Mixmaster. Deren Nutzung erfordert jedoch einen spezifischen E-Mail Client, der die Nachrichten verschlüsselt und auf eine einheitliche Länge normiert. Remailer können auch kaskadierend (vgl. Abbildung 3.13) eingesetzt werden. Dies setzt voraus, dass die zu versendende Mail eine Kette von verschlüsselten Remailer-Adressen spezifiziert. Jeder Remailer entfernt den Eintrag, der mit seinem öffentlichen Schlüssel verschlüsselt wurde, und sendet die Nachricht an die Adresse, die in dem von ihm entschlüsselbaren Eintrag spezifiziert ist. Bei einer Remailer-Kaskade kennt nur der erste Remailer der Kette den Absender der Mail und der letzte kennt die Adresse des Mailempfängers. Üblicherweise verwendet man jedoch nur einen Remailer.

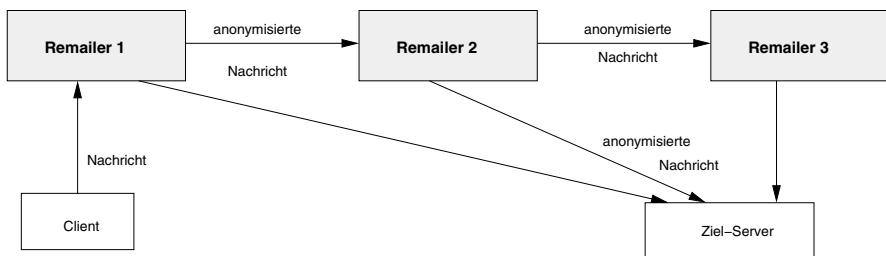


Abbildung 3.13: Kaskadierende Remailer

MIME

Verwendet man zur Codierung von E-Mailnachrichten, wie heute vielfach üblich, das Multipurpose Internet Mail Extension Protokoll (MIME), so ist zu beachten, dass die MIME-Nachrichten neben den Daten auch Befehle beinhalten können, die bei der interpretativen Bearbeitung der Mail zur Ausführung kommen. Zulässige Befehle sind beispielsweise die Initiierung einer Dateübertragung mittels FTP, der anonyme Dateitransfer mittels TFTP oder ein Dateizugriff. Bei der Ausführung dieser Kommandos wird auf das lokale Dateisystem des Mail-Empfängers zugegriffen. Es ist somit wichtig, die zugänglichen Dateibereiche so zu schützen, dass ein Angreifer weder unzulässige Schreib- noch unerwünschte Lese-Zugriffe erhält. Gelingt es ihm zum Beispiel eine präparierte `.rhost`-Datei auf das System des Mail-Empfängers zu übertragen, so kann er anschließend über ein `rlogin` Zugriff auf dessen System erlangen. Erweiterungen des E-Maildienstes um Sicherheitsfunktionen (vgl. Abschnitt 14.6) werden unter anderem durch PGP und S/MIME zur Verfügung gestellt.

File Transfer (FTP)

Dateitransfer

Das File Transfer Protocol (FTP) wird zur Dateiübertragung verwendet und bietet unterschiedliche Übertragungsmodi (z.B. Text- und Binärmodus) so-

wie Datencodierungen an. FTP ist ein Client-Server-basiertes Protokoll, mit dessen Hilfe Daten, die vom Server verwaltet werden, von Clients geladen (download) oder Daten zum Server übertragen (upload) werden können. FTP setzt man häufig zum Zugriff auf öffentlich verfügbare Dateien ein. Diese werden dann gewöhnlich über ein anonymes FTP zur Verfügung gestellt, das keine echte Authentifikation des Benutzers erfordert. Beim anonymen FTP identifiziert sich der Benutzer üblicherweise durch die Angabe seines Namens bzw. seiner Benutzerkennung. Beim anonymen FTP übertragen heutige Browser meist ein Passwort, das den jeweiligen Browser kennzeichnet (z.B. mozilla@), daneben besteht aber i.d.R. auch noch die Möglichkeit, die in den Voreinstellungen des Browsers eingetragene E-Mail-Adresse des Nutzers zu übertragen.

Sicherheitsprobleme von FTP

Da sich bei einem anonymen FTP beliebige Clients auf einem FTP-Server einloggen können, müssen die Dateibereiche des Servers vor Schreibzugriffen geschützt werden. Andernfalls könnte ein Angreifer beliebige Dateien auf dem Server speichern und sich dadurch den Zugang als berechtigter Benutzer eröffnen. Lesende Zugriffe sollten ebenfalls stark beschränkt werden, insbesondere sollte keinesfalls die Passwortdatei des FTP-Servers zugreifbar sein, um das gezielte Durchführen eines Passwort-Crackangriffs zu verhindern.

anonymes FTP

Ein besonderes Sicherheitsrisiko stellt eine Variante des FTP-Protokolls dar, nämlich das Trivial File Transfer Protokoll (TFTP). TFTP ist ein UDP-basierter File Transfer Dienst, der einen Systemzugang ohne Authentifikation ermöglicht. TFTP wird von plattenlosen Rechnern beim Booten benötigt. Der Dienst sollte deshalb so konfiguriert werden, dass nur der Transfer der zum Booten erforderlichen Verzeichnisse erlaubt ist. In allen anderen Umgebungen sollte der TFTP-Dienst nicht zur Verfügung stehen.

TFTP

Wie bei den zuvor behandelten Protokollen, so werden auch bei der Nutzung des FTP eine ganze Reihe sensibler Informationen übertragen, die den Nutzer betreffen. FTP arbeitet sitzungsorientiert und unterscheidet zwischen Kontrollnachrichten sowie den eigentlichen Daten. Zur Übermittlung der Kontrollnachrichten stellt FTP eine eigene Kontrollverbindung her, die über die gesamte Sitzung bestehen bleibt, während für jede Sende- und Empfangsaktion eine neue Datenverbindung aufgebaut wird. Sowohl bei der Kontrollverbindung als auch bei jedem Aufbau einer Datenverbindung wird die IP-Adresse des Clients übertragen. Daneben werden beim Aufruf von FTP-Kommandos weitere kritische Daten als Parameter gesendet. Dazu gehört das *User*-Kommando, das den Aufrufer über die Angaben eines ASCII-Strings identifiziert, das *Pass*-Kommando, das das Nutzer-Passwort

Privacy

als Parameter enthält sowie das *Acct*-Kommando, dessen Parameter die Nutzer-Kennung identifiziert.

Anonymisierung

Da der Benutzername, das Passwort und die Kennung bei einem nicht-anonymen FTP notwendig sind, um den Nutzer gegenüber dem FTP-Server zu authentifizieren, ist deren Anonymisierung gegenüber dem Kommunikationspartner nicht möglich. Auch die IP-Adresse wird benötigt, um die Datenverbindung zum Client aufzubauen. Mit einer Verschlüsselung der Header-Daten des FTPs ist jedoch eine wirksame Anonymisierung gegenüber Dritten erreichbar. Bei einem anonymen FTP kann die als Passwort verwendete E-Mail Adresse anonymisiert und z.B. durch eine fiktive E-Mail-Adresse ersetzt werden, um Rückschlüsse auf den Nutzer zu verschleieren.

Telnet-Dienst

Terminal-Zugang

Der Telnet-Dienst stellt einen einfachen Terminal-Zugang zu einer Maschine über eine bidirektionale TCP-Verbindung zur Verfügung. Telnet dient in erster Linie zum Aufbau einer Verbindung mit einem entfernten Komandointerpreter. Beim Verbindungsauftakt ruft der Telnet-Dämon die Login-Prozedur der Maschine zur Identifikation und Authentifikation des zugreifenden Benutzers auf.

Sicherheitsprobleme von Telnet

Sniffer

Die zur Authentifikation benötigten Passwörter und Daten werden im Klartext übertragen. Unter Verwendung von Sniffer-Programmen ist es daher für Angreifer sehr einfach, Telnet-Verbindungen zu belauschen und die beim Verbindungsauftakt übermittelten Passwörter zu beobachten. Gegen das Abhören der im Klartext übertragenen Passwörter schützt deren Verschlüsselung (vgl. Kapitel 7) oder die Verwendung von Einmal-Passwörtern (vgl. Kapitel 10).

3.5 Web-Anwendungen

In heutigen IT-Systemen sind Web-basierte Anwendungen, kurz Web-Anwendungen, sehr weit verbreitet. Sie ermöglichen eine einfache, Web-basierte Zusammenarbeit zwischen einem Dienstanbieter und den Nutzern. Typischerweise stellt der Dienstanbieter seine Web-Anwendungen über einen Web-Server oder ein Portal, beispielsweise einem Apache oder IIS-Server, zur Verfügung. Web-Anwendungen werden meist in Programmiersprachen wie PHP, JSP, Perl oder Python programmiert. Sie werden auf dem Web-Server ausgeführt und verarbeiten dabei sehr häufig Daten, die in SQL-Datenbanken abgelegt sind, die im Intranet des Dienstanbieters verwaltet werden. Die Nutzung eines Dienstes ist für Clients sehr einfach mittels des HTTP Protokolls möglich.

3.5.1 World Wide Web (WWW)

Ein sehr großer Teil heutiger Sicherheitsprobleme in Unternehmen resultiert aus verwundbaren Web-Anwendungen. Bevor wir auf die wichtigsten Schwachstellen von Web-Anwendungen eingehen, erläutern wir kurz einige technische Grundlagen, wie das HTTP-Protokoll, dynamische Web-Seiten oder aber auch das Cookie-Konzept.

Das World Wide Web (WWW) integriert bestehende Internetdienste durch eine einheitliche Adressierung und Bedienung sowie durch ein standardisiertes Dokumentenformat. Man unterscheidet WWW-Objekte, -Server und -Browser. WWW-Objekte können Dokumente oder beliebige Datenbestände wie Nachrichten und Datenbanken sein. Spezielle WWW-Objekte sind die Web-Seiten, die in der Hypertext Markup Language (HTML) geschrieben und mittels des Hypertext Transport Protokolls (HTTP) übertragen werden. Web-Seiten sind Multimediadokumente, die insbesondere Verweise (engl. *link*) auf andere WWW-Objekte enthalten können. Durch diese Verweise entsteht ein weltweites Geflecht von Objekten, woraus der Name Web (deutsch *Netz*) abgeleitet wurde. Hypertextdokumente beinhalten häufig ausführbaren Code, wie zum Beispiel Java-Applets, der in Form von mobilem Code von Web-Clients herunter geladen und ausgeführt werden kann.

Web-Browser wie der Microsoft Internet Explorer oder Firefox stellen Dienste zum Navigieren im WWW, zur Darstellung von Seiten sowie zur Interaktion mit Servern zur Verfügung. Mittels eines Browsers können WWW-Objekte von einem Server abgerufen, aber auch interaktive Benutzereingaben an den Server weitergeleitet werden. Letzteres ermöglicht die Verwendung des WWW zur Abwicklung interaktiver Aktivitäten. Beispiele hierfür sind Datenbankabfragen oder der elektronische Einkauf mit einem interaktiven Aushandeln von Vertragsbedingungen. Aufgrund der Zustandslosigkeit des HTTP-Protokolls muss bei jedem Zugriff auf ein WWW-Objekt eine TCP-Verbindung zwischen Client und Server aufgebaut werden. Zur Effizienzsteigerung speichert man deshalb häufig das benötigte Objekt in einem Cache eines lokalen Proxy-Servers, der als Stellvertreter für den Web-Server fungiert. Bei mehrmaligen Zugriffen auf das Objekt kann somit auf den aufwändigen Verbindungsaufbau verzichtet werden. Um Objekte von einem Server zum Browser übertragen zu können, ist es notwendig, Objekte eindeutig zu benennen. Die Benennung erfolgt über den Uniform Resource Locator (URL), bzw. die URI (Uniform Resource Identification).

Browser

URL

Dynamische Web-Seiten

Ursprünglich waren HTML-Seiten statische Seiten. Heute ist es jedoch üblich, dass HTML-Seiten dynamisch erzeugt werden, um beispielsweise über aktuelle Daten des Aufrufers den Seiteninhalt zu beeinflussen. Zu den ersten

Technologien, die für die Gestaltung dynamischer Seiten verwendet wurden, gehört die CGI-Skriptsprache (Common Gateway Interface). Dynamische Webseiten werden hierbei über CGI-Skripte erzeugt. Ein CGI-Script kann in einer beliebigen interpretierbaren Programmiersprache erstellt sein. CGI in Kombination mit Perl wird für kleinere Anwendungen noch wie vor genutzt, für komplexere Anwendungen kommen aber heute in der Regel Technologien wie PHP, Active Server Pages (ASP.net) von Microsoft, Java Server Pages (JSP) zum Einsatz. Im Folgenden erläutern wir die generelle Vorgehensweise bei der Erstellung dynamischer Seiten am Beispiel von CGI.

Ablauf

Ein Skript-Programm zur Erzeugung dynamischer Seiten wird in der Regel in dem Verzeichnis `cgi-bin` des Servers abgelegt und innerhalb einer HTML-Seite über einen URL identifiziert. CGI-Programme werden mit den Rechten des Web-Servers ausgeführt, d.h. je mehr Rechte dieser Prozess besitzt, desto mehr Rechte haben auch die CGI-Skripte. Ein Client kann die Skript-Ausführung über den Zugriff auf eine HTML-Seite initiieren. Gängige Vorgehensweisen sind hier der parameterlose Aufruf über eine URL oder über ein Formular. Ein parameterloser Aufruf beispielsweise des Skripts `/cgi-bin/hello.cgi` über eine HTML-Seite könnte wie folgt aussehen:

```
<a href="/cgi-bin/hello.cgi" > Test aufrufen < /a>
```

Ein formularbasierter Aufruf könnte in folgender Form ablaufen:

```
<form action="/cgi-bin/search.pl" method="get">
```

Das hierbei aufgerufene CGI-Skript erzeugt ein auszufüllendes Formular im Client-Browser, das der Client bzw. der Benutzer, der die Seite angefordert hat, ausfüllen und an den Server zurücksenden kann. Nach Erhalt der Antwort führt der Server das CGI-Skript aus, in dem beispielsweise mit den Angaben des Clients eine Anfrage an eine Datenbank beim Server gestartet wird (vgl. Beispiel 3.7). Die Eingabedaten werden vom Server in speziellen Variablen z.B. `QUERY_STRING` gespeichert, die im Skript abgefragt werden können.

Unterschied zu mobilem Code

An dieser Stelle möchten wir auf den wichtigen Unterschied zwischen mobilem Code bzw. aktiven Inhalten und CGI-Skripten hinweisen. Aktive Inhalte werden über den lokalen Browser auf den Rechner des Zugreifers heruntergeladen und dort vom Browser ausgeführt; sie stellen also eine Bedrohung für den Rechner des Clients dar. Demgegenüber werden CGI-Skripte auf dem Web-Server ausgeführt und nur das Ergebnis der Ausführung wird an den Client-Rechner übertragen. Hierbei ist also zunächst der Server bedroht, nicht der Client.

Beispiel 3.7 (CGI-Szenario)

Ein Szenario, in dem ein Client über ein Formular eine dynamische HTML-Seite erzeugt und mittels eines CGI-Skripts eine Datenbankanfrage initiiert, ist in Abbildung 3.14 zusammengefasst. Der Benutzer lädt hierbei zunächst eine HTML-Seite mit einem Formular vom Web-Server in seinen Browser. Er füllt das Formular mit Daten aus (1), die in diesem Szenario eine Suchanfrage an den Datenbankserver auf der Seite des Web-Servers spezifizieren. Das ausgefüllte Formular wird (2) an den Web-Server zurück übertragen, der das im Formular angegebene CGI-Skript (3) aufruft. Das CGI-Programm setzt die Client-Daten in eine Datenbankanfrage um und leitet diese an die

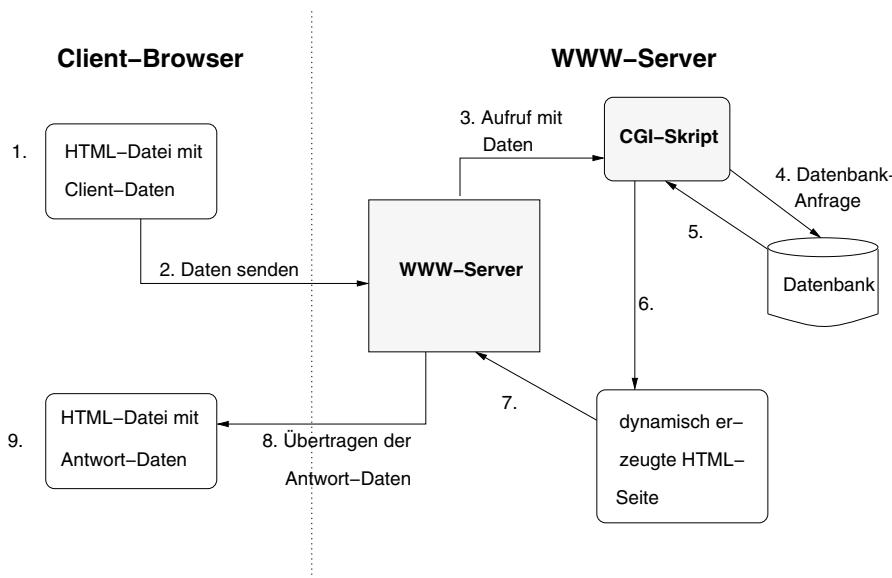


Abbildung 3.14: CGI-Szenario

Datenbank (4) weiter. Mit den von der Datenbank zurückgegebenen Werten (5) erzeugt das Skript eine neue HTML-Seite (6), die über den Web-Server (7) an den Client-Browser (8) übermittelt wird. Der Browser stellt die Suchergebnisse der Anfrage am Bildschirm des zugreifenden Benutzers dar (9).



Wie bereits weiter oben angesprochen, wird heute in vielen Fällen PHP zur Erzeugung dynamischer Seiten verwendet. Die Syntax von PHP⁷ ist an C bzw. C++ angelehnt. Analog zu CGI werden auch PHP-Skripte serverseitig interpretiert. Demgegenüber interpretiert die Scriptsprache Javascript den

⁷ Ursprünglich stand PHP für Personal Home Page Tools jetzt steht es für Hypertext Preprocessor.

Code clientseitig. Beim Einsatz von PHP sendet der Web-Server seine Ausgabe an den Browser des aufrufenden Clients zurück. PHP wird in der Regel in der so genannten LAMP-Kombination (Linux-Apache-MySQL-PHP) eingesetzt.

Wie auch bei CGI oder anderen zur dynamischen Erzeugung von Seiten eingesetzten Scriptsprachen, so liegen auch bei PHP die größten Sicherheitsprobleme im Bereich der Programmierung und hier insbesondere in der mangelhaften oder fehlenden Überprüfung von Eingabe- und Ausgabewerten, wodurch es zu Bufferoverflow-, Cross-Site-Scripting- oder aber auch SQL-Injection (siehe nachfolgende Abschnitte) kommen kann.

Cookies

*zustandsloses
Protokoll*

Cookie

Cookie-Einsatz

Das HTTP-Protokoll ist, wie gesagt, zustandslos, so dass jede Anfrage als ein eigenständiges Ereignis behandelt wird und auch bei einem wiederholten Seitenzugriff alle erforderlichen Verwaltungsmaßnahmen wie beispielsweise eine Passwort-Authentifikation beim Zugriff auf eine Web-Seite durchgeführt werden müssen. Die Browser-Software erledigt wiederholte Authentifikationen automatisch, indem die benötigten Passwörter zwischen gespeichert und anschließend transparent für den Benutzer zum Server übertragen werden. Mittels des Cookie-Konzepts wird diese Vorgehensweise des automatischen und transparenten Sammelns von Informationen verallgemeinert, um eine zustandsbehaftete HTTP-Sitzung zu emulieren. Ein Cookie ist eine vom Server generierte Datenstruktur, in die der Server Informationen über den zugreifenden Benutzer codiert, so dass der Server diese Informationen beim wiederholten Zugriff auswerten kann. Cookies werden dazu auf dem Rechner des zugreifenden Benutzers gespeichert. Auf Unix-Systemen werden Cookies typischerweise in einer Datei im Heimverzeichnis des Benutzers und unter Windows typischerweise unter C:\windows\cookies abgelegt. In Beispiel 3.8 werden die Struktur einer Cookie-Datei sowie die darin festgehaltenen Informationen erläutert. Cookies enthalten keinen ausführbaren Code, sondern nur Daten, die zwar vom Server gelesen, aber nicht auf dem Rechner des Benutzers ausgeführt werden.

Ursprünglich sollten Cookies das elektronische Einkaufen erleichtern, indem ein Benutzer aus einem Angebot Waren auswählt, die er kaufen möchte. Der Server speichert die Kennungen dieser Produkte beim Nutzer und kann diese Informationen wieder abrufen, um die Bestellung automatisch, also sehr bequem für den Käufer, auszufüllen. Mit Cookies sollte es Web-Servern ermöglicht werden, sich auf die (vermuteten) Bedürfnisse eines Benutzers einzustellen. So kann ein Betreiber von Web-Diensten mithilfe von Cookies einen Benutzer beim Betreten der Startseite seines Servers identifizieren und nachfolgende Zugriffe auf andere Seiten eindeutig diesem Benutzer zuordnen. Auf diese Weise ist der Betreiber in der Lage, ein Nutzungsprofil

des Kunden zu erstellen. Ein solches Profil kann vielfältige Informationen über den Kunden liefern und ihn so zur Zielperson z.B. für Werbebotschaften machen, die in WWW-Seiten eingeblendet werden. Gibt der Benutzer dann noch seinen Namen bekannt, was im Rahmen einer Bestellung auf natürliche Weise erfolgt, kann dem Profil sogar ein Name zugeordnet werden.

Beispiel 3.8 (Cookie-Datei)

Ein Auszug aus einer Cookie-Datei könnte wie folgt aussehen:

Domäne	Zugriffe	Zeit	Name	Wert
.altavista.com	TRUE / FALSE	946641601	AV_UID	d4099387c9c
.spiegel.de	TRUE / FALSE	946684800	RMID	839f01ad364fe
www.heise.de	FALSE / FALSE	1293753600	ticker	T=guninski

Tabelle 3.2: Cookie-Datei (Auszug)

Ein Cookie besteht aus sieben Informationsfeldern. Das erste Feld enthält den Namen derjenigen Domäne, die den Cookie erzeugt und gespeichert hat und die dessen Daten potentiell lesen kann. Die Zugriffsbeschränkungen zum Lesen der Cookie-Informationen werden mit den nächsten drei Einträgen erfasst. Falls alle Rechner der Domäne Zugriff auf den Cookie besitzen dürfen, ist das zweite Informationsfeld mit TRUE belegt. Danach wird der Pfad (z.B. '/') der Domäne angegeben, in dem der Cookie gültig ist. Der vierte Eintrag spezifiziert, ob der Zugriff auf die Cookie-Datei nur bei einer mit SSL (vgl. Kapitel 14.4) gesicherten Verbindung zulässig ist. Die Lebensdauer (als Unix-Zeitangabe), der Name des Cookies und dessen Wert vervollständigen die Informationsfelder.

Datenstruktur



Same Origin Policy

Das Sicherheitsmodell für Web-Anwendungen wurde bereits 1991 von Netscape für den damaligen Netscape Navigator die Same Origin Policy eingeführt. Dieses Sicherheitskonzept hat sich bis heute in allen Browsern durchgesetzt und ist das Standardmodell heutiger Browser. Hintergrund für das einfache Regelwerk ist die sehr weit verbreitete Nutzung von Skriptsprachen wie JavaScript oder VBScript in Web-Anwendungen. Das heißt, dass in Web-Seiten sehr häufig Skriptcode eingebettet wird, der dann im Kontext des jeweiligen Browsers des Clients zur Ausführung kommt. Durch die zunehmende Verwendung von Konzepten wie Frames, Inner Frames

Sicherheitsmodell

und Popup-Fenstern kann ein Skript zusätzlich auch auf andere Dokumente im Internet zugreifen. Im Zusammenhang mit Ajax (Asynchronous JavaScript and XML) verstärkt sich die Problematik. Ajax ermöglicht zusätzlich noch eine asynchrone Datenübertragung zwischen dem Browser und dem Web-Server mittels *XMLHttpRequest* Anfragen des Skripts. Mit derartigen Anfragen kann das Skript asynchron Daten an dem Web-Server übertragen und auch selber Daten vom Server empfangen; die Web-Seite muss dazu nicht komplett neu geladen werden.

Same Origin Policy

Mit der Same Origin Policy wird nun das Ziel verfolgt, einem Skript, das im Browser im Kontext einer Webseite ausgeführt wird, nur eingeschränkte Zugriffe auf weitere Datenobjekte im Web zu gestatten. Die Policy schränkt die weiteren lesenden Zugriff des Skripts auf die Inhalte von solchen Seiten ein, die von der gleichen Quelle (engl. *same origin*) stammen wie die Seite, in der das Skript eingebettet ist. Der Ursprung eines Datenobjekts wird dabei über die Kombination von drei Attributen charakterisiert: das Protokoll, der Port und die Domäne.

Betrachten wir als Beispiel ein Skript, das in die Seite <http://www.tum.de/cms/index.php> eingebettet ist. Folgende Zugriffe des Skripts sind gemäß der Same Origin Policy erlaubt bzw. verboten:

- Der Zugriff auf <http://www.tum.de/studenten/index.php> ist erlaubt, da die Domäne (hier www.tum.de), das Protokoll (hier HTTP) und auch der Port (hier Port 80, HTTP) identisch sind.
- Der Zugriff auf <http://www.tum.de/index.html> ist ebenfalls erlaubt.
- Der Zugriff auf <https://www.tum.de/index.html> ist jedoch nicht erlaubt, da anstelle des ursprünglichen HTTP-Protokolls hier das HTTPS Protokoll genutzt wird.
- Der Zugriff auf <http://www.tum.de:8080/index.html> ist ebenfalls nicht erlaubt, da anstelle des ursprünglichen Ports 80 der Port 8080 angesprochen wird.
- Auch der Zugriff auf die Seite <http://www.in.tum.de/index.php> ist nicht erlaubt, da anstelle der Domäne www.tum.de die Domäne www.in.tum.de angesprochen wird.

XSS Angriffe

Die Einhaltung der Policy-Regeln wird vom jeweiligen Browser der Clients kontrolliert. Da die Policy-Regeln jedoch sehr einfach sind, können sie auch ebenso einfach von Anwendungen umgangen werden. So kann beispielsweise ein HTML-Dokument über das `script`-Objekt JavaScripte von fremden Domains einbinden. Diese Skripte erhalten dieselben Rechte wie das umgebende JavaScript. Angriffe wie Cross Site Scripting (XSS) (s.u.)

nutzen dies aus, um die Same Origin Policy zu umgehen. Die Policy wird auch beispielsweise dann umgangen, wenn eine Ajax-Anwendungen auf eine entfernte Web-Seite über einen lokalen Proxy aus der eigenen Domäne zugreift. Die einfachen Same Origin Policy Regeln sind somit für heutige Web-Anwendungsszenarien unzureichend, da sie sehr einfach umgangen werden können.

3.5.2 Sicherheitsprobleme

Im Folgenden gehen wir auf einige der wichtigsten Sicherheitsprobleme in Web-basierten Anwendungen etwas genauer ein. Eine ausführliche Beschreibung der zugrunde liegenden Web-Technologie und der Sicherheitsprobleme von Web-Anwendungen findet man unter anderem in [169].

Mögliche Verwundbarkeiten und Probleme

Web-Anwendungen bestehen häufig aus drei Komponenten, nämlich der Client-, der Web-Server- und der SQL-Datenbank-Komponente. Auf der Client-Seite kommuniziert ein Nutzer mit dem Web-Server über das Internet unter Nutzung herkömmlicher HTTP-Verbindungen und unter Nutzung eines Browsers auf dem Client-Rechner. Ein Web-Server ist in der Regel mittels Firewalls zumindest rudimentär vor einigen Angriffsklassen geschützt. Ein solcher Server bietet in der Regel mehrere Web-Anwendungen zur Nutzung durch Clients an. Der Web-Server kommuniziert seinerseits mit Servern im internen Unternehmensnetz, wie zum Beispiel mit SQL-Datenbank-Servern.

Komponenten

Schwachstellen und Verwundbarkeiten ergeben sich sowohl für den Client-Rechner, als auch für den Web-Server und die internen Datenbanken, als auch für die Daten, die über unterschiedliche Kommunikationsnetze zwischen diesen Komponenten ausgetauscht werden. Die Kommunikationsverbindung zwischen Client und Web-Server ist allen klassischen Internet-Angriffen wie beispielsweise Sniffing, Man-in-the-Middle, DNS-Rebinding, Cache-Poisoning oder auch Session Hijacking ausgesetzt. Client-Rechner sind einer Vielzahl von Angriffen wie beispielsweise durch Phishing, Web-Server-Spoofing, Cross Site Scripting (siehe unten) oder auch Session Riding ausgesetzt.

Schwachstellen

Sehr stark zugenommen haben in den letzten Jahren Angriffe auf Client-Rechner, die man als Drive-by-Download Angriffe bezeichnet. Dies charakterisiert Angriffe, bei denen der Nutzer unbeabsichtigt, so zugesagen im Vorbeifahren (Drive-by) Schadsoftware auf seinen Rechner herunter lädt, ohne dass er aktiv dieses Herunterladen initiiert hat. Ein derartiger Angriff kann allein durch das Ansehen einer Web-Seite oder einer E-Mail auftreten. Die Angriffe nutzen häufig Schwachstellen in Browser-Konfigurationen

Drive-by-
Download

aus, um unter Nutzung aktiver Skriptsprachen Schadcode auf dem Client-Rechner zur Ausführung zu bringen. Häufig werden derartige unfreiwillige Downloads auch im Huckepack-Prinzip zusammen mit einer gewünschten Anwendungen auf den Client-Rechner heruntergeladen und zur Ausführung gebracht. Eine restriktive Konfigurierung der Browser, die Einspielen der neuesten Browser-Updates sowie auch eine restriktive Aktivierung von Skript-Sprachen sind wirksame Mittel, um die Angriffsmöglichkeiten für derartige Angriffe zu reduzieren.

unsichere Programmierung

Weitere Sicherheitsprobleme ergeben sich insbesondere auch für den Web-Server und die Web-Anwendungen, die auf dem Server zur Ausführung kommen. Hauptursachen für Sicherheitsprobleme bei Web-Anwendungen sind einmal mehr Fehler bei deren Programmierung. Dazu gehören fehlende Eingabefilterungen oder Format-String Angriffe ebenso, wie eine fehlende oder unzureichende Authentifikation bzw. Zugriffskontrolle. Derartige Fehler können durch Buffer-Overflow-Angriffe (siehe Abschnitt 2.2), Spoofing-Angriffe oder spezielle Angriffe, wie Cross Site Scripting-Angriffe (siehe unten), ausgenutzt werden. Entsprechende Angriffe können sowohl Daten, die beim Betreiber des Web-Servers verwaltet werden, kompromittieren, als auch direkt zu Angriffen auf die Rechner der Nutzer, also der Clients führen.

Verwundbar sind aber auch die internen Datenbanken, die über die Web-Anwendungen angesprochen werden. Die größten Schwachstellen und Angriffspunkte sind hierbei die so genannten SQL-Injection Angriffe (siehe unten) oder auch Command-Injection Angriffe, die wiederum auf unsicheres Programmieren zurückzuführen sind.

Allgemeine Probleme

Das HTTP-Protokoll sieht keinerlei Maßnahmen zur Authentifizierung der beteiligten Partner, zur Verschlüsselung des Datenverkehrs oder zur Sicherstellung der Datenintegrität vor. Damit sind Angriffe, wie das Maskieren von Serveradressen oder das Vortäuschen gefälschter URLs durch einen Client, möglich. Sensible Informationen wie Kreditkartennummern können von Angreifern abgefangen bzw. modifiziert und wiedereingespielt werden.

Caching

Aus der Protokollierung der Zugriffshistorie von Benutzern können sich zusätzliche Probleme ergeben, die zu einer Bedrohung der Privatsphäre eines Benutzers führen. Handelt es sich bei dem Web-Client zum Beispiel um ein öffentlich zugängliches Terminal, das von unterschiedlichen Benutzern nacheinander genutzt wird, so ist diese Bedrohung offensichtlich. Es ist ein Leichtes, sich mittels der protokollierten Daten einen Überblick über die Zugriffe vorheriger Benutzer zu verschaffen. Gravierendere Bedrohungen der Privatsphäre können sich einerseits aus der Verwen-

dung von Cookies und andererseits generell aus der Nutzung des Web ergeben.

Obwohl mit Cookies „nur“ Daten und nicht aktive Inhalte übertragen werden, und sie deshalb ein vergleichsweise geringes Gefährdungspotential für die Sicherheit eines Endsystems darstellen, sollte man die mit dem Cookie Konzept verbundenen Bedrohungen keinesfalls vernachlässigen. Anhand der gespeicherten Cookie-Informationen ist es Web-Servern möglich, Nutzungsprofile zu erstellen. Die Cookie-Informationen, die auch benutzerbezogene Passwörter für Web-Seiten umfassen können, werden in einer Datei im Dateisystem auf dem Client-Rechner gespeichert. Dadurch sind diese Daten Angriffen ausgesetzt, die sich aus der Ausführung von aktiven Inhalten oder durch unautorisierte Zugriffe als Folge eines Buffer-Overflow Angriffs ergeben können. Generell problematisch am Cookie-Konzept ist, dass, sofern der Nutzer nicht explizit das Setzen von Cookies verhindert, der Datenaustausch mittels Cookies vollständig im Hintergrund erfolgt, ohne dass der Benutzer über Inhalte, Zweck, Umfang, Speicherdauer oder Zugriffsmöglichkeiten auf die Cookie-Datei informiert wird. Ein weiteres Problem entsteht, wenn ein Cookie zur späteren Identifizierung des Benutzers dienen soll. Der deutsche Gesetzgeber verlangt in seit 1997 seinem Informations- und Kommunikationsdienste-Gesetz, bekannt unter dem Namen Multimediaigesetz, dass Diensteanbieter ihre Benutzer davon unterrichten, wenn ein solcher Cookie gesetzt werden soll. Dies wird bisher allerdings kaum beachtet. Das Anzeigen einer Warnung, wie im Browser-Fenster des Benutzers, reicht nämlich zur Erfüllung der gesetzlichen Pflichten nicht aus.

Zugriffsprofile sind aber auch sonst relativ einfach zu erstellen, da mit jedem Zugriff auf eine HTML-Seite über das verwendete HTML-Protokoll eine Vielzahl von Informationen, die die Privatsphäre des Zugreifers verletzen können, übertragen werden. Eine HTML-Anfrage eines Clients besteht aus mehreren Headern und dem eigentlichen Nachrichtenkörper. Zu den Header-Feldern mit kritischen Informationen gehört der *Referer*-Eintrag. Dabei handelt es um die URL derjenigen Seite, aus deren Kontext der aktuelle Zugriff auf eine Seite des Servers erfolgt ist. Das *Referer*-Feld liefert somit dem Server Kontextinformationen über seinen Client. Ein Beispiel dafür ist die Information, dass der Aufruf über eine Suchmaschine erfolgt ist. In diesem Fall enthält der *Referer*-Eintrag die ganze Trefferliste (mit u.U. interessanten Hinweisen auf Konkurrenzangebote) sowie auch die Suchbegriffe, woraus sich Rückschlüsse auf die Wünsche des Nutzers ziehen lassen. Über die verschiedenen *Accept*-Felder werden die bevorzugten Zeichensätze, die Sprache, Kodierungsverfahren etc. des Absenders festgelegt. Aus diesen Daten lässt sich bereits eine recht genaue Charakterisierung des Nutzers ableiten. Informationen

Cookie-Probleme

Benutzerprofil

über die IP-Adresse des Clients werden häufig in den Feldern *Client-IP*, *X-Forwarded* und *Cache Control* übertragen. Das *User-Agent*-Feld identifiziert den anfragenden User-Agenten, und der *From*-Eintrag enthält, falls verwendet, die E-Mail-Adresse des Benutzers, der dem User Agenten assoziiert ist. Einen Eindruck von der Fülle an Informationen, die bei jedem Web-Zugriff über den Zugreifer preisgegeben werden, gewinnt man durch den Zugriff auf Testsuiten wie <http://privacy.net/analyze> oder <http://www.browserspy.dk>.

Web-Server-Spoofing

Web-Spoofing

Das Maskieren von Web-Server Adressen, das so genannte Web-Spoofing, ist aufgrund häufig fehlender Authentifikationsmaßnahmen sehr einfach möglich. Bei einem Spoofing-Angriff präsentiert der Angreifer seinem Opfer eine präparierte Web-Seite, mit der er die Originalseite maskiert.

URL-Überprüfung

Um Spoofing-Angriffe zu erkennen, können sicherheitsbewusste Benutzer natürlich die URL des Servers inspizieren, zu dem eine Verbindung aufgebaut wird. Wenn jedoch, wie es laut RFC 1738 gestattet ist, in einer URL anstelle eines Rechnernamens (Hostname) auch eine IP-Adresse angegeben wird, führt dies zu einer deutlichen Erschwernis bei der Überprüfung der Server-Identität.

Typejacking

Eine weitere Schwierigkeit bei der benutzerseitig durchzuführenden Identitätsprüfung einer URL besteht dabei, dass URLs häufig aus langen Hostnamen bestehen. Ein beliebter Angriff, man nennt das Typejacking, besteht darin, den korrekten Namen des Servers minimal abzuändern, um auf diese Weise die Benutzer auszutricksen, die sich mit einem oberflächlichen visuellen Abgleich der Namen begnügen. Ein Beispiel ist ein Angriff, bei dem der paypal-Server unter dem Namen paypai maskiert wurde. Bei vielen Fonts sind die Kleinbuchstaben i und l kaum zu unterscheiden, so dass die minimale Änderung der URL leicht zu übersehen ist. Häufig werden Benutzer dazu verleitet, auf derartige gefälschte Seiten zu gehen, indem ihnen eine E-Mail mit dem Link auf die gefälschte Seite zugesandt wird. Da viele E-Mailprogramme den Benutzern das direkte Anklicken eines solchen Links in einer Mail gestatten, ohne dass der Benutzer also die korrekte Adresse noch einmal explizit im Browser eingeben muss, ist ein solcher Spoofing-Angriff häufig erfolgreich.

SSL-Verbindung

Aber selbst wenn die beteiligten Partner eine vermeintlich sichere SSL-Verbindung (vgl. Seite 778) mit dem dafür erforderlichen Austausch von Zertifikaten (vgl. Seite 398) miteinander aufbauen, ist es für einen Angreifer möglich, diese Kommunikationsbeziehung zu unterwandern und den Server zu maskieren. Ursache dafür ist, dass die Informationen, aufgrund derer ein Benutzer glaubt, dass eine sichere Verbindung aufgebaut ist, dem Benutzer nicht direkt zukommen, sondern ihm über seinen Browser ange-

zeigt wird. Dazu gehört das Anzeigen des SSL-Icons (das Schloss) sowie das Aufblenden von Fenstern mit Warnhinweisen und das Anzeigen von Zertifikatinformationen. Ein Angreifer kann dem Client eine authentische Server-Seite zusammen mit einer vermeintlich sicheren SSL-Verbindung vorspiegeln, indem in der gefälschten Seite ein ebenfalls gefälschter SSL-Icon auf der Statusleiste eingeblendet wird. In [182] ist eine detaillierte Beschreibung eines möglichen Angriffs zu finden. Der Angriff setzt lediglich voraus, dass der Benutzer JavaScript aktiviert hat. Da der SSL-Icon und alle damit zusammenhängenden Aktionen vom Angreifer kontrolliert werden, kann er auch dafür sorgen, dass beim Anklicken des Icons durch den Benutzer ein gefälschtes Zertifikatsfenster geöffnet wird, das dem Benutzer die gefälschte Zertifikatinformation vorgaukelt. Ein gutgläubiger Benutzer wird daraufhin davon ausgehen, dass seine Daten (z.B. Passwörter, Kreditkarteninformationen, vertrauliche Firmeninterne) verschlüsselt zum vermeintlich authentischen Server übertragen werden. In Wahrheit erfolgt keine Verschlüsselung und der Angreifer kommt unmittelbar in den Besitz der Daten. Da eine Vielzahl von Anwendungen im Zusammenhang mit dem elektronischen Handel (e-Commerce, Business-to-Business (B2B), e-Government) darauf beruhen, dass die Benutzer darauf vertrauen können, mit dem authentischen Web-Server Daten auszutauschen, kommt den Web-Spoofing Angriffen eine große Bedeutung zu.

Zur Abwehr solcher Spoofing Angriffe kann man verschiedene, meist aber eher unzureichende Techniken einsetzen. Die Etablierung einer Public-Key Infrastruktur (PKI) (vgl. Abschnitt 9.1.3) reicht nicht aus, wie das Beispiel im Zusammenhang mit SSL-Verbindungen verdeutlicht hat. Da die meisten Spoofing-Angriffe darauf beruhen, dass JavaScript im Browser des Benutzers aktiviert ist, ist der erste nahe liegende Lösungsansatz der, JavaScript standardmäßig zu deaktivieren. Jedoch ist ein allgemeines Surfen heutzutage kaum noch ohne aktives JavaScript möglich, so dass eine solche Lösung nicht benutzungsfreundlich ist und damit wohl auf sehr geringe Akzeptanz bei den Benutzern stoßen wird. Die Ursache für die Spoofing-Problematik liegt darin, dass Angreifer die Seiten so manipulieren können, dass sie auch die Statusinformationen (z.B. SSL-Icon) der von ihnen gefälschten Web-Seite vollständig kontrollieren. Das bedeutet, dass zur Abwehr neue Browser-Konzepte benötigt würden, die gewährleisten, dass Statusinformationen nicht mehr fälschbar sind. Dies könnte man beispielsweise dadurch erreichen, dass ein vertrauenswürdiger Pfad vom Browser zum Benutzer etabliert wird. Ein Ansatzpunkt hierfür wäre die Einführung von Sicherheitslabels, mit denen der Browser die von ihm generierte Statusinformation annotiert. Solange der Angreifer nicht in der Lage ist, im Vornherein bestimmen zu können, welche Information durch den Browser annotiert werden

Abwehr?

wird, kann er auch keine Web-Seite vorab oder dynamisch mit gefälschten Statusinformationen versehen.

Forschungsbedarf

Lösungskonzepte zur wirksamen Abwehr von Web-Spoofing Angriffen werden zurzeit noch erforscht (u.a. [181]). In Anbetracht der Zunahme elektronisch abgewickelter Geschäftsprozesse mit Partnern, die häufig nur (wenn überhaupt!) mittels Zertifikaten authentifiziert werden, besteht im Bereich der Gewährleistung der Vertrauenswürdigkeit von Web-Seiten noch ein dringender Forschungs- und Entwicklungsbedarf.

Sicherheitsprobleme bei HTML5

HTML5

HTML5 (vgl. www.w3.org/TR/html5) wird seit 2014 zur Verfügung. Das Ziel der seit 2007 gestarteten Arbeiten zur Spezifikation von HTML5 ist es u.a., interaktive und 3D-Anwendungen ohne zusätzliche Browser-Plugins zu unterstützen. So lassen sich beispielsweise Videos in HTML5 direkt abspielen und man kann auf die Nutzung von Flash verzichten. Mit dem Übergang zu HTML5 werden Web-Applikationen zunehmend vergleichbar mit Standardanwendungen, die auch ohne online-Webzugriff ihre Funktionalität voll erbringen. Um dies zu erreichen, erhalten HTML5 Anwendungen Zugriff auf das lokale Dateisystem oder aber auch auf lokale Datenbanken. Google Chrome unterstützt beispielsweise bereits etliche HTML5 Funktionen und der Chrome Webstore ist, in Analogie zum Apple App-Store, eine Art Marktplatz für HTML5-Anwendungen.

Lokale Datenhaltung

Ein zentrales Konzept von HTML5 ist es, umfangreiche Datensätze von Servern lokal auf dem Web-Client abzulegen. HTML5 bietet hierfür mehrere Konzepte, um Daten, die vom Server stammen, temporär oder auch persistent auf dem Web-Client mittels JavaScript zu speichern und darauf in nachfolgenden Sitzungen wiederum unter Nutzung von JavaScript erneut zuzugreifen. Dazu können, ähnlich wie in HTML durch das Cookie-Konzept bekannt, Session-Information im *SessionStore* abgelegt werden. Im Gegensatz zu Cookies können diese Datenobjekte aber bis zu 5 MB groß werden. Diese gespeicherten Daten verlieren beim Browser-Neustart ihre Gültigkeit. Demgegenüber sind Daten, die im *LocalStorage* abgelegt werden, persistent. Sie können eine Größe von bis zu 10 MB umfassen. Der Zugriff auf diese Datenobjekte ist für alle Nutzer, die auf einem gemeinsam genutzten Rechner arbeiten, möglich, so dass sich hieraus Bedrohungen für die Integrität und Vertraulichkeit der gespeicherten Daten ergeben. Mit dem Konzept des *DatabaseStorage* werden noch weiterreichende Möglichkeiten eröffnet und es können lokale Datenbanken auf dem Client angelegt werden, auf die mittels JavaScript zugegriffen werden kann. Der Vorteil dieses Konzeptes ist es, dass Webanwendungen auch offline ablaufen können, wenn die dafür erforderlichen Daten lokal verfügbar gemacht sind. Gleichzeitig sind derar-

tige Datenbanken aber auch vielfältigen Gefährdungen durch die bekannten SQL-Injection Angriffe ausgesetzt.

Im lokalen Client-Speicher können mit HTML5 zudem base64-codierte Bilddateien abgelegt und über JavaScript eingebettet in eine HTML-Datei ausgelesen und übertragen werden. Angreifer können sich dieses Konzept zu Nutze machen und über XSS-Lücken gezielt Bilddateien persistent auf der lokalen Festplatte eines Client-Rechners ablegen, die zum Beispiel strafrechtlich relevantes Bildmaterialien beinhalten können, ohne dass der Benutzer dies merkt.

Die skizzierten Merkmale von HTML5, wie das persistente Ablegen großer Datenmengen auf der Festplatte des Client-Rechners und das wiederholte Auslesen dieser Daten durch den Server mittels JavaScript bei Bedarf bietet viele Möglichkeiten, beispielsweise über XSS-Lücken persistenten Schadcode auf den Client abzulegen. Dies eröffnet also Angreifern die Möglichkeit, langlebige Schadsoftware auf Client-Rechnern zu platzieren. Dieser Schadcode existiert insbesondere auch dann noch und kann weiterhin seine Schadfunktion entfalten, wenn die XSS-Lücke in der Web-Anwendung bereits geschlossen worden ist. Zudem könnten Betreiber von Web-Sites die persistene Datenspeicherung gezielt nutzen, um auf dem Client-Rechner Nutzer-Daten zu sammeln und zu speichern, die Auskunft beispielsweise über das Surfverhalten des Nutzers geben. Dies wären bedenkliche Eingriffe in die Privatsphäre des Nutzers.

HTML5 erfordert somit zum einen neue Kontroll- und Überprüfungsmaßnahmen, aber auch ein diszipliniertes Programmieren bei der Entwicklung von HTML5-Anwendungen, um zu gewährleisten, dass durch Speicherung von personenbezogenen Daten auf dem Client keine Verletzung von Datenschutzauflagen erfolgt. Zudem müssen die Daten, die die Web-Anwendung selbst zur Erbringung ihrer Funktionalität als Eingabeparameter vom Client-Rechner zur Verarbeitung benötigt, sorgfältig geprüft werden, da deren Schadcode-Freiheit und Korrektheit aufgrund der vielfältigen Manipulationsmöglichkeiten keinesfalls sicher gestellt ist. HTML5 bietet somit eine Reihe interessanter Merkmale, um aus Web-Applikationen voll funktionsfähige, lokal ausführbare Anwendungssoftware zu machen. Gleichzeitig bergen dieses neuen Merkmale aber auch nicht unerhebliche zusätzliche Sicherheitsrisiken, die bei der Erstellung von HTML5-Anwendungen zu beachten sind.

3.5.3 OWASP Top-Ten Sicherheitsprobleme

Um der steigenden Bedeutung der Sicherheit von Web-Anwendungen für die Sicherheit von Unternehmen Rechnung zu tragen, wurde das OWASP-Projekt (Open Web Application Security Project) initiiert. OWASP ist eine

persistenten Schadcode

Angriffe

Fazit

OWASP

Vereinigung von Firmen und Organisationen, wie Sun Microsystems, IBM, Swiss Federal Institute of Technology oder auch British Telecom, die das Ziel haben, die Sicherheit von Web-Anwendungen zu verbessern. Mit den OWASP Top-Ten stellt das Konsortium ein Dokument zur Verfügung, das einen Überblick über die wichtigsten Schwachstellen und Angriffe auf Web-Anwendungen umfasst und ca. alle drei Jahre aktualisiert wird. Das Dokument wird durch Sicherheitsexperten aus aller Welt erstellt. Seit 2008 gelten die Top-Ten sogar als offizielle Vorgaben für den Payment Card Industry (PCI) Data Security Standard, so dass bei Code Reviews nachgewiesen werden muss, dass geeignete Maßnahmen zur Abwehr der in den Top-Ten gelisteten Schwachstellen und Verwundbarkeiten getroffen wurden. Im November 2017 lag ein noch nicht verabschiedeter Request for Comments vor (siehe <https://github.com/OWASP/Top10/>). Nachfolgend gehen wir auf ausgewählte Schwachstellen, die über die Zeit nicht an Aktualität verloren haben, etwas näher ein.

1. Injection, insbesondere SQL-Injection-Schwachstellen

SQL-Injection
Injection-Angriffe sind möglich, wenn Eingabedaten eines Benutzers auf der Seite des Web-Servers von einem Interpreter als Kommando oder Anfrage (z.B. Datenbank-Query) bzw. als ein Teil solcher Aktionen verstanden und entsprechend ausgeführt werden. Derartige Schwachstellen ermöglichen es, dass ein Angreifer beispielsweise den Zugriff auf Daten in Datenbanken, die vom Server betrieben werden, erhält. Daten können gelesen, manipuliert und es können sogar komplexe Datenbank-Befehle initiiert werden. Injection-Schwachstellen nehmen derzeit die erste Position der Top-Ten Schwachstellen ein. Auf diese Klasse von Schwachstellen gehen wir weiter unten noch etwas genauer ein.

2. Broken Authentication

Die hier angesprochene Schwachstelle betrifft den unzureichenden Schutz von Authentisierungs-Credentials, Session-Tokens etc. Dadurch ist es Angreifern möglich, die Identität eines Benutzers zu übernehmen (Spoofing). Unter der gefälschten Identität kann der Angreifer mit den Rechten des Nutzers unautorisiert auf Informationen zugreifen, Daten manipulieren und Code zur Ausführung bringen. Schwachstellen dieser Klasse nehmen weiterhin stark zu und belegen den Platz zwei der aktuellen Liste.

3. Cross Site Scripting (XSS)

XSS
Cross Site Scripting Schwachstellen liegen immer dann vor, wenn Daten, die von einem Nutzer eingegeben und an den Web-Server gesendet werden, vom Server in ungeprüfter bzw. ungefilterter Form an die Browser von Nutzern zurückgesendet und durch den Web-Browser weiter verarbeitet werden. XSS-Schwachstellen ermöglichen einem Angreifer,

Scriptcode im Webbrowser des Nutzers, also des Opfers, auszuführen. Mit derartigen Angriffen können ganze Nutzer-Sitzungen übernommen (hijacking), Web-Seiten gefälscht und missbraucht, Phishing Angriffe durchgeführt oder aber auch der Web-Browser zur Ausführung von scriptbasierter Malware missbraucht werden. Auf XSS-Angriffe gehen wir weiter unten noch etwas genauer ein.

4. Information Leakage

Applikationen können in unbeabsichtigter Weise Informationen über Konfigurationen, interne Abläufe oder Personen preisgeben. Ein Beispiel ist die Ausgabe von sehr detaillierten Fehlermeldungen, die dem Angreifer viele Informationen preisgeben.

Die Top-Ten der Web-Verwundbarkeiten umfassen viele Schwachstellen, die man durch ein sicheres Programmieren, durch Eingabefilterung und durch den konsequenten Einsatz von Identitäts- und Zugriffskontrollen beseitigen könnte. Abschließend gehen wir auf zwei der wichtigsten Klassen von Schwachstellen heutiger Web-Anwendungen, nämlich Cross-Site-Scripting und SQL-Injection-Angriffe, noch etwas genauer ein.

Cross-Site-Scripting (XSS)

Bei dynamisch erstellten Web-Seiten besteht ganz allgemein die Gefahr, dass über nicht korrekt überprüfte Eingaben, durch die die Web-Seite dynamisch gestaltet wird, Schadsoftware in Form von bösartigen Skripten eingeschleust wird. Im Gegensatz zu den weiter vorne beschriebenen CGI-Problemen sind hiervon alle Benutzer betroffen, in deren Browser eine derartig manipulierte Web-Seite angezeigt wird, d.h. die diese Seite abrufen. Die in die Web-Seite eingebetteten Skriptbefehle werden vom Browser des Benutzers automatisch ausgeführt, falls dieser die entsprechende Skriptsprache, wie zum Beispiel JavaScript, aktiviert hat. Ein Benutzer kann auf vielfältige Weise dazu verleitet werden, bösartigen Code auszuführen. Sehr häufig wird dazu von Angreifern ausgenutzt, dass Benutzer vertrauensvoll Links in Web-Seiten, E-Mails oder News-Postings anklicken.

Cross-Site-
Scripting

Cross-Site-Scripting Angriffe zählen auch heute noch zu den am häufigsten auftretenden Sicherheitslücken in Web-Anwendungen. Charakteristisch für XSS-Angriffe ist, dass ein Angreifer seinen Schadcode auf einem Server so platziert, dass bei Zugriffen auf den Server dieser Schadcode an den zugreifenden Nutzer weitergeleitet wird. Typischerweise wird die Schadsoftware in JavaScript geschrieben, so dass dieser Code im Benutzer-Browser zur Ausführung gelangt. Ein typisches Beispiel für ein solches Szenario ist ein Schwarzes Brett, das vom Server verwaltet wird. Ein Angreifer sendet eine Nachricht, die über das Schwarze Brett verbreitet werden soll und fügt

Schadcode in die Nachricht mit ein. Prüft der Server die erhaltene Nachricht nicht auf Korrektheit, führt er also keine Eingabefilterung durch, so wird die Nachricht mit dem integrierten Schadcode an die Nutzer weitergereicht, die diese Nachricht vom Schwarzen Brett abrufen. Wenn JavaScript im Browser des Opfers aktiviert ist, so wird der Schadcode transparent für den Nutzer ausgeführt. Da diese Angriffe Webseiten-übergreifend durchgeführt werden, nennt man sie Cross-Site-Scripting Angriffe (XSS)⁸. Mit dem Schadcode versucht ein Angreifer häufig Identifizierungsinformationen vom Opfer zu stehlen, wie zum Beispiel SessionIDs oder Cookies. Ein typisches Beispiel eines solchen Angriffs unter Nutzung von CGI ist es im Folgenden dargestellt.

Beispiel 3.9 (XSS-Angriff)

Betrachten wir zur Veranschaulichung eines Cross-Site-Scripting Angriffs das Szenario aus der Abbildung 3.15 . Auf der linken Seite befindet sich der Rechner eines beliebigen Internetnutzers, der Client A, der zum Ziel dieses Angriffs werden soll. Beteiligt an dem Szenario ist ferner eine Web-Seite eines Angreifers, die beispielsweise folgenden Link enthält:

```
<A HREF='http://example.com/comment.cgi?  
mycomment=<SCRIPT> Schadens-Code </SCRIPT>'>  
Click here </A>
```

Das Ziel des Angreifers ist es, eine XSS-Schwachstelle auf dem Server *example.com* auszunutzen, um ihn dazu zu bringen, den in dem Link integrierten Schadens-Code an den Client A zu schicken. Eine beliebte Vorgehensweise ist beispielsweise auszunutzen, dass ein Server, im Beispiel der Server *example.com*, eine Fehlermeldung an den Aufrufer zurücksendet, wenn eine angeforderte Seite auf dem Server nicht gefunden wird. Das Problem ist, dass Server vielfach in diese Fehlermeldung die gesamte URL, die zum Fehler geführt hat, integrieren.

In Teil (1) der Abbildung wird dies für unser Beispiel verdeutlicht. Das Anklicken (Schritt 1) der vom Angreifer präparierten URL durch den Nutzer am Client-Rechner A führt dazu, dass eine Anfrage nach einer bestimmten Seite zum *example.com* Server gesendet wird (Schritt 2). Als Absender der Anfrage tritt der Client-Rechner A auf. Da die gewünschte Seite auf dem Server natürlich nicht gefunden wird, sendet dieser an den Client A eine Fehlermeldung der nachfolgenden Art (Schritt 3) zurück.

```
<HTML> 404page not found <script> ... </script>
```

Abbildung 3.16 veranschaulicht den Effekt von Schritt 3 auf dem Client-Rechner A. Die dynamisch erzeugte Seite mit dem Fehlercode wird im

Fehlermeldung

Schadens-Code

⁸ Das naheliegende Akronym dieses Angriffs ist CSS. Um Verwechslungen mit der gleichlautenden Abkürzung für Cascading Style Sheets zu vermeiden, wird aber das Kürzel XSS verwendet.

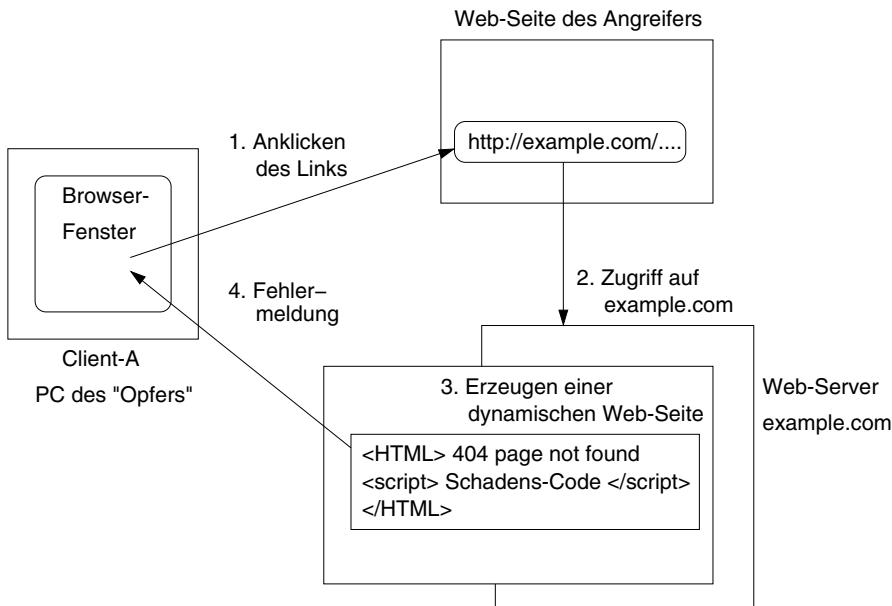


Abbildung 3.15: Cross-Site-Scripting Angriff (Teil 1)

Browser des Client A angezeigt (Schritt 5) und falls der entsprechende Benutzer Java-Script aktiviert hat, werden die (böswilligen) Script-Befehle, die der Angreifer in diese Fehlermeldung eingeschleust hat, auf dem Client A ausgeführt. Die Script-Befehle könnte beispielsweise dazu dienen, Authentifizierungs-Cookies, die auf dem Client A abgespeichert sind, an den Rechner des Angreifers (Schritt 7 in der Abbildung 3.16) zu versenden. Damit wäre der Angreifer in der Lage, sich gegenüber dem Server, der dem Client A diese(s) Cookie(s) ausgestellt hat, als Client A auszugeben und in dessen Namen tätig zu werden (z.B. unberechtigterweise Dienste zu nutzen oder Downloads auf Kosten des Client A durchzuführen).



Zu beachten ist, dass ein Angreifer mit einer XSS-Attacke auch eine über SSL gesicherte und authentifizierte Verbindung zwischen einem Benutzer-Rechner und einem Web-Server ausnutzen kann, um unter der Identität des authentifizierten Web-Servers den böswilligen Script-Code zum Opfersystem zu transferieren. Dies ist möglich, da der Code bereits vor dem Aufbau der SSL-Verbindung in eine Web-Seite platziert wird. So kann dieser Code beispielsweise Cookies modifizieren, die von dem vermeintlich vertrauenswürdigen Web-Server auf dem Rechner des Benutzers gespeichert werden. Dies ist natürlich immer dann problematisch, wenn der Web-Server Daten aus dem Cookie verwendet, um dynamische Web-Seiten zu generieren. Je-

SSL-Verbindungen

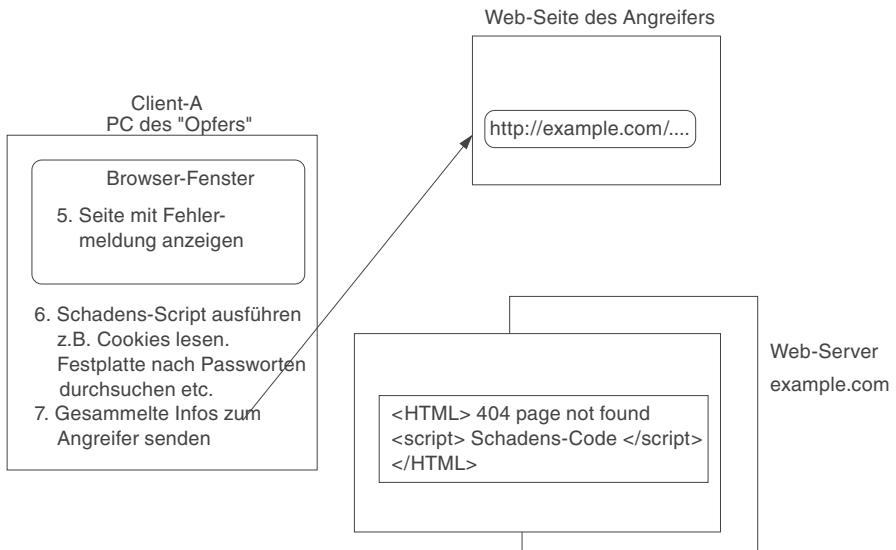


Abbildung 3.16: Cross-Site-Scripting Angriff (Teil 2)

des Mal, wenn der Benutzer auf den Web-Server zugreift, auch wenn er dies dann über vertrauenswürdige Links tut, wird das modifizierte Cookie übertragen.

Abwehr

Es ist in erster Linie die Aufgabe von Entwicklern von Web-Anwendungen durch Maßnahmen des sicheren Programmierens dafür zu sorgen, dass es nicht möglich ist, durch fehlerhaft oder gar nicht überprüfte Eingaben bösartige Scripte in die Seite einzuschleusen. So sollten problematische Meta-Zeichen aus dem Eingabestrom herausgefiltert und die zulässigen Eingaben sollten vollständig über reguläre Ausdrücke beschrieben werden. Durch die Deaktivierung von Skriptsprachen im Browser kann sich ein Benutzer vor der Ausführung von Skriptbefehlen im Browser schützen.

SQL-Injection

SQL-Injection

SQL-Injection Angriffe nutzen ebenfalls, wie oben bereits angesprochen, mangelhafte oder fehlende Eingabeüberprüfung aus, um Angriffe durchzuführen. Wie der Name des Angriffs bereits suggeriert, werden Schwachstellen von Web-Servern ausgenutzt, um über präparierte Datenbankanfragen Schadcode in eine SQL-Datenbank einzuschleusen, um zum Beispiel Daten in der Datenbank zu verändern oder auszulesen. Ein mögliches Angriffsziel kann aber auch sein, über eine Shell die Kontrolle über den Datenbankserver zu erlangen. Ein typischer Ablauf eines solchen Angriffs könnte wie folgt aussehen. Ein Benutzer gibt seine Daten in einem Webformular ein und der Web-Server setzt diese Daten in SQL-Statements ein. Falls hierbei der Server die SQL-Anfrage aus den ungeprüften Textblöcken des Benutzers zu-

sammensetzt und als String zum Datenbankserver weiterleitet, so kann ein Angreifer zusätzliche SQL-Kommandos in die Eingabedaten einfügen, die dann auf der SQL-Datenbank ausgeführt werden. Ein solcher Angriff könnte dann vereinfacht wie folgt aussehen:

- Eine zuässige Eingabe in ein Webformular könnte beispielsweise die Frage nach dem Alter einer bestimmten Person, deren Name ebenfalls einzugeben ist, sein:

```
$name = ''max''  
SELECT age FROM user WHERE name='max'
```

- Ein SQL-Injection Angriff könnte nun versuchen, nicht nur den Name einer Person anzugeben, sondern zusätzlich noch ein auszuführendes Kommando zu spezifizieren. Dieser Angriff hat Erfolg, wenn die SQL-Datenbank die Eingabe des Namens nicht prüft, sondern sämtliche Eingaben ungefiltert akzeptiert. Eine gezielt manipulierte Eingabe, die einen eingeschleusten SQL-Befehl enthält, könnte zum Beispiel wie folgt aussehen:

```
$name = ''max'; SELECT salary FROM personal  
WHERE name='fritz'
```

- Die Anfrage an die SQL-Datenbank ist somit zusammengesetzt aus der korrekten Altersabfrage und der eingeschleusten Anfrage nach dem Gehalt einer bestimmten Person:

```
$name = ''max'; SELECT salary FROM personal  
WHERE name='fritz'
```

```
SELECT age FROM user WHERE name = ''max'';
```

Mit diesem einfachen Angriff verschafft sich der Angreifer Zugriff auf weitere Informationen in der Datenbank.

Abhängig davon, welche SQL-Befehle eingeschleust werden, kann der Angreifer an interne Informationen gelangen (z.B. select-Befehl), Informationen verfälschen (z.B. insert, update, delete), die Datenbank zerstören (z.B. drop table, drop database) oder aber auch auf dem Hostsystem Systembefehle ausführen (z.B. System.exec("format c: ")).

Für die Abwehr derartiger Angriffe gilt wiederum, dass eine Filterung der Eingabedaten unumgänglich ist. Ein weiteres Lösungskonzept bieten die *Prepared Statements*. Ein solches Kommando ist ein Template mit Platzhaltern für Parameter, das in der Datenbank hinterlegt wird. SQL-Injection-Angriffe sind nicht mehr möglich, da an dem in der Datenbank hinterlegten Template keine Änderungen vorgenommen werden können und die Parameterwerte bei der konkreten Nutzung durch den Datenbankserver geprüft werden.

Fazit

Die Sicherheitsprobleme der Internet-Protokollfamilie spiegeln eine für IT-Systeme typische Situation wider. Ursprünglich für einen eher eingeschränkten Benutzerkreis aus dem Forschungsumfeld entworfen, spielen Sicherheitsfragen im Design der Protokolle keine Rolle. Durch die Weiterentwicklung der Protokolle und durch deren erweitertes Einsatzspektrum treten aber Sicherheitsanforderungen und Bedrohungen immer mehr in den Vordergrund, so dass die Design- und natürlich auch die Implementierungs mängel zu untragbaren Risiken werden. Verbesserungen lassen sich auf zwei Wegen erzielen. Zum einen kann man durch zusätzliche generische Dienste und Protokolle fehlende Sicherheitsfunktionen zu den bestehenden Protokollen hinzufügen. Dieser Weg wird unter anderem durch das Secure Socket Layer Protokoll (SSL) und mit den IPsec-Protokollen AH und ESP beschritten. Die andere Möglichkeit besteht in der Entwicklung spezifischer Sicherheitsdienste auf der Anwendungsebene wie Pretty Good Privacy (PGP) für sichere E-Mail oder der Entwicklung von Standards für sichere Web-Services. Kapitel 14 geht auf diese Protolle detailliert ein.

4 Security Engineering

Das Kapitel gibt in Abschnitt 4.1 zunächst einen allgemeinen Überblick über den Entwicklungsprozess zur Konstruktion sicherer Systeme, der in den nachfolgenden Abschnitten detaillierter ausgearbeitet wird. Es handelt sich hierbei um Maßnahmen und Methoden, die einer noch nicht methodisch ausgearbeiteten Disziplin, dem Security Engineering, zuzuordnen sind (vgl. u.a. [5]). Abschnitt 4.2 stellt Techniken zur Erfassung der relevanten Systemeigenschaften vor und Abschnitt 4.3 erläutert Vorgehensweisen zur Ermittlung des Schutzbedarfs für ein derart beschriebenes System. Der Schutzbedarf orientiert sich an Schäden, die auftreten können. Mit Methoden der Bedrohungs- und Risikoanalyse, wie sie in den Abschnitten 4.4 und 4.5 erläutert werden, erfasst man Bedrohungen, die solche Schäden bewirken können und schätzt das Risiko für deren Eintreten ein. Der anschließende Abschnitt 4.6 beschäftigt sich mit der Erstellung von Sicherheitsstrategien, die die zu gewährleistenden Sicherheitseigenschaften des Systems beschreiben, sowie mit der Erstellung einer Sicherheitsarchitektur zu deren Umsetzung. Da auf diese beiden Aspekte in späteren Kapiteln des Buches noch detailliert eingegangen wird, beschränken wir uns hier auf einen groben Überblick. Abschnitt 4.7 stellt die Sicherheitsgrundfunktionen vor. Zur Realisierung der Grundfunktionen und für den Einsatz von dazu benötigten Mechanismen und Verfahren werden in 4.8 Leitlinien angegeben, die bei der Konstruktion qualitativ hochwertiger Systeme zu beachten sind. Abschließend wird in Abschnitt 4.9 mit dem Security Development Lifecycle (SDL) ein Ansatz vorgestellt, der eine Konkretisierung des allgemein beschriebenen Vorgehensmodells darstellt und auf die Praxis der Entwicklung sicherer Software zugeschnitten ist.

Das sichere Programmieren ist ein Teilaспект des Security Engineering. Hierzu gibt es bereits eine Reihe von Best-Practice Empfehlungen und goldenen Regeln, worauf ein Entwickler bei der Erstellung sicherer Software achten sollte. Wir gehen im Folgenden hierauf nicht weiter ein. Der interessierte Leser sei für eine detailliertere Auseinandersetzung mit der Thematik u.a. auf das Buch von McGraw [117] verwiesen.

4.1 Entwicklungsprozess

Dedizierte Methodiken zur Konstruktion sicherer IT-Systeme im Sinne eines systematischen Security Engineerings wurden bislang kaum entwickelt. In den letzten Jahren gab es jedoch vermehrt Arbeiten, die sich mit Methoden und Werkzeugen zur Modell-getriebenen Konstruktion sicherer Systeme beschäftigt haben. Die dabei verfolgte Vorgehensweise des Model-driven Security Engineerings (u.a. [24]) konzentriert sich auf den Aspekt der Zugriffskontrolle. Es werden meist UML-basierte Modelle verwendet, die erweitert werden, um Sicherheitsanforderungen zu beschreiben. Unter Nutzung von Transformationswerkzeugen wird aus den Modell-Beschreibungen automatisch ausführbarer Code zur Durchführung von Zugriffskontrollen generiert. Die entsprechenden Arbeiten sind interessante Ansätze, die jedoch nur einen Teil der Fragestellung abdecken, die bei der ingenieurmäßigen Konstruktion sicherer Systeme zu lösen ist. Wir konzentrieren uns im Folgenden deshalb auf allgemeine Konstruktionsprinzipien und insbesondere auf die Phasen und Prozesse, die den gesamten Lebenszyklus von Software und von Systemen abdecken.

4.1.1 Allgemeine Konstruktionsprinzipien

Prinzipien Bereits 1975 haben Saltzer und Schroeder in [157] allgemeine Prinzipien entwickelt, die bei der Konstruktion sicherer Systeme zu beachten sind, und die auch heutzutage noch Gültigkeit besitzen. Zu diesen Prinzipien zählen u.a. das Erlaubnis-, das Vollständigkeits- und das Need-To-Know-Prinzip sowie das Prinzip der Benutzerakzeptanz.

Erlaubnis Das Erlaubnisprinzip (engl. *fail-safe defaults*) fordert, dass grundsätzlich jeder Zugriff zunächst verboten ist (default deny) und nur durch eine explizite Erlaubnis ein Zugriffsrecht gewährt werden kann.

Vollständigkeit Das Vollständigkeitsprinzip (engl. *complete mediation*) besagt, dass jeder Zugriff auf seine Zulässigkeit zu prüfen ist. Ein System, in dem Zugriffsrechte zur Nutzung von Dateien vergeben werden und eine Rechteüberprüfung nur beim Öffnen einer Datei durchgeführt wird, erfüllt beispielsweise das Vollständigkeitsprinzip nicht. In solchen Systemen ist es möglich, dass ein Benutzer eine Datei über lange Zeiträume geöffnet hält und damit das Zugriffsrecht darauf nutzt, obwohl der Besitzer der Datei zwischenzeitlich dem Benutzer dieses Recht bereits entzogen haben kann.

Need-to-know Das Prinzip der minimalen Rechte (engl. *need-to-know*) fordert, dass jedes Subjekt nur genau die Zugriffsrechte erhalten darf, die es zur Erfüllung seiner Aufgaben benötigt. Systeme, in denen ein Superuser unbeschränkte Rechte besitzt, sind ein offensichtliches Beispiel für die Nicht-Einhaltung des Need-to-know Prinzips.

Das Prinzip der Benutzerakzeptanz (engl. *economy of mechanism*) fordert, dass die eingesetzten Sicherheitsmechanismen einfach zu nutzen sein müssen und dass sie automatisch und routinemäßig angewendet werden.

Akzeptanz

Das Prinzip des offenen Entwurfs (engl. *open design*) besagt, dass die Verfahren und Mechanismen, die beim Entwurf eines Systems Verwendung finden, offen gelegt (*no security through obscurity*) werden müssen, da die Sicherheit eines Systems nicht von der Geheimhaltung spezieller Verfahren abhängig sein darf. So sollte zum Beispiel die Sicherheit kryptografischer Verfahren nicht darauf beruhen, dass man das Verschlüsselungsverfahren nicht kennt.

offener Entwurf

Bei der Konstruktion von Betriebssystemen werden die sicherheitsrelevanten Dienste und Maßnahmen häufig zusammengefasst und von den übrigen Systemteilen isoliert realisiert. Man spricht hier von Systemen mit einem Sicherheits-Kern (engl. *security kernel*). Die vertrauenswürdigen Sicherheitsdienste nennt man die Trusted Computing Base.

Sicherheitskern

4.1.2 Phasen

Die Konstruktion eines Systems entspricht einem phasenstrukturierten, iteriert durchgeführten Vorgehen, das in Abbildung 4.1 skizziert ist und eine Planungs-, Ausführungs-, Prüfungs- und Anpassungsphase umfasst.

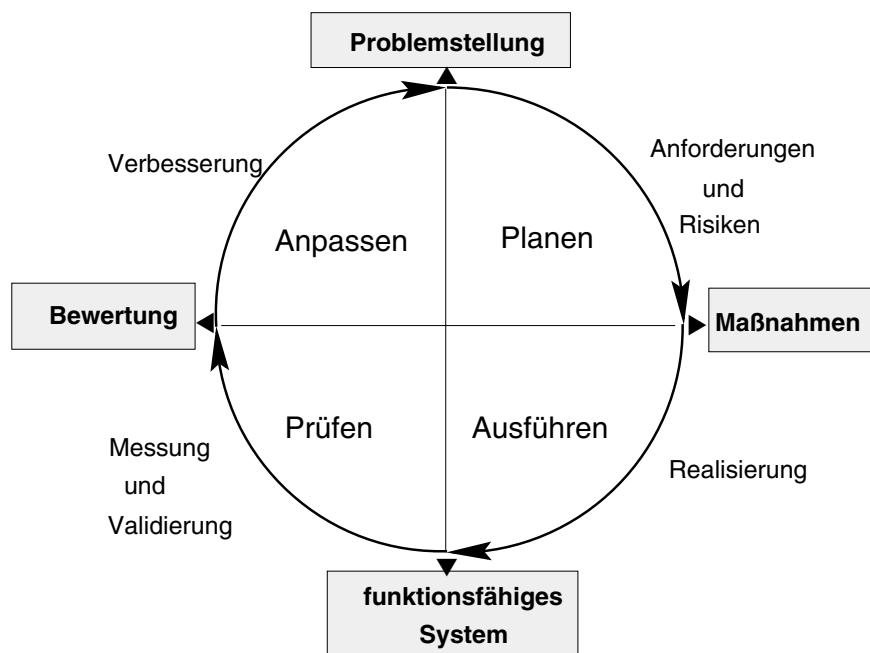


Abbildung 4.1: Vorgehen

Planen	In der Planungsphase werden die funktionalen Anforderungen des zu entwickelnden Systems sowie dessen Einsatzumgebung festgelegt und die Risiken und Bedrohungen, denen das System in der entsprechenden Umgebung ausgesetzt ist, werden analysiert und bewertet. Ausgehend von dieser Analyse werden die Sicherheitsanforderungen formuliert und die Systemarchitektur zur Erfüllung der Sicherheitsanforderungen wird entworfen. Zur Realisierung der Architektur werden Maßnahmen, Dienste und Protokolle benötigt, durch deren Einsatz ein funktionsfähiges System konstruiert wird. Dieses realisierte System ist zu validieren und es ist zu prüfen, ob es die gestellten Anforderungen erfüllt. Anhand einer Bewertung der Prüfergebnisse sind Verbesserungen festzulegen und durchzuführen. Dies kann zur Folge haben, dass Anforderungen ergänzt, verfeinert oder präzisiert werden müssen. Dies kann wiederum dazu führen, dass Bedrohungen und deren Bewertung revidiert und wirksamere Maßnahmen oder kostengünstigere Verfahren zur Realisierung eingesetzt werden müssen.
Ausführen	
Prüfen	
Anpassen	

Die Aufrechterhaltung eines geforderten Sicherheitsniveaus erfordert einen dynamischen, iterierenden Prozess der kontinuierlichen Überwachung der Einhaltung der Schutzziele und ggf. der Anpassung der festgelegten Sicherheitsstrategie des Systems an geänderte Gegebenheiten. Das mit den eingesetzten Maßnahmen jeweils erreichte Sicherheitsniveau ist dem technologischen Wandel unterworfen, so dass auch die eingesetzten Maßnahmen, wie beispielsweise kryptografische Verfahren, kontinuierlich auf ihre aktuelle Stärke zu überprüfen sind.

Abbildung 4.2 fasst die Phasen zusammen, die bei einer methodischen Konstruktion sicherer IT-Systeme zu durchlaufen sind.

4.1.3 BSI-Sicherheitsprozess

Die in Abbildung 4.2 skizzierten Entwicklungsphasen sind vordringlich für die Entwicklung und Umsetzung neuer sicherer Anwendungen und IT-Systeme gedacht. Sie stellen damit eine ergänzende Sicht zu den in den IT-Grundschutz-Katalogen des BSI vorgeschlagenen Sicherheitsprozess dar, der darauf abzielt, bereits bestehende IT-Infrastrukturen abzusichern. Abbildung 4.3 veranschaulicht den BSI-Sicherheitsprozess. Neben der operativen und taktischen Ebene, die auch in Abbildung 4.2 in verfeinerter Form zu finden ist, definiert der BSI-Prozess noch eine strategische Ebene, auf der allgemeine organisatorische Leitlinien und Managementstrukturen festzulegen sind. Auf diese Ebene gehen wir im Folgenden aber, wie bereits angesprochen, nicht näher ein.

Die Erstellung des Sicherheitskonzepts gemäß dem BSI-Prozess umfasst die Strukturanalyse mit der Feststellung der funktionalen Anforderungen und

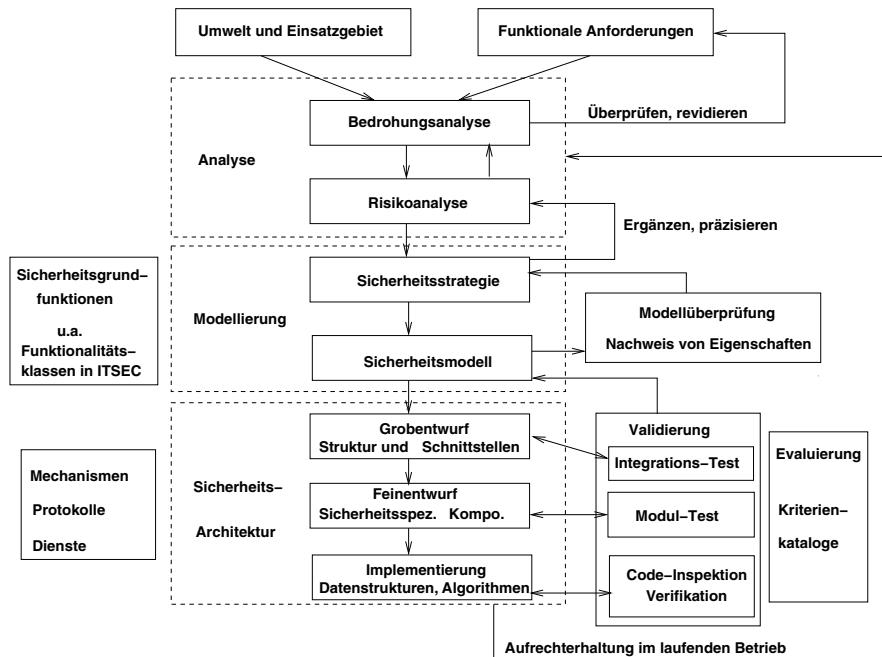


Abbildung 4.2: Entwicklungsphasen

der Systemgegebenheiten sowie die Bedrohungs- und Risikoanalyse, falls ein hoher Schutzbedarf ermittelt wurde. Ansonsten entfallen diese Phasen und es erfolgt direkt eine Modellierung und ein Basis-Sicherheitscheck mit einem Soll-Ist-Abgleich.

Notwendigkeit eines Sicherheitskonzepts

Auf die Notwendigkeit der Entwicklung und Umsetzung eines Sicherheitskonzeptes in Unternehmen wird auch durch die im Mai 2018 in Kraft getretene EU-Datenschutzgrundverordnung hervorgehoben. In Artikel Artikel 32, Abs. 1 heißt es dort:

rechtliche Rahmen

Unter Berücksichtigung des Stands der Technik, der Implementierungskosten und der Art, des Umfangs, der Umstände und der Zwecke der Verarbeitung sowie der unterschiedlichen Eintrittswahrscheinlichkeit und Schwere des Risikos für die Rechte und Freiheiten natürlicher Personen treffen der Verantwortliche und der Auftragsverarbeiter geeignete technische und organisatorische Sicherheitsmaßnahmen, um ein dem Risiko angemessenes Schutzniveau zu gewährleisten; [...].

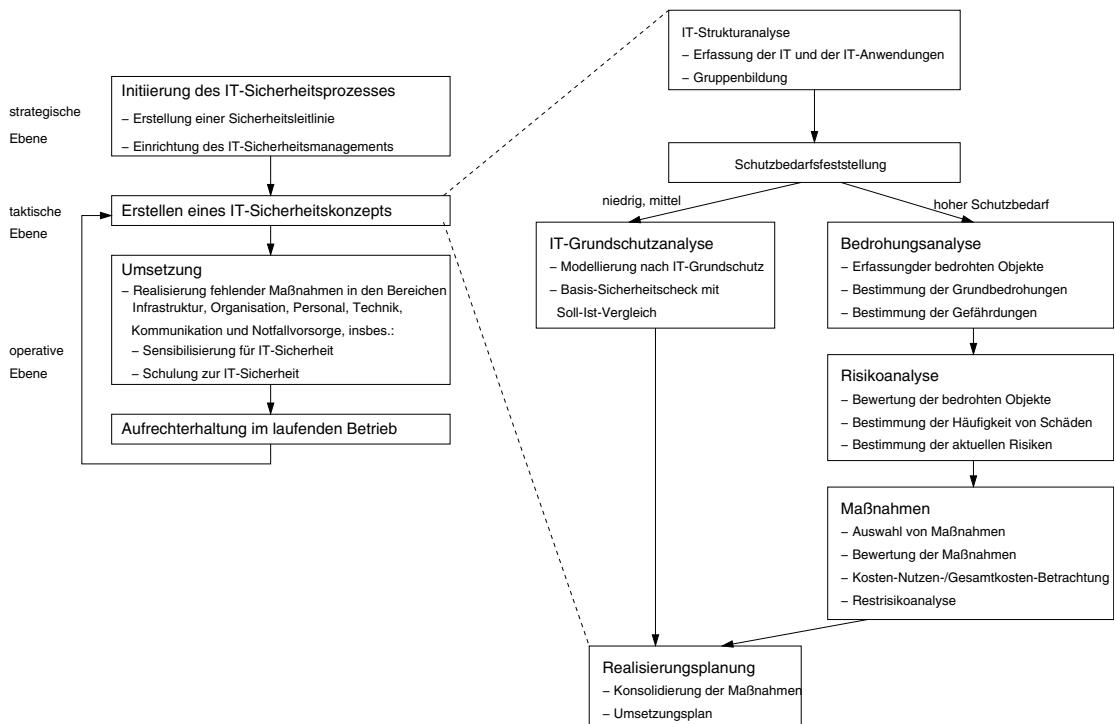


Abbildung 4.3: BSI-Sicherheitsprozess

Telemediengesetz

Auch im Telemediengesetz (TMG)¹, das 2007 das Teledienstegesetz (TDG) und das Teledienstedatenschutzgesetz (TDDSG) abgelöst hat, lassen sich entsprechende Anforderungen an ein zu erststellendes Sicherheitskonzept in Unternehmen, die Teledienste anbieten, ableiten. Das Telemediengesetz richtet sich an Diensteanbieter, die eigene oder fremde Teledienste zur Nutzung bereithalten oder den Zugang zur Nutzung vermitteln. Ein solcher Diensteanbieter ist bereits jede Organisation, die auf einer Webseite ein Kontaktformular zum Ausfüllen anbietet. Bei diesem Kontaktformular handelt es sich um einen Teledienst². Das Telemediengesetz gibt den datenschutzrechtlichen Rahmen vor, in dem Diensteanbieter die personenbezogenen Daten der Nutzer erheben, verarbeiten und nutzen dürfen. In § 13 Abs. 3 Nr. 3 TMG (Pflichten des Diensteanbieters) heißt es:

Der Diensteanbieter hat durch technische und organisatorische Vorrkehrungen sicherzustellen, dass, [...] der Nutzer Teledienste gegen Kenntnisnahme Dritter geschützt in Anspruch nehmen kann.

¹ vgl. <http://www.gesetze-im-internet.de/tmg/>

² vgl. B. Lorenz: Die Anbieterkennzeichnung im Internet, 2007, S. 88

Die Durchführung einer Risikoanalyse und die Etablierung eines Sicherheitskonzepts inklusive eines Risikomanagements ist seit Inkrafttreten des Gesetzes zur Kontrolle und Transparenz im Unternehmensbereich (KonTraG) im Mai 1998 auch vom Gesetzgeber für Vorstände von Aktiengesellschaften und für die Geschäftsführung einer GmbH gefordert. Durch das KonTraG sind diese verpflichtet, ein Risikomanagementsystem zu etablieren und erweiterte Berichtspflichten gegenüber dem Aufsichtsrat zu erfüllen. Im § 91 Abs. 2 Aktiengesetz (AktG) heißt es:

Der Vorstand hat geeignete Maßnahmen zu treffen, insbesondere ein Überwachungssystem einzurichten, damit den Fortbestand der Gesellschaft gefährdende Entwicklungen früh erkannt werden. [...]

Wie ein Risikomanagementsystem konkret auszustalten ist, hat der Gesetzgeber jedoch nicht ausdrücklich bestimmt. Der Gesetzesbegründung ist jedoch zu entnehmen, dass Risikomanagementsysteme Elemente wie ein Frühwarnsystem, ein internes Überwachungssystem einschließlich einer Revision und ein Controlling enthalten müssen. Neben der Aufstellung eines Sicherheitskonzepts einschließlich eines Modells und einer Policy sind Kontrollmaßnahmen notwendig, um die IT-bezogenen Teile des jeweiligen Unternehmens bei einer Revision sowie in deren Vorfeld zu überprüfen.

Im Folgenden werden die wesentlichen Schritte und Phasen eines Sicherheitsprozesses bzw. Entwicklungsprozesses genauer diskutiert.

4.2 Strukturanalyse

Zunächst müssen die funktionalen Eigenschaften des Systems, seine Einsatzumgebung und sein Verwendungszweck erfasst werden. Dazu sind die benötigten bzw. die vorhandenen und einzusetzenden Systemkomponenten und -dienste sowie deren Funktionalität zu beschreiben. Die Spezifikation der Anforderungen entspricht einem Pflichtenheft, in dem funktionale Anforderungen sowie die Leistungs-, Zuverlässigkeit- und Sicherheitsanforderungen des zu entwickelnden Systems festgehalten werden. Die Sicherheitsanforderungen sind eng verknüpft mit der vorgesehenen Einsatzumgebung des Systems. Die nachfolgende Beschreibung orientiert sich stark an den Vorgaben der IT-Grundschutz-Kataloge des BSI.

Der erste Schritt besteht darin, eine grafische Übersicht, einen Netztopologieplan, zu erstellen, in dem die eingesetzten Komponenten und deren Vernetzung erfasst werden. Beispiele für solche Komponenten sind Client- und Server-Systeme, Netzkomponenten wie Hubs, Switches und Router sowie Drucker, die über das Netz betrieben werden. Typische Netzwerkverbindun-

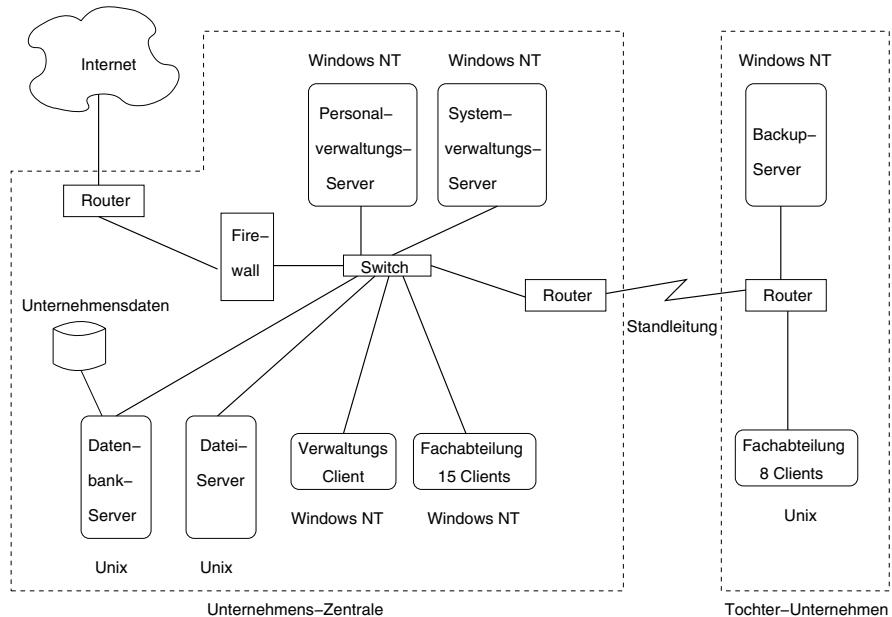


Abbildung 4.4: Beispiel eines Netztopologieplans für ein Unternehmensnetz (Quelle BSI)

gen sind lokale Netze wie Ethernet, Backbone-Technologien und Funknetze wie WLAN oder Bluetooth. Wichtige Bestandteile des Topologieplans sind die vorhandenen Verbindungen, über die die betrachtete Organisation mit der Außenwelt verbunden ist. Beispiele hierfür sind Einwahl-Zugänge über ISDN, Internet-Anbindungen über Router, Funkstrecken oder Mietleitungen beispielsweise zu entfernten Gebäuden oder Tochterunternehmen. Um die Komplexität eines solchen Topologieplans möglichst zu reduzieren, sollten gleichartige Komponenten (gleicher Typ, gleichartig konfiguriert, gleiche Anwendungen) gruppiert werden.

Beispiel 4.1 (Netztopologieplan für ein Unternehmensnetz)

In Abbildung 4.4 ist ein Beispiel eines Topologieplans³ dargestellt. Der Plan zeigt eine Unternehmensinfrastruktur, in der das Tochterunternehmen und die Zentrale über eine Standleitung miteinander verbunden sind. In der Zentrale gibt es sowohl Unix- als auch Windows NT-Server für unterschiedliche Aufgabenbereiche wie die Personalverwaltung oder die Dateiverwaltung. Da Mitarbeiterarbeitsplätze in den verschiedenen Arbeitsbereichen vom gleichen Typ sind, kann man sie im Topologieplan zu Clustern von Clients, wie Windows NT-Clients für die Verwaltung oder Unix-Clients für Fachabteilungen, zusammenfassen.



³ Quelle: BSI Grundschutz-Katalog

Ergänzend zur Netztopologie sind für jede Komponente und für jede Verbindung deren Charakteristika festzuhalten (z.B. tabellarisch). Dazu gehört bei IT-Komponenten deren eindeutige Identifikationsnummer, technische Daten über die Hardwareplattform, das Betriebssystem, die Art der Netzanbindung sowie die physikalische Netzadresse. Für Netzverbindungen gehört hierzu die Angabe der Art der Verbindung (z.B. Glasfaser), die maximale Datenübertragungsrate oder aber auch die verwendeten Netzwerkprotokolle (z.B. Ethernet-Protokolle, TCP/IP). Neben diesen eher technischen Daten sollte aber auch erfasst werden, welche Anwendungen welchen der aufgelisteten IT-Komponenten zuzuordnen sind und ob die Anwendung sicherheitskritische Daten (z.B. personenbezogene Daten) verarbeitet. Es bietet sich an, auch diese Informationen tabellarisch festzuhalten. Beispiele für relevante Anwendungen sind die Personaldatenverwaltung, Reisekostenabrechnung, die Systemverwaltung, E-Mail-Dienste oder eine zentrale Dokumentenverwaltung.

Information

Um im nächsten Schritt die Sicherheitsziele bzw. -anforderungen für das IT-System formulieren zu können, ist dessen Schutzbedarf festzustellen. Dazu muss zunächst die Gefährdungslage ermittelt und typische Schadensszenarien einzelner Anwendungen müssen untersucht werden. Daraus kann dann der Schutzbedarf der einzelnen IT-Systeme sowie der gesamten Infrastruktur abgeleitet werden.

Schutzbedarf

4.3 Schutzbedarfsermittlung

Das Ziel der Schutzbedarfsermittlung besteht darin, anhand möglicher Schäden und Beeinträchtigungen zu entscheiden, welchen Schutzbedarf eine Anwendung bzw. ein IT-System im Hinblick auf die Vertraulichkeit, Integrität und Verfügbarkeit besitzt.

4.3.1 Schadensszenarien

Da ein Schutzbedarf meist nur sehr schwer quantifizierbar ist, beschränkt man sich in der Regel auf eine qualitative Aussage. Grundlage hierfür können die Schutzbedarfskategorien sein, wie sie in den IT-Grundschutz-Katalogen des BSI festgelegt sind. Tabelle 4.1 fasst diese Kategorien zusammen.

Kategorie

Um den jeweiligen Schutzbedarf eines Systems entsprechend dieser Kategorien zu ermitteln, ist es sinnvoll, der Bedarfsermittlung die Schadensszenarien des IT-Grundschutz-Kataloges zugrunde zu legen. Nachfolgend werden

Kategorien	Erläuterung
niedrig bis mittel	Die Schadensauswirkungen sind begrenzt und überschaubar.
hoch	Die Schadensauswirkungen können beträchtlich sein.
sehr hoch	Die Schadensauswirkungen können ein existentiell bedrohliches, katastrophales Ausmaß annehmen.

Tabelle 4.1: Schutzbedarfskategorien

sechs typische Szenarien überblicksartig dargestellt. Auf diese Szenarien wendet man dann die Schutzkategorien an, die weiter unten noch erläutert werden. Szenarien

Schadensszenario 1: Verstoß gegen Gesetze/Vorschriften/Verträge

Gesetze

Beispiele für relevante Gesetze sind das Grundgesetz, das Bürgerliche Gesetzbuch, das Bundesdatenschutzgesetz, die EU Datenschutzgrundverordnung und die Datenschutzgesetze der Länder, das Sozialgesetzbuch, das Betriebsverfassungsgesetz, das Urheberrechtsgesetz oder auch das Informations- und Kommunikationsdienstegesetz (IuKDG). Da die gesetzlichen Rahmenbedingungen gerade im Bereich der Daten- und Informationsverarbeitung einem relativ häufigen Wandel unterliegen, ist es unabdingbar, dass die Sicherheitsverantwortlichen sich fortlaufend über die aktuelle Gesetzeslage informieren.

Vorschriften

Relevante Vorschriften sind Verwaltungsvorschriften, Verordnungen und Dienstvorschriften.

Verträge

Verträge sind unter anderem Dienstleistungsverträge im Bereich Datenverarbeitung oder auch Verträge zur Wahrung von Betriebsgeheimnissen.

Selbstbestimmungsrecht

Schadensszenario 2: Beeinträchtigung des informationellen Selbstbestimmungsrechts

Beispiele hierfür sind die unzulässige Erhebung personenbezogener Daten ohne Rechtsgrundlage oder ohne Einwilligung, die unbefugte Kenntnisnahme bei der Datenverarbeitung bzw. der Übermittlung von personenbezogenen Daten, die unbefugte Weitergabe personenbezogener Daten, die Nutzung von personenbezogenen Daten zu einem anderen als dem bei der Erhebung zulässigen Zweck und die Verfälschung von personenbezogenen Daten in IT-Systemen oder bei der Übertragung.

Schadensszenario 3: Beeinträchtigung der persönlichen Unversehrtheit
Anwendungen und Systeme, für die ein solches Szenario relevant ist, sind beispielsweise medizinische Überwachungs- und Diagnosesysteme, Verkehrsleitsysteme oder Flugzeugsteuerungssysteme.

Unversehrtheit

Schadensszenario 4: Beeinträchtigung der Aufgabenerfüllung
Beispiele für dieses Szenario sind Fristversäumnisse durch verzögerte Bearbeitung von Verwaltungsvorgängen, verspätete Lieferungen aufgrund verzögerter Bearbeitung von Bestellungen, oder auch die fehlerhafter Produktion aufgrund falscher Steuerungsdaten.

Aufgabenerfüllung

Schadensszenario 5: negative Auswirkungen
Solche Auswirkungen umfassen zum Beispiel Schäden, die zu Ansehensverlusten führen, die die gesellschaftliche oder wirtschaftliche Stellung eines Einzelnen oder eines Unternehmens gefährden, oder geschäftsschädigend sind.

negative Auswirkungen

Schadensszenario 6: finanzielle Auswirkungen
Beispiele hierfür sind die unerlaubte Weitergabe von Forschungs- und Entwicklungsergebnissen, die Manipulation von finanzwirksamen Daten in einem Abrechnungssystem, der Ausfall eines Produktionssystems und ein dadurch bedingter Umsatzverlust, Einsichtnahme in Marketingstrategiepapiere oder Umsatzzahlen, der Ausfall eines Web-Portals, oder auch der Zusammenbruch des Zahlungsverkehrs einer Bank, Diebstahl oder Zerstörung von Hardware.

finanzielle Auswirkungen

Zu beachten ist, dass ein Schaden auf mehrere Kategorien zutreffen kann. So kann beispielsweise der Ausfall eines Dienstes sowohl zur Beeinträchtigung der Aufgabenerfüllung als auch zu finanziellen Einbußen und zum Imageverlust führen.

Im nächsten Schritt werden nun die Schutzbedarfskategorien auf diese sechs Szenarien angewandt. Das heißt, es wird für jedes Szenario angegeben, welche Bedingungen gelten müssen, wenn ein Schutzbedarf der Kategorie niedrig, mittel oder sehr hoch entsteht.

4.3.2 Schutzbedarf

niedrig - mittel

Die BSI-IT-Grundschatz-Kataloge empfiehlt einen Schutzbedarf der Schutzkategorie niedrig bis mittel festzulegen, wenn die nachfolgend angegebenen Aussagen für die sechs Szenarien zutreffen.

Szenario	Schaden und Folgen
(1)	Verstöße gegen Vorschriften und Gesetze haben nur geringfügige Konsequenzen. Es kommt zu geringfügigen Vertragsverletzungen mit nur geringen Konventionalstrafen.
(2)	Eine Beeinträchtigung des informationellen Selbstbestimmungsrechts würde durch den Einzelnen als tolerabel eingeschätzt werden. Ein möglicher Missbrauch personenbezogener Daten hat nur geringfügige Auswirkungen auf die gesellschaftliche Stellung oder die wirtschaftlichen Verhältnisse des Betroffenen.
(3)	Eine Beeinträchtigung der persönlichen Unversehrtheit erscheint nicht möglich.
(4)	Die Beeinträchtigung der Aufgabenerfüllung würde von den Betroffenen als tolerabel eingeschätzt werden. Die maximal tolerierbare Ausfallzeit ist größer als 24 Stunden.
(5)	Eine geringe bzw. nur interne Ansehens- oder Vertrauensbeeinträchtigung ist zu erwarten.
(6)	Der finanzielle Schaden bleibt für die Institution tolerabel.

Zur Gegenüberstellung geben wir nachfolgend noch die Kategorien für den Schutzbedarf *sehr hoch* an. In analoger Weise sind natürlich auch die Klassen für den Schutzbedarf *hoch* festgelegt; der Leser sei hierfür auf die IT-Grundschatz-Kataloge verwiesen.

sehr hoch

Der Schutzbedarf eines Systems fällt demgegenüber in die Kategorie sehr hoch, falls die nachfolgenden Aussagen zutreffen.

Szenario	Schaden und Folgen
(1)	Es liegt ein fundamentaler Verstoß gegen Gesetze und Vorschriften vor oder es kommt zu Vertragsverletzungen, deren Haftungsschäden ruinös sind.
(2)	Eine bedeutende Beeinträchtigung des informationellen Selbstbestimmungsrechts des Einzelnen scheint möglich. Ein möglicher Missbrauch personenbezogener Daten würde für die Betroffenen den gesellschaftlichen und wirtschaftlichen Ruin bedeuten.
(3)	Gravierende Beeinträchtigungen der persönlichen Unversehrtheit sind möglich. Gefahren für Leib und Leben können auftreten.
(4)	Die Beeinträchtigung der Aufgabenerfüllung wird von allen Betroffenen als nicht tolerabel eingeschätzt. Die maximal tolerierbare Ausfallzeit ist kleiner als eine Stunde.
(5)	Eine landesweite bzw. internationale Ansehens- und Vertrauensbeeinträchtigung ggf. sogar existenzgefährdender Art ist denkbar.
(6)	Der finanzielle Schaden ist für die Institution existenzbedrohend.

Vereinfacht kann man sagen, dass bei einer Schutzbedarfsermittlung systematisch Fragen der Art: *was wäre wenn, ...* zu stellen und zu beantworten sind.

Nachdem der Schutzbedarf ermittelt ist, muss der Ist-Zustand der IT-Infrastruktur, also der herrschende Sicherheitsstandard, bestimmt werden. Zudem müssen die Defizite zwischen dem Ist- und dem Sollzustand, der dem Schutzbedarf Rechnung trägt, erarbeitet werden. Mit einer Bedrohung- und Risikoanalyse kann man ermitteln, wodurch Schäden potentiell entstehen können und wie hoch das Risiko ist, das damit verbunden ist. In diesem Schritt sind also Ursachen und Randbedingungen für das Eintreten von Schadensfällen zu hinterfragen.

4.4 Bedrohungsanalyse

In der Bedrohungsanalyse sind die potentiellen organisatorischen, technischen und benutzerbedingten Ursachen für Bedrohungen, die Schäden hervorrufen können, systematisch zu ermitteln. Die möglichst vollständige Erfassung der Bedrohungen eines Systems ist eine schwierige Aufgabe, die fundierte Kenntnisse über Sicherheitsprobleme und Schwachstellen existierender Systeme und deren Dienste erfordert.

Zur Unterstützung einer methodischen Vorgehensweise zur Erfassung der relevanten Bedrohungen kann man einen matrix- oder einen baumorientierten Analyseansatz wählen.

4.4.1 Bedrohungsmatrix

Matrix

Bei dieser Vorgehensweise klassifiziert man zunächst die Gefährdungsbereiche; sie bilden die Zeilen der Matrix. Eine mögliche Klassifikation von Gefährdungsbereichen, die die wichtigsten Problembereiche abdeckt, ist die folgende:

externe Angriffe

1. Bedrohungen durch externe Angriffe:

Externe Bedrohungen ergeben sich durch Aktionen eines Angreifers, die er ohne die Hilfe des bedrohten technischen Systems durchführt. Zu den externen Bedrohungen zählen physische Zerstörung von Systemkomponenten sowie Diebstahl von Ressourcen wie beispielsweise von Magnetbändern, Festplatten oder ganzen Geräten wie Notebooks und Smartphones.

interne Angriffe

2. Bedrohungen der Datenintegrität und der Informationsvertraulichkeit:

Unter diesen Problembereich fallen Angriffe, die gezielt versuchen, interne Kontrollen zu umgehen oder Schutzmechanismen unwirksam zu machen. Beispiele hierfür sind direkte Zugriffe auf Speichermedien unter Umgehung der Zugriffskontrolle des zugrunde liegenden Betriebssystems oder die Verwendung verdeckter Informationskanäle, um vertrauliche Informationen auszutauschen.

Denial-of-Service

3. Bedrohungen der Verfügbarkeit und der Ressourcennutzung:

Bedrohungen dieses Gefährdungsbereichs betreffen die unautorisierte Ressourcennutzung wie beispielsweise den Zugriff auf teure Ein/Ausgabegeräte oder die Nutzung von Rechenkapazität, bzw. die eine „normale“ Ressourcenbelegung übersteigende Nutzung von Ressourcen.

Abstreiten

4. Abstreiten durch geführter Aktionen:

Die Bedrohungen dieser Gefährdungsklasse betreffen zum einen den Bereich der Abrechnung genutzter Ressourcen wie Speicher, E/A-Geräte

oder CPU-Zeit. Zum anderen betreffen sie die Abrechnung in Anspruch genommener Dienste und die daraus resultierenden rechtsverbindlichen, vertraglichen Folgen, wie das Durchführen von Banktransaktionen oder das Abwickeln elektronischer Einkäufe.

5. Missbrauch erteilter Berechtigungen:

Rechtemissbrauch

Bedrohungen dieser Klasse treten auf, wenn ein autorisierter Benutzer die ihm erteilten Berechtigungen und das in ihn gesetzte Vertrauen missbraucht, indem er zum Beispiel Daten gezielt manipuliert.

Die Spalten einer Bedrohungsmatrix werden durch die potentiellen Auslöser von Bedrohungen definiert. Hierunter fallen die Systemadministratoren bzw. Operatoren, die Programmierer, die System- oder Anwendungssoftware erstellen, Benutzer, die intern oder über externe Medien auf das System zugreifen. Auslöser können ferner die verwendeten Dienste, Protokolle und Ausführungsumgebungen wie zum Beispiel E-Mail-Dienste, die TCP/IP-Protokollfamilie oder die Java Virtual Machine (JVM) zur Ausführung von Java-Applets sein.

Auslöser

In die Matrix trägt man potentielle Angriffsszenarien ein. Ein Auszug einer solchen Matrix ist mit Tabelle 4.2 gegeben. In den Tabelleneinträgen sind jedoch nur Beispiele für Bedrohungen festgehalten.

	Programmierer	interner Benutzer	externer Benutzer	mobiler Code
externe Angriffe	u.a. Vandalismus	Beobachten der Passworteingabe	–	–
interne Angriffe	direkter Speicherzugriff	logische Bomben	Passwort knacken	Viren
Verfügbarkeit	Speicher belegen	Prozesse erzeugen	Netzlast erzeugen	Monopolisieren der CPU

Tabelle 4.2: Auszug aus einer Bedrohungsmatrix

4.4.2 Bedrohungsbaum

Bedrohungsbaum

Die potentiellen Bedrohungen für ein System kann man auch mittels Bedrohungs- bzw. Angriffsbäumen (engl. *attack tree*) erfassen. Die Technik der Bedrohungsanalyse mittels Bedrohungsbäumen ist sehr stark angelehnt an die Analyse von Fehlern durch Fehlerbäume im Bereich der Systemzuverlässigkeit. Die generelle Vorgehensweise dabei lässt sich wie folgt zusammenfassen. Die Wurzel eines Bedrohungsbäumes definiert ein mögliches Angriffsziel und damit eine mögliche Bedrohung des Systems. Der Bedrohungsbaum für das gewählte Angriffsziel wird nun so aufgebaut, dass zunächst in der nächsten Ebene des Baumes Zwischenziele definiert werden, die zur Erreichung des Gesamtziels beitragen. Um festhalten zu können, welche der entsprechenden Zwischenziele gemeinsam erfüllt sein müssen und bei welchen es ausreicht, dass mindestens eines davon erfüllt ist, werden UND- bzw. ODER-Knoten verwendet. Die Äste eines UND-Knotens beschreiben die Bedrohungen, die zusammen eintreten müssen, damit diejenige Bedrohung eintritt, die den Wurzelknoten bzw. das Subziel charakterisiert. Bei einem ODER-Knoten werden die möglichen Alternativen erfasst, die zum Eintreten der Bedrohung an dem Elternknoten führen können. Die Blätter des Baumes beschreiben jeweils einen einzelnen Angriffsschritt und die möglichen Pfade von den Blättern zur Wurzel beschreiben unterschiedliche Wege zum Erreichen des globalen Angriffsziels an der Wurzel.

UND/ODER-Knoten

Da in vielen Szenarien ähnliche Angriffe und Bedrohungen auftreten, kann man bei der Erstellung von Bedrohungsbäumen zunächst allgemeine Angriffsziele behandeln. Mögliche allgemeine Angriffsziele sind:

1. Das Vortäuschen der Identität eines autorisierten Benutzers, also ein Maskierungssangriff.
2. Das Verändern der Betriebsssoftware zum Beispiel durch Buffer-Overflows, Viren oder Trojanische Pferde. Dadurch können Schutzmechanismen außer Kraft gesetzt und sensible Informationen Unbefugten zugänglich werden, Daten verändert oder gelöscht werden oder auch der Benutzer bei der Durchführung von Arbeiten gezielt behindert werden (Denial of Service).
3. Der unautorisierte lesende oder schreibende Zugriff auf gespeicherte Daten und Informationen.
4. Die unautorisierte Veränderung oder Kenntnisnahme von sensiblen Informationen, die über Kommunikationswege ausgetauscht werden.
5. Die Erstellung von Kommunikations- und Bewegungsprofilen von Kommunikationspartnern.

Angriffsziel

Anhand eines einfachen Beispiels wird im Folgenden die Erstellung eines Bedrohungsbäumes demonstriert.

Beispiel 4.2 (Bedrohungsbäum)

Betrachten wir ein Szenario mit einem mobilen Endgerät, zum Beispiel ein Smartphone, auf das entweder direkt durch physischen Besitz des Gerätes oder über ein Kommunikationsnetz zugegriffen werden kann. Das Angriffsziel bestehe nun darin, einen Maskierungsangriff durchzuführen, indem das Authentifikationsprogramm des mobilen Endgeräts überlistet wird, ohne jedoch den Code dieser Prozedur zu ändern oder die Prozedur vollständig zu umgehen. Das Ändern des Codes oder das vollständige Umgehen der Authentifikation sind weitere Angriffsziele, die in separaten Bedrohungsbäumen zu erfassen sind. Ist das Ziel festgelegt, so müssen die möglichen Pfade zum Erreichen des Ziels hergeleitet werden. Bei der Erstellung eines Bedrohungsbäumes ist es wichtig, dass dessen Entwicklung nachvollziehbar ist und Bedrohungen möglichst vollständig erfasst werden. In den Bedrohungsbäumen wurden deshalb spezielle Zwischenknoten als Entscheidungsknoten (gestrichelte Kästen) aufgenommen, um den Verfeinerungsprozess zu systematisieren. Im vorliegenden Beispielszenario werden beispielsweise lokale und entfernt durchgeführte Angriffe getrennt untersucht.

Maskierung

Betrachten wir nun den linken Zweig des Beispielbaumes weiter. Als Subziele (grau hinterlegte Kästen in Abbildung 4.5) definieren wir zwei neue Angriffsziele, nämlich zum einen das korrekte Login und zum anderen das Aneignen des mobilen Geräts. Da es sich um UND-Knoten handelt, müssen beide Teilziele erfüllt sein, um das übergeordnete Ziel zu erreichen. In der grafischen Darstellung werden die alternativen Wege zum Erreichen des Ziels als ODER-Knoten repräsentiert. ODER-Knoten sind die Default-Attributierung. Sie werden im Graf nicht explizit annotiert, während die Kanten zu den UND-Knoten explizit im Graf annotiert werden. In unserem Szenario erfordert beispielsweise ein erfolgreiches lokales Anmelden am Endgerät, dass der Angreifer sowohl im Besitz des Endgerätes als auch in der Lage ist, die Login-Prozedur fehlerfrei abzuwickeln.

Subziel

Ein möglicher Angriffspfad in dem Szenario ist in der Abbildung 4.6 zu sehen. Der einzige Angriffsschritt, der zum Erreichen des Ziels notwendig ist, besteht hierbei lediglich darin, dass der Angreifer sich vorübergehend das mobile Gerät aneignet. Dies kann auf vielfältige Weise möglich sein, wie z.B. während eines geschäftlichen Meetings, wenn das Gerät unbeaufsichtigt auf dem Tisch liegen bleibt. Wenn keine Authentifikation des Benutzers erforderlich ist, kann sich der Angreifer nun direkt einloggen.

Angriffspfad

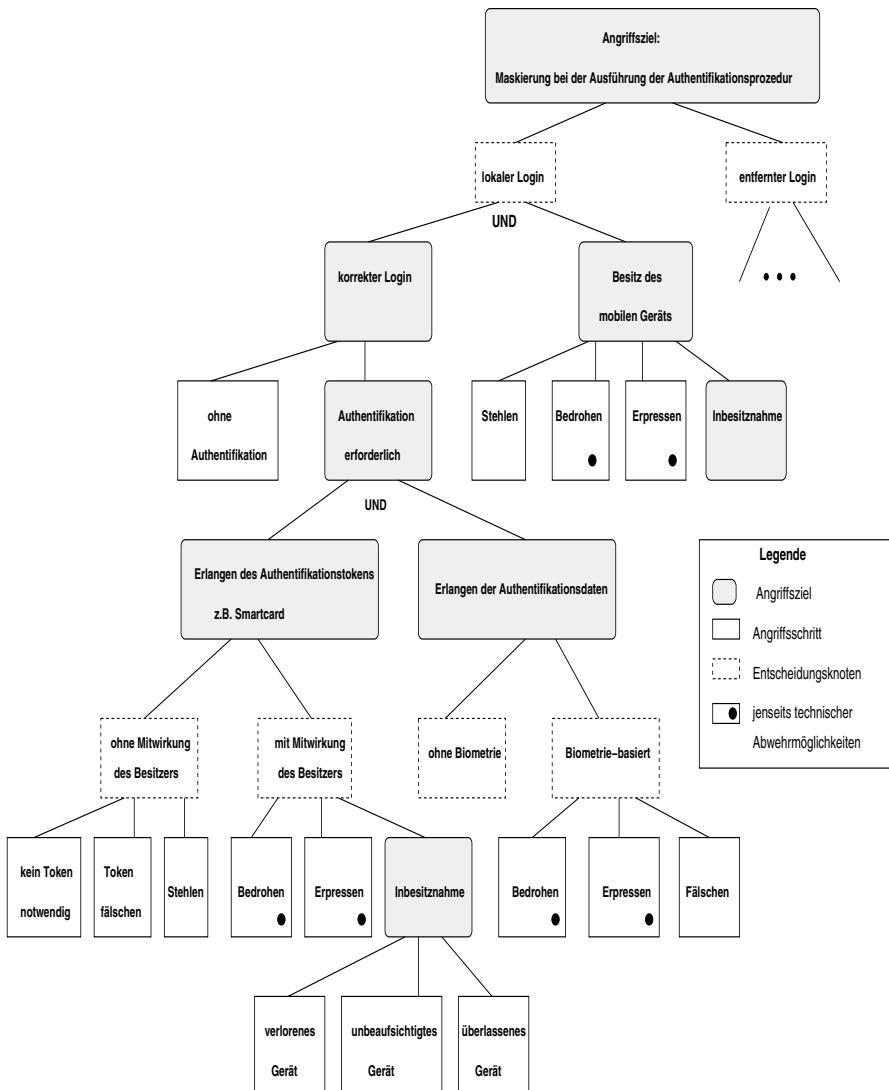


Abbildung 4.5: Bedrohungsbäume für einen Maskierungsangriff

An dem Bedrohungsbau lässt sich unmittelbar ablesen, dass ein Maskierungsangriff deutlich erschwert wird, falls man zur Authentifikation biometrische Merkmale des Gerätebesitzers oder den Besitz eines persönlichen Authentifikationstokens fordert. Bedrohungsbäume helfen somit auch dabei, unmittelbar notwendige und effektive Maßnahmen zur Abwehr von Bedrohungen auf systematische Weise zu identifizieren.



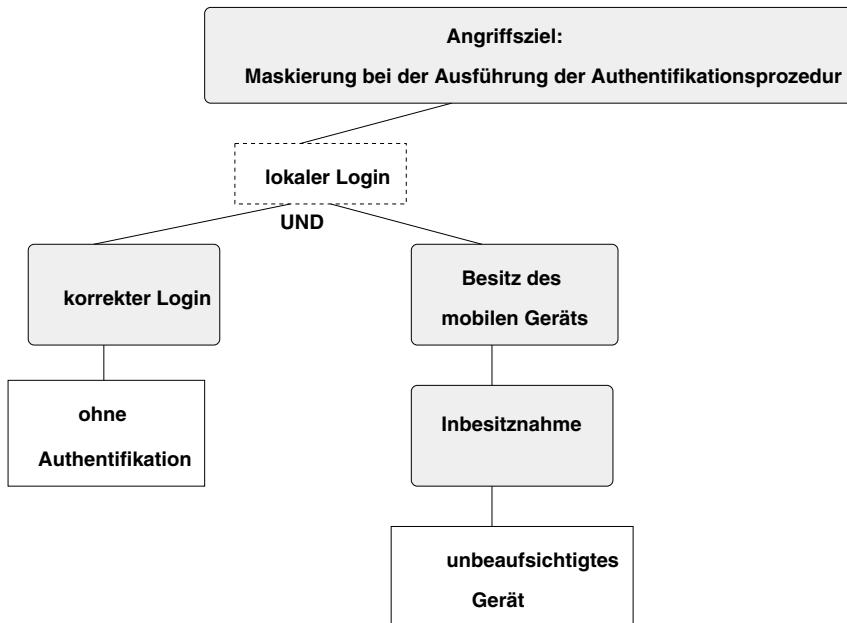


Abbildung 4.6: Ein möglicher Angriffspfad

Zur kompakteren Darstellung von Bedrohungsbäumen verwendet man auch häufig eine textuelle Notation. Anhand des obigen Beispiels wird im Folgenden die textuelle Darstellung erklärt.

Notation

Beispiel 4.3 (Bedrohungsbauum textuell)

Bei der textuellen Darstellung eines Bedrohungsbauums listet man die ODER- und UND-Teilbäume systematisch auf. Nachfolgend ist ein Ausschnitt des Baums aus Abbildung 4.5 textuell dargestellt. Daran sollte die prinzipielle Vorgehensweise deutlich werden. Beginnend mit der Wurzel des Baums, dem Angriffsziel, werden alle direkten Nachfolger der Wurzel auf einer Beschreibungsebene dargestellt. Da es sich in dem vorliegenden Beispiel um ODER-Knoten handelt, wird die Ebene als ODER-Ebene annotiert. Jede Ebene des Baums wird durch ein weiteres Einrücken bei der textuellen Aufschreibung nachgebildet.

textuelle Darstellung

Ziel: Maskierung bei Ausführung der Authentifikationsprozedur

ODER **Subziel 1:** lokaler Login

UND **Subziel 1.1:** korrekter Login

ODER **Subziel 1.1.1:** Authentifikation ist nicht gefordert

Subziel 1.1.2: ohne Authentifikation

UND **1.1.2.1:** Erlangen des Authentifikationstokens

1.1.2.2: Erlangen der Authentifikationsdaten

Subziel 1.2: Besitz des mobilen Gerätes

ODER Stehlen

Bedrohen des Besitzers

Erpressen

Subziel 1.2.1: Inbesitznahme des Gerätes

ODER verlorenes Gerät

unbeaufsichtigtes Gerät

überlassenes Gerät

Subziel 2: entfernter Login

Aus dieser Aufschreibung ist unmittelbar erkennbar, dass das globale Angriffsziel erreicht werden kann, wenn entweder das Subziel 1 oder das Subziel 2 erfolgreich erreicht wird. Subziel 1 wiederum erfordert, dass sowohl Teilziel 1.1 als auch 1.2 erzielt sind, und so weiter.



Angriffsbäume sind eine geeignete Technik, um sich auf eine systematische Weise die verschiedenen Wege zum Erreichen eines Angriffsziels klar zu machen. Dies ist wichtig, da häufig nur einige nahe liegende Bedrohungen betrachtet und Abwehrmaßnahmen allein dagegen ergriffen werden. Das ist vergleichbar mit der Absicherung des Zugangs zu einem Haus. Konzentriert man sich dabei allein auf die Haustür als einzige Schwachstelle und baut komplexe Sicherheitsschlösser ein, so ist einem Gelegenheitseinbrecher dieser Angriffsweg versperrt. Ist es aber gleichzeitig möglich, über ein offen stehendes Kellerfenster in das Haus einzudringen, nutzen die punktuell angewendeten Abwehrmaßnahmen nichts. Das folgende Beispiel soll diesen Sachverhalt anhand eines typischen Szenarios in heutigen IT-Systemen noch einmal verdeutlichen.

Beispiel 4.4 (Bedrohungsszenario für eine verschlüsselte Nachricht)

Szenario

Betrachten wir zwei Kommunikationspartner Alice (A) und Bob (B), die verschlüsselte Daten austauschen. Gegeben sei eine Nachricht M , die mit einem gemeinsamen geheimen Schlüssel K_{AB} verschlüsselt und als verschlüsselte Nachricht übertragen wurde. Ein Angreifer habe nun das Ziel, Kenntnis von der ursprünglichen Nachricht M zu erlangen.

Ziel: Lesen von M :

ODER **Subziel 1:** Dechiffrieren der verschlüsselten Nachricht M

ODER Knacken

Kryptoanalyse

Subziel 2: Bestimmung des Schlüssels K_{AB}

ODER **Subziel 2.1:** Zugriff auf Speicher von A

Subziel 2.2: Zugriff auf Speicher von B

Subziel 2.3: Verleite A zur Preisgabe von K_{AB}

Subziel 2.4: Verleite B zur Preisgabe von K_{AB}

Subziel 3: Empfänger entschlüsselt für Angreifer

ODER **Subziel 3.1:** Spoofing von A

Subziel 3.2: Wiedereinspielung, Entschlüsselung durch B

Subziel 3.3: Lesen von M direkt aus Speicher von A

Subziel 3.4: Lesen von M direkt aus Speicher von B

In heutigen IT-Systemen wird ein sehr starkes Augenmerk darauf gerichtet, dass zur Verschlüsselung von Nachrichten kryptografisch starke Verfahren mit langen Schlüsseln (vgl. Kapitel 7) eingesetzt werden, um Angriffe zum Erreichen des Subziels 1 abzuwehren. Wie man an obigem Baum aber deutlich sieht, steht einem Angreifer eine Vielzahl weiterer Möglichkeiten, nämlich alle Alternativen der Subziele 2 und 3, offen, die vertrauliche Nachricht M zu lesen. Es werden somit sehr viel umfassendere Abwehrmaßnahmen als allein die starke Verschlüsselung benötigt, um sicherzustellen, dass der Angreifer sein primäres Ziel nicht erreicht.



Bedrohungsbäume lassen sich auch zur systematischen Erfassung der Risiken, die mit den Bedrohungen verbunden sind, heranziehen.

4.5 Risikoanalyse

Sind die Bedrohungen erfasst, so muss in der Risikoanalyse (engl. *risk assessment*) deren Bewertung durchgeführt werden. Zur Bewertung der Risiken sind Wahrscheinlichkeiten für das Eintreten der verschiedenen Bedrohungen zu ermitteln und der potentielle Schaden, der dadurch verursacht werden kann, ist abzuschätzen. Sei S die Schadenshöhe (das Schadensausmaß) und E die Eintrittswahrscheinlichkeit. Dann ergibt sich ein quantitativer Wert für das Risiko durch $R = S * E$.

Risikobewertung

Bei der Bestimmung der Schadenshöhe S unterscheidet man zwischen primären und sekundären Schäden. Zu den primären Schäden zählen u.a. Schäden durch einen Produktivitätsausfall, Wiederbeschaffungs-, Personal- oder Wiederherstellungskosten. Diese Schäden lassen sich i.d.R. noch relativ einfach quantifizieren. Sehr viel schwieriger ist eine Quantifizierung

der sekundären Schäden. Hierunter fallen Schäden durch einen möglichen Imageverlust, durch einen Vertrauensverlust bei Kunden und Geschäftspartnern etc.

Die Eintrittswahrscheinlichkeiten ergeben sich aus dem geschätzten Aufwand, den ein Angreifer zur Durchführung eines Angriffes treiben muss, und einer Einschätzung des möglichen Nutzens, den er aus einem erfolgreichen Angriff ziehen könnte. Um den erforderlichen Aufwand für einen Angriff abzuschätzen, führt man häufig Penetrationstests (siehe Seite 197) durch, um die Anfälligkeit des Systems gegen klassische Angriffe abzutesten. Darüber hinaus bedient man sich meist öffentlich zugänglicher Informationen über bekannte Angriffsmuster, über häufig durchgeführte Angriffe sowie über frei verfügbare Exploits. Entsprechende Informationen sind z.B. über CERT-Webseiten, wie beispielsweise die Seite des CERTs des DFN-Vereins unter <http://www.dfn-cert.de/>, oder über die Seite des OWASP-Projekts⁴ zugänglich. In die Bewertung der Eintrittswahrscheinlichkeit muss weiterhin eine Abschätzung des Ausmaßes möglicher Schäden und eine Einschätzung der möglichen Motive eines Angreifers einfließen. Es ist somit klar, dass die Risikobewertung sehr stark von dem zugrunde liegenden Modell, das die möglichen Angreifer beschreibt, abhängt.

Angreifer-Modell

In einer Risikoanalyse werden unterschiedliche Angreifer-Modelle erstellt, um die verschiedenen Gefährdungslagen adäquat zu erfassen. In einem Angreifer-Modell sind unter anderem der Angreifertyp (z.B. Skript Kiddie, Hacker, Mitarbeiter, Wirtschaftsspion) und das zur Verfügung stehende Budget zu beschreiben. Ein sehr geringes Finanzbudget steht beispielsweise Skript Kiddies, Mitarbeitern und meist auch Hackern zur Verfügung, während professionelle IT-Kriminelle und Wirtschaftsspione eher über ein mittleres Budget und Regierungsbehörden typischerweise sogar über ein sehr großes Budget verfügen können. Ein Angreifer ist ferner charakterisiert durch seine Kenntnisse, die von nicht vorhandenen Kenntnissen bis hin zu detailliertem Insiderwissen reichen können. Schließlich sollte in ein Angreifer-Modell auch einfließen, welche Ziele ein Angreifer verfolgt. Diese können von Experimentierfreude und einfacher Neugier, über reines Gewinnstreben bis hin zu Rache reichen.

4.5.1 Attributierung

Sind die Bedrohungen durch einen Baum erfasst, so können deren Risiken als Attributierungen der Knoten des Baumes angegeben werden.

⁴ Open Web Application Security Project

Definition 4.1 (Attributierung)

Gegeben sei ein Bedrohungsbaum T , mit der Menge K seiner Knoten. Sei A die endliche Menge von Attributen. Eine Abbildung $f : K \rightarrow A$, oder $f : K \rightarrow 2^A$, nennen wir eine Attributierung von T .

□

Beispiele für Elemente der Attributmenge A sind die Kosten und der Aufwand zur Durchführung eines Angriffs oder auch die Notwendigkeit zum Einsatz spezieller Hardware bzw. spezieller Geräte, wie Lasercutter für Angriffe auf Smart Cards, um das Angriffsziel zu erreichen.

Beispiel 4.5 (Einfache Attributierung)

Gegeben sei die Menge $A = \{U, M\}$, wobei U für unmöglich und M für möglich steht. Die Attributierungsabbildung f sei wie folgt definiert:

Für jeden ODER-Knoten $k \in K$ gilt:

- (1) $f(k) = M \iff \exists k' \in suc(k) : f(k') = M$ und
- (2) $f(k) = U \iff \forall k' \in suc(k) : f(k') = U$, wobei

$suc(k)$ die Menge aller Nachfolger-Knoten von k im Baum beschreibt.

Für jeden UND-Knoten $k \in K$ gilt:

- (1) $f(k) = M \iff \forall k' \in suc(k) : f(k') = M$ und
- (2) $f(k) = U \iff \exists k' \in suc(k) : f(k') = U$.

Mögliche Angriffe sind alle diejenigen Pfade von Blättern zur Wurzel, die mit M attribuiert sind.

▲

Zusammen mit den Varianten der UND- bzw. ODER-Knoten zur Erfassung von Bedrohungen steht mit der Attributierung ein Instrumentarium zur „Berechnung“ von Risiken zur Verfügung. Typischerweise werden Kombinationen aus verschiedenen Attributen verwendet, um eine Einschätzung der möglichen Risiken zu erhalten. Ein Beispiel hierfür ist die Suche nach den billigsten Angriffen, die ohne Bedarf an spezifischen Geräten und ohne spezifisches Know-how durchführbar sind. Wünschenswert wäre hier eine Werkzeugunterstützung, die es erlaubt, Anfragen an die Bedrohungsbäume mit unterschiedlichen Angriffsmerkmalen zu formulieren. Ein solches Werkzeug sollte ferner mit der Antwort bereits einen Maßnahmenkatalog von Abwehrmöglichkeiten gegen die Bedrohungen vorschlagen können. Eine derartige Unterstützung fehlt jedoch zurzeit leider noch.

Beispiel 4.6 (Risikoberechnung)

Gegeben sei der Ausschnitt aus dem Angriffsbaum von Abbildung 4.7, mit dem das Risiko, dass ein Benutzerpasswort ausgespäht werden kann,

analysiert wird. In diesem Szenario gehen wir davon aus, dass das bedrohte IT-System ein Heim-PC mit Internetanschluss ist. Das bedeutet, dass man sich bei der Erstellung von Angreifer-Modellen im Wesentlichen auf externe Angreifer beschränken kann.

Angreifer-Modell

Unserer Risikoanalyse legen wir deshalb ein einfaches Modell zugrunde und gehen von externen Angreifern aus, denen keine oder nur sehr geringe Ressourcen zur Angriffs durchführung zur Verfügung stehen und die nur geringe fachliche Kenntnisse besitzen. Entsprechend ist das Risiko, dass das Passwort direkt beim Benutzer ausgespäht werden kann, als gering einzuschätzen. Demgegenüber ist das Risiko als sehr hoch zu bewerten, dass ein Angreifer Kenntnis von unverschlüsselt übertragenen Passwörtern erlangt. Ein Angreifer kann dazu beispielsweise frei verfügbare Sniffer-Programme verwenden, um auf einfachste Weise übertragene Passwörter abzufangen. Zur Durchführung eines solchen Angriffs wird weder spezielles Know-how noch werden spezifische Ressourcen benötigt.

Deutlich geringer schätzen wir für dieses Angreifer-Modell die Gefahr ein, dass es einem Angreifer gelingt, gespeicherte Passwörter von dem PC auszulesen. Obwohl wir hierbei von einem wenig fachkundigen Benutzer ausgehen, der eine Standardvariante von Windows verwendet und keine spezifischen Maßnahmen getroffen hat, sein System vor Angriffen von außen zu schützen, muss ein Angreifer hier mehr Aufwand treiben, um zum Erfolg zu kommen. Das Risiko ist dennoch hoch, da auch hierfür frei verfügbare Exploits oder Angriffe über Viren und Würmer nutzbar sind, um gezielt Daten von der Festplatte des Benutzers auf den Rechner des Angreifers zu transferieren.

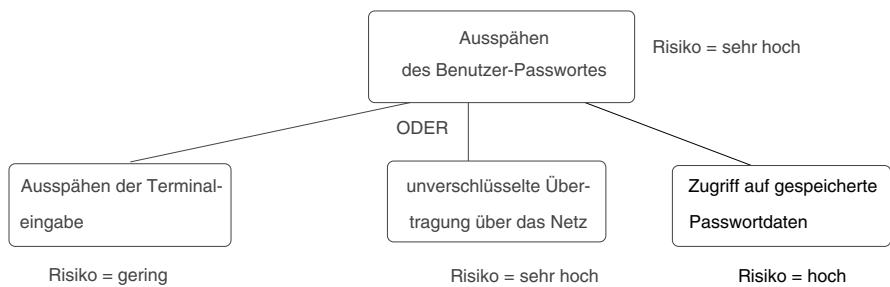


Abbildung 4.7: Beispiel einer Risikoberechnung

Aus der Risikoanalyse lässt sich unmittelbar ableiten, dass sogar bei einem so einfachen Angreifer-Modell ein sehr hohes Risiko besteht, dass das Benutzer-Passwort ausgespäht werden kann. Daraus lässt sich unmittelbar der Schluss ziehen, dass Passwörter nur verschlüsselt übertragen werden

dürfen und tunlichst nicht auf der Festplatte im Klartext abgelegt sein sollten, um wenigstens die einfachsten Angriffe abzuwehren.



BSI-Risikoanalyse

Die Durchführung einer Risikoanalyse ist nicht einfach und erfordert großen technischen und organisatorischen Sachverstand. In den IT-Grundschutz-Katalogen des BSI wird eine Risikoanalyse deshalb nur als Ergänzung für solche Systemteile empfohlen, deren Sicherheitsanforderungen über die niedrigen bis mittleren Anforderungen hinausgehen (vgl. Abbildung 4.3), oder wenn das Unternehmen, die Behörde oder Institution wichtige Anwendungen oder Komponenten betreibt, die nicht in den IT-Grundschutz-Katalogen des BSI behandelt werden. Die Kataloge enthalten für niedrige und mittlere Sicherheitsbedürfnisse Standard-Sicherheitsmaßnahmen aus den Bereichen Organisation, Personal, Infrastruktur und Technik, die bei normalen Sicherheitsanforderungen in der Regel angemessen und ausreichend zur Absicherung von typischen IT-Infrastrukturen sind. Das BSI hat im Jahre 2004 eine Methodik zur Risikoanalyse auf der Basis der IT-Grundschutz-Kataloge erarbeitet und veröffentlicht⁵, die darauf abzielt, möglichst nahtlos als Ergänzung der IT-Grundschutz-Analyse eingesetzt zu werden.

ergänzende
Analyse

Das Vorgehensmodell baut auf einer Strukturanalyse, einer Ermittlung des Schutzbedarfs sowie einer Modellierung des Systems gemäß Grundschutzzvorgaben auf (vgl. Abbildung 4.8) und identifiziert zunächst alle hochschutzbedürftigen Zielobjekte und ignoriert alle Bausteine⁶, die für diese Zielobjekte nicht relevant sind. Beispielsweise kann auf die Bausteine 3.3 *Notfallvorsorge-Konzept* und 3.8 *Behandlung von Sicherheitsvorfällen* meist verzichtet werden, wenn in der Risikoanalyse nur Teilbereiche behandelt werden, die einen normalen Schutzbedarf in Bezug auf Verfügbarkeit haben.

Vorgehen

Als Ergebnis dieser Schritte liegt eine Tabelle vor, in der die Bausteine aufgeführt sind, die für die hochschutzbedürftigen Zielobjekte relevant sind. Da jedem Baustein aus den IT-Grundschutz-Katalogen eine Liste von Gefährdungen zugeordnet ist, kann im nächsten Schritt ein Überblick über die Gefährdungslage gewonnen werden. Diese ist um zusätzliche Gefährdungen zu erweitern, da die Grundschatzkataloge keine Gefährdungen aufführt, die in üblichen Einsatzszenarien nur unter sehr speziellen Voraussetzungen zu einem Schaden führen oder sehr gute Fachkenntnis, Gelegenheiten und Mittel eines Angreifers voraussetzen. Hat das Zielobjekt in einem der Berei-

Gefährdungslage

⁵ siehe https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/ITGrundschutzstandards/standard_1003_pdf.pdf?__blob=publicationFile

⁶ Bausteine enthalten einen Überblick über die Gefährdungslage und die Maßnahmenempfehlungen für eine betrachtete Komponente.

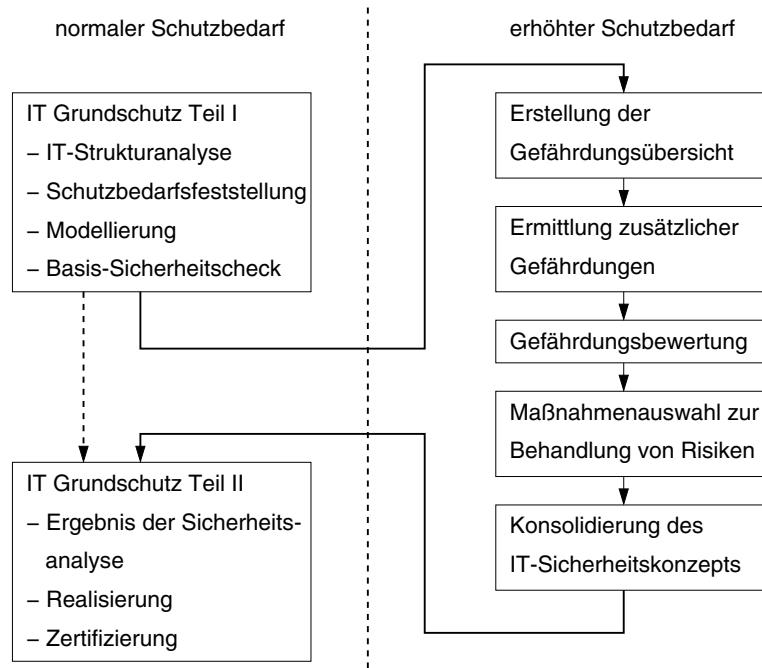


Abbildung 4.8: Ergänzende Risikoanalyse (Quelle: BSI-Grundschutz-Kataloge (jetzt BSI-Standard 100-3))

che Vertraulichkeit, Integrität und Verfügbarkeit den Schutzbedarf sehr hoch oder hoch, sollten vorrangig solche Gefährdungen gesucht werden, die diesen Grundwert beeinträchtigen. Laut dem Vorgehensmodell des BSI sollten bei der Ermittlung zusätzlicher Gefährdungen die folgenden Fragestellungen berücksichtigt werden:

- Von welchen möglichen Ereignissen aus dem Bereich höhere Gewalt droht besondere Gefahr für das IT-System?
- Welche organisatorischen Mängel müssen auf jeden Fall vermieden werden, um die IT-Sicherheit zu gewährleisten?
- Welche menschlichen Fehlhandlungen können den sicheren IT-Betrieb besonders beeinträchtigen?
- Welche speziellen Sicherheitsprobleme können beim jeweils betrachteten Zielobjekt durch technisches Versagen entstehen?
- Welche besondere Gefahr droht durch vorsätzliche Angriffe von Außentätern?

*zusätzliche
Gefährdungen*

Risiko-Reduktion

Im nächsten Schritt wird für jedes Zielobjekt geprüft, ob die bereits umgesetzten oder im IT-Sicherheitskonzept vorgesehenen IT-Sicherheitsmaßnah-

men einen ausreichenden Schutz bieten. Da sich bei der Gefährdungsbewertung meist mehrere Gefährdungen ergeben werden, denen die Maßnahmen aus den IT-Grundschutz-Katalogen nicht ausreichend entgegenwirken, müssen zusätzliche Maßnahmen zur Risiko-Reduktion eingeführt werden, die in der Regel mit erheblichen Kosten verbunden sein werden, so dass eine Kosten/Nutzen-Analyse (auch betriebswirtschaftlich unterlegt) sinnvoll ist. Abschließend muss das IT-Sicherheitskonzept konsolidiert werden, falls bei der Behandlung von verbleibenden Gefährdungen zusätzliche Maßnahmen zu den Standard-Sicherheitsmaßnahmen hinzugefügt wurden.

4.5.2 Penetrationstests

Um die Erfolgsaussichten eines vorsätzlichen Angriffs einschätzen zu können, ist es nützlich, Penetrationstests durchzuführen. In einem solchen Test wird das Angriffsverhalten eines vorsätzlichen Innen- oder Außentäters simuliert, wobei versucht wird zu ermitteln, welche Schwachstellen ausgenutzt und welche Schäden verursacht werden können. Bei der Simulation eines Innentäters verfolgt man in der Regel einen Whitebox-Ansatz. Das heißt man geht davon aus, dass der Angreifer über detaillierte Kenntnisse über die interne Struktur, die Anwendungen und Dienste verfügt. Demgegenüber wird ein Außentäter meist über einen Blackbox-Ansatz simuliert, bei dem davon ausgegangen wird, dass der Angreifer höchstens über wenig und leicht zugängliche Informationen über das System verfügt.

Penetrationstest

Typische Penetrationstests umfassen:

- Angriffe durch Erraten von Passwörtern oder Wörterbuchattacken,
- Angriffe durch Aufzeichnen und Manipulieren des Netzverkehrs,
- Angriffe durch Einspielen gefälschter Datenpakete oder
- Angriffe durch Ausnutzen bekannter Software-Schwachstellen (Makro-Sprachen, Betriebssystemfehler, Remote-Dienste etc.).

Ein Penetrationstest durchläuft typischerweise mehrere Schritte, die im Folgenden skizziert werden.

Vorgehensweise

1. Zunächst erfolgt i.d.R. eine Recherche im Internet, um frei zugängliche Informationen über den Untersuchungsgegenstand, wie die Menge der erreichbaren IP-Adressen, zu ermitteln.
2. Mittels Portscannern oder ähnlichen Analyse-Tools werden die von dem Zielsystem angebotenen Dienste, bzw. die geöffneten Ports ermittelt, die Rückschlüsse darauf zulassen, welche Anwendungen und Dienste an diese Ports gekoppelt sein könnten.

3. Im nächsten Schritt wird versucht, mittels Fingerprinting-Methoden Informationen über das verwendete Betriebssystem, dessen Version, über den verwendeten Internet-Browser, die zugrunde liegende Hardware und auch über sonstige Anwendungen, die auf dem Zielsystem installiert sind, in Erfahrung zu bringen.
4. Unter Rückgriff auf das Expertenwissen in Schwachstellendatenbanken wird dann versucht, Schwachstellen, die über die auf dem Zielsystem installierte Betriebs- und Anwendungssoftware bekannt sind, zu identifizieren.
5. Es wird dann versucht, derart identifizierte Schwachstellen auszunutzen, um systematisch unberechtigten Zugriff auf das Zielsystem zu erlangen oder um weitere Angriffe vorzubereiten.

rechtlicher Rahmen

Gerade der letzte Schritt des oben aufgeführten groben Rasters verdeutlicht, dass die Durchführung eines Penetrationstest nicht unproblematisch ist. So ist sicherzustellen, dass beim Eindringen in ein System durch das Testteam dessen produktiver Betrieb nicht nachhaltig gestört und dass keine irreparablen Schäden durch die Manipulation von Daten entstehen. Da mit dem Durchführen eines Penetrationstests datenschutzrechtliche Belange verknüpft sind, darf ein solcher Test nur in genauer Absprache mit dem Betreiber des Zielsystems und dem zuständigen Datenschutzbeauftragten erfolgen. Auch andere Gesetzesvorschriften, wie das Gesetz über den Schutz von zugangskontrollierten Diensten und von Zugangskontrolldiensten (ZKDSG) können sehr leicht durch Penetrationstests verletzt werden. Unter das ZKDSG fallen zugangskontrollierte Dienste wie ein passwortgeschützter WWW- oder FTP-Server. Da es das Ziel eines Penetrationstests ist zu versuchen, vorhandene Schutzmechanismen zu umgehen, ist ein Verstoß gegen das ZKDSG kaum vermeidbar. Ein Penetrationstester, der ohne sich die Erlaubnis des Betroffenen eingeholt zu haben, einen Exploit zum Remote-Zugriff auf einem passwortgeschützten Webserver besitzt, verhält sich ordnungswidrig und kann mit einer Geldbuße bis zu 50.000 Euro konfrontiert werden [60]. Eine ausführliche Darstellung der gesetzlichen Randbedingungen für die Durchführung eines Penetrationstest sowie dessen organisatorischer und technische Durchführung ist in der Studie *Durchführungskonzept für Penetrationstests*⁷ des BSI zu finden.

Anhand der Bedrohungs- und Risikoanalyse sowie der Ergebnisse von Penetrationstests sind die funktionalen Anforderungen zu überprüfen und gegebenenfalls zu revidieren bzw. zu ergänzen.

⁷ Erhältlich unter https://www.bsi.bund.de/ContentBSI/Publikationen/Studien/pentest/index_htm.html

4.6 Sicherheitsarchitektur und Betrieb

Aus den Ergebnissen der Bedrohungs- und Risikoanalyse ergibt sich der Ist-Zustand bezüglich des geltenden Sicherheitsstandards des Systems. Aus dem Abgleich dieses Ist-Zustandes mit dem Soll-Zustand, der den Schutzbedarf beschreibt, ist abzuleiten, welche Maßnahmen zur Abwehr der Bedrohungen erforderlich sind. Die entsprechenden Maßnahmen sind zu klassifizieren. Wesentliche Klassifikationskriterien hierfür sind die Wichtigkeit der bedrohten Komponenten und die Kosten und der Aufwand, den die benötigten Kontroll- und Schutzmaßnahmen verursachen. Dazu gehören die Kosten für die Beschaffung zusätzlicher Hardware sowie Kosten im Sinne von Effizienzeinbußen, die sich durch Verzögerungen infolge der durchzuführenden Kontrollen ergeben. So lassen sich mit zusätzlicher Hardware in Form von redundanten Komponenten Angriffe auf die Verfügbarkeit einer Komponente abwehren oder zusätzliche Rechner, die als Firewall-Rechner eingesetzt werden, ermöglichen es, unautorisierte Netzzugriffe abzuweisen.

Ist-Zustand

4.6.1 Sicherheitsstrategie und Sicherheitsmodell

Die erforderlichen Maßnahmen zur Erfüllung des Schutzbedarfs sind in einer Sicherheitsstrategie bzw. einem Sicherheitsregelwerk (engl. *security policy*) informell oder präzise formalisiert zu erfassen. Da Klassen von Anwendungen sehr ähnliche Sicherheitsbedürfnisse besitzen, können zu deren Abdeckung allgemeine Sicherheitsgrundfunktionen eingesetzt werden. Auf die wesentlichen Grundfunktionen wird in Abschnitt 4.7 genauer eingegangen. Kriterienkataloge (vgl. Abschnitt 5) wie die internationalen Common Criteria definieren darüber hinaus Funktionsklassen für dedizierte Anwendungsszenarien, so dass ein Anwender zunächst klären muss, ob seine Sicherheitsanforderungen bereits von einer dieser Klassen abgedeckt werden bzw. welche Kombination von Grundfunktionen benötigt wird.

Regelwerk

Ausgehend von der Spezifikation der Sicherheitsanforderungen ist ein Sicherheitsmodell des Systems zu konstruieren, das die geforderten funktionalen und sicherheitsbezogenen Eigenschaften auf einem hohen Abstraktionsniveau modelliert. Das Modell ermöglicht es, von Realisierungseigenschaften zu abstrahieren und die Gewährleistung der geforderten Eigenschaften auf der Basis der Modellfestlegungen zu überprüfen. Wird ein formales Modell konstruiert, so ist es möglich, Eigenschaften des Systems formal nachzuweisen. Anhand des Modells sind die festgelegten Eigenschaften zum Beispiel in Bezug auf Vollständigkeit und Konsistenz zu überprüfen und die Sicherheitsstrategie ist gegebenenfalls zu präzisieren oder anzupassen. Kapitel 6 erklärt die wichtigsten Ansätze zur formalen Modellierung von Sicherheitseigenschaften und diskutiert die Einsatzfelder der jeweiligen Modellierungsansätze.

Modell

4.6.2 Systemarchitektur und Validierung

Architektur

Das Sicherheitsmodell ist durch einen Architektur-Grobentwurf zu konkretisieren, in dem die Systemkomponenten mit ihren Schnittstellen und Abhängigkeiten festgelegt werden. Der Grobentwurf identifiziert die zu schützenden Komponenten und definiert die Sicherheitskomponenten, die sicherheitsspezifische Funktionen durchführen. Diese Komponenten werden im Feinentwurf verfeinert und so detailliert spezifiziert, dass ein präziser Rahmen für die Implementierung, d.h. für die Codierung der Komponenten, festgelegt ist. In der Implementierungsphase sind die benötigten Datenstrukturen und Algorithmen zu spezifizieren. Dazu kann auf Standardmechanismen wie kryptografische Verfahren, Passwortschutz oder Zugriffskontrolllisten, auf Protokolle wie Schlüsselverteilungs- oder Authentifikationsprotokolle, sowie auf Dienste wie Zugriffskontrollen oder Auditing zurückgegriffen werden.

Validierung und Evaluation

Testen

Das implementierte System ist methodisch zu testen und zu evaluieren bzw. die sicherheitsrelevanten Funktionen sind, wenn möglich, zu verifizieren. Zur methodischen Validierung müssen Testziele, -pläne und -verfahren festgelegt und dokumentiert werden. Die Vollständigkeit der Testszenarien und Testläufe im Hinblick auf die Abdeckung der relevanten Problembereiche ist zu begründen und die Testergebnisse sind zu bewerten. Mit Modul- und Integrationstests ist die fehlerfreie Integration korrekter Modulbausteine in das System nachzuweisen. Tests auf der Implementierungsebene sollten Code-Inspektionen, wie ein Code Review, umfassen.

Evaluieren

Das implementierte System zusammen mit seiner Dokumentation und Einsatzumgebung kann abschließend einer Evaluierung durch Dritte gemäß einem nationalen oder internationalen Kriterienkatalog unterzogen werden. Durch die Evaluierung wird dem System eine Sicherheitsklassifikation erteilt, die sowohl Betreibern als auch Anwendern ein festgelegtes Maß an Sicherheit zusichert.

4.6.3 Aufrechterhaltung im laufenden Betrieb

Monitoring

Der Prozess des Security Engineerings endet nicht, wie vielfach fälschlicherweise angenommen, mit dem erfolgreichen Abschluss der Evaluation und der Installation des Systems beim Kunden bzw. Anwender. Vielmehr ist natürlich auch während des laufenden Betriebs zu prüfen, ob die verwendeten Sicherheitsmaßnahmen ausreichend sind. Notwendig ist ein Monitoring und Kontrollieren der Systemaktivitäten möglichst ohne Unterbrechung, um insbesondere auch auf neue Bedrohungen schnell reagieren zu können. Hierzu stehen bereits eine Vielzahl sowohl frei verfügbare (Freeware) als auch kommerzielle Analyse-Tools zur Verfügung.

Für einen Sicherheitsverantwortlichen bzw. Administrator ergibt sich jedoch das Problem, das jedes dieser Werkzeuge eine eigene Bedienoberfläche besitzt, spezifisch konfiguriert werden muss und einen ganz bestimmten Funktionsumfang aufweist, der nur einen Teil der benötigten Analysefunktionalität abdeckt. Die kontinuierliche Überprüfung des Sicherheitszustands heutiger Systeme wird deshalb häufig sträflich vernachlässigt, da ein hoher Aufwand erforderlich ist, die verwirklichten Sicherheitsmaßnahmen fortlaufend daraufhin zu überprüfen, ob sie noch ausreichend sind, die festgelegte Sicherheitspolicy zu erfüllen.

Vielzahl von Tools

4.7 Sicherheitsgrundfunktionen

Mit den Sicherheitsgrundfunktionen, die in diesem Abschnitt vorgestellt werden, steht eine Art „Baukasten“ von allgemeinen Maßnahmen zur Abwehr von Bedrohungen zur Verfügung. Diese Grundfunktionen sind entsprechend der individuellen Anforderungen des zu konstruierenden Systems zu kombinieren und zu realisieren.

Identifikation und Authentifikation

Zur Abwehr von Bedrohungen, die sich aus Maskierungsangriffen ergeben, sowie zur Abwehr unautorisierter Zugriffe müssen Subjekte und sicherheitsrelevante Objekte eindeutig identifizierbar sein. Subjekte müssen ferner in der Lage sein, ihre Identität nachzuweisen, also sich zu authentifizieren, so dass Authentifikationsmerkmale festzulegen sind. Beispiele hierfür sind Geheimnisse (Passwörter) oder charakteristische biometrische Merkmale wie Fingerabdrücke.

Identität

Die Sicherheitsanforderungen legen fest, ob und wenn ja, welche Subjekte zwar zu identifizieren sind, sich jedoch nicht auch authentifizieren müssen. Dies kann zum Beispiel beim Zugriff auf öffentlich verfügbare Informationen der Fall sein. Für die Sicherheitsgrundfunktion muss weiterhin festgelegt sein, wann, das heißt bei welchen Aktionen oder in welchen Zugriffskontexten, eine Identifizierung und Authentifikation durchzuführen ist. Herkömmlicherweise erfolgt durch ein Betriebssystem eine Authentifikation nur beim Systemzugang, also beim Login, so dass alle nachfolgend durchgeföhrten Aktionen auf der Gültigkeit dieser Kontrolle beruhen. Für sicherheitsrelevante Aktionen kann es aber wünschenswert sein, dass Subjekte vor jeder Ausführung einer solchen Aktion ihre Identität nachweisen, so dass eine entsprechende Kontrolle festzulegen ist. So ist es zum Beispiel sinnvoll, jede Transaktion in einer Homebanking-Anwendung individuell zu authentifizieren.

Authentifikation

Fehlerbehandlung

Schließlich muss zur Abwehr systematischer Angriffsversuche angegeben werden, welche Aktionen bei einer nicht erfolgreichen Identifikation bzw. Authentifikation durchzuführen sind. Beispiele hierfür sind das Protokollieren der Aktionen in einem Logfile oder das Sperren der Kennung nach mehrmaligem, vergeblichen Zugangsversuch.

Rechteverwaltung

Die Rechteverwaltung liefert die Basis zur Abwehr von Bedrohungen der Integrität und Vertraulichkeit von Objekten infolge unautorisierter Zugriffe. Subjekte besitzen Rechte zum Zugriff auf Objekte. Aus den Sicherheitsanforderungen ist zu entnehmen, welche Rechte für zu schützende Objekte festzulegen sind. Beispiele hierfür sind Lese/Schreib-Rechte auf Dateien, das Suchen und Einfügen von Objekten in Datenbanken oder das Ausführen von Methoden auf Objekten in objektorientierten Systemen wie Push oder Pop auf Keller-Objekten. Die Vergabe von Zugriffsrechten unterliegt meist einer Dynamik, so dass Subjekten Rechte auch entzogen werden können. Für die Rechteverwaltung ist festzulegen, welche Rechte dynamisch vergeben werden können und unter welchen Voraussetzungen Rechte vergeben bzw. geändert werden dürfen. In den meisten im Einsatz befindlichen Mehrbenutzerbetriebssystemen (u.a. Unix und seine Derivate) wird beispielsweise festgelegt, dass der Eigentümer einer Datei, in der Regel ist dies der Erzeuger der Datei, das Recht besitzt, Zugriffsrechte an seinen Dateien zu vergeben und auch wieder zurückzunehmen (Owner-Prinzip).

Rechte

Vergabe

Wahrnehmung

In der Rechteverwaltung ist darüber hinaus festzulegen, unter welchen Umständen ein Subjekt ein vergebenes Recht wahrnehmen darf. Beispiele hierfür sind die Einführung von Rollen, so dass Subjekte Rechte nur im Kontext der Aktivität einer spezifischen Rolle ausführen dürfen (z.B. Einrichten von Benutzerkennungen nur in der Rolle des Kennungsadministrators). Beschränkungen zur Nutzung von Rechten können auch funktionale Kontexte oder Tageszeiten betreffen. In Unix-basierten Systemen ist es beispielsweise mit dem Konzept der *suid*-Rechte möglich, Zugriffsrechte temporär für den Kontext der Ausführung von Operationen zu vergeben (siehe Seite 636).

Rechteprüfung

Zugriffskontrolle

Die Abwehr von unautorisierten Zugriffen erfordert eine Kontrolle der Zugriffe. Mit der Sicherheitsgrundfunktion der Rechteprüfung ist festzulegen, wann, d.h. bei welchen Aktionen, eine Überprüfung stattfinden soll und welche Kontrollen jeweils durchzuführen sind. Gemäß des Vollständigkeitsprinzips sollte jeder Zugriff kontrolliert werden. Davon weicht man jedoch aus Aufwandsgründen häufig ab. Wie bereits angesprochen, wird in herkömmlichen Betriebssystemen nur beim Öffnen einer Datei die entspre-

chende Berechtigung überprüft. Nachfolgende Zugriffe auf die Datei müssen nur noch einen Konformitäts-Test durchlaufen (z.B. nur Lese-Zugriff auf eine zum Lesen geöffnete Datei). Eine erneute Berechtigungsüberprüfung findet nicht statt. Konformitätsprüfungen sind Beispiele für Kontrollmaßnahmen, die mit der Rechteprüfung festzulegen sind. Sie basieren darauf, dass aufgrund einer Berechtigungsprüfung ein qualifizierter Zugriffsausweis ausgestellt wurde, dessen Besitz zur Durchführung von Zugriffen gemäß der Qualifikation des Ausweises berechtigt. Beispiele für solche Ausweise sind das File Handle unter NFS, der File-Descriptor in Unix Systemen oder der Object-Handle in Windows-Betriebssystemen.

Neben den durchzuführenden Kontrollen ist mit der Rechteprüfung auch festzulegen, unter welchen Randbedingungen es Ausnahmen von der Rechteprüfung geben soll, und welche Aktionen bei unautorisierten Zugriffsversuchen durchzuführen sind. In herkömmlichen Betriebssystemen werden häufig Prozesse, die im Systemmodus ausgeführt werden, als besonders vertrauenswürdig betrachtet, so dass diese von der Rechtekontrolle ausgenommen sind. Ausnahmen von Rechteüberprüfungen sind problematisch und sollten nur sehr restriktiv und beherrschbar eingesetzt werden. Beispiele von Aktionen, die man als Reaktion auf einen unautorisierten Zugriffsversuch durchführen kann, sind das Protokollieren des Zugriffsversuchs in einem Logbuch, die Ausgabe permission denied für den zugreifenden Benutzer oder auch das Auslösen eines Alarms auf einer Operator-Konsole.

Ausnahme

Beweissicherung

Zur Abwehr von Angriffen, die versuchen, durchgeführte Aktionen im Nachhinein abzustreiten, ist eine Beweissicherung notwendig. Diese wird auch zur nachträglichen Analyse von Systemen benötigt, um Angriffe aufzudecken, die darauf abzielen, Rechte zu missbrauchen. Mit der Grundfunktion der Beweissicherung ist festzulegen, welche Ereignisse zu protokollieren und welche Informationen dabei jeweils zu erfassen sind. Die wichtigsten festzuhaltenden Informationen betreffen die Identität des Subjekts, das das protokolierte Ereignis auslöst, die Angabe der Objekte und Operationen, die von dem Ereignis betroffen sind und den Zeitpunkt, zu dem das Ereignis ausgelöst wurde. Mit der Beweissicherung ist darüber hinaus festzulegen, welche Subjekte unter welchen Randbedingungen auf die protokollierten Informationen zugreifen dürfen und nach welchen Kriterien die Daten ausgewertet werden sollen. Zugriffe auf protokolierte Daten sind restriktiv zu gewähren, um eine nachträgliche unautorisierte Modifikation zu verhindern.

Protokollierung

Eine spezielle Ausprägung der Beweissicherung hat sich in den letzten Jahren unter dem Begriff Computer-Forensik etabliert. In Abschnitt 1.4

Forensik

wurden bereits Werkzeuge und Vorgehensweisen vorgestellt, die hierfür wichtig sind.

Wiederaufbereitung

gemeinsame
Betriebsmittel

Maßnahmen zur Wiederaufbereitung von gemeinsam, aber exklusiv benutzbaren Betriebsmitteln sind notwendig, um Angriffe auf die Informationsvertraulichkeit abzuwehren. Diese Grundfunktion legt fest, für welche Betriebsmittel eine Wiederaufbereitung durchzuführen ist und wann diese Aufbereitung zu erfolgen hat. Beispiele für solche Betriebsmittel sind der Prozessor mit seinen Registersätzen oder Seitenkacheln des Arbeitsspeichers. Die gemeinsam genutzten Prozessorregister sind beim Scheduling von Prozessen zu bereinigen, falls der Prozessor einem anderen Prozess zugeteilt wird. Seitenkacheln des Arbeitsspeichers werden im Rahmen der virtuellen Speicherverwaltung nacheinander von unterschiedlichen Prozessen genutzt. Die Speicherverwaltung kann diese Bereiche direkt beim Auslagern einer Seite aus dem Hauptspeicher auf den Hintergrundspeicher bereinigen oder diese Bereinigung so lange aufschieben, bis eine neue Seite in die Kachel eingelagert wird. Unterbleibt eine Wiederaufbereitung gemeinsam genutzter Objekte, so ist der direkte Zugriff auf gespeicherte Informationen unter Umgehung der Rechteprüfung möglich. Zu beachten ist, dass aus Effizienzgründen in heutigen Betriebssystemen auf eine derartige konsequente Wiederaufbereitung häufig verzichtet wird. Zumindest für sicherheitskritische Bereiche erfolgt das aber beispielsweise unter Windows durch die virtuelle Speicherverwaltung dieser Systeme. Speicherseiten, die als sicherheitskritisch eingestuft werden (z.B. Seiten, die zu den Laufzeit-Kellern (Stack) der Prozesse gehören), werden in einer speziellen Liste verwaltet und vor ihrer Wiederverwendung mit Nullen überschrieben.

Verfügbarkeit

Gewährleistung der Funktionalität

Beispiele

Die Gewährleistung der Verfügbarkeit von Systemdiensten und damit die Abwehr von Denial-of-Service Angriffen ist für viele Anwendungsbereiche sicherheitsrelevant. Diese Grundfunktion legt fest, welche Funktionalität welcher Systemkomponenten mit welcher Priorität gewährleistet werden muss und unter welchen Randbedingungen ganz oder teilweise auf Funktionalität verzichtet werden kann. Zugriffs- und Zugangskontrollen sind Beispiele von Systemdiensten, deren Funktionalität stets gewährleistet werden sollte, während die Funktionalität von Oberflächenanimationen eine niedrige Priorität erhalten kann, da die Aufrechterhaltung dieser Funktionalität für die Einhaltung der Sicherheitsanforderungen unerheblich ist.

4.8 Realisierung der Grundfunktionen

Zur Realisierung der im vorherigen Abschnitt eingeführten Sicherheitsgrundfunktionen sind Sicherheitsmechanismen und -verfahren einzusetzen. In diesem Abschnitt werden allgemeine Richtlinien für den Einsatz von Mechanismen formuliert. Bei der Auswahl eines Realisierungskonzepts, -verfahrens oder eines Protokolls ist zu prüfen, ob mit diesem die Umsetzung der im Folgenden angegebenen, sicherheitsrelevanten Aspekte möglich ist. Dedizierte Konzepte und Verfahren erläutern wir in nachfolgenden Kapiteln.

Richtlinien

Aspekte, die beim Einsatz von Mechanismen zur Realisierung der Grundfunktion der Identifikation und Authentifikation zu beachten sind, betreffen die Garantien der Eindeutigkeit der Identität von Subjekten und Objekten, die Fälschungssicherheit der Identität sowie den Aufwand, der zur unautorisierten Erlangung einer Identität notwendig ist. Hohe Sicherheitsanforderungen verlangen den Einsatz von Mechanismen, die eine hohe Fälschungssicherheit garantieren und die nur unter hohem Aufwand zu überwinden sind.

Authentifikation

Bei der Realisierung der Rechteverwaltung ist zu untersuchen, ob mit den eingesetzten Mechanismen die Sicherheitsanforderungen vollständig erfasst und insbesondere die Zugriffsrechte in der erforderlichen Weise vergeben werden können. Verwendet man beispielsweise ein Gruppenkonzept zur Vergabe von Zugriffsrechten an Gruppen von Subjekten und besitzt ein Subjekt zu jedem Zeitpunkt alle Rechte aller Gruppen, in denen es Mitglied ist, so lassen sich damit Sicherheitsanforderungen nicht erfüllen, die eine Rechtewahrnehmung nur dann gestatten, wenn das ausführende Subjekt in genau der Gruppe aktiv ist, an die das Recht vergeben wurde. Neben der Vollständigkeit ist auch der Aspekt der Widerspruchsfreiheit der Rechtevergabe von Bedeutung. Falls mit den eingesetzten Mechanismen eine widersprüchliche Rechtevergabe möglich ist, so muss festlegbar sein, wie ein solcher Widerspruch aufzulösen ist, indem zum Beispiel eine negative Rechtevergabe, d.h. eine Rechteverweigerung, stets eine höhere Priorität als eine Rechteerteilung besitzt.

Rechteverwaltung

Aspekte, die bei der Realisierung der Grundfunktion der Rechteüberprüfung von Bedeutung sind, betreffen den Zeitpunkt der Rechteprüfung, also die Aktualität der Kontrollentscheidung, sowie die Integrität und Verfügbarkeit der zur Entscheidungsfindung benötigten Daten. Die Verfügbarkeit der Entscheidungsdaten kann beispielsweise in verteilten Umgebungen durch Netzwerkausfälle oder Netzüberlastung infrage gestellt sein, so dass Replikationstechniken einzusetzen sind. Die Integrität von Entscheidungsdaten ist bei deren Übertragung über ein Transportmedium bedroht, so dass wirksame Mechanismen zur Erkennung von unautorisierten Manipulationen einzusetzen sind.

Rechteprüfung

Beweissicherung

Fragestellungen im Zusammenhang mit der Realisierung der Grundfunktion der Beweissicherung betreffen die Korrektheit der protokollierten Daten und die Vollständigkeit der Beweissicherung. Es sind Mechanismen einzusetzen, die nicht umgangen werden können und die korrekt arbeiten, so dass ein Angreifer keine Aufzeichnung fehlerhafter Daten veranlassen kann. Im Hinblick auf die Gewährleistung der Vollständigkeit der Protokollierung sind unter anderem Maßnahmen erforderlich, die verhindern, dass beim Überlauf der Protokolldatei Ereignisse entgegen der Sicherheitsanforderungen nicht mehr protokolliert werden.

Verfügbarkeit

Mechanismen zur Realisierung der Verfügbarkeit von Komponenten sowie der Gewährleistung der Systemfunktionalität betreffen Erkennungs- und Behebungs- sowie Fehlervermeidungs- oder Fehlerverhinderungsmaßnahmen. Mit den gewählten Fehlererkennungsmaßnahmen ist zu gewährleisten, dass alle Möglichkeiten, durch die der entsprechende Fehler auftreten kann, behandelt werden. Maßnahmen zur Fehlerbehebung wie beispielsweise der Abbruch von Teilberechnungen dürfen zu keiner Beeinträchtigung derjenigen Komponenten führen, deren korrekte Funktionalität mit höchster Priorität gefordert wird. Für Fehlervermeidungs- und Fehlerverhinderungsmaßnahmen ist deren Einfluss auf die Funktionalität des Systems zu untersuchen. Durch die entsprechenden Maßnahmen dürfen sicherheitskritische Operationen in ihrer Funktionalität nicht beeinflusst werden.

Grundschutz**Grundschutz**

Wesentlich konkretere Bausteine zur Umsetzung einer Sicherheitsstrategie werden in den IT-Grundschutz-Katalogen des BSI definiert. Jeder dieser Bausteine entspricht einem typischen Bereich. Beispiele hierfür sind Stand-alone Systeme (u.a. Laptops, Unix-, Windows-PC, Telearbeit), vernetzte Infrastrukturen (u.a. Client-Server-Rechner, Firewalls, Verkabelung), bauliche Einrichtungen, Kommunikationskomponenten (u.a. Fax, ISDN) und Vorgehensmodelle (u.a. Viren-Schutzkonzept, Notfallversorgung, Datensicherungskonzept). Für jeden Baustein werden typische Bedrohungen und Eintrittswahrscheinlichkeiten beschrieben. Daraus werden die erforderlichen Abwehrmaßnahmen aus den Bereichen Infrastruktur, Personal, Organisation, Hard- und Software, Kommunikation und Notfallvorsorge abgeleitet. Es handelt sich dabei um Standardtechniken.

Schichten

Da die Abbildung der individuellen Bedürfnisse auf diese vorgefertigten Bausteine eine komplexe Aufgabe ist, gruppieren das Handbuch verschiedene Sicherheitsaspekte zu Schichten und benennt die für die jeweiligen Schichten notwendigen Bausteine.

Schicht 1 erfasst übergeordnete Aspekte. Typische Bausteine dieser Schicht sind das IT-Sicherheitsmanagement, die Organisation, das Datensicherungskonzept und das Virenschutzkonzept. Typische Bausteine der Schicht 2, der Infrastruktur, sind die Module Gebäude, Räume, Schutzschränke und häuslicher Arbeitsplatz. Schicht 3 beschreibt Komponenten, wobei typische Bausteine Stand-alone Systeme wie z.B. Unix-System, Laptops, Netze und TK-Anlagen sind. Schicht 4 beschreibt die Netze. Typische Bausteine dieser Schicht sind heterogene Netze, Netz- und Systemmanagement und Firewalls. Schicht 5 umfasst schließlich Anwendungen mit typischen Bausteinen wie E-Mail, WWW-Server, Faxserver und Datenbanken.

Abschließend wird mit dem Security Development Lifecycle (SDL) der Firma Microsoft ein methodischer Ansatz vorgestellt, der eine Konkretisierung des allgemein beschriebenen Vorgehensmodells darstellt, aber auf die Praxis der Entwicklung sicherer Software zugeschnitten ist.

4.9 Security Development Lifecycle (SDL)

Als Konsequenz der zunehmenden Sicherheitsprobleme, die durch fehlerhafte Programmierung auftreten, wurden pragmatische Prozesse entwickelt, um die Qualität der Software zu erhöhen. Bei der Firma Microsoft hat dies seinen Niederschlag in dem SDL-Prozess (Security Development Lifecycle) gefunden, bei dem drei Paradigmen der sicheren Programmierung berücksichtigt werden. Diese sind die Grundsätze des *Secure by Design*, *Secure by Default* (sicher aufgrund von Standardeinstellungen) und *Secure in Deployment* (Sicherheit auch bei der Auslieferung und Bereitstellung). Mit dem Paradigma des *Secure by Design* wird angestrebt, dass Software über den gesamten Entwicklungszyklus hinweg so entworfen wird, dass sie in der Lage ist, sich selbst und die verarbeiteten Informationen vor Angriffen zu schützen bzw. um in zumindest begrenztem Umfang auch Angriffen Stand zu halten (angriffstolerantes System). Das Paradigma *Secure by Default* greift die Prinzipien von Saltzer und Schroeder auf und fordert u.a. eine minimale Berechtigungsvergabe und die Einhaltung des Erlaubnisprinzips. *Secure by Deployment* fordert die Bereitstellung von Anleitungen und Werkzeugen, die den Anwender bei dem sicheren Einsatz der Software unterstützen. Die Umsetzung der skizzierten Paradigmen hat zu einer Veränderung des Software-Entwicklungs-Zyklus geführt, indem Maßnahmen zur Umsetzung der Paradigmen in den Entwicklungs-Prozess bei Microsoft integriert wurden. Abbildung 4.9 veranschaulicht die Phasen des Entwicklungsprozesses mit den integrierten Sicherheitsmaßnahmen.

SDL

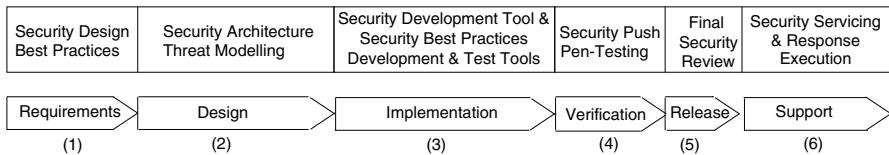


Abbildung 4.9: Sicherheit integriert in den Software-Entwicklungs-Prozess

4.9.1 Die Entwicklungsphasen

Phasen

In der Anforderungsphase (1) erfolgt die Identifikation der Sicherheitsanforderungen und Schutzziele der zu erstellenden Software und deren Schnittstelle zu und sichere Interoperation mit vorhandenen Softwareprodukten.

In der Entwurfsphase (2) werden im Sinne einer Trusted Computing Base diejenigen Komponenten identifiziert, die für die Sicherheit grundlegend sind. Zusätzlich wird die Software-Architektur definiert (z.B. Schichtenarchitektur mit Verfeinerungsschritten). In dieser Phase erfolgt auch eine Bedrohungs- und Risikomodellierung gemäß der STRIDE und DREAD-Methodik (s.u.).

In der Implementierungsphase (3) werden Tools und Test-Methoden zur Vermeidung von Sicherheitsschwachstellen eingesetzt, sowie Code-Reviews durchgeführt. Beispiele für verwendete Tools sind die von Microsoft entwickelten Werkzeuge PREfix, PREfast und Fuzzingtools zum Testen von APIs mittels gezielt konstruierter, ungültiger Eingaben.

In der Überprüfungsphase (4) (Verification) liegt die Software mit vollem Funktionsumfang vor und es beginnen die Beta-Tests mit Benutzern. Gleichzeitig wird eine systematische Suche nach Sicherheitsmängeln (genannt Security Push) mittels erneuter Code-Reviews und Sicherheitstests (Pen-Testing) durchgeführt.

In der Veröffentlichungsphase (5) (Release) durchläuft die Software eine abschließende Sicherheitsüberprüfung (FSR, Final Security Review), um zu klären, ob sie reif ist, an den Kunden ausgeliefert zu werden. In dieser Phase werden u.a. auch Penetrationstests durchgeführt.

Um während der Lebenszeit (6) der ausgelieferten Produkte in der Lage zu sein, auf neu entdeckte Fehler und Schwachstellen schnell und umfassend reagieren zu können, muss ein Prozess definiert sein, der festlegt, wie Berichte über Schwachstellen auszuwerten sind und wie darauf reagiert wird, zum Beispiel durch eine Veröffentlichung der Schwachstelle, durch eine Aktualisierung der Tools zur Codeanalyse oder durch einen erneuten Code-Review.

4.9.2 Bedrohungs- und Risikoanalyse

Die systematische Durchführung von Bedrohungs- und Risikoanalysen mittels der STRIDE und DREAD-Methoden ist ein integraler Bestandteil der Entwurfsphase des SDL-Prozesses.

STRIDE-Bedrohungsanalyse

STRIDE ist ein Acronym und steht für Spoofing Identity, Tampering with Data, Repudiation, Information Disclosure, Denial of Service und Elevation of Privilege. Das bedeutet, dass mit dem STRIDE Ansatz versucht wird, Angriffe, die diesen Angriffsklassen zuzurechnen sind, zu identifizieren, zu bewerten und geeignete Maßnahmen zur Abwehr der Angriffe festzulegen. Bei der Angriffsklasse, die auf die Veränderung von Daten abzielt (Tampering with Data), fokussiert der Ansatz vordringlich auf Angriffe zur Veränderung persistent gespeicherter Daten, wie Passwörter oder Log-Dateien, sowie auf Angriffe auf die Integrität von Datenpaketen, die über Netze übertragen werden. Repudiation-Angriffe umfassen beispielsweise nicht durchgeführte Bestellungen. Eine unberechtigte Informationsweitergabe (Information Disclosure) ist beispielsweise ein unautorisierte Datenbank-Zugriff oder aber auch der Zugriff und die Weitergabe von lokalen Daten. Zu den Angriffen mit dem Ziel, die eigenen Berechtigungen zu erhöhen (Elevation of Privileges), zählen u.a. Buffer-Overflow- oder auch SQL-Injection-Angriffe.

Für jede Angriffsklasse legt die STRIDE-Methode eine Menge von Best-Practice Abwehrmaßnahmen fest, wie beispielsweise starke Mehrfaktor-Authentisierung (vgl. Kapitel 10), Verschlüsselung, digitale Signaturen und Hashfunktionen (vgl. Kapitel 7 und 8).

Einsatz von STRIDE

In der Phase der Bedrohungsanalyse wird eine zu analysierende Software bzw. ein System unter Bezugnahme auf die sechs Angriffsklassen (S,T, R, I, D und E) auf Sicherheitsschwachstellen untersucht und es werden in diesem Schritt auch bereits diejenigen Maßnahmen identifiziert, die zum Schließen der Lücke geeignet sind. Das zu analysierende System wird dazu in seine relevanten Komponenten aufgeteilt und für jede dieser Komponenten wird eine individuelle Bedrohungsanalyse durchgeführt. Zur Modellierung der Systeme und Komponenten dienen grafische Darstellungen (z.B. unter Nutzung der UML), mit denen mögliche Datenflüsse (u.a. Netzverbindungen, Named Pipes), Datenspeicher (u.a. Dateien, Datenbanken), Prozesse und Interaktoren (u.a. Benutzer, Server, Web-Dienste) beschrieben werden. Für jedes Element der grafischen Darstellung (Prozesse, Datenspeicher, Datenflüsse und Interaktoren) sind diejenigen Bedrohungen der STRIDE-Klassen

zu identifizieren, die auf das Element zutreffen können (z.B. für Datenflüsse sind es Bedrohungen der Klassen Tampering, Disclosure oder auch Denial of Service).

Beispiel 4.7 (STRIDE-Analyse einer Web-Anwendung)

Web-Anwendung

Abbildung 4.10 zeigt vergrößert die Anwendung der STRIDE-Methode auf eine allgemeine Web-Anwendung.

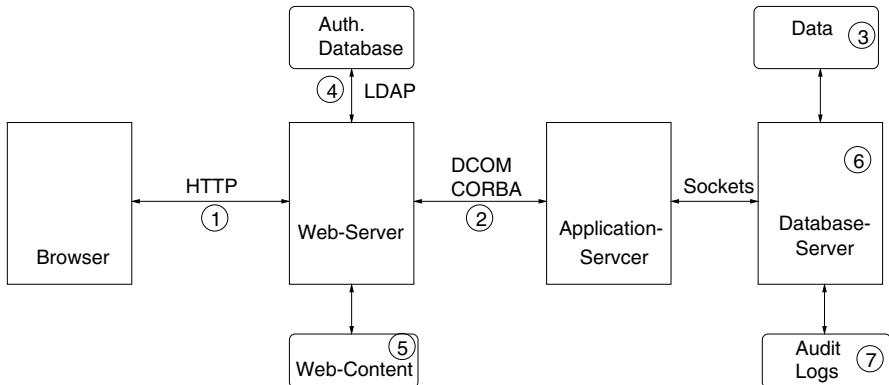


Abbildung 4.10: STRIDE-Analyse einer Web-Anwendung

Die Datenflüsse, die in dem Bild mit 1 und 2 markiert sind, sind von Identitätsdiebstahl (Spoofing), Datenmanipulation (Tampering), aber auch unberechtigter Informationsweitergabe (Information disclosure) bedroht. Der Datenspeicher, der die Datenbank des Datenbankservers repräsentiert (3) ist Bedrohungen in Bezug auf die Integrität der Daten (Tampering) und deren Vertraulichkeit (Information disclosure) ausgesetzt, während der Datenfluss, der die Verbindung zwischen dem Web-Server und dem LDAP-Verzeichnis mit den Authentisierungsdaten beschreibt (4), anfällig ist für Bedrohungen in Bezug auf Identitätsdiebstahl (Spoofing), den Verlust der Informationsvertraulichkeit (Information disclosure) und einer unberechtigten Rechte-Aneignung (Elevation of Privileges). Die Datenbasis, die den Web-Content speichert (5), ist Bedrohungen in Bezug auf die Datenintegrität (Tampering) ausgesetzt, während die Log-Datenbank (7) zusätzlich auch anfällig ist für Repudiation-Angriffe. Für den Datenbankserver (6) sind Spoofing-Angriffe, aber auch Angriffe auf die Vertraulichkeit (Information disclosure) und Angriffe zur Rechte-Aneignung zu erwarten.

Eine ausführlichere Beschreibung der Vorgehensweise zur Erhebung des Schutzbedarfs und zur Durchführung einer STRIDE-basierten Bedrohungs- und Risikoanalyse anhand des Beispiels findet man Online⁸. Die dort an-

⁸ <http://msdn.microsoft.com/de-de/library/cc405496.aspx>

gewandte Vorgehensweise entspricht dem in diesem Kapitel eingeführten allgemeinen Vorgehen. So wird zur systematischen Durchführung einer Bedrohungsanalyse ebenfalls die Nutzung der Methode der Bedrohungsbäume verwendet.



Risikoanalyse

Die qualifizierte Risikobewertung inklusive der Priorisierung der identifizierten Bedrohungen erfolgt anschließend mittels der so genannten DREAD-Methode. DREAD steht für die Kriterien **D**amage, **R**eproducibility, **E**xploitability, **A**ffected und **D**iscoverability. Mit dem Kriterium des Schadenspotentials (**D**amage) versucht man den Schaden zu qualifizieren, der durch das Ausnutzen der Lücke entstehen könnte, die durch die Bedrohungsanalyse identifiziert wurde. Mit Reproduzierbarkeit bewertet man den Schwierigkeitsgrad zur Reproduktion des Angriffs und die Ausnutzbarkeit bewertet die Schwierigkeit, einen entsprechenden Angriff durchzuführen. Das Kriterium **Affected** gibt eine qualitative Einschätzung wie groß der Kreis der potentiell betroffenen Benutzer ist und die Auffindbarkeit qualifiziert den Schwierigkeitsgrad, mit dem die Sicherheitslücke aufgedeckt werden kann.

DREAD

Die Methode schlägt für jedes Kriterium eine Qualifizierung nach hoch (3), mittel (2) und gering (1) vor und erstellt eine Risikobewertung als Summe der Einzelbewertungen. Das Risiko einer Bedrohungen wird als hoch eingestuft, wenn bei der Risikoanalyse ein Wert zwischen 12 und 15 ermittelt wurde, ein Wert zwischen 8 und 11 wird als ein mittleres Risiko und ein Wert zwischen 5 und 7 wird als geringes Risiko angesehen. In Tabelle 4.3 ist ein Beispiel für eine Risikobewertung angegeben.

Mit dem SDL-Prozess und den entwickelten Methoden zur Bedrohungs- und Risikoanalyse sind wichtige Schritte zur Umsetzung eines durchgehenden Security Engineering-Prozesses für die industrielle Praxis der Software-Entwicklung gemacht worden.

Bewertung	Hoch (3)	Mittel (2)	Gering (1)
(D) Schadenspotenzial	Der Angreifer kann das Sicherheitssystem untergraben und volle Berechtigungen erlangen.	Verbreitung sicherheitsrelevanter Informationen ist möglich	Verbreitung unbedeutender Informationen ist möglich
(R) Reproduzierbarkeit	Der Angriff kann jederzeit reproduziert werden.	Der Angriff kann reproduziert werden, jedoch nur in einem Zeitfenster.	Der Angriff ist auch mit Kenntnis der Sicherheitslücke sehr schwer zu reproduzieren.
(E) Ausnutzbarkeit	Ein Programmieranfänger kann den Angriff in kurzer Zeit durchführen.	Ein erfahrener Programmierer kann den Angriff durchführen.	Nur eine erfahrene Person mit eingehendem Fachwissen kann den Angriff ausführen.
(A) Betroffene Benutzer	Alle Benutzer sind betroffen; Standardkonfiguration, Schlüsselkunden	Einige Benutzer sind betroffen; keine Standardkonfiguration	Ein sehr geringer Prozentsatz von Benutzern ist betroffen; unbekannte Funktion, betrifft anonyme Benutzer
(D) Auffindbarkeit	Der Angriff wird über öffentlich zugängliche Medien erklärt. Die Sicherheitslücke findet sich in einer viel verwendeten Funktion und ist leicht wahrnehmbar.	Die Sicherheitslücke befindet sich in einem selten verwendeten Teil des Produkts. Die bösartige Verwendbarkeit ist nur mit einigem Aufwand erkennbar.	Der Fehler ist unbekannt und es ist unwahrscheinlich, dass Benutzer das Schadenspotenzial erkennen.

Tabelle 4.3: DREAD-Analyse

5 Bewertungskriterien

Zur methodischen Bewertung der Sicherheit informationstechnischer Systeme wurden nationale und internationale Kriterienkataloge entwickelt. Kriterienkataloge stellen ein Bewertungsschema zur Verfügung, das es erlaubt, die Sicherheit unterschiedlicher Systeme, die aber eine ähnliche Funktionalität besitzen, zu vergleichen. Die Kriterien definieren damit eine Art Metrik zur Bewertung der Sicherheit eines Systems und dienen gleichzeitig als Leitlinien zur Entwicklung und Konstruktion sicherer Systeme. Die Evaluierung und Bewertung erfolgt durch unabhängige Evaluierungsstellen, die Zertifikate¹ ausstellen, die dem Benutzer eines zertifizierten Produktes ein evaluiertes Maß an Sicherheit zusichern (engl. *assurance*).

Die ersten derartigen Kriterien waren die Trusted Computer System Evaluation Criteria des amerikanischen Verteidigungsministeriums, die wir in Abschnitt 5.1 näher betrachten. Anschließend gehen wir in Abschnitt 5.2 auf die deutschen IT-Kriterien und in Abschnitt 5.3 auf die harmonisierten, europäischen ITSEC-Kriterien ein. Durch die stetig ansteigende Globalisierung wächst der Bedarf nach weltweiten Evaluationskriterien und nach Evaluierungszertifikaten, die international anerkannt werden. Mit der Entwicklung der Common Criteria wurde diesen Anforderungen Rechnung getragen. Abschnitt 5.4 fasst die wesentlichen Aspekte dieses internationalen Kriterienkataloges zusammen.

5.1 TCSEC-Kriterien

Die ältesten Kriterien zur Bewertung der Sicherheit von IT-Systemen sind die Trusted Computer System Evaluation Criteria (TCSEC) [53], die Anfang 1980 am US National Computer Security Center entwickelt und 1985 um die Trusted Network Interpretation [128] zur Einbeziehung vernetzter Systeme ergänzt wurden. Die TCSEC-Kriterien sind bekannt als das Orange Book, das nach der Farbe des Katalogumschlages benannt wurde. Auch wenn die TCSEC-Kriterien heute für die Evaluierungspraxis keine Rolle mehr spielen, so sind doch viele Konzepte und Ansätze heutiger, noch im

¹ Ein amtlich bestätigter Nachweis.

Einsatz befindlicher Kriterienkataloge auf die TCSEC und die im Folgenden vorgestellten IT- bzw. ITSEC-Kriterien zurück zu führen.

5.1.1 Sicherheitsstufen

Sicherheitsstufen

Die TCSEC legen mit den Stufen D, C, B und A sowie deren weiteren Untergliederungen sieben Hierarchiestufen zur Klassifikation der Sicherheit eines Systems fest. Die niedrigste Stufe D ist nicht weiter untergliedert und gilt für Systeme, die keinen bzw. nur minimalen Schutz gewährleisten.

Stufe C: Benutzerbestimbarer Schutz (discretionary)

Stufe C1

Eine Bewertung nach Stufe C erfordert, dass Zugriffsrechte durch Benutzer individuell vergeben werden können. Eine Einstufung gemäß C1 besagt, dass die Rechtesfestlegungen nur grobgranular erfolgen können. Die C1-Einstufung erfordert Maßnahmen zur Kontrolle der Identität von Benutzern und zum Schutz dieser Kontrollfunktionen. An die Dokumentation und durchzuführenden Systemtests werden nur geringe Anforderungen gestellt, so dass lediglich sicherzustellen ist, dass keine offensichtlichen Möglichkeiten zur Umgehung der Kontrollen existieren.

Stufe C2

Eine Einstufung nach C2 besagt, dass Benutzern differenzierte Möglichkeiten zur Rechtesfestlegung zur Verfügung stehen, so dass Zugriffsrechte auch an einzelne Benutzer vergeben oder auch Zugriffsverbote festgelegt werden können. Zusätzlich zu den Maßnahmen der C1-Einstufung werden Dienste gefordert, so dass die Aktionen einzelner Benutzer abrechenbar sind und Zugriffe auf zu schützende Daten überwacht werden (Auditing). Die durchzuführenden Tests müssen auch den Schutz der Audit- und Authentifikationsdaten umfassen.

Stufe B: Systembestimmter Schutz (mandatory)

Stufe B1

Mit dem Übergang zu den B-Klassen werden Sicherheitsstufen und Sicherheitsklassifikationen (engl. *labeling*) für Subjekte und Objekte eingeführt. Systeme, die nach Stufe B1 bewertet werden, umfassen die Anforderungen der Stufe C2 und gewährleisten zusätzlich systembestimmte Zugriffsrstruktionen. Dazu ist eine geordnete Menge von Sensitivitätsklassen festzulegen. Häufig wird die Menge $SC = \{ \text{streng geheim, geheim, vertraulich, öffentlich} \}$ mit der linearen Ordnung streng geheim > geheim > vertraulich > öffentlich verwendet. Weiterhin ist jedem Objekt o eine Sicherheitseinstufung (engl. *classification*) $sc(o)$ und jedem Subjekt s eine Sensitivitätsklasse (engl. *clearance*) $sc(s)$ zuzuordnen. Zugriffe auf Objekte werden durch systemglobale Regeln beschränkt. Ein bekanntes Beispiel hierfür ist die Beschränkung von Lesezugriffen auf ein Objekt o derart, dass ein Lesen höchstens dann zulässig

ist, wenn die Clearance des zugreifenden Subjekts s größer oder gleich der Sicherheitsklassifikation von o ist, also $sc(s) \geq sc(o)$. Da dadurch gewährleistet ist, dass beim Lese-Zugriff keine hoch eingestufte Information zu einem niedrig eingestuften Subjekt fließt, nennt man die Beschränkung auch die „no read-up“ Regel (vgl. Abschnitt 6.2.4).

Systembestimmte (mandatorische) Beschränkungen dominieren die benutzerbestimmbaren Rechtevergaben. Das heißt, dass eine benutzerbestimmt vergebene Zugriffserlaubnis ungültig ist, wenn der Zugriff durch eine systemglobale Regel verboten ist. Eine B1-Einstufung erfordert weiterhin die informelle Beschreibung eines Sicherheitsmodells zusammen mit einer Erläuterung, warum die festgelegten Sicherheitsmaßnahmen ausreichend sind.

Eine Einstufung nach B2 erfordert ein formales Sicherheitsmodell sowie Maßnahmen zur Erkennung bzw. Behandlung verdeckter Kanäle und zur Einrichtung eines vertrauenswürdigen Pfades (engl. *trusted path*) beim Login. Ein vertrauenswürdiger Pfad etabliert eine direkte Kommunikation zwischen dem Benutzer und den vertrauenswürdigen Sicherheitskomponenten TCB (engl. *trusted computing base*). Ein solcher Pfad wird normalerweise über die Festlegung einer spezifischen, vordefinierten Kontrollsequenz (z.B. einer Tastenkombination wie Control-Alt-Delete unter Windows) initiiert.

Stufe B2

Die Stufe B3 stellt insbesondere zusätzliche Anforderungen an die Trusted Computing Base (TCB), also an die Menge der sicherheitsrelevanten Komponenten und Dienste. Die TCB muss klein, getestet und gegen unbefugte Zugriffe geschützt sein und alle sicherheitsrelevanten Aktionen sind zu überwachen.

Stufe B3

Die Stufe A erfordert keine funktionalen Erweiterungen der Stufe B3, jedoch wird ein formaler Nachweis der Einhaltung spezifizierter Sicherheitseigenschaften gefordert. Nur sehr wenige Systeme wurden nach A evaluiert. Ein Beispiel ist die Netzwerkkomponente und die zugehörige Kommunikationssoftware MLS LAN Version 2.1 der Firma Boeing.

Stufe A

Tabelle 5.1 fasst die funktionalen Anforderungen der unterschiedlichen Klassifikationsstufen des Orange Books noch einmal zusammen.

5.1.2 Kritik am Orange Book

Einer der Hauptkritikpunkte, die an den TCSEC geübt werden, betrifft die einseitige Ausrichtung auf zentrale Betriebssysteme, so dass offene Kommunikationssysteme oder dedizierte Anwendungen nur unzureichend erfasst werden. Weiterhin betonen die Orange Book-Kriterien sehr stark Vertraulichkeitseigenschaften von Systemen. Da diese jedoch in der festgelegten Form meist nur für militärische Einsatzumgebungen wichtig sind, werden

zentrale BS

Vertraulichkeit

C1	C2	B1	B2	B3		
					Subjekt Identifikation und Authentifikation	Identifikation und Authentifikation
					Trusted Path	
					Benutzerbestimmbare Kontrolle	
					Systembestimmte Kontrolle	
					Attribute für Subjekte u. Objekte	
					Integrität der Attribute	
					Anzeigen der Attributwerte	Rechteverwaltung
					Export über Single-level Kanal	
					Export über Multi-level Kanal	
					Export der Attribut-Daten	
					Benutzer-lesbare Ausgabe	
					Zugriffskontrollen	Rechte-Kontrolle
					Abrechenbarkeit	
					Dokumentation und Auswertung von Audit-Daten	Audit
					Monitoring sicherheitskritischer Ereignisse	
					Wiederaufbereitung	Wiederaufbereitung

keine Anforderungen neue oder erweiterte Anforderungen keine zusätzlichen Anforderungen

Tabelle 5.1: Funktionale Anforderungen des Orange Books

Anforderungen des kommerziellen Bereichs nicht ausreichend berücksichtigt. Als weiterer Kritikpunkt gilt die Vernachlässigung von benutzerspezifischen Sicherheitsinteressen zugunsten der Interessen von Herstellern und Betreibern von IT-Systemen. Im Vordergrund der sicherheitsrelevanten Maßnahmen stehen Funktionen und Dienste zur Protokollierung von Daten, wohingegen für den Benutzer auch Dienste zur Reduktion der Informationsansammlung von großer Bedeutung sind. Ein schwerwiegender Kritikpunkt betrifft die fehlende Trennung zwischen der Sicherheitsfunktionalität und der Qualität, mit der die Funktionalität erbracht wird. Das heißt, dass die TCSEC-Kriterien nur bewerten, dass eine Funktionalität erbracht wird, jedoch nicht erfassen, mit welcher Wirksamkeit Bedrohungen durch die eingesetzten Mechanismen abgewehrt werden.

Heutige Systeme werden nicht mehr nach den TCSEC-Kriterien evaluiert, sondern man richtet sich in der Regel nach den internationalen Common Criteria (siehe Abschnitt 5.4).

5.2 IT-Kriterien

Einige der erwähnten Kritikpunkte am Orange Book wurden in den deutschen IT-Kriterien [92], wegen des grünen Umschlages auch als Grünbuch

Hersteller-orientiert

Funktionalität-orientiert

Grünbuch

bekannt, und in den europäischen ITSEC-Kriterien [93] (s.u.) aufgegriffen. Durch die Einführung von Funktionsklassen und Qualitätsstufen werden die Systemfunktionalität und die Qualität der eingesetzten Mechanismen getrennt bewertet. Die Funktionsklassen umfassen die Festlegungen der TCSEC-Kriterien.

5.2.1 Mechanismen

Zur Bewertung der eingesetzten Sicherheitsmechanismen ist eine Skala definiert, die die Stärke der Mechanismen klassifiziert. Die Analyse der Mechanismen hat zum Ziel, Schwächen aufzudecken, die auch durch eine korrekte Implementierung der Sicherheitsfunktionen, die diese Mechanismen verwenden, nicht behebbar sind. Diese Skala ist durch die Stufen ungeeignet, schwach, mittel, stark, sehr stark festgelegt. Ein ungeeigneter Mechanismus ist nicht wirksam, während ein schwacher zumindest zur Abwehr unabsichtlicher Verstöße gegen die Sicherheitsanforderungen geeignet ist. Er stellt jedoch kein ernsthaftes Hindernis gegen absichtliche Verstöße dar. Ein Mechanismus wird als mittelstark eingestuft, wenn er Schutz gegen absichtliche Verstöße bietet, dieser Schutz jedoch mit mittelgroßem Aufwand von Personen mit normalen Kenntnissen über das System überwunden werden kann. Ist ein Mechanismus nur mit großem Aufwand oder durch Zuhilfenahme aufwändiger Hilfsmittel zu überwinden, so bietet er einen starken Schutz gegen absichtliche Verstöße. Falls beim derzeitigen Stand der Technik der Mechanismus nur mit sehr großem Aufwand und mit sehr aufwändigen Hilfsmitteln zu überwinden ist, so schützt er sehr stark gegen absichtliche Sicherheitsverstöße.

Klassifikation

Beispiel 5.1 (Authentifikationsmechanismen)

Zur Bewertung der Stärke der Mechanismen, die in IT-Systemen zur Realisierung der Identifikations- und Authentifikationsgrundfunktion eingesetzt werden, wird die Wahrscheinlichkeit untersucht, mit der eine nicht korrekte Identifikation möglich ist. Ein schwacher Mechanismus ermöglicht eine nicht korrekte Identifikation mit einer Wahrscheinlichkeit kleiner als $\frac{1}{10^2}$, ein mittelstarker mit einer Wahrscheinlichkeit kleiner als $\frac{1}{10^4}$, ein starker mit einer Wahrscheinlichkeit kleiner als $\frac{1}{10^6}$ und ein sehr starker Mechanismus mit einer Wahrscheinlichkeit kleiner als $\frac{1}{10^8}$.



5.2.2 Funktionsklassen

Menge von
Grundfunktionen

Klassen F1 - F5

Klasse F6

Klasse F7

Klasse F8

Klasse F9

Klasse F10

Eine Funktionsklasse kann man als eine Zusammenfassung von Sicherheitsgrundfunktionen beschreiben, die die Sicherheitsanforderungen einer Klasse von IT-Systemen abdeckt.

Die IT-Kriterien legen 10 Funktionsklassen fest, wobei die Klassen F1 – F5 hierarchisch strukturiert und von den Klassen des Orange Books abgeleitet sind. D.h. F1 ist abgeleitet von C1 und F5 ist abgeleitet von B3/A1. Die Funktionsklasse F6 stellt besondere Anforderungen an die Integrität von Daten, wie sie u.a. in Datenbanken oder Software-Entwicklungsumgebungen erforderlich sind. Hierzu gehört die Einführung von Rollen und Objekttypen, die Festlegung eines Trusted Path beim Login sowie die Protokollierung sicherheitsrelevanter Zugriffe auf Daten.

Die Klasse F7 stellt hohe Anforderungen in Bezug auf die Verfügbarkeit, wie sie beispielsweise in Prozesssteuerungsrechnern benötigt werden. Gefordert werden Maßnahmen zur Fehlerbehandlung und -überbrückung, zum Zurücksetzen abgebrochener Berechnungen sowie zur Gewährleistung der Verklemmungsfreiheit.

Die Anforderungen der Klasse F8 beziehen sich auf die Datenintegrität während der Kommunikation, so dass eine Authentifikation bei jeder Interaktion oder der Einsatz fehlererkennender Codes sowie Maßnahmen zur Erkennung von Wiedereinspielungen gefordert werden.

Die Klasse F9 stellt Anforderungen an die Vertraulichkeit von Daten bei deren Übertragung und ist beispielsweise für spezifische Verschlüsselungsgeräte relevant.

Die Funktionsklasse F10 schließlich umfasst Vertraulichkeits- und Integritätsanforderungen vernetzter Systeme. An Grundfunktionen werden unter anderem die Authentifikation der Kommunikationspartner, eine Ende-zu-Ende Verschlüsselung sowie die Protokollierung relevanter Aktionen gefordert.

Ein System kann auch die Anforderungen mehrerer Funktionsklassen erfüllen.

5.2.3 Qualität

Für die Bewertung der Qualität eines Systems werden unterschiedliche Einzelaspekte untersucht und bewertet. Dazu gehören die Qualität der Darstellung der Sicherheitsanforderungen sowie deren Konsistenz und die Qualität der Spezifikation der zu evaluierenden Systemteile zusammen mit der Qualität der verwendeten Werkzeuge. Weitere Aspekte betreffen die

Stärke der verwendeten Sicherheitsmechanismen und deren Eignung zur Erfassung der gestellten Sicherheitsanforderungen, die Qualität des Herstellungsvorgangs einschließlich der verwendeten Werkzeuge, die Qualität der durchgeführten Tests und die Fähigkeiten des Personals. Auch die Dokumentation sowie deren Korrektheit und Verständlichkeit wird hinsichtlich ihrer Qualität untersucht und evaluiert.

Zur Bewertung der Qualität legen die IT-Kriterien acht Qualitätsstufen fest, die von unzureichend bis hin zu formal verifiziert reichen. Die Stufe Q0 stellt keine Qualitätsanforderungen und ist für Systeme reserviert, deren messbare Qualität nicht für eine der höheren Stufen ausreichend ist. Die Stufe Q1 charakterisiert eine geringe Qualitätsstufe. Das System ist nur grob auf Korrektheit geprüft, bietet keinen oder nur geringen Schutz vor Manipulationen und verwendet Mechanismen mit ausreichender Stärke. Die Stufe Q2 repräsentiert eine zufrieden stellende Qualität, da das System eingehend auf Korrektheit geprüft wird und einen zufrieden stellenden Schutz gegen Manipulationen bietet. Die Qualitätsstufe Q2 erfordert, dass mittelstarke Mechanismen zur Realisierung der Sicherheitsfunktionen verwendet werden. Die Stufen Q3 bis Q5 bezeichnen eine gute bis sehr gute Qualität, da bei der Erstellung des IT-Systems zur Analyse von dessen Korrektheit semiformale oder formale Techniken eingesetzt werden und ein Konsistenzabgleich zwischen Quellcode und Spezifikation erfolgt. Q3 – Q5 erfordern den Einsatz starker Mechanismen. Systeme der Qualitätsstufen Q6 und Q7 bieten schließlich einen ausgezeichneten, formal verifizierten Schutz mit Mechanismen, die im Fall von Q6 sehr stark bzw. im Fall von Q7 praktisch nicht überwindbar sind.

Hersteller können ihre Produkte anhand der Kriterien von unabhängigen, akkreditierten Evaluierungsstellen analysieren und bewerten lassen. Als Ergebnis der Evaluation wird ein Zertifikat ausgestellt, das zum einen die Funktionsklassen, die von dem evaluierten System erfüllt werden, und zum anderen eine Qualitätsstufe enthält.

5.3 ITSEC-Kriterien

Die europäischen ITSEC (Information Technology Security Evaluation Criteria) Kriterien wurden 1991 als harmonisierte Kriterien der Länder Großbritannien, Frankreich, Niederlande und Deutschland erarbeitet. Die ITSEC-Kriterien legen analog zu den deutschen IT-Kriterien Funktionsklassen mit Sicherheitsanforderungen für spezifische Anwendungsklassen fest. Die Bewertung der Qualität der erbrachten Funktionen wird jedoch im Vergleich zu den Qualitätsstufen der IT-Kriterien noch weiter aufgeteilt und zwar (1) in die Bewertung der korrekten Funktionsweise des Evaluierungsgegenstandes

Qualitätsstufen

IT-Zertifikat

Europäische Kriterien

Wirksamkeit

und (2) in die Bewertung der Wirksamkeit der eingesetzten Mechanismen. Die Wirksamkeit wird durch eine Skala zur Bewertung der Stärke von Sicherheitsmechanismen durch die Stufen niedrig, mittel und hoch festgelegt.

5.3.1 Evaluationsstufen

Evaluationsstufen

Stufe E1

Die ITSEC-Kriterien definieren sieben Evaluationsstufen E0 bis E6, wobei jede dieser Stufen den Grad an Vertrauen ausdrückt, der in die Korrektheit der Funktionalität des Evaluierungsgegenstandes gesetzt wird. Die Evaluationsstufen sind hierarchisch strukturiert. Abbildung 5.1 gibt einen groben Überblick über die Stufen. Stufe E0 wurde nicht betrachtet, da sie eine unzureichende Vertrauenswürdigkeit beschreibt. Zur Erreichung der Stufe E1 müssen die Sicherheitsanforderungen für das IT-System formuliert sein. Das heißt, dass die Sicherheitsstrategie sowie die Bedrohungen des Systems beschrieben werden müssen. Die Funktionalität der einzusetzenden Sicherheitsmechanismen ist in einer informellen Beschreibung des Architekturentwurfs zu erfassen. Zur Evaluierung sind funktionale Tests

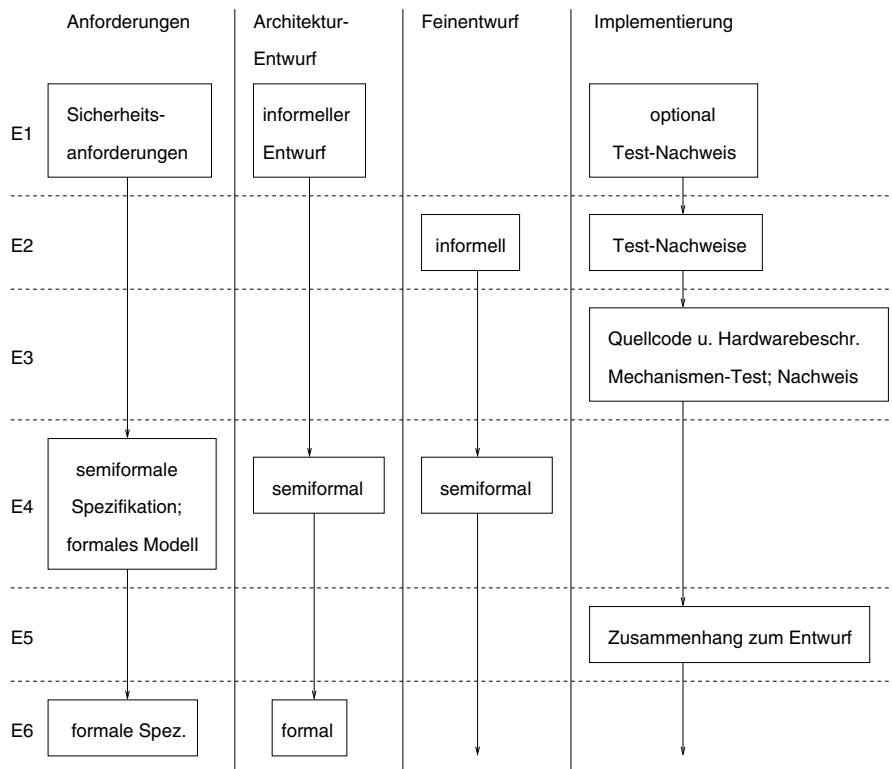


Abbildung 5.1: Evaluierungsstufen der ITSEC-Kriterien

durchzuführen. Falls bereits Tests durchgeführt wurden und eine Testdokumentation zur Verfügung steht, so muss diese die Testziele, -verfahren und -ergebnisse beinhalten.

Eine Einstufung nach E2 erfordert zusätzlich die informelle Beschreibung des Feinentwurfs des Systems, also die Beschreibung der Realisierung aller sicherheitsspezifischen und -relevanten Funktionen. Die E2-Stufe erfordert weiterhin Testdokumentationen und eine Bewertung der Testergebnisse relativ zu den gestellten Sicherheitsanforderungen.

Die Evaluierungsstufe E3 stellt insbesondere zusätzliche Anforderungen an die Implementierung. Es wird eine Beschreibung gefordert, die den Zusammenhang und die Übereinstimmung zwischen Quellcode bzw. den Hardware-Konstruktionszeichnungen einerseits und den Komponenten des Feinentwurfs andererseits beschreibt. Mit dem Übergang auf die höheren Stufen werden zunehmend semiformale bzw. formale Techniken zur Beschreibung von Anforderungen bzw. des Systemdesigns gefordert. Semiformale Spezifikationen können Spezifikationssprachen wie SDL oder grafische Darstellungen sein wie Datenflussdiagramme, Entity-Relationship-Diagramme oder Zustandsübergangsdiagramme.

Eine E4-Einstufung erfordert ein formales Sicherheitsmodell zur präzisen Beschreibung der Sicherheitseigenschaften. Darüber hinaus müssen für die Beschreibung der eingesetzten, sicherheitsspezifischen Funktionen semiformale Notationen verwendet werden, und es ist eine semiformalen Beschreibung des Architekturentwurfs sowie des Feinentwurfs erforderlich.

Mit dem Übergang zur Stufe E5 erfolgt der Übergang von beschreibenden Dokumenten zu erklärenden. Für den Implementierungsteil wird insbesondere gefordert, dass der Zusammenhang zwischen Quellcode und den Komponenten des Feinentwurfs erklärt wird. Die höchste Evaluierungsstufe E6 erfordert schließlich den Einsatz formalen, mathematisch fundierter Techniken zur Spezifikation der Sicherheitsanforderungen sowie zur Beschreibung des Architekturentwurfs.

5.3.2 Qualität und Bewertung

Die Bewertung der Qualität eines evaluierten Systems besteht aus einem Paar, bestehend aus einer Evaluationsstufe, also einer Bewertung der Korrektheit der Funktionalität, und aus einer Einstufung der Wirksamkeit der verwendeten Mechanismen. So besagt beispielsweise die Zertifizierungsstufe (E2, mittel), dass mit zufriedenstellenden Mitteln die Korrektheit der Systemfunktionalität dargelegt und die Stärke der dafür eingesetzten Realisierungsmechanismen mit mittel bewertet wurde.

Stufe E2

Stufe E3

Stufe E4

Stufe E5

Stufe E6

Korrektheit und
Wirksamkeit

Der Zusammenhang zwischen den Qualitätsstufen der IT-Kriterien und den Evaluationspaaren der ITSEC-Kriterien ist in Tabelle 5.2 dargestellt.

	unzu-reichend	gering	zufrieden-stellend	gut bis sehr gut	ausge-zeichnet
IT	Q0	Q1	Q2	Q3–Q5	Q6, Q7
ITSEC	E0 niedrig	E1 mittel	E2 hoch	E3–E5 hoch	E6 hoch

Tabelle 5.2: Qualitätsstufen der IT und ITSEC-Kriterien

Das deutsche Signaturgesetz schreibt beispielsweise vor, dass Komponenten, die zur Erzeugung der Signaturschlüssel verwendet werden, eine Einstufung nach (E4, hoch) besitzen müssen, damit der Schlüssel zur Erstellung qualifizierter digitaler Signaturen (vgl. Abschnitt 8.2) eingesetzt werden kann.

Fazit

Die IT- und ITSEC-Kriterien stellen Fortschritte gegenüber den TCSEC-Kriterien dar, aber sie konzentrieren sich noch immer sehr auf hierarchisch verwaltete, zentrale Systeme. Auch berücksichtigen sie anwenderspezifische Interessen noch immer unzureichend und erfordern einen mit hohem Aufwand durchzuführenden Evaluationsprozess, dessen Ergebnisse eine zu geringe Aussagekraft besitzen. Weiterhin gilt, dass mit diesen Kriterien Gefahren, die sich aus der Verwendung unzuverlässiger Werkzeuge ergeben, noch immer nicht genügend erfasst werden. Nachteilig für einen globalen Wettbewerb ist darüber hinaus, dass die Zertifikate, die auf der Basis der jeweiligen nationalen Kataloge und auch des europäischen Katalogs ausgestellt werden, nicht weltweit anerkannt sind.

5.4 Common Criteria

Common Criteria

Mit den Common Criteria for Information Technology Security Evaluation (CC) [145]², den gemeinsamen Kriterien für die Prüfung und Bewertung der Sicherheit von Informationstechnik wurden Kriterien entwickelt, die als internationaler Standard dienen und damit weltweit einheitlich anerkannt und gültig sind. Sie sind für die Bewertung der Sicherheitseigenschaften praktisch aller informationstechnischer Produkte und Systeme geeignet.

² Aktuelle Informationen findet man unter <http://www.commoncriteriaportal.org/>.

Die erste Version (Version v1.0) der Common Criteria wurde bereits 1996 veröffentlicht und nach einer ausführlichen Testphase durch die Version v2.0 im Jahre 1998 ersetzt. Die Version v2.1 wurde ab 1999 als ISO-Standard ISO 15408 eingesetzt. Im Jahr 2006 wurde die CC Version 2.3 veröffentlicht und ist bei der ISO unter der Nummer 15408 als ein internationaler Standard anerkannt. Die CC Version 3.1 wurde im September 2006 verabschiedet und wurde in Deutschland im Bundesanzeiger vom 23.02.2007 offiziell bekannt gegeben. Die Version 3.1 ist bei der ISO zur Standardisierung eingereicht worden.

Hersteller sowie Vertreiber von IT-Produkten können in Deutschland beim Bundesamt für Sicherheit in der Informationstechnik (BSI) Sicherheitszertifikate auf Grundlage der CC beantragen. Wie bei einer ITSEC-Evaluierung, werden auch die CC-Evaluierungen von akkreditierten und vom BSI lizenzierten Prüfstellen durchgeführt.

5.4.1 Überblick über die CC

Die CC sind eine Weiterentwicklung und Harmonisierung der europäischen ITSEC-Kriterien, des Orange-Book (TCSEC) der USA und der kanadischen Kriterien (CTCPEC). Abbildung 5.2 stellt die Abhängigkeiten zwischen den verschiedenen nationalen und internationalen Kriterienkatalogen dar.

Einordnung

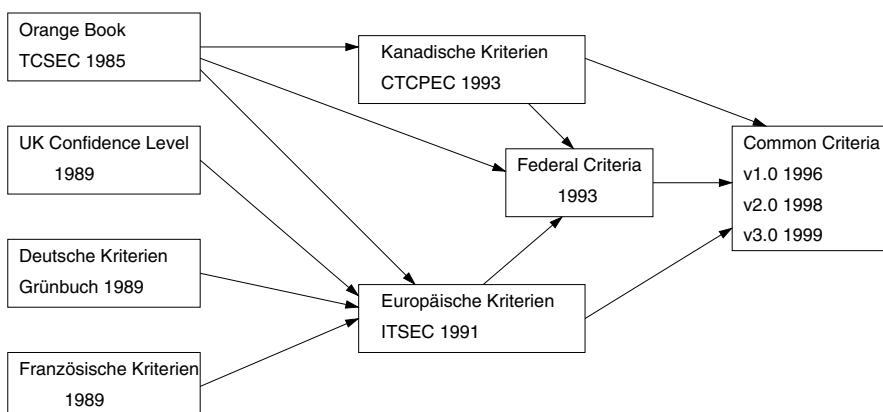


Abbildung 5.2: Abhängigkeiten zwischen Kriterienkatalogen

Analog zu den bereits erläuterten Kriterienkatalogen dienen die CC dazu, Vertrauen in die Wirksamkeit von IT-Sicherheitsfunktionen von evaluierten IT-Systemen zu schaffen. Das CC-Zertifikat gilt als entsprechender Nachweis für die Vertrauenswürdigkeit des evaluierten Systems. Aus Sicht eines Anwenders ist es darüber hinaus aber auch wünschenswert zu wissen, ob die zertifizierte Sicherheitsfunktionalität seinen Bedürfnissen entspricht und

Schutzprofil

die Qualität der Sicherheit seiner jeweiligen Bedrohungslage und dem Wert seiner Daten angemessen ist. Mit dem in den Common Criteria verankerten Konzept der Schutzprofile (engl. *protection profile*(PP)) wird versucht, diesen Anforderungen Rechnung zu tragen. Unabhängig von existierenden Produkten können Anwender ihre Anforderungen und Bedürfnisse mittels derartiger Schutzprofile zum Ausdruck bringen.

EAL

Die Evaluierungsstufen EAL1 – EAL7 (Evaluation Assurance Level) der Common Criteria sind sehr eng an die ITSEC-Stufen angelehnt. Die Evaluierungsstufe EAL2 entspricht der ITSEC-Stufe E1, die Stufe EAL3 der Stufe E2 usw. bis schließlich zur Evaluierungsstufe EAL7, die der ITSEC Stufe E6 entspricht. Die Stufe EAL1 wurde zusätzlich eingeführt, um den Zugang zur Evaluierung zu erleichtern. Auf Grund vieler Gemeinsamkeiten zwischen ITSEC und CC ist gewährleistet, dass alle Zertifikate auf der Basis der ITSEC auch nach der Einführung der CC ihren Wert behalten und vergleichbar zu CC-Evaluationsergebnissen sind.

Die Common Criteria bestehen aus drei Teilen, auf die im Folgenden etwas näher eingegangen wird.

Introduction and General Model**Teil 1**

Der Teil 1 enthält einen Überblick über die erforderlichen Dokumente, die für eine Evaluierung vorzulegen sind. In Anlehnung an die TCSEC sind die Sicherheitsanforderungen eines Evaluierungsgegenstandes (EVG) (engl. *Target of Evaluation, TOE*) zunächst unabhängig von dessen Einsatzumgebung in Schutzprofilen (Protection Profile) (siehe dazu Abschnitt 5.4.3) zu beschreiben.

Mit Beginn der Evaluierung werden die Sicherheitsanforderungen des Schutzprofils in spezielle Sicherheitsvorgaben (engl. *security target* (ST)) für diesen EVG überführt. In den Sicherheitsvorgaben werden über die Informationen eines Schutzprofils hinaus, noch weitere Informationen und Beschreibungen über die genaue Einsatzumgebung und den Gegenstand der Evaluierung hinzugefügt. Liegt für den EVG kein passendes Schutzprofil vor, so können die Sicherheitsvorgaben direkt formuliert werden. Dieses Vorgehen kombiniert den aus den nordamerikanischen Kriterien bekannten Ansatz der Verwendung von Schutzprofilen mit der Verwendung von Sicherheitsvorgaben wie sie in den ITSEC-Kriterien festgelegt sind. Die Spezifikationen für die Erstellung von Schutzprofilen und Sicherheitsvorgaben werden in Anhängen des Teils 1 erläutert.

Vor der Evaluierung des EVG erfolgt zunächst eine separate Evaluierung der Sicherheitsvorgaben, in der geprüft wird, ob die Bedrohungen, die voraussichtliche Einsatzumgebung, die Sicherheitsziele und die Sicherheitsanforderungen auf einander abgestimmt sind. Dadurch soll unter anderem

verhindert werden, dass in den Sicherheitsvorgaben Sicherheitsanforderungen aufgeführt werden, die zu keiner Erhöhung der Sicherheit führen und damit nur eine Erhöhung der Komplexität und des Aufwandes nach sich ziehen würden. Mit einer erfolgreichen Evaluation der Sicherheitsvorgaben legt man die Grundlage für die eigentliche Evaluation des EVGs.

Security Functional Requirements

Der Teil 2 enthält eine Beschreibung der Kriterien und Anforderungen an Sicherheitsgrundfunktionen (siehe Abschnitt 5.4.2) analog zu den ITSEC-Funktionsklassen. Die Sicherheitsanforderungen an die Funktionalität sind in Klassen (Classes) strukturiert und innerhalb einer Klasse in Familien (Families) aufgeteilt. Jede Familie besitzt mindestens eine Komponente (Component), in der die Sicherheitsanforderungen an die konkrete Funktionalität (u.a. Identifikation und Authentifizierung, Zugriffskontrolle und Beweissicherung) beschrieben werden. Die in diesem Teil angegebenen Sicherheitsanforderungen spiegeln allgemein anerkanntes und akzeptiertes Expertenwissen wider. Dies stellt ein empfohlenes Angebot für die Beschreibung der Funktionalität eines Produktes bzw. Systems dar, von dem jedoch in begründeten Fällen abgewichen werden kann. Das heißt, dass die Kriterien es auch erlauben, eigene Sicherheitsanforderungen zu formulieren. Im Anhang finden sich Hintergrundinformationen zu dem Funktionskatalog und es werden Zusammenhänge zwischen Bedrohungen, Sicherheitszielen und funktionalen Anforderungen aufgezeigt.

Teil 2

Security Assurance Requirements

Der Teil 3 enthält schließlich die Beschreibung der Kriterien zur Evaluierung von Schutzprofilen sowie der Sicherheitsvorgaben. Um die Problematik der Re-Zertifizierung abzuschwächen, ist hier auch ein Konzept festgelegt (Assurance Maintenance), das beschreibt, unter welchen Randbedingungen das Evaluierungsergebnis auch dann noch gültig ist, wenn nach Abschluss der Evaluierung Änderungen vorgenommen werden.

Teil 3

Die Sicherheitsanforderungen an die Vertrauenswürdigkeit sind wie bei den Sicherheitsanforderungen an die Funktionalität mittels Klassen, Familien und Komponenten strukturiert. Sie bestehen aus den (1) Anforderungen an den Entwickler, (2) Anforderungen an Inhalt und Form der Prüfnachweise und den (3) Anforderungen an den Evaluator. Analog zu den Sicherheitsgrundfunktionen entsprechen auch die Sicherheitsanforderungen an die Vertrauenswürdigkeit einem allgemein anerkannten Expertenwissen und können, falls sie auf den Evaluierungsgegenstand nicht angemessen anwendbar sind, erweitert werden. In Abschnitt 5.4.4 geben wir einen Überblick über die wichtigsten Klassen und Familien.

EAL

Aus dem Zusammenfügen der Vertrauenswürdigkeitskomponenten ergibt sich eine Evaluationsstufe für Vertrauenswürdigkeit (Evaluation Assurance Level, EAL) (vgl. Tabelle 5.5). Wie bereits gesagt, umfassen die CC sieben EAL-Stufen. Ziel einer Common Criteria-Evaluierung ist die Bestätigung, dass die vom Hersteller behauptete Sicherheitsfunktionalität wirksam ist. Da die Sicherheitsleistung insbesondere durch die Ausnutzbarkeit vorhandener Schwachstellen unwirksam werden kann, ist die Analyse der Schwachstellen ein zentrales Prüfziel. Mit wachsenden EAL-Stufen erhöht sich der zu leistende Analyseaufwand, um auch komplexere Schwachstellen aufdecken zu können. Für den Fall, dass ein EVG in speziellen Bereichen über die Anforderungen der beantragten Evaluationsstufe hinausgehende Prüfanforderungen erfüllt, kann dies als besondere Anstrengung, die der Hersteller geleistet hat, positiv hervorgehoben werden (Augmentation).

Die Anforderungen an die Funktionalität sowie an die Vertrauenswürdigkeit können in Schutzprofilen für bestimmte Produkttypen (z.B. Firewalls, Smartcards) zusammengefasst werden. Diese Schutzprofile enthalten einen ausführlichen Beschreibungsteil, in dem u.a. das Sicherheitskonzept beschrieben und die Bedrohungen den Anforderungen gegenübergestellt werden.

Tabelle 5.3 fasst im Überblick zusammen, welche Informationen und Hilfestellungen die drei Teile der CC für die drei Zielgruppen der CC, nämlich Anwender, Entwickler und Evaluatoren zur Verfügung stellt.

CEM

Eine weitere wichtige Grundlage für die gegenseitige Anerkennung von CC-zertifizierten Produkten ist die Entwicklung der *Gemeinsamen Evaluationsmethodologie* (Common Evaluation Methodology, CEM) auf der Basis der CC. Wenn sichergestellt ist, dass von den beteiligten Stellen dieselben Kriterien und dasselbe Evaluationsverfahren in gleicher Weise angewendet werden, dann können Nutzer und Anbieter ein ausreichendes Vertrauen in die Evaluationsergebnisse anderer (z.B. nicht-nationaler) Zertifizierungsstellen haben. Die zugrunde liegenden Sicherheitskriterien beschreiben primär, *was* evaluiert und die Methodologie *wie* evaluiert werden soll.

Wie die CC, so gliedern sich auch die CEM in mehrere Teile. Der Teil *Introduction and General Model* erläutert den Evaluationsansatz und Grundprinzipien, die bei einer CC-Evaluation zu beachten sind. Der Teil *Evaluation Methodology* stellt die eigentliche Evaluationsmethodologie dar. Hier werden Evaluationsmethoden, Prozeduren und Techniken für die Evaluation von Schutzprofilen, von Sicherheitsvorgaben (Security Targets) und für die Vertrauenswürdigkeitsstufen (Evaluation Assurance Level) beschrieben.

	Anwender	Entwickler	Evaluatoren
Teil 1	Hintergrund-informationen und zum Nachschlagen. Allgemeine Anleitung für ein PP.	Hintergrund-informationen, Hilfestellung bei der Formulierung von Sicherheitsspezifikationen für den EVG.	Hintergrund-informationen und zum Nachschlagen. Allgemeine Anleitung für PP und ST.
Teil 2	Anleitung zur Spezifikation von Anforderungen an Sicherheitsfunktionen.	Hilfestellung bei der Formulierung von funktionalen Spezifikationen für den EVG.	Hilfestellung bei der Feststellung, ob ein EVG die postulierten Sicherheitsfunktionen wirksam erreicht.
Teil 3	Anleitung bei der Festlegung der erforderlichen Vertrauenswürdigkeitsstufen.	Hilfestellung bei der Spezifikation der Anforderungen an die Vertrauenswürdigkeit des EVG.	Hilfestellung bei Feststellung der Vertrauenswürdigkeit des EVG und bei der Prüfung und Bewertung von PP und ST.

Tabelle 5.3: Anwendung der CC für die jeweilige Zielgruppe

5.4.2 CC-Funktionsklassen

Nachfolgend geben wir einen groben Überblick über die Funktionsklassen, die in den CC ausführlich beschrieben sind.

Die Klasse FAU (Sicherheitsprotokollierung) befasst sich mit der Sicherheitsprotokollierung. Zur Sicherheitsprotokollierung gehören das Erkennen, Aufzeichnen, Speichern und Analysieren von Informationen im Zusammenhang mit sicherheitsrelevanten Aktivitäten. Die dabei entstehenden Protokollaufzeichnungen können untersucht werden, um zu ermitteln, welche sicherheitsrelevanten Aktivitäten ausgeführt wurden und wer (welcher Benutzer) dafür verantwortlich ist.

Die Klasse FCO (Kommunikation) stellt zwei Familien bereit, die sich speziell mit der Sicherstellung der Identität der am Datenaustausch beteiligten Seiten befassen. Diese Familien beziehen sich auf die Sicherstellung der Identität eines Urhebers der übertragenen Informationen (Urheberschaftsbeweis) und die Sicherstellung der Identität des Empfängers der übertragenen

Klasse FAU

Klasse FCO

Informationen (Empfangsbeweis). Diese Familien stellen sicher, dass ein Urheber nicht bestreiten kann, die Nachricht verschickt zu haben und dass auch der Empfänger den Empfang nicht leugnen kann.

Klasse FCS

Die Klasse FCS (Kryptografische Unterstützung) wird dann verwendet, wenn der EVG kryptografische Funktionen, die als Hardware, Firmware und/oder Software implementiert sein können, enthält. Die Klasse FCS besteht aus zwei Familien, nämlich FCS_CKM (Kryptografisches Schlüsselmanagement) und FCS_COP (Kryptografischer Betrieb). Die Familie FCS_CKM behandelt die Aspekte des Managements kryptografischer Schlüssel, während sich die Familie FCS_COP mit der Anwendung dieser kryptografischen Schlüssel während des Betriebs befasst.

Klasse FDP

Die Klasse FDP (Schutz der Benutzerdaten) ist in vier Gruppen von Familien aufgeteilt, die auf die Benutzerdaten in einem EVG während des Imports, Exports und der Speicherung sowie auf direkt mit Benutzerdaten verknüpfte Sicherheitsattribute eingehen. Dazu gehören Sicherheitspolitiken zum Schutz der Benutzerdaten wie eine Zugriffskontrollpolitik oder eine Informationsflusskontrolle.

Klasse FIA

Die Klasse FIA (Identifikation und Authentisierung) beschäftigt sich mit den Anforderungen an Funktionen zur Einrichtung und Verifizierung angegebener Benutzeridentitäten. Die Familien in dieser Klasse behandeln die Bestimmung und Verifizierung der Benutzeridentität, die Bestimmung von deren jeweiligen Berechtigungen zur Interaktion mit dem EVG und die richtige Zuordnung der Sicherheitsattribute zu jedem einzelnen autorisierten Benutzer.

Klasse FMT

Die Klasse FMT (Sicherheitsmanagement) dient zur Spezifikation des Managements von vielfältigen Aspekten des EVG. Dazu gehört die Festlegung unterschiedlicher Managementrollen und ihre Interaktion, wie beispielsweise die Separierung der Berechtigung, aber auch die Spezifikation des Managements von Sicherheitsattributen, die zum Beispiel Zugriffskontrollisten und Verzeichnisse der Berechtigungen einschließen.

Klasse FPR

Die Klasse FPR (Privatheit) enthält Anforderungen an die Privatheit. Diese Anforderungen stellen einen Schutz des Benutzers gegen Enthüllung und Missbrauch seiner Identität durch andere Benutzer bereit (u.a. Anonymität, Pseudonymität).

Klasse FPT

Die Klasse FPT (Schutz der Sicherheitsfunktionen des EVG) enthält Familien von funktionalen Anforderungen an die Integrität und das Management der Mechanismen, die die Sicherheitsfunktionen bereitstellen, sowie an die Integrität von Daten der Sicherheitsfunktionen.

Klasse FRU

Die Klasse FRU (Betriebsmittelnutzung) stellt drei Familien zur Unterstützung der Verfügbarkeit geforderter Betriebsmittel, wie zum Beispiel

Rechenfähigkeiten und/oder Speicherfähigkeiten, bereit. Die Familie Fehlertoleranz stellt den Schutz gegen Nichtverfügbarkeit von Fähigkeiten bei Fehlern, die durch den EVG verursacht wurden, bereit. Die Familie Priorität der Dienste stellt sicher, dass die Betriebsmittel den wichtigeren oder zeitkritischeren Aufgaben zugeordnet werden und nicht durch Aufgaben niedriger Priorität vereinnahmt werden können. Die Familie Betriebsmittelzuteilung stellt Begrenzungen für den Gebrauch der verfügbaren Betriebsmittel bereit und schützt deshalb den Benutzer vor einer Vereinnahmung der Betriebsmittel.

Die Klasse FTA (EVG-Zugriff) spezifiziert funktionale Anforderungen zur Kontrolle der Einrichtung einer Benutzersitzung.

Klasse FTA

Die Familien der Klasse FTP (Vertrauenswürdiger Pfad/Kanal) stellen Anforderungen an einen vertrauenswürdigen Kommunikationspfad zwischen Benutzern und den Sicherheitsfunktionen bereit, sowie an einen vertrauenswürdigen Kommunikationskanal zwischen den TSF und anderen vertrauenswürdigen IT-Produkten.

Klasse FTP

Wie bereits weiter vorne ausgeführt, stellen die in den aufgelisteten Klassen erfassten funktionalen Beschreibungen ein Angebot dar, mit denen ein Hersteller die Sicherheitsanforderungen an sein Produkt bzw. System auf eine standardisierte Weise beschreiben kann.

5.4.3 Schutzprofile

Schutzprofile (protection profiles) bieten eine anerkannte Lösung für Standard-Sicherheitsprobleme einer Klasse von Produkten. Sie werden implementierungsunabhängig festgelegt und können in Verfeinerungsschritten durch die daraus ableitbaren Sicherheitsvorgaben auf einen konkreten Evaluationsgegenstand zugeschnitten werden. Schutzprofile lassen sich als eine Verallgemeinerung der Sicherheitsvorgaben gemäß CC auffassen. Sie beschreiben ein Sicherheitskonzept. In Schutzprofilen werden Anforderungen an die Funktionalität sowie an die Vertrauenswürdigkeit zusammengefasst, wodurch eine definierte Menge von Sicherheitszielen vollständig abgedeckt wird. Schutzprofile müssen vollständig, konsistent und technisch stimmig sein. Dies wird in einem eigenen Evaluierungsprozess nachgewiesen und durch ein Zertifikat bestätigt.

Protection Profile

Abbildung 5.3 beschreibt den strukturellen Aufbau eines Schutzprofils, wie er durch die CC vorgegeben wird.

Struktureller Aufbau

Die Einführung soll das Schutzprofil eindeutig identifizieren und einen allgemeinen Überblick über das Schutzprofil geben, so dass ein Anwender entscheiden kann, ob dieses Schutzprofil für ihn von Interesse ist. In einem Schutzprofil sind die allgemeinen IT-Sicherheitseigenschaften, aber auch die

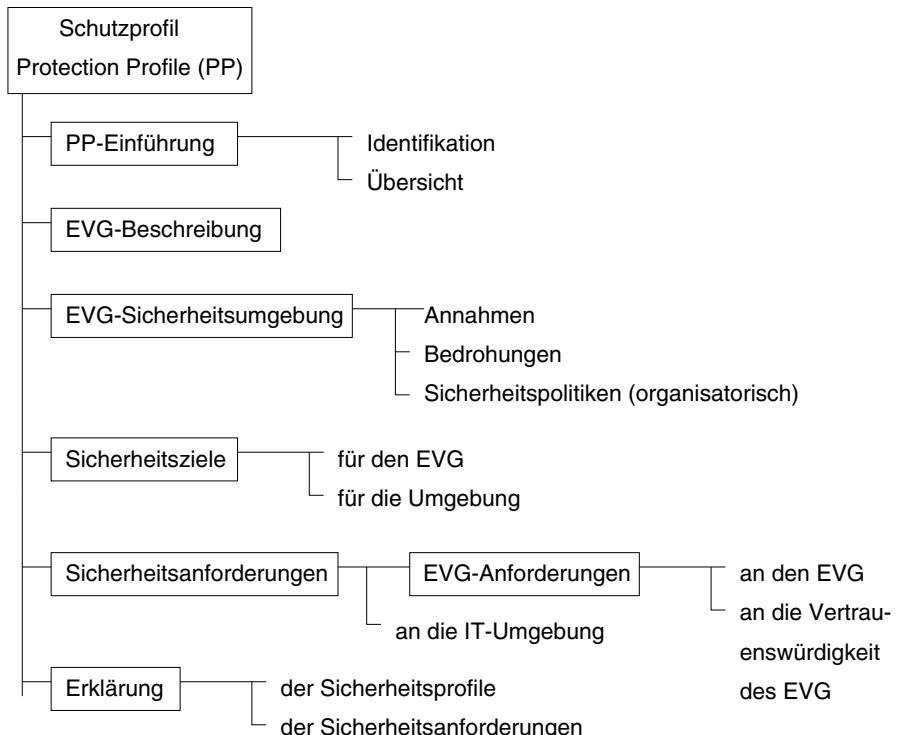


Abbildung 5.3: Struktureller Aufbau eines Schutzprofils

Grenzen der Benutzung aufzuzeigen. Das IT-Sicherheitskonzept beschreibt sowohl den Wert der Daten und deren Verarbeitung als auch die Annahmen an eine typische Einsatzumgebung. Enthalten sind auch einzuuhaltende gesetzliche Auflagen, vorgeschriebene Sicherheitsstandards sowie alle abzuwehrenden Bedrohungen auf die zu schützenden Werte. Hierbei wird eine Art Risikoanalyse durchgeführt, wobei die Bedrohungen zu den zu schützenden Werten in Beziehung gebracht werden sollen. Die Sicherheitspolitiken beschreiben Annahmen über den sicheren Betrieb des Evaluierungsgegenstandes, die sicherstellen, dass die Sicherheitsziele effektiv erfüllt werden (u.a. materieller Schutz des EVG, verbindende Aspekte z.B. in Netzen, administrative Aspekte).

Sicherheitsziele

Die Sicherheitsziele müssen detaillierte Angaben darüber machen, wie den Bedrohungen entgegengewirkt werden soll bzw. die Sicherheitspolitiken erfüllt werden sollen. Es wird dabei zwischen IT-Sicherheitszielen und Nicht-IT-Sicherheitszielen unterschieden, wobei die letzteren durch Anforderungen an die Einsatzumgebung erreicht werden sollen. Dabei soll jeder relevanten Bedrohung durch mindestens ein Sicherheitsziel entgegengewirkt werden.

Ein Schutzprofil beschreibt den Sicherheitsbedarf für eine bestimmte Produktgruppe in Form von Sicherheitsanforderungen. Sie dienen als Basis für die Spezifikation der Produktfunktionalität und als Basis für die Evaluierung und Zertifizierung mit dem Ziel, die Erfüllung aller Anforderungen zu bestätigen. Zur Erstellung der Anforderungen für ein Profil kann auf die CC-Funktionsklassen zurückgegriffen werden, da diese abgestimmte, evaluierbare und in sich konsistente Sicherheitsanforderungen enthalten. Das heißt, Verfasser eines Schutzprofils erhalten mit den CC Rahmenvorgaben, die sie dabei unterstützen, ihre jeweiligen Sicherheitsanforderungen vollständig und widerspruchsfrei zu spezifizieren. Die Anforderungen an die Vertrauenswürdigkeit sind in Form der 7 Evaluation Assurance Levels (EAL-Stufen) vordefiniert und können zur Vergleichbarkeit genutzt werden.

Die Forderung nach einem Erklärungsteil im Schutzprofil zwingt den Verfasser, eine erste Konsistenz- und Vollständigkeitsanalyse durchzuführen und die Angemessenheit aller Anforderungen darzulegen.

Auf dem Portal der Mitglieder der Common Criteria findet man eine Vielzahl von bereits veröffentlichten Protection Profiles³, die u.a. Geräte zur Zugriffskontrolle, Datenbanken, Smartcards, Schlüsselmanagement, Betriebssysteme und Produkte zur Erstellung digitaler Signaturen umfassen. Im Zuge der Entwicklungen rund um die neuen elektronischen Pässe in Deutschland, die biometrische Merkmale in Form von einem digitalen Gesichtsbild und ab 2007 auch einen digitalen Fingerabdruck enthalten, hat das deutsche BSI ein Protection Profile⁴ für maschinenlesbare Reisedokumente erstellt. Das Schutzprofil definiert die Sicherheitsziele und Anforderungen für den kontaktlosen Chip, das ist der Evaluierungsgegenstand (EVG), eines maschinenlesbaren Reisedokumentes unter Berücksichtigung der Anforderungen der internationalen zivilen Luftfahrtorganisation (ICAO⁵). Das Protection Profile definiert Sicherheitsanforderungen (Basic Access Control und Extended Access Control) für den EVG, die der Vertrauenswürdigkeitsstufe EAL4+ entsprechen.

Ende 2005 wurde ebenfalls vom BSI ein Protection Profile⁶ für die elektronische Gesundheitskarte (eGK) festgelegt. Der Evaluierungsgegenstand ist eine Smartcard und das Schutzprofil definiert die Sicherheitsdienste, die durch diese Karte bereitgestellt werden, basierend auf den Regularien des deutschen Gesundheitssystems (u.a. Gesetz zur Modernisierung der Gesetzlichen Krankenversicherung (GKV-Modernisierungsgesetz), Sozial-

³ Siehe <http://www.commoncriteriaportal.org/pps/>

⁴ Siehe <http://www.commoncriteriaportal.org/files/ppfiles/PP0026b.pdf>

⁵ International Civil Aviation Organisation

⁶ Siehe <http://www.commoncriteriaportal.org/files/ppfiles/PP0020b.pdf>

gesetzbuch (SGB)). Das Schutzprofil spezifiziert Sicherheitsanforderungen für die Vertrauenswürdigkeitsstufe EAL4+.

5.4.4 Vertrauenswürdigkeitsklassen

Im Folgenden geben wir einen Überblick über die Vertrauenswürdigkeitsklassen und -familien, die im Teil 3 der Common Criteria ausführlich dargestellt werden. Tabelle 5.4 stellt die Vertrauenswürdigkeitsklassen, -familien und Kurzformen für jede Familie im Überblick dar.

Klasse ACM

Die Klasse ACM (Konfigurationsmanagement) (CM) hilft sicherzustellen, dass die Integrität des EVG erhalten bleibt. Das CM verhindert nicht-autorisierte Modifikationen, Hinzufügungen oder Löschungen im EVG und schafft somit die Vertrauenswürdigkeit, dass der für die Prüfung und Bewertung verwendete EVG und die entsprechende Dokumentation dem lieferbaren EVG und dessen Dokumentation entsprechen. Die Familie CM-Automatisierung (ACM_AUT) legt den Automatisierungsgrad fest, der bei der Kontrolle der Konfigurationsteile zur Anwendung kommt. Die Familie CM-Leistungsvermögen (ACM_CAP) definiert die Eigenschaften des Konfigurationsmanagementsystems und die Familie CM-Anwendungsbereich (ACM SCP) zeigt die EVG-Bestandteile auf, die vom Konfigurationsmanagementsystem kontrolliert werden müssen.

Klasse ADO

Die Klasse ADO (Auslieferung und Betrieb) definiert Anforderungen an die Maßnahmen, Prozeduren und Normen, die sich mit der Sicherheit der Auslieferung, der Installation und des Betriebs des EVG befassen. Sie stellt ferner sicher, dass der vom EVG gebotene Sicherheitsschutz während Transport, Installation, Anlauf und Betrieb nicht verletzt wird. Die Familie Auslieferung (ADO_DEL) umfasst die Prozeduren zur Erhaltung der Sicherheit während des Transports des EVG zum Benutzer. Sie umfasst spezielle Prozeduren oder Operationen, die zum Nachweis der Authentizität der gelieferten EVG erforderlich sind. Die Familie Installation, Generierung und Anlauf (ADO_IGS) erfordert, dass die Kopie des EVG vom Systemverwalter so konfiguriert und aktiviert wird, dass sie die gleichen Schutzeigenschaften wie die EVG-Masterkopie aufweist.

Klasse ADV

Die Klasse ADV (Entwicklung) definiert Anforderungen zur schrittweisen Verfeinerung von der EVG-Übersichtsspezifikation in den ST bis zur tatsächlichen Implementierung. Jede dieser Darstellungen liefert Informationen, die dem Evaluator helfen festzustellen, ob die funktionalen Anforderungen des EVG erfüllt sind. Die Familie ADV_FSP beschreibt die funktionale Spezifikation und muss eine vollständige und getreue Umsetzung der Sicherheitsanforderungen an den EVG sein. Die Familie ADV_HLD ist eine Entwurfsspezifikation auf höchster Stufe, die die funktionale Spezifikation

Klasse	Familie	Abkürzung
ACM: Konfigurationsmanagement	CM-Automatisierung CM-Leistungsvermögen CM-Anwendungsbereich	ACM_AUT ACM_CAP ACM SCP
ADO: Auslieferung und Betrieb	Auslieferung Installation, Generierung und Anlauf	ADO_DEL ADO_IGS
ADV: Entwicklung	Funktionale Spezifikation Entwurf auf hoher Ebene Darstellung der Implementierung Interna der EVG- Sicherheitsfunktionen Entwurf auf niedriger Ebene Übereinstimmung der Darstellung Sicherheitsmodell	ADV_FSP ADV_HLD ADV_IMP ADV_INT ADV_LLD ADV_RCR ADV_SPM
Klasse AGD: Handbücher	Systemverwalterhandbuch Benutzerhandbuch	AGD ADM AGD_USR
ALC: Lebenszyklus- Unterstützung	Sicherheit bei der Entwicklung Fehlerbehebung Lebenszyklus-Beschreibung Werkzeuge und Techniken	ALC_DVS ALC_FLR ALC_LCD ALC_TAT
ATE: Testen	Testabdeckung Testtiefe Funktionale Tests Unabhängiges Testen	ATE_COV ATE_DPT ATE_FUN ATE_IND
AVA: Schwachstellenbewertung	Analyse der verdeckten Kanäle Missbrauch Stärke der EVG- Sicherheitsfunktionen Schwachstellenanalyse	AVA_CCA AVA_MSU AVA_SOF AVA_VLA

Tabelle 5.4: Vertrauenswürdigkeitsklassen und -familien

verfeinert. Der Entwurf auf hoher Ebene gibt die Grundstruktur der TSF und der wichtigsten Hardware-, Firmware- und Softwareelemente an. Die Darstellung der Implementierung (ADV_IMP) ist die am wenigsten abs-

trakte Darstellung. Sie erfasst die Details der internen Arbeitsweise in der Form von Quellcode, Hardwarekonstruktionszeichnungen usw. Die Interna der EVG-Sicherheitsfunktionen (ADV_INT) spezifizieren die erforderliche interne Strukturierung der TSF. Die Familie Entwurf auf niedriger Ebene (ADV_LLD) ist eine detaillierte Entwurfsspezifikation, die den Entwurf auf hoher Ebene bis auf einen Detaillierungsgrad verfeinert, der als Grundlage zur Programmierung und/oder Hardwarekonstruktion benutzt werden kann. Die Übereinstimmung der Darstellung (ADV_RCR) ist ein Nachweis der Übereinstimmung zwischen allen benachbarten Paaren verfügbarer Darstellungen. Sicherheitsmodelle (ADV_SPM) sind strukturierte Darstellungen der Sicherheitspolitiken und dienen der Schaffung einer stärkeren Vertrauenswürdigkeit, dass die funktionale Spezifikation mit den Sicherheitspolitiken und schließlich mit den funktionalen Anforderungen an den EVG übereinstimmt.

Klasse AGD

Die Klasse AGD (Handbücher) definiert Anforderungen, die die Verständlichkeit, Abdeckung und Vollständigkeit der vom Entwickler bereitgestellten Betriebsdokumentation betreffen. Die Anforderungen an das Systemverwalterhandbuch (AGD_ADM) helfen sicherzustellen, dass Systemverwalter und Operatoren des EVG detaillierte Informationen zur sicheren Verwaltung des EVG und zum wirksamen Gebrauch der TSF-Privilegien und TSF-Schutzfunktionen erhalten. Die Anforderungen an das Benutzerhandbuch (AGD_USR) helfen sicherzustellen, dass die Benutzer in der Lage sind, den EVG sicher bedienen zu können, indem gefordert wird, dass es die notwendigen Hintergrundinformationen und spezielle Informationen zum korrekten Gebrauch der EVG-Schutzfunktionen bereitstellt.

Klasse ALC

Die Klasse ALC (Lebenszyklus-Unterstützung) definiert Anforderungen an die Vertrauenswürdigkeit, die durch Verwendung eines klar festgelegten Lebenszyklus-Modells für alle Schritte der EVG-Entwicklung erreicht wird, einschließlich Fehlerbehebungsprozeduren, sowie des korrekten Gebrauchs von Werkzeugen und Techniken und der zum Schutz der Entwicklungsumgebung angewendeten Sicherheitsmaßnahmen. Die Familie Sicherheit bei der Entwicklung (ALC_DVS) deckt die materiellen, organisatorischen, personellen und andere in der Entwicklungsumgebung ergriffene Sicherheitsmaßnahmen ab. Die Fehlerbehebung (ALC_FLR) stellt sicher, dass die von den EVG-Konsumenten entdeckten Fehler aufgezeichnet und korrigiert werden, solange der EVG vom Entwickler unterstützt wird. Die Lebenszyklus-Beschreibung (ALC_LCD) gibt an, dass die technischen Praktiken, die ein Entwickler bei der Herstellung des EVG anwendet, die Gesichtspunkte und Aktionen beinhalten, die in den Anforderungen an die Entwicklungsverfahren und die Unterstützung des Betriebs identifiziert sind. Die Familie Werkzeuge und Techniken (ALC_TAT) befasst sich mit der Not-

wendigkeit, die für die Analyse und Implementierung des EVG verwendeten Entwicklungswerzeuge zu definieren.

Die Klasse ATE (Testen) legt Anforderungen an das Testen dar, die nachweisen, dass die TSF die funktionalen EVG-Sicherheitsanforderungen erfüllen. Die Familie Testabdeckung (ATE_COV) behandelt die Vollständigkeit der vom Entwickler mit dem EVG durchgeführten funktionalen Tests, d.h. sie befasst sich mit dem Grad, bis zu dem die EVG-Sicherheitsfunktionen getestet sind. Die Testtiefe (ATE_DPT) befasst sich mit dem Detaillierungsgrad, bis zu dem der Entwickler den EVG testet. Mittels funktionalen Tests (ATE_FUN) wird ermittelt, dass die TSF die Eigenschaften aufweist, die zur Erfüllung der in deren ST enthaltenen Anforderungen notwendig sind. Die Familie unabhängiges Testen (ATE_IND) spezifiziert den Grad, bis zu dem das funktionale Testen von einer anderen Person als dem Entwickler (zum Beispiel einer dritten Person) ausgeführt werden muss.

Klasse ATE

Die Klasse AVA (Schwachstellenbewertung) behandelt insbesondere solche Schwachstellen, die bei Konstruktion, Betrieb, Missbrauch oder falscher Konfiguration des EVG entstehen. Die Analyse der verdeckten Kanäle (AVA_CCA) zielt auf die Entdeckung und Analyse unerwünschter Kommunikationskanäle ab, die zur Verletzung der vorgesehenen TSP ausgenutzt werden können. Die Missbrauchsanalyse (AVA_MSU) untersucht, ob ein Systemverwalter oder Benutzer, der sich in den Handbüchern auskennt, unter normalen Umständen in der Lage ist festzustellen, ob Konfiguration und Betrieb des EVG unsicher sind. Die Analyse der Stärke der EVG-Sicherheitsfunktionen (AVA_SOF) betrifft EVG-Sicherheitsfunktionen, die auf Wahrscheinlichkeits- oder Permutationsmechanismen (zum Beispiel Passwort oder Hashfunktion) beruhen. Die Familie Schwachstellenanalyse (AVA_VLA) umfasst die Identifikation von Fehlern, die möglicherweise in den einzelnen Verfeinerungsstufen der Entwicklung eingeführt wurden. Sie führt zur Definition von Penetrationstests.

Klasse AVA

Die erklärten Vertrauenswürdigkeitsklassen und -familien bilden die Grundlage zur Festlegung der Evaluierungsstufen EAL1 – EAL7. Tabelle 5.5 verdeutlicht den Zusammenhang zwischen den Vertrauenswürdigkeitsfamilien (Zeilen) und den Evaluierungsstufen (Spalten).

Jede Nummer in dieser Matrix gibt eine spezifische Vertrauenswürdigkeitskomponente an. Die Zunahme an Vertrauenswürdigkeit von einer EAL-Stufe zur nächsten wird durch Austausch gegen eine hierarchisch höhere Vertrauenswürdigkeitskomponente derselben Vertrauenswürdigkeitsfamilie (d.h. zunehmende Schärfe, größerer Anwendungsbereich und/oder zunehmende Testtiefe) und durch Hinzufügen von Vertrauenswürdigkeitskomponenten aus anderen Vertrauenswürdigkeitsfamilien (d.h. Hinzufügen neuer Anforderungen) erreicht. Die CC erlauben es aber auch, Vertrauenswür-

Klasse	Familien	EAL1	EAL2	EAL3	4	5	6	7
Konfigurationsmanagement	ACM_AUT				1	1	2	2
	ACM_CAP	1	2	3	4	4	5	5
	ACM_SCP			1	2	3	3	3
Auslieferung und Betrieb	ADO_DEL		1	1	2	2	2	3
	ADO_IGS	1	1	1	1	1	1	1
Entwicklung	ADV_FSP	1	1	1	2	3	3	4
	ADV_HLD		1	2	2	3	4	5
	ADV_IMP				1	2	3	3
	ADV_INT					1	2	3
	ADV_LLD				1	1	2	2
	ADV_RCR	1	1	1	1	2	2	3
	ADV_SPM				1	3	3	3
Handbücher	AGD ADM	1	1	1	1	1	1	1
	AGD_USR	1	1	1	1	1	1	1
Lebenszyklus-Unterstützung	ALC_DVS			1	1	1	2	2
	ALC_LCD				1	2	2	3
	ALC_TAT				1	2	3	3
Testen	ATE_COV		1	2	2	2	3	3
	ATE_DPT			1	1	2	2	3
	ATE_FUN		1	1	1	1	2	2
	ATE_IND	1	2	2	2	2	2	3
Schwachstellenbewertung	AVA_CCA					1	2	2
	AVA_MSU			1	2	2	3	3
	AVA_SOF		1	1	1	1	1	1
	AVA_VLA		1	1	2	3	4	4

Tabelle 5.5: Vertrauenswürdigkeitsfamilien und Evaluierungsstufen

digkeitskomponenten (aus Vertrauenswürdigkeitsfamilien, die noch nicht in der EAL enthalten sind) zu einer EAL hinzuzufügen, oder aber Vertrauenswürdigkeitskomponenten (gegen andere, hierarchisch höher stehende Vertrauenswürdigkeitskomponenten derselben Vertrauenswürdigkeitsfamilie) in einer EAL auszutauschen. Von den in den CC definierten Vertrauenswürdig-

keitskonstrukten können nur die EAL mit einem Zusatz versehen werden. Ein Zusatz bringt die Verpflichtung seitens des Antragstellers mit sich, die Nützlichkeit und den zusätzlichen Wert der der EAL hinzugefügten Vertrauenswürdigkeitskomponente zu rechtfertigen.

Das vom BSI definierte Schutzprofil für den kontaktlosen Chip in Reisedokumenten (vgl. Seite 231) fordert beispielsweise die Vertrauenswürdigkeitsstufe EAL4+. Die über den Umfang von EAL4 hinausgehenden Anforderungen betreffen die zusätzlichen Komponenten ADV_IMP.2, ALC_DVS.2, AVA_MSU.3 und AVA_VLA.4.

Aufgrund der bislang positiven Erfahrungen bei Evaluierungen und der Vereinheitlichung der Kriterien zu einem weltweiten Standard bleibt zu hoffen, dass sich die Common Criteria weiterhin bewähren und dass sie für viele Produkte verwendet werden. Nur so kann der Verbraucher verlässliche Zusicherungen über die zu erwartende Sicherheitsleistung von Systemen bzw. Komponenten erhalten und diese Produkte vertrauensvoll einsetzen. Dies ist eine wichtige Voraussetzung für die Akzeptanz und den großflächigen Einsatz von Softwareprodukten in sicherheitskritischen privaten, firmen- bzw. behördeninternen sowie öffentlichen Bereichen.

5.5 Zertifizierung

IT- bzw. ITSEC- oder Common Criteria-Zertifikate werden in Deutschland vom BSI, dem Bundesamt für Sicherheit in der Informationstechnik, als „ein amtlich bestätigter Nachweis der Sicherheitsleistungen eines Produkts“ ausgestellt. Bestandteil des Zertifizierungsverfahrens ist eine technische Prüfung, die von einer vom BSI anerkannten Prüfstelle (u.a. TÜV) durchgeführt werden kann. In den USA ist die NSA, die National Security Agency, für die Evaluierung von IT-Produkten bzw. -Systemen zuständig.

Zertifikate

Die Kriterienkataloge erlauben es, sowohl Hardwarekomponenten als auch Software für unterschiedliche Anwendungsbereiche zu evaluieren. Evaluierungsgegenstände können sein:

Evaluierungs-
gegenstände

- Rechner, Router, Firewalls, PC-Karten etc.
- Kryptosysteme, Betriebssysteme, Datenbanken, Anwendungssoftware,
- vollständige IT-Systeme in konkreter Einsatzumgebung,
- Werkzeuge zur Produktentwicklung und
- Produktkombinationen: Rechner zusammen mit Peripherie-Geräten und Anwendungen.

An Zertifizierungsarten unterscheidet man die Zertifizierung eines fertigen Produktes, die entwicklungsbegleitende Zertifizierung sowie die Re-

Zertifizierungsart

Zertifizierung eines bereits zertifizierten Produktes. Ein Zertifikat kann sich nie auf einen ganzen Produkttyp beziehen, sondern gilt stets entweder für eine bestimmte Version oder für ein Release eines Produktes. Aufgrund der hohen Änderungsfrequenz von Software sind Re-Zertifizierungen in kurzen Abständen notwendig.

Zertifizierungsbericht

Als Ergebnis der Evaluierung wird ein Zertifizierungsbericht erstellt. Dieser Zertifizierungsbericht ist ein Dokument, das die Sicherheitseigenschaften des Evaluierungsgegenstandes relativ zu den aufgeführten Bedrohungen beschreibt, die Wirksamkeit der eingesetzten Sicherheitsmechanismen bewertet und mit der Vergabe einer Evaluierungsstufe den Grad des Vertrauens in die Korrektheit der Funktionalität des Produkts bescheinigt. Weiterhin enthält ein solcher Bericht Anforderungen an die Installation und Einsatzumgebung des Evaluierungsgegenstandes sowie eine Beschreibung der erkannten, inhärenten Schwachstellen zusammen mit Angaben über mögliche, wirksame Gegenmaßnahmen.

zertifizierte Produkte

Beispiele für BSI-zertifizierte Produkte⁷ sind Betriebssysteme sowohl für den Mainframe- als auch für den Workstation- bzw. PC-Bereich, VirenScanner, Notebooks, Datenübertragungssoftware, Firewall-Komponenten oder auch Chipkartenlesegeräte.

⁷ Siehe auch https://www.bsi.bund.de/DE/Themen/ZertifizierungundAnerkennung/ZertifizierungnachCCundITSEC/ZertifizierteProdukte/zertifizierteprodukte_node.html

6 Sicherheitsmodelle

Das Kapitel führt in Abschnitt 6.1 ein einfaches Klassifikations- und Bewertungsschema ein, das als Grundlage zur Beschreibung der nachfolgend vorgestellten formalen Sicherheitsmodelle dient. In Abschnitt 6.2 beschäftigen wir uns zunächst mit der Klasse der Zugriffskontrollmodelle und stellen mit dem Zugriffsmatrix-Modell, dem rollenbasierten Modell und dem Bell-LaPadula Modell die wichtigsten Repräsentanten dieser Klasse vor. Abgesehen vom Bell-LaPadula Modell behandeln die Ansätze dieser Klasse schwerpunktmäßig Datensicherheitsaspekte. Demgegenüber stehen bei der Klasse der Informationsflussmodelle, die im Abschnitt 6.3 betrachtet werden, Fragen der Informationssicherheit im Vordergrund. Wir beschränken uns in diesem Kapitel auf die Vorstellung der wichtigsten Ansätze zur formalen Modellierung von Systemen und Anwendungen. Modelle und Techniken zur Modellierung von Protokollen werden in diesem Kapitel nicht betrachtet.

6.1 Modell-Klassifikation

Die im Kapitel 5 erläuterten Kriterienkataloge zur Bewertung der Sicherheit von Systemen haben verdeutlicht, dass ein hoher Qualitätsstandard bei der Konstruktion von Systemen nur dann erreichbar ist, wenn man von einer präzisen, d.h. einer semiformalen oder formalen Beschreibung der Sicherheitseigenschaften des zu entwickelnden Systems bzw. der Anwendung ausgeht und ein Sicherheitsmodell konstruiert. Der Begriff des Modells wird im Folgenden im Sinne von „Abstraktion von“ verwendet. Ein Modell beschreibt die Eigenschaften eines realen Systems auf einem hohen Abstraktionsniveau.

Die Frage nach der Nützlichkeit eines spezifischen Modells für ein Anwendungsproblem muss sich demnach daran orientieren, welche problemspezifischen Anforderungen bestehen und welche Möglichkeiten das jeweilige Modell bietet, diese Anforderungen zu erfassen. Es stellt sich also die Aufgabe, bekannte Sicherheitsmodelle im Hinblick auf die durch sie beschreibbaren Eigenschaften zu analysieren, um sie den Anforderungen entsprechend einsetzen zu können.

Dimensionen

Modelle werden im Folgenden nach ihrer Fähigkeit klassifiziert, flexibel an anwendungsspezifische Anforderungen anpassbar zu sein und Datenintegritäts- und Vertraulichkeitsanforderungen differenziert erfassen zu können. Die Dimensionen des Klassifikationsschemas ergeben sich durch die Festlegung der zu schützenden sowie der agierenden Einheiten (Objekte und Subjekte), durch die Festlegung der zu verwaltenden Zugriffsrechte, durch die Möglichkeiten zur differenzierten Erfassung von Zugriffsbeschränkungen sowie durch das Spektrum der modellierbaren Sicherheitsstrategien. Die Bewertung der unterschiedlichen Dimensionen liefert Leitlinien und einen Rahmen für die Auswahl eines Modells für ein zu konstruierendes Anwendungssystem.

6.1.1 Objekte und Subjekte

Die zu schützenden Einheiten werden als Objekte des Modells erfasst, so dass deren Granularität von wesentlicher Bedeutung für die Sicherheits-eigenschaften des zu konstruierenden Systems ist.

Grobkörnige Modellierung

grobe Objekte

Zugriffskontrollen werden nur für die zu schützenden Einheiten, die Objekte, durchgeführt, so dass deren grobkörnige Festlegung wie beispielsweise als Dateien zu einer unvollständigen Erfassung von sicherheitsrelevanten Eigenschaften und damit zu Sicherheitslücken führen kann. Komponenten im System, die nicht als zu schützende Objekte modelliert werden, unterliegen keiner Zugriffs- bzw. Informationsflussskontrolle. Eine grobe Granularität der Objekte führt zudem dazu, dass unterschiedliche Einheiten in einem Objekt zusammengefasst werden, so dass keine Möglichkeiten mehr bestehen, für diese Einheiten individuelle Schutzfestlegungen zu treffen. Die Vergabe von Zugriffsrechten gemäß des need-to-know-Prinzips, also so, dass nur so viele Rechte vergeben werden, wie zur Erfüllung der Aufgaben nötig sind, ist ausgehend von diesen grobkörnigen Objekten schwierig bzw. kaum zu erreichen. Grobgranulare Objekte auf der Basis von Dateien haben jedoch den Vorteil, dass mit den Konzepten und Diensten klassischer Betriebssysteme eine effiziente Realisierung der Zugriffsbeschränkungen und -kontrollen möglich ist.

grobe Subjekte

Subjekte greifen auf Objekte zu, so dass diese Zugriffe zu kontrollieren sind. Eine grobgranulare Modellierung von Subjekten bedeutet, dass die Aktivitäten einzelner Benutzer nicht differenziert kontrollierbar sind. Die Benutzer des Systems werden zu wenigen, meist großen Benutzergruppen zusammengefasst. Da Zugriffsrechte jeweils an die ganze Gruppe vergeben werden, ohne die Bedürfnisse des individuellen Benutzers zu berücksichtigen, vergibt man auf diese Weise meist zu viele Rechte, was zu einem

Missbrauch von Rechten führen kann. Eine benutzerspezifische Rechtevergabe und -kontrolle ist auf der Basis grobgranularer Subjekte nur schwer oder gar nicht zu modellieren. Analog zu grobgranularen Objekten können auch grobgranulare Subjekte effizient mit den in klassischen Betriebssystemen angebotenen Konzepten implementiert werden.

Anwendungsspezifische Körnung

Zur Durchsetzung des need-to-know-Prinzips ist es erforderlich, dass Zugriffsrechte für Objekte beliebiger Granularität vergeben und kontrolliert werden können. Dies ermöglicht es, die zu schützenden Einheiten anwendungsspezifisch festzulegen und jedem Subjekt höchstens die Menge von Zugriffsrechten zu gewähren, die es zur Erfüllung seiner Aufgabe benötigt.

Objekte

Die Möglichkeit, feingranulare Subjekte zu modellieren, erlaubt es, benutzerspezifische Zugriffsrechte zu vergeben. Zusammen mit einer feingranularen Modellierung von Objekten können die Zugriffsbeschränkungen flexibel und zugeschnitten auf die spezifischen Anforderungen der Anwendung modelliert werden.

Subjekte

6.1.2 Zugriffsrechte

Die Festlegung der Zugriffsrechte hat einen großen Einfluss darauf, in welchem Maße Datenintegritätseigenschaften mit einem betrachteten Modell erfassbar sind.

Wir sagen, dass Zugriffsrechte universell sind, wenn durch sie allgemeine und nicht objektspezifische Operationen bezeichnet werden. Durch die Festlegung von universellen Rechten werden Subjekten große Freiheitsgrade gewährt, die zu Bedrohungen der Datenintegrität führen können. Beispiele für solche universellen Rechte sind die bekannten `write`-, bzw. `read`-Rechte für Dateien. Eine Schreibberechtigung für eine Datei gemäß des `write`-Rechts eröffnet unbeschränkte Möglichkeiten zur Manipulation der in der Datei enthaltenen Daten. Da klassische Betriebssysteme eine Verwaltung universeller Rechte direkt unterstützen, lassen sich die entsprechend modellierten Systeme effizient implementieren.

universelles Recht

Objektspezifische Zugriffsrechte ermöglichen es, Zugriffsmöglichkeiten auf einen festgelegten funktionalen Kontext und zugeschnitten auf das jeweilige Objekt zu beschränken. Dieser funktionale Kontext stellt sicher, dass das Objekt nur gemäß der festgelegten Semantik der Operation manipuliert wird. Beispiele hierfür sind Rechte, die den Methoden abstrakter Datenobjekte entsprechen (z.B. Push oder Pop auf einem Keller-Objekt). Objektspezifische Rechte ermöglichen die Vergabe von Zugriffsrechten, die zugeschnitten sind auf die von Subjekten durchzuführenden Aufgaben. Auf diese Weise lässt sich a priori der Schaden begrenzen, den ein Angreifer bei der Nutzung eines

objektspezifisches Recht

solchen Rechts verursachen kann. Die Implementierung von Systemen mit objektspezifischen Zugriffsrechten erfordert eine geeignete Unterstützung entweder direkt durch das zugrunde liegende Betriebssystem, oder durch eine Middleware-Schicht bzw. durch das Anwendungssystem selbst, wie zum Beispiel Datenbankmanagementsysteme.

6.1.3 Zugriffsbeschränkungen

einfach

Wir nennen eine Zugriffsbeschränkung einfach, wenn ein Subjekt für den Zugriff auf ein Objekt nur das entsprechende Zugriffsrecht besitzen muss, d.h. dass ein Zugriff nicht durch weitere Bedingungen beschränkt wird. Einfache Zugriffsbeschränkungen lassen sich effizient durch Konzepte und Dienste klassischer Betriebssysteme oder gängiger Middleware-Ausführungsumgebungen implementieren.

komplex

Wir nennen eine Zugriffsbeschränkung komplex, wenn ein zulässiger Zugriff auf ein Objekt vom Vorhandensein eines entsprechenden Zugriffsrechts sowie weiteren Bedingungen abhängig ist. Beispiele dafür sind Bedingungen über globale Variablen (z.B. Zugriff nur während der Bürostunden) oder über objektlokale Variablen (z.B. innerhalb von 24 Stunden höchstens dreimal Geld von einem Konto abheben). Klassische Betriebssysteme realisieren in der Regel ein Referenzmonitorkonzept, womit nur einfache Zugriffsbeschränkungen erfasst werden. Komplexe Überprüfungen müssen explizit durch die Anwendungsprogramme realisiert und auf die zugrunde liegende Betriebssystem-Infrastruktur abgestimmt werden (z.B. implementieren von Sichten (views) in Datenbanksystemen).

6.1.4 Sicherheitsstrategien

Wir unterscheiden zwei große Klassen von Strategien, nämlich die Klasse der Zugriffskontroll- und die der Informationsfluss-Strategien. In der Praxis werden in der Regel Zugriffskontrollstrategien angewendet. Diese untergliedern wir weiter in benutzerbestimmbare, systembestimmte und rollenbasierte Strategien.

Zugriffskontrollstrategie

DAC Strategie

Benutzerbestimmbare Zugriffskontrollstrategien (engl. *discretionary access control* (DAC)) basieren auf dem Eigentümer-Prinzip (engl. *owner*). Das bedeutet, dass der Eigentümer eines Objekts für die Rechtevergabe bzw. -rücknahme verantwortlich ist. Zugriffsrechte werden auf der Basis einzelner Objekte vergeben bzw. zurückgenommen. Das heißt, dass mit benutzerbestimmten Strategien objektbezogene Sicherheitseigenschaften, aber keine systemglobalen Eigenschaften festgelegt werden. Da bei der Festlegung der

individuellen, objektbezogenen Beschränkungen die Abhängigkeiten zwischen Objekten, wie z.B. Benutzungs-, Kommunikations- und Kooperations-abhängigkeiten, nicht betrachtet werden, besteht die Gefahr, inkonsistente Strategien zu modellieren. Eine widersprüchliche Rechtevergabe liegt beispielsweise vor, wenn ein Subjekt s durch die Berechtigung zur Ausführung einer Operation op implizit die Berechtigung zum lesenden Zugriff auf ein Objekt o erhält, andererseits für s aber explizit, d.h. durch ein negatives Recht, der lesende Zugriff auf o verboten ist.

Systembestimmte (regelbasierte) Festlegungen (MAC) spezifizieren system-globale Eigenschaften. Sie dominieren über benutzerbestimmte Rechtevergaben, so dass ein Zugriff verweigert wird, wenn es eine systembestimmte Festlegung gibt, die einen entsprechenden Zugriff verbietet, auch wenn der Zugriff gemäß des benutzerbestimmbaren Teils der Strategie erlaubt wäre. Daneben können aber auch die systembestimmten Rechtevergaben durch benutzerbestimmbare, explizite Berechtigungsverbote weiter eingeschränkt werden. Von einem zugrunde liegenden Betriebssystem sind besondere Maßnahmen und Dienste zur Verfügung zu stellen, um systembestimmte Festlegungen zu realisieren.

MAC Strategie

Im Bereich der Sicherheit von Geschäftsprozessen (innerbetriebliche Prozesse, aber auch unternehmensübergreifende Prozesse) wird verstärkt eine rollenbasierte Zugriffskontrollstrategie RBAC (Role-Based-Access-Control) verfolgt. Im Gegensatz zu herkömmlichen Modellen, die Subjekte in den Mittelpunkt des Rechtemanagements stellen, stehen in rollenbasierten Modellen die durchzuführenden Aufgaben im Vordergrund. Zugriffsrechte werden damit abhängig von und zugeschnitten auf die Aufgaben vergeben, die Subjekte im System durchführen. Subjekte erhalten dann über explizit festzulegende Rollenmitgliedschaften Berechtigungen zur Ausführung von Aufgaben.

RBAC Strategie

Informationsfluss-Strategie

Zugriffskontrollstrategien beschränken Zugriffe auf Objekte, jedoch kontrollieren sie kaum oder gar nicht, auf welche Weise ein Subjekt Informationen, die es durch Objektzugriffe erlangt, weiterverarbeiten darf, d.h. welche Informationsflüsse zulässig sind. Mit systembestimmten Strategien sind nur sehr restriktive und einfache Informationsflussbeschränkungen festlegbar. Informationsfluss-Strategien verallgemeinern diese systemglobalen Beschränkungen, indem gültige und verbotene Informationskanäle zwischen Subjekten bzw. Subjektklassen festgelegt werden.

Informationsflüsse

6.2 Zugriffskontrollmodelle

Zunächst werden zwei Ansätze vorgestellt, die die Modellierung von Aspekten der Datensicherheit ermöglichen. Einfache, systembestimmte Regeln zur Modellierung von Aspekten der Informationssicherheit charakterisieren zwei weitere Modelle, die wir anschließend betrachten.

6.2.1 Zugriffsmatrix-Modell

Das Zugriffsmatrix-Modell (engl. *access matrix model*), auch bekannt als Referenzmonitor-Modell (u.a. [73]), ist das einfachste und älteste Sicherheitsmodell. Es bietet die Möglichkeit, Objekte und Subjekte zugeschnitten auf die Anforderungen der jeweilig zu konstruierenden Anwendung festzulegen sowie Zugriffsrechte universell oder objektspezifisch zu modellieren.

Definition 6.1 (Zugriffsmatrix)

Der Schutzzustand eines Systems zum Zeitpunkt t wird durch eine $|S_t| \times |O_t|$ -Matrix M_t modelliert, wobei gilt:

- die Spalten der Matrix werden durch die endliche Menge O_t der Objekte und
- die Zeilen der Matrix werden durch die endliche Menge S_t der Subjekte zum Zeitpunkt t definiert.
Subjekte können auch Objekte sein, d.h. es gilt $S_t \subseteq O_t$.
- Es gilt¹: $M_t : S_t \times O_t \longrightarrow 2^R$, wobei R die endliche Menge der Zugriffsrechte festlegt.

Ein Eintrag $M_t(s, o) = \{r_1, \dots, r_n\}$ beschreibt die Menge der Rechte, die das Subjekt s zum Zeitpunkt t an dem Objekt o besitzt.

□

Für jeden Zeitpunkt t modelliert die Zugriffsmatrix M_t die in dem betreffenden Zustand gültigen Zugriffsrechte der Subjekte an den Objekten des Systems.

Der Schutzzustand des Systems, d.h. die Matrix M_t , kann durch die Ausführung von Kommandos zur Erzeugung und zum Löschen von Subjekten bzw. Objekten oder zur Weitergabe und Rücknahme von Zugriffsrechten verändert werden. Die Matrix ist ihrerseits ein zu schützendes Objekt, so dass Berechtigungen zur Durchführung von Zustandsänderungen ebenfalls als Rechte zu modellieren und zu vergeben sind.

¹ 2^R bezeichnet die Potenzmenge der Menge R .

Beispiel 6.1 (Zugriffsmatrix)

Ein Beispiel für eine Zugriffsmatrix ist in Tabelle 6.1 angegeben. Prozesse werden als Subjekte modelliert und Dateien, aber auch Prozesse sind die Objekte. In seiner dualen Rolle kann somit ein Prozess als Subjekt Rechte wahrnehmen aber auch selber als Objekt genutzt werden. Für Prozesse sind ein Sende- und Empfange-Recht für Kommunikationsendpunkte (z.B. Sockets oder Ports), Rechte zur Ausführung von Synchronisationsoperationen wie `wait` und `signal`, sowie ein zu dem Eigentümer-Recht für Dateien analoges Recht, `control (c)`, als Zugriffsrechte modelliert. In der angegebenen Zugriffsmatrix hat der Prozess 1 das `control`-Recht an den Prozessen 2 und 4. Eine mögliche Interpretation des `control`-Rechts ist die Modellierung einer Vater-Sohn-Beziehung zwischen Prozessen.

Rechte

	Datei 1	Datei 2	Prozess 1	Proz. 2	Proz. 3	Proz. 4
Prozess 1				c, send		c
Prozess 2			wait, signal		c	
Prozess 3	read,write	write, owner		receive		send
Prozess 4		read, write			send	

Tabelle 6.1: Ausschnitt aus einer Zugriffsmatrix

Sei der Prozess 3 in Tabelle 6.1 im Auftrag von Benutzer Heinz und Prozess 4 im Auftrag des Benutzers Otto aktiv. Subjekt Heinz bzw. der für ihn tätige Prozess 3 besitzt das owner-Recht für Datei 2. D.h. er darf Rechte an dem Objekt Datei 2 an andere Subjekte weitergeben bzw. erteilte Rechte an Datei 2 wieder zurücknehmen. Subjekt Otto bzw. der für ihn aktive Prozess 4 hat kein Recht, auf das Objekt Datei 1 zuzugreifen, besitzt jedoch das Lese- und Ausführrecht für Datei 2 sowie das Recht, Nachrichten an Prozess 3 zu senden.

Rechtevergabe



Statische und dynamische Matrix

Wir unterscheiden Zugriffsmatrix-Modelle mit statischer bzw. dynamischer Zugriffsmatrix. Bei einem statischen Modell ist keine Änderung der Rechtevergabe möglich. Statische Vorgehensweisen sind zur Modellierung von Anwendungsproblemen geeignet, in denen der Rechtezustand a priori bekannt und über lange Zeiträume konstant ist. Die Sicherheitsstrategien

statische Matrix

einfacher Router, die als zustandslose Paketfilter Firewalls (vgl. Kapitel 14.1) dienen, können zum Beispiel mit statischen Zugriffsmatrix-Modellen beschrieben werden.

Beispiel 6.2 (Statische Zugriffsmatrix)

Firewall-Szenario

In Tabelle 6.2 ist ein Ausschnitt einer Zugriffsmatrix eines Paketfilter Firewalls angegeben. Sie spezifiziert die erlaubten und unzulässigen Zugriffe auf Ports durch Subjekte, die über IP-Adressen identifiziert werden. Zum besseren Verständnis enthält die Tabelle keine IP-Adressen, sondern eine Art funktionale Charakterisierung der Subjekte als FTP-Server, Mail-Rechner und als externe, d.h. als außerhalb des Intranetzes liegende Rechner. Das Fehlen eines Eintrags besagt, dass der Zugriff auf den entsprechenden Port blockiert wird.

	Port 21 (ftp)	Port 25 (smtp)	Port 53 (DNS-Anfrage)	Port 514 (rshell)	≥ 1023 (smtp)
FTP-Server	receive		send		
Mail-Rechner		receive			send
externer Rechner		send	send		

Tabelle 6.2: Sicherheitsstrategie für Port-Zugriffe

Mit der angegebenen Zugriffsmatrix ist u.a. festgelegt, dass ein Rechner, dessen Internetadresse nicht zur Domäne des zu schützenden Netzes gehört, der also ein externer Rechner ist, Mails an den Port 25 senden oder auch eine Domain-Server Anfrage an Port 53 richten darf. Wohingegen für externe Rechner das Erzeugen einer Remote Shell auf dem lokalen Rechner verboten sein sollte. D.h. der Zugriff auf Port 514 ist (natürlich) zu blockieren.



Dynamische Matrix

Elementaroperation

Zustandsübergänge, also Veränderungen der Zugriffsmatrix, werden durch die Ausführung von Kommandos modelliert, die durch eine Folge von Elementaroperationen spezifiziert werden. Zur Erklärung der allgemeinen Vorgehensweise zur Definition solcher Kommandos greifen wir im Folgenden auf die von Harrison, Ruzzo und Ullman in [77] definierten Kommandos zu-

rück. Die dort verwendeten sechs Elementaroperationen sind das Erzeugen (`create`) bzw. Löschen (`destroy`) von Objekten und Subjekten sowie Operationen zur Rechteweitergabe (`enter`) und -zurücknahme (`delete`).

Kommandos bzw. Regeln zur Veränderung des Schutzzustandes besitzen folgende allgemeine Struktur:

Kommandos

Kommando	Erläuterung
$\begin{aligned} & \text{COMMAND } \alpha(X_1, \dots, X_k) \\ & \text{IF } r_1 \text{ IN } (X_{s_1}, X_{o_1}) \text{ AND} \\ & \quad \dots \\ & \quad r_m \text{ IN } (X_{s_m}, X_{o_m}) \\ & \text{THEN } op_1, \dots, op_n \\ & \text{END;} \end{aligned}$	α ist ein Regel-Name $X_i \in \mathcal{O}_t \cup \mathcal{S}_t$ formale Parameter $r_i \in \mathcal{R}$ Rechte op_i Elementaroperation

Beispiel 6.3 (Elementaroperationen)

Die enter- und delete- Operationen sind wie folgt festgelegt.

enter u. delete

Elementaroperation	Bedingung	Wirkung
enter r into (s,o)	$s \in \mathcal{S}_t, o \in \mathcal{O}_t$	$M_{t'}(s, o) := M_t(s, o) \cup \{r\}$
delete r from (s,o)	$s \in \mathcal{S}_t, o \in \mathcal{O}_t$	$M_{t'}(s, o) := M_t(s, o) \setminus \{r\}$



Beispiel 6.4 (Kommandos)

Das Kommando `create` definiert die Rechtevergabe bei der Erzeugung von Objekten. Gemäß der Festlegung erhält der Erzeuger das Eigentümerrecht für das neu erzeugte Objekt, das ihn dazu berechtigt, Nutzungsrechte an dem Objekt an andere Subjekte weiterzugeben.

Objekt-Erzeugung

Kommando	Erläuterung
COMMAND create (<i>user</i> , <i>file</i>) create file; enter owner into (<i>user</i> , <i>file</i>) END;	Aufruf durch Subjekt <i>user</i> $\mathcal{O}_{t'} = \mathcal{O}_t \cup \{\text{file}\}$ neue Spalte in der Matrix Eigentümer-Recht an Subjekt <i>user</i> vergeben

Das Kommando `revoke_r` definiert eine Rechterücknahme, die nur für dazu Berechtigte zulässig ist.

Kommando	Erläuterung
COMMAND revoke_r (S_1, S_2, X) IF owner $\in M_t(S_1, X)$ AND $r \in M_t(S_2, X)$ THEN delete r from (S_2, X) END;	Aufruf durch S_1 Rechterücknahme



Sicherheitseigenschaften

Die Formalisierung von Schutzzuständen ermöglicht es, Sicherheitseigenschaften der so modellierten Systeme zu untersuchen. Im Folgenden werden beispielhaft drei interessante Eigenschaften untersucht.

(1) Safety-Problem

Fragestellung

Das Safety-Problem erfasst die Fragestellung, ob ausgehend von einem gegebenen Schutzzustand M_t ein Subjekt s das Recht r an dem Objekt o erhalten kann, wenn es dieses Recht im Zustand M_t noch nicht besitzt. D.h. es ist zu zeigen, dass ausgehend von Zustand M_t , mit $r \notin M_t(s, o)$, ein Zustand $M_{t'}$, mit $t' > t$ erreichbar ist, für den gilt $r \in M_{t'}(s, o)$.

Mit Definition 6.2 wird die Grundlage zur Formalisierung des Safety-Problems gelegt. Eingeführt werden zum einen Systemkonfigurationen, die ein modelliertes System zu einem gegebenen Zeitpunkt vollständig beschreiben, und zum anderen Übergänge zwischen Konfigurationen, die durch die Ausführung von Kommandos ausgelöst werden.

Definition 6.2 (Safety-Problem)

Eine Systemkonfiguration K_t zum Zeitpunkt t ist durch den Schutzzustand M_t und die in diesem Zustand existierenden Mengen von Objekten \mathcal{O}_t und Subjekten \mathcal{S}_t festgelegt, also $K_t = (M_t, \mathcal{O}_t, \mathcal{S}_t)$.

Mit $K_t \stackrel{op}{\models} K_{t+1}$ bezeichnen wir den Konfigurationsübergang, der durch die Ausführung des Kommandos op bewirkt wird.

Sei K_0 eine Anfangskonfiguration des Systems und für die Konfiguration K_t gelte: $r \notin M_t(s, o)$.

Das Safety-Problem reduziert sich auf die Frage, ob ausgehend von der Konfiguration K_t durch Ausführung von Kommandos op_1, \dots, op_n , also

$$K_0 \stackrel{*}{\models} K_t \stackrel{op_1}{\models} K_{t+1} \stackrel{op_2}{\models} \dots \stackrel{op_n}{\models} K_{t+n},$$

eine Konfiguration K_{t+n} erreicht werden kann, mit $r \in M_{t+n}(s, o)$.

□

Harrison, Ruzzo und Ullman haben in [77] die Unentscheidbarkeit des Safety-Problems bewiesen, indem sie das Safety-Problem auf das unentscheidbare Halteproblem von Turing-Maschinen (u.a. [81]) zurückführten. Die Unentscheidbarkeitsaussage besagt, dass es keinen Algorithmus gibt, der bei gegebener Anfangskonfiguration eines Systems mit einer beliebig festgelegten Menge von Kommandos entscheiden kann, ob ein Recht r in den Besitz eines Subjektes s gelangen kann.

Für die Praxis bedeutet dies jedoch keineswegs, dass für ein konkretes System mit einer konkret festgelegten Kommandomenge diese Frage unbeantwortbar ist. Die Unentscheidbarkeitsaussage besagt lediglich, dass es kein generelles Entscheidungsverfahren gibt, sondern von Fall zu Fall dedizierte Entscheidungsverfahren zu konstruieren sind. Theoretische Untersuchungen im Bereich der Zugriffsmatrix-Modelle haben darüber hinaus gezeigt, dass es sogar Entscheidungsverfahren für ganze Systemklassen gibt. Die entsprechenden Systeme lassen sich mit Take-Grant-Modellen beschreiben. Da sie jedoch sehr restriktiven Einschränkungen unterliegen, sind sie für die Praxis wenig relevant und werden hier nicht weiter betrachtet. Ausführliche Darstellungen der Eigenschaften und Grenzen von Take-Grant-Modellen finden sich unter anderem in [110, 51].

(2) Soll-Ist-Vergleiche

Gegeben sei ein konkretes Systemmodell mit einer festgelegten Menge von Kommandos und einem Schutzzustand M_t . Die gewünschten Sicherheitseigenschaften des modellierten Systems seien informell (oder formal) gegeben, so dass zu untersuchen ist, ob das modellierte System (Ist-Eigenschaften) die gestellten Anforderungen (Soll-Eigenschaften) erfüllt.

Konfiguration

Problemstellung

Unentscheidbarkeit

Bedeutung

Fragestellung

Im Rahmen dieser Überprüfung (bzw. Verifikation, falls die Anforderungen formal spezifiziert wurden) kann zum Beispiel untersucht werden, ob ein Subjekt in dem Modell Rechte erhalten kann, die im Widerspruch zu den festgelegten Sicherheitsanforderungen stehen.

Die matrixorientierte Erfassung der direkten Berechtigungen ermöglicht es, durch die Bildung der transitiven Hülle der Berechtigungsabhängigkeiten die indirekten Berechtigungen direkt zu bestimmen. Um Widersprüche zu den festgelegten Anforderungen aufzudecken, sind somit für alle Subjekte deren potentielle implizite Rechte zu ermitteln.

Beispiel 6.5 (Soll-Ist-Abgleich)

Für ein zu modellierendes System seien als Soll-Eigenschaften festgelegt, dass das Subjekt Joe weder explizit noch implizit das Recht `lese_Kontostand` an Objekt `Konto_Bill` erhalten darf. Die Modellierung sei so gewählt, dass auch Prozeduren als Objekte bzw. Subjekte erfasst werden. Ein Auszug des Schutzzustandes des modellierten Systems, der die Ist-Eigenschaften beschreibt, sei wie folgt gegeben:

	...	Konto_Bill	Drucke_Auszug
Joe			<code>execute</code>
Drucke_Auszug		<code>lese_Kontostand</code>	
...			

Anforderung (Soll)

Ist-Zustand

Durch die Vergabe des `execute`-Rechts an den Benutzer Joe zur Ausführung der Prozedur `Drucke_Auszug` und der gleichzeitigen Vergabe des `lese_Kontostand`-Rechts an die Prozedur `Drucke_Auszug` erhält Joe bei Ausführung der Operation `execute` implizit das Recht `lese_Kontostand` an dem Objekt `Konto_Bill`. Das heißt, dass die Ist-Eigenschaften nicht mit den Soll-Eigenschaften übereinstimmen und die Modellierung oder die Rechtevergabe zu korrigieren ist.



(3) Modellierung von Zugriffsrechten

Die Modellierung der Zugriffsrechte hat einen großen Einfluss auf die Sicherheitseigenschaften, die das modellierte System besitzt. Werden Rechte grobgranular vergeben, indem unter einer Sammelberechtigung unterschiedliche Zugriffsmöglichkeiten auf ein Objekt zusammengefasst werden, so können dadurch Sicherheitslücken relativ zu den informell gestellten Sicherheitsanforderungen entstehen. Dieser Sachverhalt wird im Folgenden anhand der Modellierung Unix-artiger Zugriffsrechte verdeutlicht.

Beispiel 6.6 (Grobgranulare Zugriffsrechte)

Die Sicherheitsanforderungen des zu modellierenden Systems, bzw. eines Ausschnittes, seien wie folgt festgelegt: Anforderungen

1. Das Verzeichnis D ist ein Objekt (z.B. die tmp-Datei), für das gilt, dass alle Benutzer des Systems das Schreibrecht an diesem Verzeichnis erhalten, also Dateien in dem Verzeichnis ablegen dürfen.
2. Die Datei f ist ein Element von D und wird vom Benutzer Joe verwaltet (owner-Recht). Es ist sicherzustellen, dass nur der Eigentümer von f die Datei f modifizieren darf.

Zu modellieren ist somit zunächst ein Schreibrecht für Verzeichnisse. Entscheidet man sich dabei für eine aus Unix bekannte Semantik dieses Zugriffsrechts (vgl. Kapitel 12.4), so bedeutet dies, dass mit der Berechtigung, schreibend auf ein Verzeichnis D zugreifen zu dürfen, gleichzeitig auch die Rechte zum Löschen einer beliebigen Datei f aus D sowie zum Hinzufügen einer Datei f' zu D verbunden ist. Das Schreibrecht fasst somit weitere Berechtigungen zu einer vergrößerten Berechtigung zusammen.

Die Menge der Zugriffsrechte sei mit $\mathcal{R} = \{\text{owner}, \text{all_write}, \text{write}\}$ definiert. Modellierung

Zur Erfüllung der Anforderung (2.) wird

$$\{\text{owner}, \text{write}\} \subseteq M_t(\text{Joe}, f)$$

festgelegt. Um die Sicherheitsanforderung (1.) mit einem Zugriffsmatrix-Modell zu erfassen, sind Schreibberechtigungen so zu vergeben, dass alle Benutzer, also auch alle zukünftigen, das Schreibrecht an dem Verzeichnis D erhalten. Dies ist jedoch allein ein modellierungstechnisches Problem, das sich durch die Definition eines geeigneten Kommandos wie folgt einfach lösen lässt.

Zur Erfüllung der Anforderung (1.) wird das `all_write`-Recht an das Objekt D selber vergeben, $\text{all_write} \in M_t(D, D)$, und das Kommando `all_user_write_for_D` wie folgt definiert:

```
COMMAND all_user_write_for_D(user)
IF all_write ∈ Mt(D, D)
THEN enter write into Mt(user, D); delete write from Mt(user, D)
END;
```

Mit dem definierten Kommando erhält jedes Subjekt, das das Kommando ausführt, temporär das Schreibrecht an dem Verzeichnis D , so dass die Sicherheitsanforderung (1.) erfüllt wird.

Problem

Aufgrund der gewählten Modellierung ist aber die Sicherheitsanforderung (2.) nicht zu gewährleisten, da ein beliebiges Subjekt $s \neq \text{Owner}$ die Datei f aus dem Verzeichnis D löschen und damit diese modifizieren kann.

Lösung

Durch eine Differenzierung des Schreibrechts an Verzeichnissen kann das Problem gelöst werden. Modelliert man das Schreib-Recht durch zwei neue Rechte, nämlich durch ein Recht zum Löschen einer Datei und durch ein Recht zum Hinzufügen einer Datei, so lassen sich Zugriffsrechte zugeschnitten auf die Anforderungen vergeben.

**Klassifikation und Bewertung****flexibel**

Das Zugriffsmatrix-Modell ist sehr flexibel einsetzbar, da sowohl Objekte als auch Subjekte beliebig granular festlegbar und Zugriffsrechte universell oder objektspezifisch modellierbar sind. Datenintegritätsanforderungen können anwendungsangepasst erfasst werden. Es liegt somit in der Hand des Modellkonstrukteurs, eine geeignete Modellierung so zu wählen, dass die gewünschten Sicherheitsanforderungen damit auch durchsetzbar sind.

Problembereiche

Da mit dem Modell die Zugriffsbeschränkungen objektbezogen festzulegen sind, besteht die Gefahr, dass inkonsistente Schutzzustände auftreten. Durch die Analyse der Eigenschaften des modellierten Systems müssen solche Inkonsistenzen erkannt und beseitigt werden. Bedingt durch die Freiheitsgrade des Modells können sehr komplexe Sicherheitseigenschaften beschrieben werden, deren Nachweise schwierig und aufwändig sind. Zurzeit stehen nur wenige Werkzeuge zur Unterstützung konsistenter Sicherheitsstrategien zur Verfügung. Das SHVT-Tool des Fraunhofer Instituts für Sichere Informationstechnologie ist ein solches Tool, das mit einem Model-Checking Ansatz arbeitet, um Sicherheitseigenschaften automatisch zu verifizieren. Das Tool kann auch zur Angriffssimulation und zum systematischen Erkennen von Schwachstellen in modellierten Systemen (vgl. [149]) verwendet werden.

Sicherheitsstrategie

Über die Modellierung von Eigentümerrechten sind benutzerbestimmbare Sicherheitsstrategien zu formulieren. Eine rollenbasierte Strategie kann höchstens rudimentär unter Nutzung des Gruppenkonzepts beschrieben werden. Sicherheitsanforderungen, die die Beschränkung von Informationsflüssen betreffen, sind mit dem Zugriffsmatrix-Ansatz nicht modellierbar. Um auch Informationsflussbeschränkungen erfassen zu können, sind Erweiterungen des Modells notwendig.

6.2.2 Rollenbasierte Modelle

Rollenkonzept

Bei einer rollenbasierten Modellierung werden die Berechtigungen zur Nutzung geschützter Komponenten direkt an Rollen und damit an Aufgaben geknüpft. Die durch Rollen modellierten Aufgaben werden von Subjekten

durchgeführt, so dass das Sicherheitsmodell festlegen muss, welche Subjekte welche Aufgaben durchführen, d.h. in welchen Rollen agieren dürfen. Führt ein Subjekt Aktionen in einer Rolle aus, so ist es aktiv in dieser Rolle. Rollenbasierte Zugriffsmodelle (engl. *role-based access control (RBAC)*) wurden 1992 von D.F Farraiolo und D.R. Kuhn in [62, 63] als formale Modelle eingeführt.

Ausgehend von den Arbeiten von Farrailo-Kuhn und dem RBAC-Rahmenwerk, das von R. Sandhu in [158] entwickelt wurde, hat die NIST² im Jahr 2000 ein vereinheitlichtes Modell³ für RBAC vorgeschlagen, das 2004 als ANSI⁴-Standard ANSI INCITS 359-2004 angenommen wurde. Über das Web-Portal der NIST stehen eine Reihe von RBAC-Fallbeispielen⁵ für verschiedene Bereiche, wie dem Bereich der IT-Infrastrukturen, dem der Banken und dem Finanzbereich, für Government-Anwendungen oder auch für Anwendungen im Gesundheitswesen zur Verfügung. Ergänzend zu dem allgemeinen Rahmenwerk wurden und werden spezielle RBAC-Standards für dedizierte Anwendungsdomänen⁶, wie das Gesundheitswesen, industrielle Kontrollsysteme, Biometrie oder auch das Militär entwickelt. Seit ihrer Einführung haben die RBAC-Modelle somit aufgrund ihrer großen Praxisrelevanz eine große Beachtung gefunden. So bieten heute nahezu alle ERP-Systeme, Contentmanagement- oder Dokumentenmanagementsysteme Rollenkonzepte zum Rechte- und Zugriffsmanagement an.

Definition 6.3 fasst die Komponenten und Abbildungen eines rollenbasierten Sicherheitsmodells zusammen.

Definition 6.3 (RBAC-Modell)

Ein rollenbasiertes Sicherheitsmodell wird durch ein Tupel

$$RBAC = (\mathcal{S}, \mathcal{O}, RL, \mathcal{P}, sr, pr, session)$$

definiert, wobei

1. \mathcal{S} die endliche Menge der Benutzer des Systems,
2. \mathcal{O} die endliche Menge der zu schützenden Objekte und
3. RL die endliche Menge von Rollen ist. Jede Rolle beschreibt eine Aufgabe und damit die Berechtigungen der Rollenmitglieder.

² National Institute of Standards and Technology

³ Siehe <http://csrc.nist.gov/rbac/sandhu-ferraiolo-kuhn-00.pdf>

⁴ American National Standards Institute

⁵ <http://csrc.nist.gov/rbac/RBAC-case-studies.html>

⁶ Vgl. <http://csrc.nist.gov/rbac/rbac-stds-roadmap.html>

4. \mathcal{P} ist die endliche Menge der Zugriffsberechtigungen und
5. sr, pr und $session$ beschreiben Relationen.

Rollenmitglied

Die Abbildung sr modelliert die Rollenmitgliedschaft eines Subjektes $s \in \mathcal{S}$,

$$sr : \mathcal{S} \longrightarrow 2^{RL}.$$

Falls $sr(s) = \{R_1, \dots, R_n\}$ gilt, dann ist das Subjekt s autorisiert in den Rollen R_1, \dots, R_n aktiv zu sein.

Berechtigung

Über die Berechtigungsabbildung

$$pr : RL \longrightarrow 2^{\mathcal{P}}$$

wird jeder Rolle $R \in RL$ die Menge derjenigen Zugriffsrechte zugeordnet, die die Mitglieder der Rolle während ihrer Rollenaktivitäten wahrnehmen dürfen.

Sitzung

Die Relation $session \subseteq \mathcal{S} \times 2^{RL}$ beschreibt Paare (s, rl) , die als Sitzungen bezeichnet werden. Es muss gelten, $rl \subseteq sr(s)$.

Für ein Subjekt $s \in \mathcal{S}$ und für eine Sitzung $(s, rl) \in session$ gilt, dass s alle Berechtigungen der Rollen $R \in rl$ besitzt.

Sei (s, rl) eine Sitzung, dann sagen wir, dass s aktiv in den Rollen $R \in rl$ ist.

□

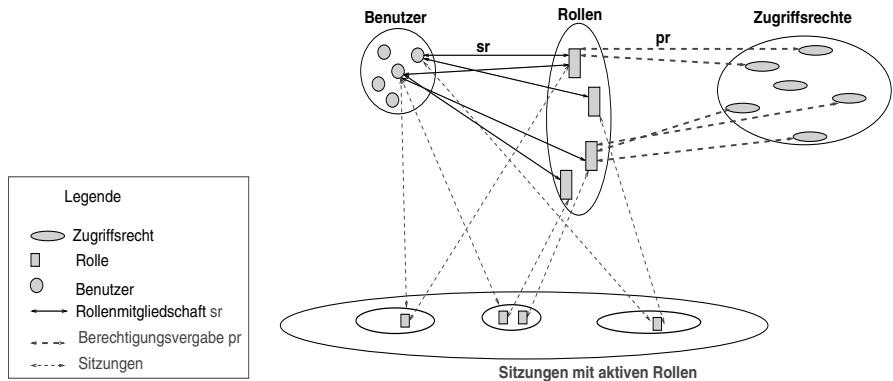


Abbildung 6.1: Zusammenhänge in einem RBAC-Modell

In einem RBAC-System kann ein Subjekt gleichzeitig in unterschiedlichen Sitzungen aktiv sein; es besitzt potentiell alle Berechtigungen aller Rollen, in denen es aktiv ist. Um Berechtigungen wahrnehmen zu können, muss ein Subjekt aktiv in einer solchen Rolle sein, für die die Berechtigung

erteilt ist. Abbildung 6.1 veranschaulicht die Zusammenhänge zwischen den eingeführten Modellierungskonzepten und Relationen.

Beispiel 6.7 (Bankszenario)

Als Beispiel für den Einsatz von RBAC betrachten wird ein einfaches Bankszenario. Gegeben sei das organisatorische Umfeld einer Zweigstelle einer Bank.

Beispiel

- Mögliche Rollen in diesem Szenario seien durch $RL = \{\text{Zweigstellenleiter}, \text{Kassierer}, \text{Kundenbetreuer}, \text{Angestellter}, \text{Kassenprüfer}, \text{Kunde}\}$ festgelegt.
- Zu schützende Objekte \mathcal{O} seien beispielsweise Kundenkonten, Personaldaten, Kreditdaten oder auch Kundendaten.
- Beispiele für Zugriffsrechte \mathcal{P} sind die Berechtigungen, ein Konto zu sperren, Kreditrahmen zu erhöhen oder zu erniedrigen oder ein Konto zu eröffnen bzw. Einzahlungen und Abhebungen auf bzw. von einem Konto durchzuführen.
- Herr Mustermann und Frau Renate seien Mitarbeiter der Bank; sie gehören zur Menge der Subjekte S .
- Herr Mustermann habe die Aufgaben eines Zweigstellenleiters zu erbringen und Frau Renate ist Kundenbetreuerin. Beide sind auch ihrerseits Kunden der eigenen Bank. Für die Rollenmitgliedschaft ergibt sich damit:

$$sr(\text{Mustermann}) = \{\text{Angestellter}, \text{Zweigstellenleiter}, \text{Kunde}\}$$

$$sr(\text{Renate}) = \{\text{Angestellter}, \text{Kundenbetreuer}, \text{Kunde}\}.$$
- Die Rechtevergabe gemäß Abbildung pr könnte nun beispielsweise wie folgt modelliert sein:

$$\begin{aligned} &\{Konto_sperren, Kreditrahmen_erhoechen\} \\ &\subseteq pr(\text{Zweigstellenleiter}), \\ &\{Einzahlung_auf_KundenKonto, Abheben_von_KundenKonto\} \\ &\subseteq pr(\text{Kunde}). \end{aligned}$$



Hierarchische, rollenbasierte Modelle

In praxisrelevanten Anwendungsumgebungen wie Unternehmen und Behörden werden Aufgaben und Zuständigkeiten innerhalb von Abteilungen oder Projekten häufig hierarchisch geordnet. Durch eine Erweiterung des RBAC-Modells um eine partielle Ordnung⁷ ≤ auf den Rollen lassen sich

*hierarchisches
RBAC*

⁷ Antisymmetrisch, transitiv und reflexiv.

auch hierarchische Berechtigungsstrukturen durch RBAC-Modelle erfassen. Über die Rollenhierarchie können Rechte vererbt werden. Seien R_1 und R_2 Rollen mit $R_1 \leq R_2$, dann besitzen die Mitglieder der Rolle R_2 mindestens auch alle Rechte der Mitglieder der Rolle R_1 . Diese Rechtevererbung kann jedoch auch problematisch sein, da dadurch ein Subjekt unter Umständen mehr Rechte erhält, als es zur Erfüllung seiner Aufgaben benötigt.

Beispiel 6.8 (Bankszenario (Forts.))

Gegeben sei erneut das Bankszenario von Seite 255 mit den Rollen $RL = \{\text{Zweigstellenleiter}, \text{Kassierer}, \text{Kundenbetreuer}, \text{Angestellter}, \text{Kassenprüfer}, \text{Kunde}\}$ festgelegt. Legt man die Rollenhierarchie kanonisch gemäß der Organisationsstruktur fest, so ergeben sich folgende direkte Abhängigkeiten zwischen den Rollen.

Hierarchie

Rollenhierarchie: $\text{Zweigstellenleiter} \geq \text{Kassenprüfer}$,
 $\text{Kassenprüfer} \geq \text{Kundenbetreuer}$,
 $\text{Kassenprüfer} \geq \text{Kassierer}$,
 $\text{Kundenbetreuer} \geq \text{Angestellter}$,
 $\text{Kassierer} \geq \text{Angestellter}$.

Zur vollständigen Darstellung der Rollenhierarchie ist die transitive Hülle der direkten Abhängigkeiten zu bilden.

Problem

Die Rollenhierarchie ist in Abbildung 6.2 veranschaulicht. Problematisch an der festgelegten Rollenhierarchie ist zum Beispiel, dass Mitglieder der Kassenprüferrolle auch alle Rechte der Rolle Kassierer erben (u.a. Einzahlen/Auszahlen von Beträgen von Kundenkonten), obwohl sie diese zur Durchführung ihrer Kontrollaufgaben nicht benötigen.

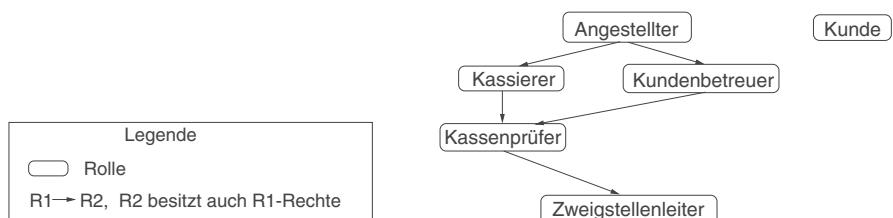


Abbildung 6.2: Rollenhierarchie des Bankbeispiels



Sicherheitseigenschaften

Im Folgenden werden Sicherheitseigenschaften von RBAC Modellen in Form von Invarianten angegeben. Diese Invarianten sind von einer Realisierung des Modells zu gewährleisten.

Notation:

Wir schreiben abkürzend $R_i \in session(s)$ genau dann, wenn es ein Paar $(s, R') \in session$ gibt, mit $R_i \in R'$.

Weiterhin schreiben wir $\{(R_i, R_j)\} \subseteq session(s)$, wenn es Paare $(s, R'), (s, R'') \in session$ gibt, mit $R_i \in R'$ und $R_j \in R''$.

1. Ein Subjekt darf nur in solchen Rollen aktiv sein, in denen es Mitglied ist: RBAC-Invarianten

$\forall s \in S$ muss gelten: $R_i \in session(s) \implies R_i \in sr(s)$.

2. Ein Subjekt nimmt nur Rechte wahr, die einer Rolle, in der es aktiv ist, zugeordnet sind.

Gegeben sei die Funktion $exec$ mit:

$$exec : S \times P \longrightarrow Boolean$$

$exec(s, p) = true : \iff s$ ist berechtigt, p auszuführen.

Dann muss gelten:

$$\forall s \in S : exec(s, p) \implies \exists R_i \in RL : R_i \in session(s) \wedge p \in pr(R_i).$$

3. In einem hierarchischen, rollenbasierten Modell gilt, dass ein Subjekt alle Rechte derjenigen Rollen erbt, die von seinen aktiven Rollen dominiert werden:

Gegeben sei die partielle Ordnung $\leq \subseteq RL \times RL$. Für alle $s \in S$ gilt:

$$exec(s, p) \implies \exists R_i \in RL : R_i \in session(s) \wedge (p \in pr(R_i) \vee \exists R_j \wedge R_j \leq R_i \wedge p \in pr(R_j)).$$

Bei einer Realisierung eines RBAC-Modells sind Realisierungsmechanismen festzulegen, so dass die zur Gewährleistung der Invarianten benötigten Informationen zur Verfügung stehen und die erforderlichen Kontrollen durchgeführt werden können.

Beschränkte Rollenmitgliedschaft

In einem RBAC-Modell ist es häufig sinnvoll bzw. notwendig, Regeln zur Beschränkung von Rollenmitgliedschaften zu formulieren.

Regeln zur statischen Aufgabentrennung (engl. *separation of duty*) betreffen den wechselseitigen Ausschluss von Rollenmitgliedschaften.

Definition 6.4 (Statische Aufgabentrennung)

Gegeben sei ein RBAC-Modell mit der Menge RL der Rollen. Die Relation SSD beschreibt die statische Aufgabentrennung und ist wie folgt festgelegt:

$$SSD \subseteq RL \times RL, \text{ mit}$$

$(R_i, R_j) \in SSD \iff$ die gleichzeitige Mitgliedschaft in den Rollen R_i und R_j ist ausgeschlossen.

statische Trennung

Invariante

Für das RBAC-Modell mit der Relation SSD muss gelten:

$$\forall R_i, R_j \in RL \text{ mit } R_i \neq R_j \quad \forall s \in S \text{ gilt :}$$

$(s \in \text{member}(R_i) \wedge s \in \text{member}(R_j)) \implies (R_i, R_j) \notin SSD$, wobei
 $\text{member}(R_i) := \{s \mid s \in S \wedge R_i \in sr(s)\}.$

□

Gemäß Definition 6.4 darf ein Subjekt s nur dann Mitglied zweier unterschiedlicher Rollen R_i und R_j sein, wenn sich die den Rollen assoziierten Aufgaben wechselseitig nicht ausschließen.

Beispiel 6.9 (Statische Aufgabentrennung)

Gegeben sei das in Beispiel 6.8 bereits eingeführte Bankszenario. Die Rollen des Kassenprüfers und Kassierers werden so verfeinert, dass sie sich auf die jeweilige Zweigstelle der Bank beziehen. Es gelte:

$R1 = \text{Kassenprüfer_von_Zweigstelle_A};$

$R2 = \text{Kassierer_in_Zweigstelle_A}.$

Aufgabentrennung

Da ein Kassenprüfer die Aufgabe hat, die Aktivitäten eines Kassierers zu überprüfen, sollte ein Subjekt nicht gleichzeitig Mitglied der Kassierer- und der Prüferrolle der gleichen Zweigstelle sein dürfen. Die Relation SSD enthält somit das Paar $(R1, R2)$, d.h. $(R1, R2) \in SSD$.

▲

Statische Aufgabentrennungen werden unabhängig von aktiven Sitzungen formuliert. Diese statische Festlegung ist jedoch für viele Anwendungsbereiche zu starr. Vielfach reicht es aus, die gleichzeitige Aktivität von Subjekten in unterschiedlichen Rollen zu untersagen, ohne jedoch die Rollenmitgliedschaft generell zu verbieten. Entsprechende Beschränkungen können durch Regeln zur dynamischen Aufgabentrennung formuliert werden.

Definition 6.5 (Dynamische Aufgabentrennung)

dynamische Trennung

Gegeben sei ein RBAC-Modell mit der Menge RL der Rollen. Die Relation DSD , die die dynamische Aufgabentrennung beschreibt, ist wie folgt festgelegt:

$$DSD \subseteq RL \times RL, \text{ mit}$$

$(R_i, R_j) \in DSD \iff \text{die gleichzeitige Aktivität eines Subjekts in der Rolle } R_i \text{ und in der Rolle } R_j \text{ ist unzulässig.}$

Für das RBAC-Modell mit der Relation DSD muss gelten:

Invariante

$$\forall R_i, R_j \in RL \quad \forall s \in S \text{ gilt} \\ \{(R_i, R_j)\} \subseteq session(s) \implies (R_i, R_j) \notin DSD.$$

□

Definition 6.5 besagt, dass ein Subjekt s nicht gleichzeitig in solchen Sitzungen aktiv sein darf, deren assoziierte Aufgaben wechselseitig ausgeschlossen durchzuführen sind.

Beispiel 6.10 (Dynamische Aufgabentrennung)

Gegeben sei erneut das in Beispiel 6.8 angegebene Bankszenario mit der dort festgelegten Menge RL von Rollen. Betrachten wir die beiden Rollen $R3 = \text{Kundenbetreuer}$ und $R4 = \text{Kunde}$. Da ein Bankangestellter, der Kundenberater ist, auch selber ein Konto auf der gleichen Bank, bei der er angestellt ist, besitzen und in der Rolle des Bankkunden aktiv sein darf, ist eine statische Aufgabentrennung hier unzweckmäßig. Andererseits sollten Manipulationsmöglichkeiten a priori beschränkt werden, so dass gefordert wird, dass ein Angestellter nicht gleichzeitig in der Rolle des Kundenbetreuers und des Kunden aktiv sein darf. Damit gilt: $(R3, R4) \in DSD$.

▲

dynamische
Aufgabentrennung

Klassifikation und Bewertung

Mit einem RBAC-Modell sind Objekte anwendungsspezifisch modellierbar und die Berechtigungen zum Zugriff auf die Objekte werden aufgabenorientiert vergeben. Mit einer RBAC-Strategie lassen sich die Nachteile klassischer, objektbezogener Sicherheitsstrategien vermeiden, da sich die Berechtigungsprofile von Rollen nur selten ändern. Auf diese Weise lassen sich die Probleme der mangelnden Skalierbarkeit objektbezogener Strategien bei dynamisch sich ändernden Subjektmengen stark reduzieren. RBAC-Modelle eignen sich gut für einen Einsatz in verteilten Systemen. Die Rollenzuordnung bzw. deren Entzug ist die Aufgabe des Systemadministrators.

Aufgaben-
orientierung

Das Modell trifft keine a priori Festlegungen hinsichtlich der Menge der zu vergebenden Zugriffsrechte, die universell oder objektspezifisch modelliert werden können. Die definierten Invarianten, einschließlich der Regeln zur Formulierung von statischen und dynamischen Beschränkungen von Rollenaktivitäten, ermöglichen eine Rechtevergabe gemäß des need-to-know Prinzips und unterstützen eine systematische Aufgabenteilung. Die eingebrachten Regeln zur Erfassung unzulässiger Abhängigkeiten zwischen Rollen sowie die Möglichkeit, Rollenhierarchien zu modellieren, sind unmittelbar auf existierende Organisations- und Verwaltungsstrukturen in Unternehmen

skalierbar

Einsatzbereich

und Behörden übertragbar, so dass in diesem Umfeld RBAC-Modelle sehr gut für große Klassen von Anwendungssystemen geeignet sind.

Die Entwicklungen rund um RBAC werden weiter fortgesetzt. Sie zielen häufig darauf ab, die Ausdrucksstärke des Modellierungsansatz zu erhöhen, indem u.a. Konzepte zur Parametrisierung von Rollen oder auch zur Kontext-basierten Zugriffskontrolle in das Modell integriert oder aber auch die Möglichkeiten zur Spezifikation von Zugriffsbeschränkungen in Form von Constraints weiter verfeinert werden.

6.2.3 Chinese-Wall Modell

Anwendungsbereich

Das Chinese-Wall Sicherheitsmodell [31], auch bekannt als Brewer-Nash Modell, wurde entwickelt, um die unzulässige Ausnutzung von Insiderwissen bei der Abwicklung von Bank- oder Börsentransaktionen oder bei der Beratung unterschiedlicher Unternehmen zu verhindern. So soll zum Beispiel durch das IT-System verhindert werden, dass ein Unternehmensberater, der Insiderinformationen über ein Unternehmen besitzt, diese Informationen verwendet, um einem Konkurrenzunternehmen Ratschläge zu erteilen. Die grundlegende Idee des Chinese-Wall Modells basiert darauf, dass die zukünftigen Zugriffsmöglichkeiten eines Subjekts durch die Zugriffe, die es in der Vergangenheit bereits durchgeführt hat, beschränkt werden. Das heißt, dass die Zulässigkeit von Zugriffen auch von der Zugriffshistorie abhängt.

Zugriffsrechte

Dem Chinese-Wall Modell liegt ein Zugriffsmatrix-Modell zugrunde, wobei die Menge der Zugriffsrechte explizit durch die universellen Rechte

$$\mathcal{R} = \{read, write, execute\}$$

Objektbaum

festgelegt ist. Die Menge \mathcal{S} der Subjekte ist durch die agierenden Personen, also die Berater, definiert. Die zu schützenden Objekte des Systems werden als Baum strukturiert. Die Blätter des Baumes sind die Objekte, die in den unterschiedlichen Unternehmen verwaltet werden. Diese Zuordnung wird durch die nächste Ebene des Baumes repräsentiert, die die interessierenden Unternehmen modelliert. Unternehmen, die in Konkurrenz zueinander stehen, werden in Interessenskonfliktklassen zusammengefasst; diese bilden die dritte Ebene des Baumes.

Sicherheitsmarken

Die Zugehörigkeit eines Objektes o zu einem Unternehmen sowie einer Interessenskonfliktklasse wird durch zwei Sicherheitsmarken $y(o), x(o)$ modelliert. Die Marke $y(o)$ beschreibt das Unternehmen, zu dem das Objekt o gehört, und $x(o)$ beschreibt die Interessenskonfliktklasse dieses Unternehmens. Für öffentlich zugängliche Information, die allen Subjekten unbeschränkt zugänglich sein kann, wird eine spezielle Markierung y_0 und eine spezielle Interessenskonfliktklasse x_0 mit $x_0 = \{y_0\}$ eingeführt. x_0 wird auch als interessensfreie Information bezeichnet.

Beispiel 6.11 (Objektbaum)

Gegeben seien die Interessenskonfliktklassen Ölgesellschaften und Banken. An Unternehmen betrachten wir die Firmen Aral und Shell sowie die Deutsche und die Dresdner Bank. Der Objektbaum für dieses Szenario ist in Abbildung 6.3 angegeben. Das Objekt $o1$ gehört zur Firma Aral und liegt in der Interessenskonfliktklasse aller Ölgesellschaften, in der z.B. auch das Objekt $o3$, das seinerseits zur Firma Shell gehört, liegt. Demgegenüber steht Objekt $o2$ in keinem Interessenskonflikt zu den Objekten $o1$ oder $o3$. Über die Einordnung des Objekts $o1$ in den Baum, sind dessen Sicherheitsklassen mit $y(o1) = \text{Aral}$ und $x(o1) = \text{Ölgesellschaft}$ festgelegt.

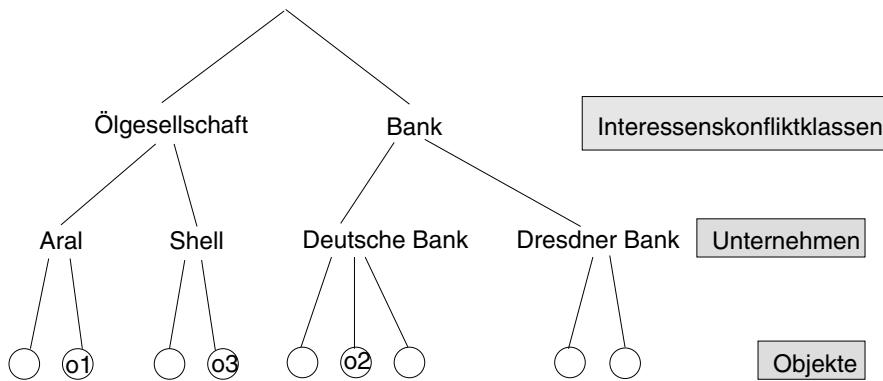


Abbildung 6.3: Objektbaum im Chinese-Wall Modell

Schutzzustand

Zusätzlich zu den benutzerbestimmbar zu vergebenden Rechten gemäß einer Zugriffsmatrix M_t wird der Schutzzustand eines Chinese-Wall Modells durch eine $|\mathcal{S}| \times |\mathcal{O}|$ -Matrix N_t bestimmt, die festhält, welche Subjekte auf welche Objekte bis zu dem betrachteten Zeitpunkt t bereits zugegriffen haben.

Schutzzustand

Definition 6.6 (Zugriffshistorie)

Gegeben sei ein System mit der Menge der Subjekte \mathcal{S} , der Menge der Objekte \mathcal{O} und der Menge der Rechte \mathcal{R} . Die Zugriffshistorie von Subjekten $s \in \mathcal{S}$ wird durch die Matrix N_t beschrieben, mit

$$N_t : \mathcal{S} \times \mathcal{O} \longrightarrow 2^{\mathcal{R}}.$$

Es gilt $N_t(s, o) = \{r_1, \dots, r_n\}$ genau dann, wenn es Zeitpunkte $t' < t$ gibt, zu denen das Subjekt s gemäß der Berechtigungen r_1, \dots, r_n auf das Objekt o zugegriffen hat.

Zugriffshistorie

Der Zugriff auf ein Objekt wird durch zwei systembestimmte Regeln beschränkt. Die Leseregel (Regel 1) reglementiert den Lesezugriff, während durch die Schreibregel (Regel 2) Beschränkungen für modifizierende Zugriffe festgelegt sind.

Lese-Regel

Regel 1:

Ein z-Zugriff, mit $z \in \{read, execute\}$ auf ein Objekt $o \in \mathcal{O}$ ist für ein Subjekt $s \in \mathcal{S}$ zum Zeitpunkt t zulässig genau dann, wenn gilt:

$$\begin{aligned} z \in M_t(s, o) \wedge \forall o' \in \mathcal{O} : \\ N_t(s, o') \neq \emptyset \implies (y(o') = y(o) \vee x(o') \neq x(o) \vee y(o') = y_0). \end{aligned}$$

Die Lese-Zugriffsbeschränkung besagt, dass ein Subjekt s zum Zeitpunkt t nur dann lesend (bzw. ausführend) auf ein Objekt o zugreifen darf, wenn das Subjekt im Zustand t das entsprechende Leserecht (bzw. Execute-Recht) besitzt, es bis dahin auf nur auf Objekte o' zugegriffen hat, die zum gleichen Unternehmen gehören (d.h. $y(o') = y(o)$), oder die Unternehmen einer anderen Interessenskonfliktklasse als der des Objekts o angehören (d.h. $x(o') \neq x(o)$), oder öffentlich zugreifbare Information beinhalten (d.h. $y(o') = y_0$).

Mauerbau

Mit dieser Zugriffsregel ist gewährleistet, dass nach dem Zugriff auf ein Objekt o durch ein Subjekt s eine s -spezifische Mauer um alle diejenigen Objekte o' errichtet wird, die zu anderen Unternehmen der gleichen Interessenskonfliktklasse gehören, zu der auch das Objekt o gehört, und auf die damit das Subjekt s nicht mehr zugreifen kann. Diese Eigenschaften kann für Chinese-Wall Modelle formal nachgewiesen werden.

Lese-Beschränkung

Satz

In einem gemäß des Chinese-Wall Modells modellierten System gilt für alle Subjekte $s \in \mathcal{S}$ folgende Aussage: Hat das Subjekt s auf ein Objekt o zugegriffen, so darf s höchstens noch auf Objekte o' zugreifen, für die gilt:

$$y(o') = y(o) \vee x(o') \neq x(o) \vee y(o') = y_0.$$

Beweis:

Annahme: Für ein Subjekt s und Objekte $o1, o2$ gilt zum Zeitpunkt t :

$$\begin{aligned} N_t(s, o1) \neq \emptyset \wedge N_t(s, o2) \neq \emptyset \wedge y(o1) \neq y(o2) \wedge \\ x(o1) = x(o2) \wedge y(o1) \neq y_0 \wedge y(o2) \neq y_0. \end{aligned}$$

Ohne Beschränkung der Allgemeinheit sei der erste Zugriff von Subjekt s auf Objekt $o1$ zum Zeitpunkt $t' \leq t$ und der erste Zugriff von s auf Objekt $o2$ sei zum Zeitpunkt t'' mit $t' \leq t'' \leq t$ erfolgt. Somit gilt: $N_{t''}(s, o1) \neq \emptyset$.

Nach Regel 1 muss für einen zulässigen o_2 -Zugriff durch s aber gelten:

$$y(o1) = y(o2) \vee x(o1) \neq x(o2).$$

Zusammen mit der Annahme gilt also:

$$(x(o1) = x(o2) \wedge x(o1) \neq x(o2) \wedge y(o1) = y(o2)) \vee \\ (y(o1) = y(o2) \wedge y(o1) \neq y(o2) \wedge x(o1) = x(o2)).$$

Diese Aussage ist immer falsch, womit die Annahme zum Widerspruch geführt worden ist.

•

Beispiel 6.12 (Subjektspezifische Mauerbildung)

Betrachten wir erneut das Beispiel aus Abbildung 6.3. Das Subjekt $s1$ habe zum Zeitpunkt t' auf das Objekt $o1$ zugegriffen. Damit gilt für alle Zeitpunkte $t > t'$, dass Subjekt $s1$ auf kein Objekt o' mit $y(o') = Shell$ zugreifen darf, da $y(o') \neq y(o1) \wedge x(o') = x(o1) = Ölgesellschaft$.

Ein Zugriff auf das Objekt $o2$ mit $x(o2) = Bank$ ist jedoch weiterhin zulässig. Ein entsprechender Zugriff erfolge zum Zeitpunkt $t'' > t'$. Die $s1$ -spezifische Mauer nach dem Zeitpunkt t'' ergibt sich wie in Abbildung 6.4 angegeben. Das bedeutet, dass das Subjekt $s1$ jetzt nur noch auf Objekte der Äste *Aral* und *Deutsche Bank* zugreifen darf.

▲

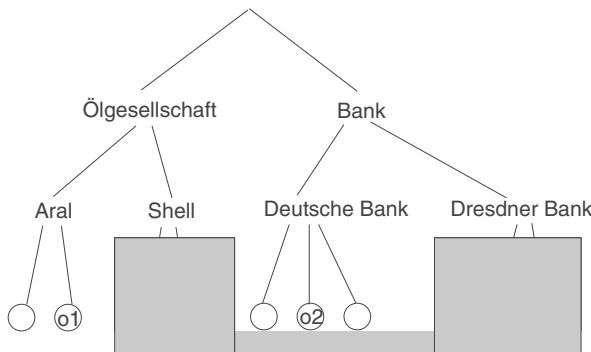


Abbildung 6.4: $s1$ -spezifische Mauer nach dem Zugriff auf $o1$ und $o2$

Mit Regel 1 ergeben sich also systembestimmte Zugriffsbeschränkungen. Diese reichen jedoch nicht aus, um Informationsflüsse zu verhindern, die zur unzulässigen Ausnutzung von Insiderinformationen führen könnten.

Beispiel 6.13 (Unerwünschter Informationsfluss)

Man betrachte erneut das in Abbildung 6.3 skizzierte Szenario mit zwei Subjekten $s1$ und $s2$. Falls Subjekt $s1$ zunächst lesend auf das Objekt $o1$, dann schreibend auf das Bank-Objekt $o2$ zugreift, woraufhin das Subjekt $s2$ die Informationen aus Objekt $o2$ liest und in das Objekt $o3$ schreibt, so tritt ein Informationsfluss vom Objekt $o1$ zum Objekt $o3$ auf. D.h. es tritt ein Informationsfluss zwischen Objekten auf, die zu Konkurrenzfirmen gehören. Für einen Schreibzugriff sind somit weitere Restriktionen festzulegen.



Schreibregel

Regel 2:

Ein Schreibzugriff auf ein Objekt $o \in \mathcal{O}$ durch ein Subjekt $s \in \mathcal{S}$ ist zum Zeitpunkt t zulässig genau dann, wenn gilt:

$$\begin{aligned} write \in M_t(s, o) \wedge \forall o' \in \mathcal{O} : \\ read \in N_t(s, o') \implies (y(o') = y(o) \vee y(o') = y_o). \end{aligned}$$

Die Schreibzugriffsbeschränkung legt fest, dass ein Schreibzugriff durch ein Subjekt s zum Zeitpunkt t auf das Objekt o zulässig ist, wenn s das Schreibrecht besitzt und bis zum Zeitpunkt t nur Lesezugriffe auf solche Objekte o' hatte, die entweder zum gleichen Unternehmen wie o gehören oder die nur unklassifizierte, also frei zugängliche Informationen beinhalteten. Die Schreibregel verhindert einen Informationsfluss zwischen Objekten, die zu unterschiedlichen Unternehmen, aber zu der gleichen Interessenskonfliktklasse gehören.

Beispiel 6.14 (Chinese-Wall Beschränkung)

Betrachten wir erneut das in Abbildung 6.3 skizzierte Szenario sowie zwei Subjekte $s1$ und $s2$. Nach einem Lesezugriff auf das Objekt $o1$ durch das Subjekt $s1$ gilt, dass ein Schreibzugriff auf das Objekt $o2$ unzulässig ist, da $y(o2) \neq y(o1)$, so dass der in Beispiel 6.13 aufgetretene, unzulässige Informationsfluss nicht mehr auftreten kann.



Klassifikation und Bewertung

Obwohl die formale Beschreibung des Chinese-Wall Modells in [31] von grobkörnigen Objekten wie Dateien sowie von Benutzern als grobkörnigen Subjekten ausgeht, ist diese Festlegung nicht notwendig vorgeschrieben, d.h. mit dem Modell lassen sich auch Systeme mit feinkörniger festgelegten Einheiten modellieren. Im Gegensatz zu den Freiheitsgraden, die der Entwickler bei der Wahl der Objekte besitzt, wird die Festlegung von einfachen, universellen Zugriffsrechten vorgeschrieben, so dass bei der Wahrnehmung von

Zugriffsrechten Benutzer oder Prozesse, die in deren Auftrag aktiv sind, die zu schützenden Objekte ohne weitere Beschränkungen manipulieren können.

Systeme, die mit dem Chinese-Wall Modell beschrieben werden, legen eine benutzerbestimmbare Strategie fest. Diese wird durch die einfachen Zugriffsregeln um systembestimmte Festlegungen erweitert. Mit dem Modell ist es jedoch nicht möglich, über die sehr restriktiven, systembestimmten Regeln hinausgehend, erlaubte und verbotene Informationskanäle individuell, abhängig von den Anwendungsanforderungen zu modellieren. Das Modell ist geeignet für Anwendungen, die keine strengen Datenintegritätsanforderungen haben und die keine über die restriktiven Beschränkungen hinausgehenden Vertraulichkeitsanforderungen stellen.

systembestimmte
Regeln

6.2.4 Bell-LaPadula Modell

Das Bell-LaPadula Modell [14, 13, 15] gilt als das erste vollständig formalisierte Sicherheitsmodell. Dem Bell-LaPadula Modell liegt ein dynamisches Zugriffsmatrix-Modell zugrunde, wobei die Zugriffsrechte durch die Menge der universellen Rechte $\{read-only, append, execute, read-write, control\}$ vorgeschrieben sind. Das *read-only*-Recht erlaubt einen reinen Lesezugriff, das *append*-Recht berechtigt dazu, Daten an ein vorhandenes Objekt anzufügen, das *read-write*-Recht erlaubt den Lese- oder Schreibzugriff und das *control*-Recht ermöglicht die Rechtewitergabe bzw. -rücknahme. Das *execute*-Recht berechtigt zur Ausführung einer ausführbaren Datei.

universelle Rechte

Als Erweiterung des zugrunde liegenden Zugriffsmatrix-Modells wird eine geordnete Menge von Sicherheitsklassen SC eingeführt, um den zu verarbeitenden Informationen sowie den agierenden Subjekten unterschiedliche Vertraulichkeitsstufen zuzuordnen. Ein Element $X \in SC$ wird durch ein Paar $X = (A, B)$ beschrieben, wobei A eine Sicherheitsmarke und B eine Menge von Sicherheitskategorien (engl. *compartment*) ist.

Sicherheitsklassen

Jedem Subjekt s wird eine Sicherheitsklasse, die so genannte Clearance $sc(s) \in SC$, und jedem Objekt o wird eine Sicherheitsklassifikation, die so genannte Classification $sc(o) \in SC$, zugeordnet. Die Clearance $sc(s)$ eines Subjekts s ist die maximale Sicherheitsstufe, die s einnehmen darf. Bei der Anmeldung (login) muss ein Subjekt seine aktuelle Clearance $sc_{akt}(s)$ angeben, wobei gilt: $sc_{akt}(s) \leq sc(s)$, d.h. die aktuelle Clearance darf die maximale Clearance nicht überschreiten

Clearance

Classification

Auf SC wird eine partielle Ordnung⁸, (SC, \leq) , festgelegt.

partielle Ordnung

⁸ Antisymmetrisch, transitiv und reflexiv.

Für alle $X, Y \in SC$, mit $X = (A, B), Y = (A', B')$ gilt:

$$X \leq Y \iff A \leq A' \wedge B \subseteq B'.$$

Beispiel 6.15 (Sicherheitsklassen)

Gegeben sei ein Krankenhauszenario, in dem die Patientenakten die zu schützenden Objekte sind. Die Menge der Sicherheitsmarken sei durch

$$\{\text{unklassifiziert}, \text{vertraulich}, \text{geheim}, \text{streng geheim}\}$$

mit der Ordnung

$$\text{unklassifiziert} \leq \text{vertraulich} \leq \text{geheim} \leq \text{streng geheim}$$

Kategorien festgelegt. Die Menge der Sicherheitskategorien sei durch

$$\{\text{Arzt}, \text{Schwester}, \text{Patient}, \text{Verwaltung}\}$$

definiert. Die Menge von Sicherheitsklassen SC könnte zum Beispiel folgende Elemente umfassen:

$$SC = \{ (\text{geheim}, \emptyset), (\text{vertraulich}, \emptyset), \\ (\text{vertraulich}, \{\text{Arzt}, \text{Schwester}\}), \\ (\text{vertraulich}, \{\text{Schwester}\}), \dots \}.$$

Aufgrund der festgelegten Ordnungsrelation gilt u.a.:

$$(\text{geheim}, \emptyset) \geq (\text{vertraulich}, \emptyset) \\ (\text{vertr.}, \{\text{Arzt}, \text{Schwester}\}) \geq (\text{vertr.}, \{\text{Schwester}\}).$$



Systembestimmte Zugriffsbeschränkungen

Der Zugriff auf Objekte wird durch zwei systembestimmte Regeln, die Simple-Security und die *-Eigenschaft, beschränkt. Die Simple-Security-Regel, auch bekannt als no-read-up Regel, besagt, dass ein Lese- oder Executezugriff auf ein Objekt o durch ein Subjekt s nur dann zulässig ist, wenn s das entsprechende Zugriffsrecht r besitzt und die Objektklassifikation kleiner oder gleich der Subjekt-Clearance ist. D.h. es muss gelten: $r \in M_t(s, o) \wedge sc(s) \geq sc(o)$.

no-read-up Die *-Eigenschaft, auch bekannt als no-write-down Regel, besagt, dass ein append-Zugriff auf ein Objekt o durch ein Subjekt s nur zulässig ist, wenn die Objektklassifikation mindestens so hoch ist, wie die Clearance des Subjekts, also $\text{append} \in M_t(s, o) \wedge sc(s) \leq sc(o)$, und dass ein Lese-Schreib-Zugriff auf ein Objekt o nur zulässig ist, wenn die Objektklassifikation gleich der Clearance des Subjekts ist, d.h. $\text{read-write} \in M_t(s, o) \wedge sc(s) = sc(o)$.

no-write-down

Durch die beiden systembestimmten Regeln sind einfach zu überprüfende Bedingungen formuliert. Diese gewährleisten, dass Informationsflüsse höchstens von unten nach oben entlang der partiellen Ordnung \leq oder innerhalb einer Sicherheitsklasse auftreten können. Abbildung 6.5 veranschaulicht die zulässigen Flüsse für ein System mit den linear geordneten Sicherheitsmarken *unklassifiziert*, *vertraulich*, *geheim*, und *streng geheim*.

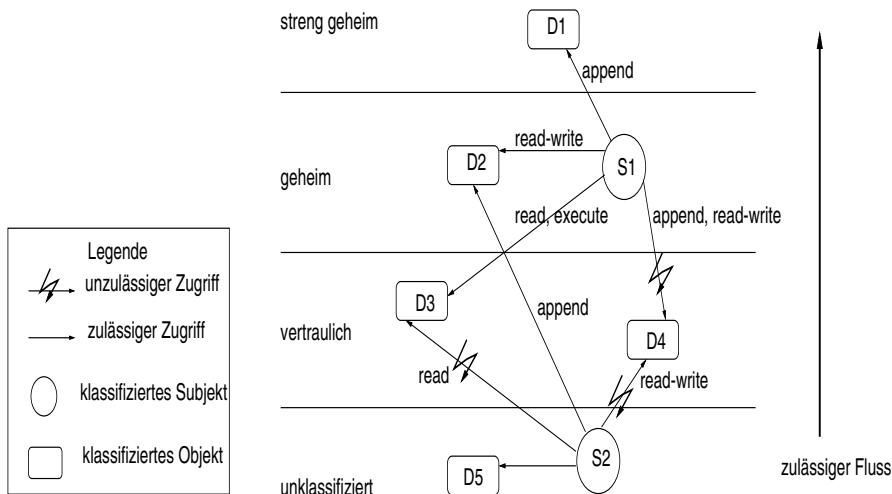


Abbildung 6.5: Zulässige Flüsse im Bell-LaPadula Modell

Beispiel 6.16 (Zulässige bzw. unzulässige Flüsse)

Gegeben sei erneut das in Beispiel 6.15 eingeführte Krankenhauszenario mit der dort festgelegten, geordneten Menge von Sicherheitsmarken und Sicherheitskategorien.

Für $s \in \mathcal{S}_t$ gelte: $sc(s) = (\text{vertraulich}, \{\text{Arzt}\})$.

Lese- bzw. Execute-Zugriffe durch s auf Objekte $o \in \mathcal{O}_t$ mit z. B.

$sc(o) = (\text{vertraulich}, \emptyset)$ oder $sc(o) = (\text{vertraulich}, \{\text{Arzt}\})$ sind zulässig, während zum Beispiel der Zugriff auf ein Objekt o' mit

$sc(o') = (\text{geheim}, \{\text{Arzt}\})$ oder $sc(o') = (\text{vertraulich}, \{\text{Patient}\})$

unzulässig ist.

Append-Zugriffe sind für s auf Objekte o mit z.B.

$sc(o) = (\text{vertraulich}, \{\text{Arzt}\})$ oder $sc(o) = (\text{geheim}, \{\text{Arzt}\})$

zulässig und Lese-Schreib-Zugriffe sind auf Objekte o mit

$sc(o) = (\text{vertraulich}, \{\text{Arzt}\})$ zulässig.



Die Praxis hat gezeigt, dass in Folge der Workflows in Unternehmen und Behörden, klassifizierte Objekte sukzessive in immer höhere Sicherheitsklassen eingestuft werden. Sie werden häufig z.B. durch einen Sachbearbeiter auf einer niedrigen Stufe erstellt. Bei einer Weiterleitung an den Vorgesetzten, der eine höhere Einstufung besitzt, müssen diese Objekte auch eine höhere Einstufung erhalten, damit auf sie in dieser Stufe modifizierend zugegriffen werden kann. Um Informationen dynamisch neu zu klassifizieren und insbesondere um Informationen wieder niedriger einzustufen, wurde das Konzept der vertrauenswürdigen Prozesse (engl. *trusted process*) eingeführt. Ein vertrauenswürdiger Prozess ist ein Subjekt, bei dem man davon ausgeht, dass es keine Aktionen durchführt, die die Sicherheitseigenschaften des Systems in Frage stellen. Derartige Subjekte, meist handelt es sich dabei um Betriebssystemprozesse, unterliegen deshalb nicht den erklärten Zugriffskontrollen.

vertrauenswürdiger Prozess

Betriebssystem-Unterstützung

Das Bell-LaPadula-Modell wird in heutigen Betriebssystemen unter der Bezeichnung Multi-Level Security (MLS) umgesetzt. Beispiele sind SELinux⁹ bzw. Red Hat Enterprise Linux 5 OS, das Mainframe Betriebssystem z/OS 1.8 von IBM, das 2007 vom BSI nach EAL4+ zertifiziert wurde¹⁰ oder aber auch das Betriebssystem Solaris 10 11/06 OS von Sun mit den integrierten Trusted Extensions, wodurch Multi-level Security unterstützt wird.

Im Folgenden geben wir zur Veranschaulichung eine mögliche Umsetzung des Bell-LaPadula-Modells in dem Betriebssystem Unix MLS an.

Beispiel 6.17 (Unix MLS)

Multi Level Secure

Sicherheitsklassifikation

Zugriffskontrolle

Das Unix System MLS (Multi-Level Security) ist eine Unix-Erweiterung um Sicherheitsklassen und -kategorien, die vom Systemadministrator festgelegt werden. Die Subjekte in Unix MLS sind Prozesse, die im Auftrag von Benutzern aktiv sind. Die Objekte sind u.a. Dateien, Verzeichnisse, inodes¹¹, Signale und Prozesse. Jedem Objekt wird eine Sicherheitsklassifikation bestehend aus einer Sicherheitsmarke und einer Menge von Kategorien zugeordnet. Jedem Benutzer des Systems ist eine Clearance zugeordnet, die auf die Prozesse, die im Auftrag eines spezifischen Benutzers im System aktiv sind, vererbt wird. Dazu wurden zum einen die login-Funktion und zum anderen die Objekt- und Prozessbeschreibungen des Unix Systems erweitert, so dass beim Login die Clearance des Benutzers angegeben werden kann.

Bei jedem Zugriff auf ein geschütztes Objekt sind die Simple-Security Ei-

⁹ Vgl. <http://www.nsa.gov/research/selinux/>

¹⁰ siehe https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Zertifizierung/7148_pdf.__blob=publicationFile

¹¹ Datei-Deskriptoren im Unix-Betriebssystemkern (vgl. Kapitel 12.4)

genschaft sowie die *-Eigenschaft des Bell-LaPadula Modells einzuhalten. Die Zugriffskontrolle überprüft zunächst, ob das zugreifende Subjekt das benötigte Zugriffsrecht besitzt. Im zweiten Schritt wird bei der Ausführung der in Tabelle 6.3 angegebenen Operationen überprüft, ob das ausführende Subjekt die notwendige Clearance besitzt. Die Einträge in Tabelle 6.3

Zugriffsoperation	Beschränkung für Subjekt s und Objekt o
read(file)	$sc(s) \geq sc(o)$
exec(file)	$sc(s) \geq sc(o)$
write(file)	$sc(s) = sc(o)$
overwrite(file)	$sc(s) = sc(o)$
append(file)	$sc(s) \leq sc(o)$
stat(i-node)	$sc(s) \geq sc(o)$
change(i-node)	$sc(s) = sc(o)$
read(directory)	$sc(s) \geq sc(o)$
search(directory)	$sc(s) \geq sc(o)$
link(directory)	$sc(s) = sc(o)$
create(directory)	$sc(s) = sc(o)$
unlink(directory)	$sc(s) = sc(o)$
read(signal/ipc)	$sc(s) \geq sc(o)$
write(signal/ipc)	$sc(s) = sc(o)$
kill(signal/ipc)	$sc(s) = sc(o)$

Tabelle 6.3: Bell-LaPadula-Regeln für Unix MLS-Kommandos

sind wie folgt zu lesen: Subjekt s darf die angegebene Operation auf dem Objekt o ausführen, wenn die Clearance des Subjekts die angegebene Bedingung erfüllt. So ist für Zugriffe, die nur lesenden Charakter haben, wie das Durchsuchen eines Verzeichnisses `search(directory)`, gefordert, dass das zugreifende Subjekt mindestens die Sicherheitseinstufung des Objekts besitzt. Auf diese Weise wird gewährleistet, dass keine hoch klassifizierte Information zu einem niedriger eingestuften Subjekt fließen kann (Simple Security Property).



Das Bell-LaPadula Modell weist einige gravierende Mängel auf, von denen zwei nachfolgend angesprochen werden. Aufgrund dieser Mängel wird das Modell zumeist mit Erweiterungen und Anpassungen in heutigen Betriebssystemen (s.o.) verwendet.

Mängel

(1) Beschränkte Ausdrucksfähigkeit

Es existieren Klassen von Anwendungsszenarien, deren informelle Sicherheitsanforderungen die intendierte Informationsfluss-Strategie erfüllen, die aber dennoch nicht modellierbar sind.

Beispiel 6.18 (Nicht modellierbares Szenario)

restriktive
Modellierung

Gegeben sei die geordnete Menge von Sicherheitsmarken wie in Beispiel 6.15 bereits angegeben. Es sei ein Server zu modellieren, der eine Operation *op* anbietet, die lesend auf eine Datei *D1* zugreift, die die Sicherheitsklassifikation *geheim* besitzt. Des Weiteren soll im Zuge der Ausführung von *op* der Server nach diesem Lese-Zugriff auf eine Datei *D2*, die die Sicherheitsklassifikation *streng* *geheim* besitzt, schreibend zugreifen.

```
Procedure op(..)
Begin
    D1.read(..);           Zugriff auf Datei D1
    ...
    D2.write(..);          Zugriff auf Datei D2
End op;
```

Die Operation *op* gewährleiste, dass bei ihrer Ausführung keine Informationen über die gelesenen Daten der Datei *D1* zurück zu dem aufrufenden Server fließen. Client-Prozesse, die die Sicherheitsklassifikation *vertraulich* besitzen, sollen die Berechtigung zur Ausführung von *op* erhalten. Die Semantik von *op* stellt sicher, dass es bei dessen Ausführung höchstens zu einem Informationsfluss von der Datei *D1* zur Datei *D2* kommt, also von der Sicherheitsklasse *geheim* zur Klasse *streng geheim*. Die gestellten Anforderungen sind somit konform zum Geist der Bell-LaPadula Strategie. Das skizzierte System lässt sich jedoch nicht mit dem Bell-LaPadula Modell beschreiben, da eine Funktionskomposition, wie sie durch die Operation *op* formuliert ist, nicht erfassbar ist. Das heißt, dass die betrachtete Operation in zwei sequentielle Operationen, nämlich einen Lese-Zugriff auf eine mit der Sicherheitsklasse *geheim* markierte Datei *D1*, gefolgt von einem Schreib-Zugriff auf eine mit *streng geheim* markierte Datei *D2*, aufzuteilen ist. Der Aufruf der Lese-Operation durch einen als *vertraulich* eingestuften Client verstößt jedoch gegen die Simple-Security Property. Der Client darf also die Zugriffsberechtigung nicht erhalten, so dass das Szenario nicht modellierbar ist.



(2) Problem des blinden Schreibens

Durch die systembestimmten Regeln ist es möglich, dass ein Subjekt schreibend auf ein höher eingestuftes Objekt zugreifen, aber anschließend die von

blindes Schreiben

ihm verursachten Veränderungen nicht lesen darf, da sonst die no-read-up Bedingung verletzt wäre. Dieses blinde Schreiben ist im Hinblick auf Integritätsanforderungen sehr problematisch, da ein Subjekt ein Objekt beliebig (z.B. unabsichtlich fehlerhaft) manipulieren kann.

Klassifikation und Bewertung

Das Bell-LaPadula Modell schreibt keine Granularität für Objekte oder Subjekte vor. Die Zugriffsrechte sind als universelle Rechte festgelegt und die benutzerbestimmbare Strategie wird durch die angegebenen, systembestimmten Regeln erweitert. Die Rechtefestlegungen, kombiniert mit den restriktiven, systembestimmten Festlegungen des Modells, schränken das Spektrum der IT-Systeme, die mit diesem Modell beschrieben werden können, jedoch sehr ein. So wird beispielsweise eine Vielzahl von Zugriffen auf Objekte durch die systembestimmten Festlegungen verboten, obwohl durch einen entsprechenden Zugriff keine Verletzung der intendierten Strategie auftreten würde. Andererseits sind die MAC-Regeln einfach zu überprüfen und effizient zu implementieren. Dies findet seinen Niederschlag in der relativ großen Anzahl kommerzieller Betriebssysteme, die die Bell-LaPadula Strategie unter der Bezeichnung MLS (Multi-level Security) realisieren (u.a. SE-Linux, z/OS 1.8 von IBM oder auch Solaris 10 11/06 OS von Sun).

restriktiv

Wegen der mit dem Modell festgelegten, universellen Rechte und der einfachen Zugriffsbeschränkungsregeln ist das Modell nicht geeignet, hohe Anforderungen an die Datenintegrität zu modellieren. Problematisch ist, dass niedrig eingestufte Subjekte höher eingestufte Objekte unbeschränkt schreibend manipulieren (blides Schreiben „nach oben“) dürfen. Zur Behebung der Integritätsprobleme wurde das Biba-Modell [25] entwickelt. Analog zum Bell-LaPadula Modell erfolgt eine Einteilung von Subjekten und Objekten in hierarchisch geordnete Integritätsklassen, wobei die Zugriffsregeln entgegengesetzt zum Bell-LaPadula Modell festgelegt werden (z.B. Lesen ist nur „nach oben“ erlaubt). Die systemglobalen Regeln verhindern eine Verletzung der Integrität von Objekten durch Modifikationen, die von niedrig eingestuften Subjekten ausgelöst werden. Die Regeln sind jedoch ebenfalls sehr restriktiv und wenig flexibel, so dass das Modell nur stark eingeschränkt einsetzbar ist.

wenig Integrität

Biba-Modell

Das Bell-LaPadula Modell ist geeignet für Anwendungsprobleme mit geringen Anforderungen an eine differenzierte, benutzerbestimmbare Zugriffskontrolle bei gleichzeitig hohen Vertraulichkeitsanforderungen, die sich mit den restriktiven, systemglobalen Regeln erfassen lassen. Dies gilt insbesondere für IT-Systeme, die hierarchisch geordnete Informationen verarbeiten und deren Benutzerstruktur durch das organisatorische Umfeld ebenfalls hierarchisch strukturiert werden kann.

6.3 Informationsflussmodelle

Bei den Informationsflussmodellen steht die Informationssicherheit im Mittelpunkt. Die Modelle beschreiben zulässige und unzulässige Informationskanäle zwischen Subjekten.

6.3.1 Verbands-Modell

Informationsflusstrategie

Mit dem Verbands-Modell [50] wird der im Bell-LaPadula Modell verfolgte Ansatz verallgemeinert, um Informationsflüsse zwischen Datenvariablen eines Programms zu beschreiben. Verbands-Modelle sind dadurch charakterisiert, dass Beschränkungen für Informationsflüsse unabhängig von den Objekten, die die Informationen repräsentieren, festgelegt werden. Das heißt, dass nicht Objektzugriffe beschränkt werden, sondern dass der Umgang mit der Information, die durch Objekte repräsentiert ist, reglementiert wird, indem zulässige und unzulässige Informationskanäle festgelegt werden.

Definition 6.7 (Verbandseigenschaft)

Verband

Ein Verband ist durch das Tupel $(SC, \leq, \oplus, \otimes)$ definiert, wobei

- SC eine endliche Menge von Sicherheitsklassen beschreibt.
- Die Flussrelation \leq definiert eine partielle Ordnung auf SC und
- die Operatoren \oplus und \otimes definieren für je zwei Sicherheitsklassen die kleinste obere bzw. die größte untere Grenze.

Die größte untere Grenze von SC kann man als unklassifizierte Information interpretieren, auf die jedes Subjekt zugreifen darf.

Für \oplus gilt: $\forall A, B, C \in SC$:

- (a) $A \leq A \oplus B$ und $B \leq A \oplus B$ und
- (b) $A \leq C \wedge B \leq C \implies A \oplus B \leq C$.

Für \otimes gilt: $\forall A, B, C \in SC$:

- (a) $A \otimes B \leq A$ und $A \otimes B \leq B$ und
- (b) $C \leq A \wedge C \leq B \implies C \leq A \otimes B$.

□

Informationsflussbeschränkungen

zulässige Flüsse

Analog zum Bell-LaPadula Modell wird jedem Subjekt s und jedem Objekt o eine Sicherheitsmarke $sc(s)$ bzw. $sc(o)$ aus SC zugeordnet. In einem Verbands-Modell dürfen Informationen stets nur innerhalb von Sicherheitsklassen oder aufwärts, gemäß der festgelegten partiellen Ordnung \leq fließen. Ein abwärts gerichteter Informationsfluss oder ein Fluss zwischen Sicher-

heitsklassen, die nicht miteinander in Beziehung stehen, ist nicht zulässig. Das bedeutet, dass Informationen genau dann von einem Objekt, das mit einer Sicherheitsklasse A markiert ist, zu einem Objekt, das mit einer Sicherheitsklasse B markiert ist, fließen dürfen, wenn gilt: $A \leq B$. Niedrig eingestufte Information kann damit unbeschränkt fließen.

Hasse-Diagramme

Eine Verbandsstruktur kann man mittels eines so genannten Hasse-Diagramms als einen Graf veranschaulichen. In einem solchen Diagramm sind die Elemente des Verbandes die Knoten des Grafen. Eine Kante zwischen zwei Knoten bezeichnet das größte der beiden Elemente am oberen Ende der Kante und das kleinste am unteren Ende.

Hasse-Diagramm

Beispiel 6.19 (Hasse-Diagramm einer Verbandsstruktur)

Seien die folgenden acht Sicherheitsklassen gegeben:

$$SC = \{(geheim, \emptyset), (geheim, \{a\}), (geheim, \{b\}), (geheim, \{a, b\}), \\ (vertraulich, \emptyset), (vertraulich, \{a\}), (vertraulich, \{b\}), \\ (vertraulich, \{a, b\})\}.$$

Das zugehörige Hasse-Diagramm ist in Abbildung 6.6 angegeben. Es gilt u.a.: $(geheim, \{a\}) \oplus (geheim, \{b\}) = (geheim, \{a, b\})$ und $(geheim, \{a\}) \otimes (geheim, \{b\}) = (geheim, \emptyset)$.

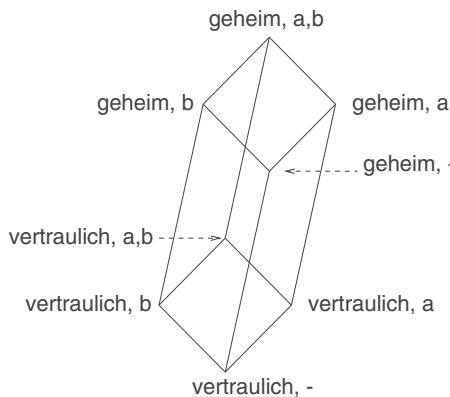


Abbildung 6.6: Hasse-Diagramm

Bedeutung der Verbandsstruktur

Die Transitivität der Relation \leq impliziert, dass jeder implizite Fluss von einer Variablen X zu einer Variablen Y , im Zeichen $X \rightarrow Y$, der aus einer Folge von Flüssen

Kontrolle direkter Flüsse

$$X = Z_0 \rightarrow Z_1 \rightarrow \cdots \rightarrow Z_n = Y$$

resultiert, zulässig ist, falls jeder direkte Fluss $Z_i \rightarrow Z_{i+1}$ zulässig ist. Das bedeutet, dass es ausreicht, die direkten Flüsse zu kontrollieren, um die Zulässigkeit von Informationsflüssen zu überprüfen. Übertragen auf die Kontrolle von Informationsflüssen in Programmen besagt dies, dass eine Folge von Anweisungen zulässig ist, falls jede einzelne Anweisung zulässig ist.

Kontrollaufwand

Die Verbandseigenschaft der Existenz des Suprenums und Infimums für je zwei Verbandselemente kann ebenfalls genutzt werden, um den Kontrollaufwand zu verringern. Betrachten wir dazu als Beispiel eine Zuweisung, die Informationsflüsse von den Variablen X_1, \dots, X_n zur Variable Y bewirkt:

$$Y \leftarrow X_1, \dots, Y \leftarrow X_n, \text{ z.B. } Y := X_1 + X_2 * X_3.$$

Dann muss für alle $i \in \{1, \dots, n\}$ gelten: $sc(X_i) \leq sc(Y)$. Anstatt diese Überprüfung für jede Variable X_i durchzuführen, ist es ausreichend, die kleinste obere Schranke $sc(X_1) \oplus \dots \oplus sc(X_n)$ zu berechnen und für diese die Bedingung zu überprüfen.

Der \otimes -Operator kann ausgenutzt werden, falls Informationsflüsse von einer Variablen zu mehreren anderen zu kontrollieren sind, also

$$Y_1 \leftarrow X, \dots, Y_n \leftarrow X.$$

Die IF-Anweisung in Programmen ist ein Beispiel für eine solche Situation, z.B. `if X then Y1 := 0; Y2 := 1 end if`. Hier reicht es, die größte untere Schranke $Y = sc(Y_1) \otimes \dots \otimes sc(Y_n)$ zu berechnen und die Zulässigkeit für diese, also $sc(X) \leq Y$, zu überprüfen.

statische Kontrollen

Falls die Sicherheitsklassifikation von Objekten über die Dauer ihrer Existenz nicht verändert wird, lassen sich die unteren bzw. oberen Schranken schon zur Übersetzungszeit berechnen, so dass Flusskontrollen bereits statisch durch den Compiler durchgeführt und aufwändige Laufzeitkontrollen vermieden werden können.

Klassifikation und Bewertung

Die Flussrelation, die durch die partielle Ordnung auf den Sicherheitsklassen definiert ist, legt eine systembestimmte Informationsfluss-Strategie fest. Mit dem Verbandsmodell werden für jede Sicherheitsklasse die erlaubten Informationsflüsse erfasst. Das bedeutet, dass alle Benutzer, die die gleiche Sicherheitsklassifikation besitzen, die gleichen Rechte zum Zugriff auf Informationen erhalten. Eine feinkörnige Modellierung von Subjekten ist damit unmittelbar verbunden mit der Festlegung einer großen Anzahl von Sicherheitsklassen. Dies hat sehr aufwändige, nicht skalierende Verwaltungsmaßnahmen bei der Realisierung der modellierten Systeme zur Folge. Demgegenüber schränkt eine grobkörnige Modellierung von Sub-

schlecht skalierend

jetten und die damit verbundene geringe Anzahl der zu verwaltenden Sicherheitsklassen die Möglichkeiten zur differenzierten Festlegung zulässiger Informationskanäle sehr ein.

Weitere Beschränkungen für den Einsatz des Verbandsmodells resultieren aus der im Wesentlichen statischen Zuordnung zwischen Objekten und ihren Sicherheitsklassen. Um die durch die Flussrelation festgelegten Anforderungen zu erfüllen, erfordert die Modellierung in der Regel eine zu hohe Einstufung von Objekten in eine Sicherheitsklasse. Dies wiederum führt dazu, dass niedrig eingestufte Subjekte nur sehr eingeschränkte Rechte erhalten, die in praktischen Anwendungen häufig nicht ausreichen, um die erforderlichen Aufgaben zu erfüllen. Um dies zu vermeiden, werden in der Praxis Subjekte häufig zu hoch eingestuft mit der Konsequenz, dass Benutzer auf diese Weise mehr Rechte erhalten, als sie zur Erledigung ihrer Aufgaben benötigen. Es wird also gegen das need-to-know-Prinzip verstößen.

starre Sicherheitsklassifikation

Zusammen mit den Schwächen im Bereich der Modellierung von Datenintegritäts-Anforderungen führen diese Eigenschaften des Modells dazu, dass es nur für sehr beschränkte Klassen von Anwendungsproblemen als Modellierungsbasis geeignet ist.

In der Literatur wurden für den Bereich der Informationsflussbeschränkung weitere, sehr unterschiedliche Modelle entwickelt, wie das Non-Deducibility-Modell von Sutherland [170] oder das Restrictiveness-Modell von McCullough [116]. Diese haben aber in der Praxis kaum Anwendung gefunden. Die entsprechenden Modelle basieren auf Beeinflussungs- und Beobachtungseigenschaften (Interference) zwischen unterschiedlich klassifizierten Benutzern eines Systems.

6.4 Fazit und Ausblick

Aus der Analyse und Klassifikation der bekannten Modelle lassen sich folgende, allgemeine Leitlinien zum Einsatz der jeweiligen Modelle ableiten.

Das Zugriffsmatrix-Modell ist geeignet für Anwendungsprobleme, die Objekte unterschiedlicher Granularität und unterschiedlichen Typs als zu schützende Einheiten benötigen und im Wesentlichen Integritätsanforderungen stellen. Wegen der objektbezogenen Modellierung der Zugriffsbeschränkungen und -kontrollen sollten die Objekte möglichst wenige Wechselwirkungen besitzen, um inkonsistente Festlegungen zu minimieren. Wegen seiner mangelhaften Skalierbarkeit eignet sich der Matrix-Ansatz schlecht zur Modellierung einer großen Menge von Subjekten, die sich in hoher Frequenz dynamisch ändert.

Zugriffsmatrix

Rollenbasiert

Rollenbasierte Modelle unterstützen eine aufgabenorientierte Modellierung von Berechtigungsvergaben. Damit eignen sie sich sehr gut für Anwendungsszenarien, in denen Geschäftsprozesse zu erfassen und Berechtigungsprofile, basierend auf organisatorischen, hierarchischen Strukturen, zu modellieren sind. Durch die Abkehr von einer auf einzelne Benutzer bezogenen Vergabe von Zugriffsrechten sind RBAC-Modelle besonders für verteilte Anwendungen sowie solche IT-Systeme geeignet, die durch eine dynamisch sich ändernde Menge von Subjekten und durch eine über lange Zeiträume gleich bleibende Menge von Aufgaben charakterisiert sind.

Chinese-Wall

Das Chinese-Wall Modell spiegelt eine kommerzielle Strategie wider, die besonders für Anwendungen im Umfeld von Unternehmensberatungen geeignet ist, um einen Informationsfluss zwischen konkurrierenden Unternehmen zu verhindern. Jedoch sind die Regeln des Modells sehr restriktiv und entsprechen unter Umständen nicht den durch die Anwendung zu modellierenden Aufgaben, da in kommerziellen Umgebungen die Datenintegrität einen hohen Stellenwert besitzt.

Bell-LaPadula

Die Festlegungen des Bell-LaPadula Modells entsprechen hierarchischen Organisationsstrukturen, wie sie insbesondere aus dem militärischen Bereich bekannt sind. Anforderungen zur Beschränkung der zulässigen Informationsflüsse entlang von Hierarchien finden sich aber auch in Unternehmen, Behörden und speziellen Anwendungsszenarien, wie medizinischen Informationssystemen. Die Bell-LaPadula Strategie wird, wie weiter oben angedeutet, als MLS-Strategie in einigen kommerziellen Betriebssystemen direkt unterstützt. Aus diesem Grund ist das Modell trotz seiner restriktiven Vorgaben auch für nicht-militärische Bereiche geeignet, um mit relativ geringem Implementierungsaufwand von Seiten der Entwickler einfache Informationsflussbeschränkungen durchzusetzen.

Verbands-Modell

Das Verbands-Modell ist für Szenarien geeignet, die in hohem Maße die Vertraulichkeit der zu verarbeitenden Information erfordern. Um die Anzahl der benötigten Sicherheitsklassen klein und damit den Verwaltungsaufwand gering zu halten, sollte die zu schützende Information hierarchisch klassifizierbar sein. Da mit einer kleinen Anzahl von Sicherheitsklassen nur noch eine geringe Differenzierung der Subjekte möglich ist, sollten Subjekte einer Klassifikationsstufe im Wesentlichen die gleichen Aufgaben, zum Beispiel lesende oder anfragende Zugriffe auf eine Menge unabhängiger Objekte, erledigen müssen. Auf diese Weise kann man gewährleisten, dass die verbandsbezogene Rechtevergabe nicht zu einer Verletzung des need-to-know-Prinzips führt. Anwendungen, die hauptsächlich als Informations-Server für unterschiedliche Benutzergruppen dienen, sind Beispiele dafür.

Ausblick

Die in diesem Kapitel vorgestellten Sicherheitsmodelle sind gut untersucht und insbesondere die RBAC-Modelle und die Multi-level-Security Modelle haben sich in vielen Bereichen in der Praxis bewährt. Durch den technologischen Wandel hin zu ubiquitär vernetzten Systemen, mit häufig ad-hoc zu etablierenden Kommunikationsverbindungen, benötigt man weitergehende Modellierungsansätze. In vielen Szenarien sind Modelle notwendig, um Vertrauen in den Kommunikations- bzw. Geschäftspartner aufzubauen. Hierfür wurden und werden Vertrauensmodelle entwickelt, die häufig darauf basieren, dass die Partner in dem zugrundeliegenden Szenario eine Bewertung über andere Partner abgeben. Es handelt sich hierbei um Ausprägungen von so genannten Reputationssystemen. Ein einfaches Beispiel eines solchen Reputationssystems kennt man von eBay. Jeder eBay Teilnehmer kann seinen Partner nach der durchgeföhrten Transaktion mit einer positiven, neutralen oder negativen Einschätzung bewerten. Diese Reputationswerte werden bei eBay zentral gespeichert und verwaltet.

Vertrauensmodell

Schwieriger wird eine Bewertung von Partnern jedoch in einem offenen Peer-to-Peer (P2P) Overlaynetz, in dem es keine zentrale Instanz zur Verwaltung von Reputationswerten gibt, sondern dies von den beteiligten Peers selbst-organisiert und dennoch vertrauenswürdig durchgeführt werden muss. Die Entwicklung von geeigneten Vertrauensmodellen in P2P-Netzen und für ubiquitär vernetzte Systeme ist ein aktueller Forschungsbereich.

P2P

Ein weiterer wichtiger Bereich der aktuellen Forschung betrifft die Entwicklung geeigneter Nutzungsmodelle für digitale Inhalte, z.B. Dokumente, aber auch Multi-Media Dateien, die in einer physisch verteilten Umgebung gemeinsam, aber unter nachweislicher Einhaltung der Sicherheitsbedürfnisse der Inhalte-Erststeller (Content-Provider) genutzt werden sollen. Bereits 2002 haben Park und Sandhu in [139] erste Ansäßen für Modelle zur Nutzungskontrolle aufgezeigt. Aktuelle Arbeiten beschäftigen sich mit Modellen zur formalen Modellierung der Nutzungskontrolle für strukturierte XML-Dokumente (u.a. [161]). Der History-basierte Ansatz zur Nutzungskontrolle (vgl. [153]) und dessen Umsetzung stellt hierzu eine Erweiterung dar. Der Ansatz berücksichtigt Kontext-Bedingungen wie die Zugriffshistorie, um Nutzungen für strukturierte Dokumente dynamisch zu regeln. Die Forschungs- und Entwicklungsarbeiten in diesem Bereich sind jedoch noch lange nicht abgeschlossen.

Nutzungskontrolle

7 Kryptografische Verfahren

Nach einer Einleitung in Abschnitt 7.1 werden in den Abschnitten 7.3 und 7.4 wichtige begriffliche und theoretische Grundlagen kryptografischer Verfahren eingeführt. Abschnitt 7.5 widmet sich ausführlich der Klasse der symmetrischen Verschlüsselungsverfahren und stellt mit dem DES-Algorithmus und seinem Nachfolger, dem AES zwei in der Praxis weit verbreitete Verfahren vor. Asymmetrische Systeme werden in Abschnitt 7.6 behandelt, der auch vertiefend auf das bekannte RSA-Verfahren eingeht. Abschnitt 7.7 gibt einen kompakten Einblick in das Gebiet der elliptischen Kurven Kryptografie (ECC). Techniken der Kryptoanalyse werden abschliessend in Abschnitt 7.8 angesprochen.

Kapitelüberblick

7.1 Einführung

Die Verschlüsselung von Nachrichten ist eine sehr alte Wissenschaft, deren Wurzeln bis in die Antike zurückreichen. Unter der Kryptografie versteht man die Lehre von den Methoden zur Ver- und Entschlüsselung von Nachrichten zum Zweck der Geheimhaltung von Informationen gegenüber Dritten (Angreifern). Die Kryptoanalyse ist die Wissenschaft von den Methoden zur Entschlüsselung von Nachrichten, ohne Zugriff auf den verwendeten Schlüssel zu haben. In der Praxis sind Kryptoanalyse und Kryptografie eng miteinander verbunden, da zur Entwicklung sicherer kryptografischer Verfahren stets auch die kritische Beurteilung der Verfahren gehört, wozu man sich der Methoden der Kryptoanalyse bedient. Die beiden Teilgebiete werden unter dem Oberbegriff Kryptologie zusammengefasst.

Kryptografie

Kryptoanalyse

Kryptologie

Steganografie

Im Unterschied zur Kryptografie zielen die Methoden der Steganografie (griech. *stegano*: geheim, *graphein*: schreiben) darauf ab, bereits die Existenz einer Nachricht zu verbergen (engl. *conceal*) und nicht nur den Informationsgehalt einer Nachricht zu verschleiern. Auf steganografische Methoden und digitale Wasserzeichen gehen wir in diesem Buch nicht vertiefend ein, sondern beschränken uns in Abschnitt 7.2 auf einen kurzen Einblick in die Fragestellung.

Historische Entwicklung

Bis vor wenigen Jahren wurden kryptografische Verfahren hauptsächlich von einem eingeschränkten, spezialisierten Benutzerkreis vordringlich im militärischen oder diplomatischen Umfeld eingesetzt, um Informationen vertraulich zu übermitteln. Eine umfassende Darstellung der kryptografischen Verfahren bis zum Jahre 1967 findet man in dem klassischen Buch *The Code-breakers* von David Kahn [96].

Caesar-Code

Zu den bekanntesten historischen Verfahren zählt der Caesar-Code, der nach dem römischen Kaiser und Feldherrn G.J. Caesar benannt ist. Dieser Code verschlüsselt einen Klartext, indem jeder Buchstabe des Klartextes durch denjenigen Buchstaben ersetzt wird, der im Alphabet drei Positionen weiter hinten auftritt (also A wird durch D, B durch E etc. ersetzt). Der Verschlüsselungsalgorithmus ist hier das Ersetzen eines Buchstabens durch einen anderen. Die Anzahl der Positionen, die zwischen zu ersetzendem und ersetztetem Zeichen liegen, hier der Wert drei, ist der verwendete Schlüssel.

Kerckhoffs-Prinzip

Der Caesar-Code ist ein einfaches Beispiel einer Chiffre, deren Sicherheit darauf beruht, dass Angreifern das verwendete Verschlüsselungsverfahren nicht bekannt ist. Unter Kenntnis des Algorithmus wäre es für einen Angreifer nämlich einfach möglich, den Schlüsselraum der Caesar-Chiffre nach dem passenden Schlüssel zu durchsuchen. Da es nur 26 verschiedene Schlüssel gibt, kann der Angreifer diese alle nacheinander auch ohne maschinelle Unterstützung durchprobieren. Die Caesar-Chiffre erfüllt damit nicht das bekannte Kerckhoffs-Prinzip, das besagt, dass die Sicherheit eines kryptografischen Verfahrens allein von den verwendeten Schlüsseln abhängen sollte. Um dies zu gewährleisten, benötigt man einen sehr großen Schlüsselraum, damit ein Angreifer den verwendeten Schlüssel nicht durch einfaches Ausprobieren herausfinden kann.

maschinelle Verschlüsselung

Mit dem Einsatz mechanischer und digitaler Verschlüsselungsmaschinen werden wachsende Anforderungen an die Sicherheit der verwendeten kryptografischen Verfahren und insbesondere an die Größe ihrer Schlüsselräume gestellt. Die Anforderungen ergeben sich einerseits aus dem stetig steigenden Bedarf an vertraulicher Informationsverarbeitung unter Nutzung offener Netze und andererseits aus dem technologischen Fortschritt der zur Verschlüsselung bzw. Kryptoanalyse eingesetzten Geräte. Die technologische Entwicklung hat ihre Anfänge bei einfachen, mechanischen Hilfsmitteln, wie dem Jefferson-Zylinder (ca. 1790), über komplexe Rotormaschinen zur mechanischen Chiffrierung, wie der berühmten ENIGMA [79] der deutschen Wehrmacht im Zweiten Weltkrieg, bis hin zu speziellen kryptografischen Chips oder Spezialrechnern, wie dem DES-Cracker [65].

Einsatzbereiche

Kryptografische Verfahren benötigt man in unterschiedlichen sicherheitsrelevanten Bereichen. Im Bereich der Identifikation und Authentifikation (Zugangskontrolle) setzt man Verschlüsselungsverfahren ein, um unter anderem Passworte, die für die Zugangskontrolle benötigt werden, verschlüsselt zu speichern. Zur Gewährleistung der Datenintegrität werden spezielle kryptografische Mechanismen (vgl. Kapitel 8) verwendet, um digitale Fingerabdrücke der Daten zu erstellen. Mit digitalen Signaturen, die ebenfalls unter Verwendung kryptografischer Verfahren erzeugt werden, kann man die Verbindlichkeit von durchgeführten Aktionen sicherstellen. Insbesondere bei der Gewährleistung der sicheren Übertragung von Daten über unsichere Kommunikationsnetze wie dem Internet oder Funknetzen werden kryptografische Verfahren eingesetzt.

Einsatzbereiche

Das Ziel des vorliegenden Kapitels ist es, einen Überblick über den Stand der Technik im Gebiet der Kryptologie zu geben. Für vertiefende Einblicke sei auf eine Vielzahl von Büchern zu dieser Thematik u.a. [118, 166, 12, 159, 36] verwiesen. Unter <http://www.CrypTool.de> kann man ein Freeware-Programm finden, das anhand von Demonstrationen eine Einführung in die gängigen symmetrischen und asymmetrischen Verschlüsselungsverfahren sowie in Kryptoanalysetechniken für diese bietet.

7.2 Steganografie

Wie eingangs bereits erwähnt, umfassen steganografische Verfahren Methoden, die darauf abzielen, die Existenz einer Nachricht zu verbergen. Man unterscheidet technische und linguistische Steganografie. Im Folgenden werden diese beiden Teilgebiete nur kurz umrissen, um einen Einblick in grundlegende Techniken zu vermitteln. Der an einer vertiefenden Behandlung der Thematik interessierte Leser sei auf die vielfältige Literatur unter anderem in [97, 12, 94, 4] verwiesen.

Verbergen

Neben dem Ziel der geheimen Nachrichtenübermittlung besitzt die Steganografie noch andere Anwendungsbereiche, die zunehmend an Bedeutung gewinnen. So kann zum Beispiel der Urheber von Mediendaten wie Bildern, Videos oder Musik unter Verwendung steganografischer Methoden gut verborgene und schwer entfernbare Copyright-Informationen oder digitale Wasserzeichen (engl. watermarking) in seine digitalen Inhalte einbetten. Industrie und Wissenschaft beschäftigen sich schon seit einigen Jahren mit dem Problem der Integration digitaler Wasserzeichen in Mediendaten, um deren Authentizität und Integrität zu prüfen. Mit der Zunahme der Verbreitung digitaler Inhalte nimmt auch die Bedeutung der Wasserzeichen-

Klassen

Watermarking

technologie rasant zu. Eine Herausforderung bei den Algorithmen, die für das Einbetten digitaler Wasserzeichen verwendet werden, ist sicherzustellen, dass die Veränderungen des markierten Mediums (z.B. ein Bild, ein Videofilm, ein Musikstück) für den Menschen nicht erkennbar sind. Weiterhin ist zu fordern, dass die Wasserzeichen robust sind und auch bei Transformationen des Mediums, z.B. durch Kompression, noch erhalten sind und ausgelesen werden können. Die Verfahren zum Einbetten digitaler Wasserzeichen müssen zugeschnitten sein auf das Medium, in das sie eingebettet werden; ein Audi-Wasserzeichen hat eine andere Charakteristik als ein Bild- oder ein Wasserzeichen für einen Videostream. Eine Beschreibung der verschiedenen Watermarking-Techniken würde den Rahmen des vorliegenden Buches sprengen. Deshalb beschränken wir uns im Folgenden nur auf einen knappen Einstieg in die Thematik der Steganographie und verweisen für eine Vertiefung der Thematik auf die reichhaltige Literatur hierzu (u.a. [7, 184]).

7.2.1 Linguistische Steganografie

Die linguistische Steganografie besteht aus zwei großen Hauptrichtungen, nämlich den unersichtlich getarnten Geheimschriften, dem so genannten Open Code, und den sichtlich getarnten Geheimschriften, den Semagrammen. Semagramme verstecken die Nachricht in sichtbaren Details einer Schrift oder Zeichnung. Ein Beispiel dafür sind kurze und lange Grashalme in einer Zeichnung, die als Morsezeichen zu interpretieren sind.

Semagramme

Open Code

Open Codes verbergen eine geheime Nachricht in einer unverfänglich aussenhenden, offen mitgeteilten Nachricht. Solche offenen Nachrichten erfordern eine Absprache zwischen dem Sender und dem Empfänger. Beispiele sind so genannte Jargon Codes (maskierte Geheimschriften), die für außenstehende Personen bedeutungslos oder unverständlich erscheinen. Ein Open Code wurde auch in dem bekannten Film *Fruhstück bei Tiffany's* verwendet (vgl. [12]). Dort half unwissentlich Audrey Hepburn alias Holly Golightly einem Gangster dessen Kokainhandel vom Gefängnis aus zu betreiben, indem sie seine angeblichen Wettermitteilungen weiter leitete, obwohl die Nachricht: „... es gibt Schnee in New Orleans“ als Wettermeldung für diese Region sicherlich unüblich ist.

Milieu Codes

Open Codes haben sich in bestimmten Berufs- und Gesellschaftsgruppen gebildet wie im Milieu der Bettler, Vagabunden und Gauner. Beispiele sind die Kommunikation zwischen Kartenspielern durch Mimik und Gestik oder das „Zinken“ von Haustüren durch Vagabunden, die Zeichen an Türen anbrachten, um vor dem Hausbesitzer (z.B. Polizist) zu warnen. Unter den Open Code fällt auch ein vorher abgesprochenes Spezialvokabular von Kommunikationspartnern, in dem bestimmte Worte eine spezifische Bedeutung haben. Man denke hier beispielsweise an das Wort Schnee für Kokain oder Gras für Haschisch.

Um Open Codes wirkungslos zu machen, versuchen deshalb Zensoren, die die Bedeutung des Codes nicht kennen, aber eine versteckte Information in einer Nachricht vermuten, diese semantisch korrekt umzuformulieren, so dass eventuell verborgene Hinweise gelöscht werden. Auf diese Weise kam es zu recht kuriosen Nachrichtenaustauschen. So soll (vgl. [12]) im Ersten Weltkrieg ein Zensor die Nachricht „father is dead“ in „father is deceased“ abgewandelt haben, worauf eine Nachricht mit dem Inhalt „is father dead or deceased“ zurückkam.

Zensur

Stichworte sind ein Teilgebiet des Jargon Codes; sie wurden hauptsächlich im militärischen Bereich verwendet. Im Zweiten Weltkrieg repräsentierte zum Beispiel eine bestimmte Formulierung in Wetter- oder Radiomeldungen ein Angriffssignal. So sollten im Wetterbericht der Japaner die Worte Ostwind und Regen, zweimal wiederholt, den Krieg mit den USA ankündigen. Weitere Techniken der linguistischen Steganografie umfassen Filterungsverfahren, wie das Herausfiltern des x-ten Zeichens nach einem bestimmten Zeichen oder des ersten Zeichens nach einem Zwischenraum.

Stichworte

7.2.2 Technische Steganografie

Zu den ältesten technischen Steganografieverfahren (u.a. [97]) zählt die Verwendung von Geheimtinte z.B. aus Milch oder Zwiebelsaft. Die damit verfasste Botschaft wird erst unter Einwirkung von UV-Licht oder durch Erwärmung sichtbar. Von dem griechischen Dichter Herodot (490-425 vor Christus) ist eine etwas skurrilere steganografische Technik überliefert. Er berichtet von einem Adeligen, der eine Geheimbotschaft auf den rasierten Kopf eines Sklaven tätowieren ließ und ihn, nachdem das Haar wieder nachgewachsen war, zu seinem Ziel sandte. Nach einer Kopfrasur war die Botschaft wieder frei lesbar. Anwendungen steganografischer Techniken in jüngerer Vergangenheit finden sich in der Schnelltelegraphie, bei der vorab gespeicherte Morsecode-Botschaften mit 20 Zeichen pro Sekunde (engl. *spurts*) übertragen wurden, oder der Microdot im Zweiten Weltkrieg, der, obwohl so klein wie z.B. ein Punkt über dem Buchstaben i, Informationen im Umfang einer DIN A4-Seite beinhalten kann.

Historisches

Steganografie in digitalen Bildern

Moderne Verfahren der technischen Steganografie bedienen sich häufig Bilddateien, um Informationen darin zu verstecken. Digitale Bilder werden als ein Feld von Punkten (engl. *pixel*) repräsentiert, wobei jedes Pixel die Lichtintensität an diesem Punkt beschreibt. Üblicherweise geschieht das Einbetten von Informationen in ein Bild durch Manipulation der Farbwerte bzw. Helligkeitsstufen einzelner Pixel. Im einfachsten Fall wird dazu das „unwichtigste“ Farbbit (engl. *least significant bit* (LSB)) verändert. Da nicht

Pixel-Felder

LSB

jedes Pixel ein geeigneter Kandidat für Manipulationen ist, werden meist Algorithmen eingesetzt, die die Eignung von Pixeln prüfen. Benachbarte Pixel, die entweder in ihren Farb- bzw. Helligkeitswerten sehr unterschiedlich (z.B. eine Grenzlinie) oder sehr ähnlich sind, scheiden als Kandidaten für Manipulationen aus, da eine Veränderung der Werte vom menschlichen Auge entdeckt werden könnte.

Bildrepräsentation

Digitale Bilder speichert man üblicherweise in 24-Bit oder in 8-Bit Dateien. Die Farbwerte der Pixel werden dabei von den drei Grundwerten rot, grün und blau abgeleitet. Jede dieser Grundfarben wird als ein Byte repräsentiert. Jedes Pixel eines 24-Bit-Bildes wird somit durch drei Bytes beschrieben, die den jeweiligen Farbanteil von rot, grün und blau angeben (z.B. 00100111 (Rotanteil) 11101001 (Grünanteil) 11001000 (Blauanteil)). Die Farbe weiß wird durch 11111111 11111111 11111111 und schwarz wird durch 00000000 00000000 00000000 repräsentiert. Liegen minimal unterschiedliche Farbtöne nebeneinander, so erscheinen die Farben der Punkte für das menschliche Auge identisch, so dass Modifikationen, die nur minimale Unterschiede hervorrufen, nicht erkannt werden.

Standardformate

Heutige Standardbildformate stellen ein Bild mit $1024 * 768$ Pixel dar. Das heißt, dass ein Bild aus ca 0.8 Millionen Bildpunkten besteht und einen Speicherbedarf von 2.4 Megabyte hat. Bei einer Manipulation des LSBs eines jeden Bytes, könnte man immerhin 292.912 Bytes an Informationen in einem solchen Bild verstecken.

Beispiel 7.1 (Verbergen eines Buchstabens)

Um den ASCII-codierten Buchstaben A (der 8-Bit ASCII-Code für A ist 10000011) zu verstecken, benötigt man nur drei Pixel eines Bildes. Seien drei Pixel durch folgende Farbwerte gegeben:

	Rot	Grün	Blau
Pixel 1	00100111	11101001	11001000
Pixel 2	00100111	11001000	11101001
Pixel 3	11001001	00100111	11101001

LSB-Modifikation

Um in den Least Significant Bits (LSB) dieser Pixel den Binärwert von A zu verbergen, reicht es in diesem Beispiel aus, drei Bits zu verändern.

	Rot	Grün	Blau
Pixel 1	0010011 <u>1</u>	1110100 <u>0</u>	11001000
Pixel 2	0010011 <u>0</u>	11001000	1110100 <u>0</u>
Pixel 3	11001001	00100111	11101001

Die unterstrichenen Bits sind die modifizierten. Bildet man aus den LSBs der drei Pixel eine Folge, so erhält man die Codierung des Buchstabens A.



Die angesprochene Technik des Versteckens von Informationen ist einfach zu implementieren, aber sehr anfällig gegenüber Veränderungen der Daten, die zum Beispiel durch Datenkompression hervorgerufen werden. Bekannte Kompressionsformate für Bilder sind JPEG, GIF oder BMP. Da das JPEG-Format (Join Picture Expert Group) eine Kompression mit Datenverlust vornimmt, ist der Einsatz der LSB-Steganografie-Technik für Bilder, die in das JPEG-Format transformiert werden, ungeeignet. Eine datenerhaltende Kompression wird demgegenüber durch das GIF (Graphic Interchange Format) oder das BMP (Bitmap) von Microsoft Windows garantiert.

Datenkompression

7.3 Grundlagen kryptografischer Verfahren

Dieser Abschnitt führt zunächst grundlegende Begriffe im Zusammenhang mit kryptografischen Verfahren ein. Anschließend erläutern wir die Anforderungen, die an heutige, sichere Verfahren gestellt werden.

7.3.1 Kryptografische Systeme

Ein kryptografisches System legt fest, wie Klartexte in Kryptotexte transformiert (verschlüsselt) werden und wie Kryptotexte wieder in Klartexte zurück transformiert (entschlüsselt) werden. Das wesentliche Ziel beim Einsatz kryptografischer Verfahren besteht in der Geheimhaltung der in der Nachricht codierten Information gegenüber Dritten (Angreifern). Ein Angreifer versucht, unter Kenntnis von Informationen über das kryptografische System, verschlüsselte Daten zu entschlüsseln. Mit Definition 7.1 führen wir die Komponenten eines kryptografischen Systems ein. Jedes in der Praxis eingesetzte Verschlüsselungssystem kann dadurch beschrieben werden.

Definition 7.1 (Kryptografisches System)

Gegeben seien zwei endliche Zeichenvorräte (Alphabete) A_1 und A_2 . Ein kryptografisches System oder kurz Kryptosystem ist gegeben durch ein

Kryptosystem

Tupel

$$\mathcal{KS} = (\mathcal{M}, \mathcal{C}, EK, DK, E, D)$$

mit

1. der nicht leeren endlichen Menge von Klartexten $\mathcal{M} \subseteq A_1^*$, wobei A_1^* die Menge aller Worte über dem Alphabet A_1 beschreibt,
2. der nicht leeren endlichen Menge von Krypto- bzw. Chiffretexten $\mathcal{C} \subseteq A_2^*$,
3. der nicht leeren Menge von Verschlüsselungsschlüsseln EK ,
4. der nicht leeren Menge von Entschlüsselungsschlüsseln DK sowie einer Bijektion $f : EK \rightarrow DK$. Die Bijektion assoziiert zu einem Verschlüsselungsschlüssel $K_E \in EK$ einen dazu passenden Entschlüsselungsschlüssel $K_D \in DK$, d.h. $f(K_E) = K_D$,
5. dem linkstotalen und injektiven Verschlüsselungsverfahren

$$E : \mathcal{M} \times EK \rightarrow \mathcal{C} \quad \text{und}$$

6. dem Entschlüsselungsverfahren

$$D : \mathcal{C} \times DK \rightarrow \mathcal{M},$$

mit der Eigenschaft, dass für zwei Schlüssel $K_E \in EK, K_D \in DK$ mit $f(K_E) = K_D$ gilt:

$$\forall M \in \mathcal{M} : D(E(M, K_E), K_D) = M.$$

□

Gemäß Definition 7.1 sind Klartexte und Chiffretexte Worte über einem endlichen Zeichenvorrat, wobei die beiden Alphabete A_1 und A_2 auch unterschiedlich sein können. Die Eigenschaft (6.) der Definition besagt, dass ein beliebiger Klartext M , der mit einem Verschlüsselungsschlüssel aus dem zulässigen Schlüsselraum verschlüsselt wurde, $E(M, K_E)$, anschließend wieder mit dem dazu passenden Entschlüsselungsschlüssel, $K_D = f(K_E)$, entschlüsselt werden kann, $M = D(E(M, K_E), K_D)$.

Angreifer

Die Festlegungen aus Definition 7.1 werden in Abbildung 7.1 veranschaulicht. Wir gehen davon aus, dass ein Angreifer auf die Chiffretexte zugreifen kann. Dies ist eine realistische Annahme, da das Abhören offener Kommunikationsnetze, insbesondere drahtloser Netze, einfach möglich ist. Mit dem abgefangenen Chiffertext und den ihm zur Verfügung stehenden Informationen versucht nun der Angreifer, den zugehörigen Klartext zu ermitteln. Beispiele für solche Informationen sind Hinweise, dass die Klartexte in deutscher Sprache verfasst sind, so dass über die statistischen Eigenschaften der

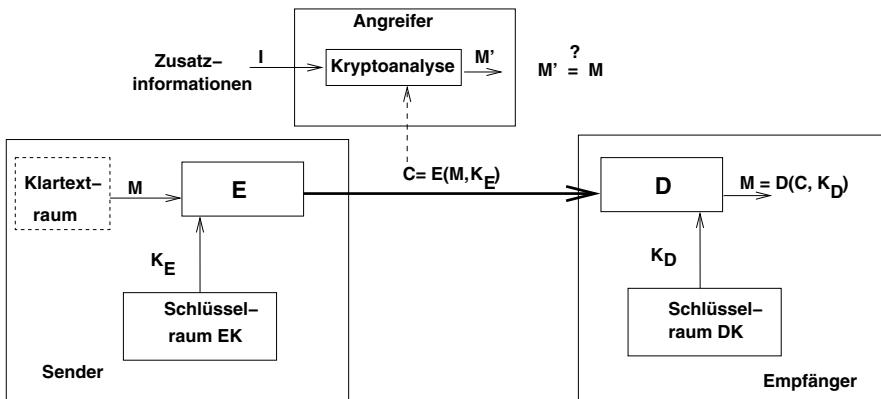


Abbildung 7.1: Komponenten eines Kryptosystems

Sprache Anhaltspunkte für eine Kryptoanalyse vorliegen (z.B. Auftreten von Bigrammen, wie ei und st, oder von Trigrammen, wie sch).

Beispiel 7.2 (Caesar-Chiffre)

Wie bereits einleitend erwähnt, ist die Caesar-Chiffre eine der bekanntesten einfachen Chiffren. Sie gehört zur Klasse der Verschiebungschiffren. Die Chiffre arbeitet auf dem lateinischen Alphabet und ersetzt zyklisch jedes Zeichen der zu verschlüsselnden Nachricht durch das im Alphabet um drei Positionen weiter hinten liegende Zeichen: also A durch D, B durch E, usw. sowie X durch A, Y durch B und Z durch C.

Verallgemeinert man die Caesar-Chiffre, indem man eine beliebige Anzahl x von Verschiebepositionen zulässt, so kann die damit beschriebene Familie von Chiffren mit den in Definition 7.1 eingeführten Begriffen wie folgt charakterisiert werden. Die Mengen \mathcal{M} der Klartexte sowie \mathcal{C} der Chiffretexte sind durch das lateinische Alphabet festgelegt. Um eine Verwechslung mit den in der Definition eingeführten Notationen zu vermeiden, wird in diesem Beispiel das lateinische Alphabet bestehend aus Kleinbuchstaben verwendet. Es gilt also:

$$A_1 = A_2 = \{a, \dots, z\}.$$

Jeder Buchstabe wird eindeutig mit einer Zahl aus der Menge $\{0, \dots, 25\}$ assoziiert, wobei gilt, a wird der Zahl 0, b der Zahl 1 und z wird mit der Zahl 25 assoziiert.

Der Schlüsselraum wird durch die Menge der möglichen Verschiebepositionen definiert, also $DK = EK = \{1, \dots, 26\}$. Das Verschlüsselungsverfahren kann als eine Addition modulo 26 beschrieben werden¹.

Caesar-Chiffre

¹ Für die Rechenregeln der Modulo-Arithmetik vgl. Definition 7.10.

Mit den getroffenen Festlegungen gilt für einen Klartext $M = M_1, \dots, M_r$, $M_i \in A_1$ für $i \in \{1, \dots, r\}$:

$$E(M_i) = (M_i + K_E) \bmod 26 = C_i, \text{ wobei } K_E \in EK.$$

Das Entschlüsselungsverfahren macht die Verschiebung wieder rückgängig, also

$$D(C_i) = (C_i + K_D) \bmod 26, \text{ mit}$$

$$K_D = f(K_E) = -K_E \text{ und } C_i \in A_2, \text{ für } i \in \{1, \dots, r\}.$$



hybride Systeme

Wir unterscheiden zwei Klassen kryptografischer Systeme, nämlich die symmetrischen und die asymmetrischen. Beide Klassen von Verfahren werden in heutigen IT-Systemen häufig in Kombination eingesetzt. Man spricht von hybriden Systemen.

Symmetrische Verfahren

symmetrisches Kryptosystem

Charakteristisch für symmetrische Kryptosysteme (engl. *secret key*) ist, dass Ver- und Entschlüsselungsschlüssel gleich oder leicht voneinander abzuleiten sind. Drückt man diesen Sachverhalt mit den Festlegungen von Definition 7.1 aus, so heißt dies, dass die Bijektion f die Identitätsfunktion ist oder einen einfach zu berechnenden Zusammenhang zwischen K_E und K_D beschreibt.

Schlüssel

In einem symmetrischen System müssen zwei Kommunikationspartner einen gemeinsamen, geheimen Schlüssel (symmetrisch) verwenden, wenn sie miteinander vertraulich kommunizieren möchten. Das bedeutet, dass sie sich vorab über diesen gemeinsamen Schlüssel verständigen müssen. Die Sicherheit einer Kommunikation mittels symmetrischer Verfahren hängt also nicht nur von der kryptografischen Stärke der verwendeten Verfahren ab, sondern auch von der sicheren Aufbewahrung und Verwaltung der geheimen Schlüssel durch die Kommunikationspartner sowie von der sicheren initialen Übermittlung der verwendeten, gemeinsamen Schlüssel.

Protokoll

Kommunikationsprotokoll

Eine vertrauliche Kommunikation unter Nutzung eines symmetrischen Kryptosystems durchläuft grob die folgenden Protollschrifte:

1. Die Kommunikationspartner Alice A und Bob B vereinbaren einen gemeinsamen, geheimen Schlüssel $K_E = K_D = K_{A,B}$.

2. Um einen Klartext M von Alice zu Bob zu versenden, verschlüsselt Alice den Text M mittels $K_{A,B}$, also $C = E(M, K_{A,B})$, und sendet C an Bob.
3. Bob entschlüsselt C mittels $K_{A,B}$, also

$$M = D(C, K_{A,B}) = D(E(M, K_{A,B}), K_{A,B}).$$

Abschnitt 7.5 geht vertiefend auf symmetrische Verschlüsselungsverfahren ein und stellt mit dem Data Encryption Standard (DES) und dem Advanced Encryption Standard (AES) bekannte und in der Praxis häufig eingesetzte Verfahren vor. Auf Lösungen für das Problem des sicheren Schlüsselaustausches gehen wir in Kapitel 9 genauer ein.

Asymmetrische Verfahren

Mitte der 70er Jahre wurde zeitgleich von Ralph Merkle sowie von Diffie und Hellman [55] mit den asymmetrischen Kryptosystemen (engl. *public key cryptography*) eine neue Klasse kryptografischer Verfahren vorgeschlagen. Die Kernidee asymmetrischer Verfahren besteht darin, dass jeder Kommunikationspartner ein Schlüsselpaar bestehend aus seinem persönlichen, geheimen Schlüssel (engl. *private key*) und einem öffentlich bekannt zu gebenden Schlüssel besitzt. Auch in diesen Systemen existieren also geheime Schlüssel, die sicher zu verwalten sind. Im Gegensatz zu den symmetrischen Verfahren müssen diese aber nicht sicher ausgetauscht werden.

asymmetrisches Kryptosystem

Schlüsselpaar

Kommunikationsprotokoll

Eine vertrauliche Kommunikation unter Nutzung eines asymmetrischen Kryptosystems durchläuft grob die folgenden Protollschrifte:

Protokoll

1. Die Kommunikationspartner Alice A und Bob B erzeugen ihre eigenen Schlüsselpaare (K_E^A, K_D^A) bzw. (K_E^B, K_D^B) , wobei die Schlüssel K_D^A und K_D^B die jeweiligen privaten Schlüssel sind.
2. Die Schlüssel K_E^A und K_E^B werden öffentlich bekannt gegeben. Diese können zum Beispiel in einer öffentlichen Datenbank oder von einem öffentlich zugänglichen Schlüssel-Server verwaltet werden.
3. Will nun Alice einen Klartext M , der für Bob bestimmt sein soll, verschlüsseln, so verwendet sie dazu Bobs öffentlichen Schlüssel, also: $E(M, K_E^B) = C$, und sendet C an Bob.
4. Bob entschlüsselt C mit seinem privaten Schlüssel, also: $D(C, K_D^B) = M$.

Zu den bekanntesten Vertretern der Klasse der asymmetrischen Systeme gehört das RSA-Verfahren [152]. Abschnitt 7.6 gibt einen vertiefenden

Einblick in asymmetrische Kryptosysteme und geht eingehend auf das RSA-Verfahren ein.

7.3.2 Anforderungen

Kerckhoffs-Prinzip

Für praxisrelevante Kryptosysteme wird gefordert, dass sie das Kerckhoffs-Prinzip erfüllen. Das heißt, dass die Sicherheit des Systems nicht von der Geheimhaltung der Ver- und Entschlüsselungsfunktion abhängen darf. Geht man also davon aus, dass die Algorithmen zur Ver- und Entschlüsselung öffentlich zugänglich sind, so ergibt sich daraus die nächste allgemeine Anforderung, die besagt, dass der Schlüssel mit dieser Kenntnis nicht (oder nur sehr schwer) zu berechnen sein darf.

*grosser
Schlüsselraum*

Für die Caesar-Chiffre haben wir bereits gesehen, dass deren Schlüsselraum nur die Größe 26 hat, so dass der korrekte Schlüssel durch Ausprobieren einfach herleitbar ist. Dies verdeutlicht die nächste allgemeine Forderung. Sie besagt, dass der Schlüsselraum EK sehr groß sein sollte, damit er nicht mit vertretbarem Aufwand durchsucht werden kann. Nun stellt sich natürlich die Frage, wie groß ein Schlüsselraum sein muss, damit das Verfahren einem Angriff standhält, der unter aus Sicht des Angreifers vertretbaren Kosten (u.a. Rechenzeit und Speicherkapazität) den Schlüsselraum durchsucht.

Beispiel 7.3 (Schlüsselraumsuche)

56-Bit Schlüssel

Betrachten wir einen Schlüssel der Länge 56-Bit, wie er beim klassischen DES-Verfahren (vgl. Kapitel 7.5.4) eingesetzt wird. Bei 56-Bit Schlüsseln erhalten wir einen Schlüsselraum EK mit 2^{56} unterschiedlichen Schlüsseln.

Bereits im Jahr 1998 wurde von der EFF (Electronic Frontier Foundation) für ein Budget von weniger als 250 000 US Dollar ein Spezialrechner bestehend aus ca. 1500 Chips entwickelt, die so genannte Deep Crack Maschine², die einen Brute-Force Angriff auf den DES durchführte. Für einen verschlüsselten Text wurde dazu der gesamte Raum der möglichen DES-Schlüssel, also 2^{56} Schlüssel, durchprobiert und versucht, den verschlüsselten Text zu entschlüsseln. Der Spezialrechner benötigte hierfür lediglich 56 Stunden, da die 1500 Chips den Schlüsselraum parallel durchsucht haben.

Zur effizienten Durchführung von Kryptoanalysen wurden in 2006 in dem COPACOBANA-Rechner³ (Cost-Optimized Parallel COde Breaker) 120 Field-Programmable Gate Arrays (FPGAs) zu einer Maschine zusammengebaut. Mit den 120 FPGAs wurde der DES-Schlüsselraum ebenfalls parallel durchsucht. Da hierfür Standardchips verwendet wurden, konnte eine Brute-Force Analyse für weniger als 10 000 US Dollar durchgeführt werden und

² https://w2.eff.org/Privacy/Crypto/Crypto_misc/DESCracker/HTML/19980716_eff_des_faq.html

³ <http://www.copacobana.org/>

sie dauerte im Durchschnitt 6.4 Tage. In dem Nachfolgeprojekt COPACOBANA RIVYERA konnte dies sogar noch verbessert werden. In weniger als einem Tag konnte ein DES-Schlüssel geknackt werden, wobei 128 Xilinx Spartan-3 5000 FPGAs zum Einsatz kamen.

Die Ergebnisse veranschaulichen, dass ein 56-Bit Schlüssel, also ein Schlüsselraum von 2^{56} Schlüsseln, nicht ausreichend ist für heutige Anforderungen.



In Deutschland veröffentlicht die Bundesnetzagentur⁴ jährlich eine Empfehlung zu den als für die Erstellung qualifizierter digitaler Signaturen geeignet eingestuften kryptografischen Algorithmen und den empfohlenen Schlüssellängen. Für symmetrische Kryptoverfahren gelten Schlüssellängen von 128 Bit, wie sie vom aktuellen NIST-Kryptostandard AES unterstützt werden, als ausreichend sicher bis zum Jahr 2036⁵. Für RSA, als dem bekanntesten Vertreter asymmetrischer Verfahren, wird eine Schlüssellänge von mindestens 2048 und für sicherheitskritische Anwendungen wird eine Schlüssellänge von 4096 Bit empfohlen. Tabelle 7.6 auf der Seite 354 fasst die von der NIST empfohlenen Schlüssellängen für symmetrische und asymmetrische Verfahren zusammen.

*Empfohlene
Schlüssellängen*

7.4 Informationstheorie

In diesem Abschnitt werden grundlegende theoretische Konzepte erläutert, die die Basis zum tiefer gehenden Verständnis heutiger kryptografischer Systeme bilden. Wir werden dazu auf die Informationstheorie, die sich mit der Beschreibung des Informationsgehalts von Nachrichten beschäftigt sowie auf Begriffsbildungen zur theoretischen Beschreibung von Sicherheitseigenschaften eingehen. Beide Bereiche wurden maßgeblich von C. E. Shannon [163, 164] geprägt.

*theoretische
Konzepte*

7.4.1 Stochastische und kryptografische Kanäle

Bei der Übertragung von Nachrichten von einer Quelle über einen Kanal zu einem Empfänger versucht man in der Regel zu erreichen, dass eine versandte Nachricht auch vollständig ankommt. Bei der Übertragung kann es im Kanal jedoch zu Störungen kommen. Sind diese Störungen zufällig und nicht systematisch, d.h. dass sie nicht prinzipiell beschrieben und durch den Einsatz geeigneter Maßnahmen seitens des Empfängers behoben werden können, so spricht man von einem stochastisch gestörten Kanal.

*stochastischer
Kanal*

⁴ <http://www.bundesnetzagentur.de>

⁵ Vgl. <http://www.keylength.com/en/3/>

Gesetzmäßigkeiten

Zufällige Störungen kann man im Allgemeinen nicht im Einzelnen vorhersagen, allerdings kann man mit Hilfe der Wahrscheinlichkeitsrechnung Gesetzmäßigkeiten des Kanals beschreiben. Wichtig für den Empfänger ist dabei, dass er Aussagen über die a priori Verteilung der Quelle machen kann. Das heißt, dass er Kenntnisse darüber besitzt, mit welcher Wahrscheinlichkeit die Quelle eine empfangene Nachricht geschickt haben könnte. So kann ein Empfänger zum Beispiel aufgrund der Tatsache, dass eine Quelle in deutscher Sprache sendet, eine Vielzahl von Buchstabenkombinationen ausschließen. Wenn der Empfänger die ursprüngliche Nachricht nicht kennt und diese gestört empfängt, so kann er auf der Basis der ihm verfügbaren Informationen zu deren Rekonstruktion nur eine Schätzung durchführen.

Verhalten eines Kanals

In der Nomenklatur der Wahrscheinlichkeitsrechnung bedeutet dies folgendes. Gegeben sei eine Nachrichtenquelle Q , die Nachrichten M_1, \dots, M_n sendet, wobei jede Nachricht M_i mit einer Wahrscheinlichkeit p_i gesendet wird. Es gelte $\sum_{i=1}^n p_i = 1$.

Seien C_1, \dots, C_k alle Nachrichten, die der Empfänger beobachten kann. Das Verhalten des Kanals lässt sich nun mittels bedingter Wahrscheinlichkeiten p_{ij} beschreiben, wobei $p_{ij} = P(C_j | M_i)$ die bedingte Wahrscheinlichkeit beschreibt, dass der Kanal die Nachricht C_j ausgibt, falls die Nachricht M_i gesendet wurde.

Maximum-Likelihood Methode**Schätzverfahren**

Mit der Kenntnis über die Wahrscheinlichkeitsverteilung p_i und der bedingten Wahrscheinlichkeiten p_{ij} kann der Empfänger die Originalnachricht schätzen. Sei C_j die empfangene Nachricht, dann lässt sich die Wahrscheinlichkeit, dass M_i gesendet wurde, mit der Maximum-Likelihood Funktion wie folgt berechnen:

$$P(M_i | C_j) = P(C_j | M_i)P(M_i).$$

Bestimmt man nun ein i' derart, dass die bedingte Wahrscheinlichkeit dieses i' größer gleich den anderen bedingten Wahrscheinlichkeiten ist, also

$$P(M_{i'} | C_j) \geq P(M_i | C_j), \quad \text{für alle } 1 \leq i \leq n,$$

so erhält man mit $M_{i'}$ die Nachricht, die mit größter Wahrscheinlichkeit gesendet wurde.

Kryptografischer Kanal**kryptografischer Kanal**

Im Bereich der Kryptografie werden Kryptotexte über kryptografische Kanäle übertragen, die Ähnlichkeiten mit stochastisch gestörten Kanälen aufweisen. Dies wird deutlich, wenn man den Informationsempfänger eines

stochastischen Kanals als Angreifer in einem Kryptosystem auffasst. Die Analogie ergibt sich dann unmittelbar daraus, dass ein Angreifer versucht, anhand der übermittelten Daten sowie seiner Kenntnisse über die Nachrichtenquelle die ursprüngliche Nachricht zu ermitteln (Kryptoanalyse). Die Störungen eines kryptografischen Kanals sind jedoch nicht unabsichtlich und stochastisch, sondern vielmehr gezielt vom Absender des Kryptotextes verursacht. Der Sender ist ja bemüht, die Nachricht durch Störungen, das ist hier die Chiffrierung, so zu sichern, dass nur der rechtmäßige Empfänger sie entschlüsseln und den Klartext lesen kann. Zur Erzeugung derartiger Störungen benötigt man neben den Chiffrierfunktionen auch noch Schlüssel.

Mit diesen Vorbemerkungen ergibt sich nun für einen Angreifer die folgende Situation: Seien nun wieder M_i die gesendeten Nachrichten und C_j die Chiffretexte, die ein Angreifer beobachten kann. Interpretiert der Angreifer den Kanal als stochastischen Kanal und kennt er die Verteilung der verwendeten Schlüssel, so kann er die bedingten Wahrscheinlichkeiten für deren Auftreten berechnen und die Kryptoanalyse als Maximum-Likelihood Schätzung durchführen. Im Gegensatz zur Decodierung eines gestörten Signals ist der durch den Angreifer zu leistende Aufwand hierbei jedoch wesentlich höher, da der Sender bestrebt sein wird, die Kryptoanalyse möglichst schwierig zu machen. Für den Angreifer ist es somit wichtig, möglichst viele Informationen über die Nachrichtenquelle zur Verfügung zu haben, um seinen Analyseaufwand zu reduzieren. Wichtige Begriffe in diesem Zusammenhang sind die Entropie und die Redundanz einer Quelle.

Kryptoanalyse

7.4.2 Entropie und Redundanz

Für die Analyse von Nachrichten ist nicht nur die Wahrscheinlichkeit ihres Auftretens von Bedeutung, sondern vor allem ihr Informationsgehalt. Intuitiv wird man von einer Nachricht, die selten auftritt, annehmen, dass sie einen höheren Informationsgehalt besitzt als eine, die häufig auftritt. Vor diesem Hintergrund wird der Informationsgehalt einer Nachricht als negativer Logarithmus (zur Basis 2) ihrer Auftretens-Wahrscheinlichkeit angegeben, also

$$I(M_i) = \log\left(\frac{1}{p_i}\right) = -\log(p_i), \text{ gemessen in Bits/Zeichen.}$$

Informationsgehalt

Will man nun den mittleren Informationsgehalt einer Quelle ermitteln, die unkorrelierte Nachrichten versendet, so gewichtet man die Nachrichten mit

Entropie

der Wahrscheinlichkeit ihres Auftretens und summiert diese auf.

$$H(p_1, \dots, p_n) = - \sum_{i=1}^n p_i \log(p_i).$$

Man bezeichnet H als die Entropie der Nachrichtenquelle. Die Entropie misst den Informationsgehalt, den man im Durchschnitt erhält, wenn man die Quelle beobachtet. Sie ist damit ein Maß für die Unbestimmtheit der Quelle. Die Entropie erlaubt Aussagen über die Unbestimmtheit der erzeugten Nachrichten, falls die Quelle nicht beobachtet werden kann.

maximale Entropie

In einem kryptografischen System ist es natürlich wünschenswert, Angreifern möglichst wenig Informationen über die Quelle zu liefern, also von einer Quelle mit einer großen Entropie auszugehen. Den größten Wert nimmt H dann an, wenn die Nachrichten gleich verteilt gesendet werden, d.h. wenn gilt:

$$p_i = \frac{1}{n} \text{ und } H_{\max} = \log(n).$$

Diesen Wert bezeichnet man als die maximale Entropie.

bedingte Entropie

Beobachtet man nun empfangene Nachrichten, so kann man den mittleren Informationsgehalt einer Nachricht M_i unter der Voraussetzung, dass ein Chiffertext C_j empfangen wurde, berechnen. Ermittelt man den Mittelwert über alle $M \in \mathcal{M}$ aus der Menge der Klartextnachrichten unter der Bedingung, dass ein $C \in \mathcal{C}$ empfangen wurde, und summiert diesen Mittelwert über alle C aus der Menge der Chiffertexte, so erhält man die bedingte Entropie:

$$H(\mathcal{M} | \mathcal{C}) = - \sum_{M,C} P(C)P(M | C) \log(P(M | C)).$$

Die bedingte Entropie sagt aus, wie unbestimmt eine Quelle noch ist, nachdem bereits eine Nachricht beobachtet wurde.

Redundanz

Je mehr eine Quelle von der maximalen Entropie abweicht, desto günstiger stehen die Chancen für einen Kryptoanalysten, aus einer Menge von Klartexten den Wahrscheinlichsten auszuwählen, der zum beobachteten Kryptotext passt. Diese Abweichung wird durch die Redundanz erfasst. Die Redundanz ist die Differenz zwischen maximaler und tatsächlicher Entropie, also $D = H_{\max} - H$.

Beispiel 7.4 (Redundante Quelle)

Betrachten wir als Beispiel eine Nachrichtenquelle, die das lateinische Alphabet mit 26 Buchstaben sowie ein Leerzeichen umfasst. Die maximale Entropie dieser Quelle ist

lateinisches Alphabet

$$H_{\max} = \log(27) \sim 4,75.$$

Erzeugt die Quelle aber Nachrichten als Zeichenfolgen, deren Verteilung derjenigen Verteilung entspricht, die das Auftreten von sinnvollen Texten in der deutschen Sprache modelliert, so ergibt sich für die Entropie ein Wert von ungefähr 1.6 und damit eine Redundanz von mehr als 3. Mit anderen Worten heißt das, dass die deutsche Sprache eine Redundanz von über 66% besitzt.



7.4.3 Sicherheit kryptografischer Systeme

Mit den eingeführten Festlegungen können wir nun Definitionen für die Sicherheit eines kryptografischen Systems angeben. Wir unterscheiden zwischen der absoluten sowie der praktischen Sicherheit.

Sicherheit

Absolute Sicherheit

Die Definition der absoluten Sicherheit stammt von Shannon [163]. Shannon geht von der Tatsache aus, dass dem Angreifer die a priori Verteilung der Quelle bekannt ist und er versucht, unter Ausnutzung der a posteriori Verteilung (die Wahrscheinlichkeit, dass C_j empfangen wird, wenn M_i gesendet wurde) für ihn relevante Informationen herzuleiten. Wünschenswerterweise sollte dies dem Angreifer aber nicht gelingen.

Definition 7.2 (Absolute Sicherheit)

Shannon definiert ein Kryptosystem als absolut sicher, wenn die a priori Verteilung für jede Nachricht gleich ihrer a posteriori Verteilung ist, d.h. wenn gilt:

$$P(M) = P(M | C).$$



Die Definition der absoluten Sicherheit besagt, dass der Angreifer durch das Abhören des Kanals keinerlei Informationen erhalten kann. Verwendet man nun die Maximum-Likelihood Methode, so kann man die Maximum-Likelihood Funktion maximieren, da sie wegen der Gleichheit der a priori und a posteriori Verteilung unabhängig vom empfangenen Chiffretext stets die Nachricht mit der höchsten a priori Wahrscheinlichkeit liefert. Ersetzt man in den jeweiligen Formeln die bedingte Wahrscheinlichkeit durch die a priori Wahrscheinlichkeit, so ergibt sich, dass die bedingte Entropie gleich der maximalen Entropie ist. Daraus folgt, dass ein Angreifer in einem absolut sicheren Kryptosystem allein durch das Beobachten der Quelle keine Information über den verwendeten Schlüssel gewinnen kann.

absolut sicher

keine Informationspreisgabe

Eigenschaften absolut sicherer Kryptosysteme

Schlüsselraum

Aus der Eigenschaft $P(M) = P(M \mid C)$ eines absolut sicheren Systems folgt für einen festen Klartext M und alle Chiffretexte C , dass gilt:

$$P(C \mid M) = P(C) > 0.$$

Das bedeutet, dass es zu jedem beobachteten C einen Schlüssel geben muss, mit dem der zugehörige Klartext M in C überführt wurde. Da aber ein Schlüssel einen Klartext nicht auf zwei unterschiedliche Chiffretexte abbilden kann, folgt, dass es mindestens so viele Schlüssel wie Chiffretexte C gibt. Aufgrund der Injektivität der Chiffrierabbildung gibt es aber auch mindestens so viele Chiffretexte wie Klartexte. Daraus folgt, dass es in einem absolut sicheren Kryptosystem mindestens so viele Schlüssel wie Klartexte geben muss.

One-time-pad

In der Praxis erweist sich nur das so genannte One-time-pad als absolut sicheres Kryptosystem. Dabei handelt es sich um ein Chiffrierverfahren, das einen Klartext mit einer zufällig erzeugten Schlüsselfolge verschlüsselt, die dieselbe Länge wie der Klartext besitzt. Aufgrund der Zufälligkeit des Schlüssels ist es nicht möglich, durch Abhören des Kanals Informationen zu erhalten. Da die einzelnen Zeichen der Schlüsselfolge zufällig und unabhängig voneinander generiert werden, ist es aber auch dem berechtigten Empfänger der Nachricht nicht möglich, die Schlüsselfolge selber zu generieren. Das bedeutet, dass die Folge auf einem sicheren Weg ausgetauscht werden muss, was mit einem sehr großen Aufwand verbunden ist. Dadurch ist das Verfahren für praktische Anwendungen weitestgehend unbrauchbar. Eine der wenigen bekannten Anwendungen des One-time-pads ist die verschlüsselte Übertragung der Daten über den heißen Draht zwischen Moskau und Washington.

Praktische Sicherheit

Messen des Analyseaufwands

In der Praxis betrachtet man bei der Frage nach der Sicherheit eines Systems auch den Aufwand, der für eine Kryptoanalyse zu leisten ist. So sind beispielsweise für die Analyse einer einfachen Verschiebung, die alle $26!$ Permutationen des lateinischen Alphabets zulässt, im Durchschnitt

$$\frac{26!}{2} > 2 \cdot 10^{26}$$

Entschlüsselungsoperationen durchzuführen, wenn man den Klartext mittels vollständiger Suche finden will. Verwendet man einen großen Schlüsselraum, aber gleichzeitig eine Sprache mit einer typischen Wahrscheinlichkeitsverteilung, so kann man in jedem Analyseschritt den Schlüsselraum um eine Menge von Möglichkeiten reduzieren. Deshalb darf bei als praktisch si-

Partitionierung

cher geltenden Systemen der Schlüsselraum nicht leicht partitionierbar sein. Würde es zum Beispiel gelingen, den Schlüsselraum bei jedem Analyse-schritt zu halbieren, so reduzierte sich bei n Schlüsseln der Suchaufwand von n auf $\log n$.

Wie wir gesehen haben, wird die Aufgabe der Kryptoanalyse erleichtert, wenn statistische Eigenschaften des Klartextes zur Analyse herangezogen werden können. Deshalb sollte jedes Chiffretextzeichen von möglichst vielen Klartextzeichen sowie dem gesamten Schlüssel abhängen, damit die statistischen Besonderheiten des Klartextes ausgeglichen werden. Zur Ge-winnung relevanter Analyseinformationen werden dann große Einheiten von Chiffretext benötigt. Diese Eigenschaft kennzeichnet Shannon als das Prinzip der Diffusion.

Diffusion

Weiterhin wird für ein praktisch sicheres Kryptosystem gefordert, dass der Zusammenhang zwischen Klartext, Schlüssel und Chiffretext so komplex wie möglich zu gestalten ist, um so die Kryptoanalyse zu erschweren. Diese Eigenschaft kennzeichnet Shannon als das Prinzip der Konfusion. Die Prinzipien der Diffusion und Konfusion sind maßgebend für die praktische Sicherheit von Kryptosystemen.

Konfusion

Eine wichtige Rolle spielen auch so genannte Klartextangriffe. Bei dieser Klasse von Angriffen geht man davon aus, dass dem Kryptoanalysten Paare, bestehend aus Klartext und zugehörigem Chiffretext, vorliegen. Ist der Zusammenhang zwischen den Paaren und den verwendeten Schlüsseln offensichtlich, wie dies zum Beispiel bei der Caesar-Chiffre (vgl. Beispiel 7.2) der Fall ist, bietet das verwendete Verfahren keinen Schutz vor einem solchen Klartextangriff. Von einem praktisch sicheren Verfahren ist somit zu fordern, dass es Klartextangriffen widersteht.

Klartextangriffe

Nach diesen Vorüberlegungen können wir nun die folgende Definition für die praktische Sicherheit eines kryptografischen Systems angeben.

Definition 7.3 (Praktische Sicherheit)

Ein kryptografisches Verfahren wird als praktisch sicher bezeichnet, wenn der Aufwand zur Durchführung der Kryptoanalyse die Möglichkeiten eines jeden denkbaren Analysten übersteigt und die erforderlichen Kosten den erwarteten Gewinn bei weitem übertreffen.

praktisch sicher

□

Mit den Festlegungen von Definition 7.3 bleibt nun noch zu klären, wie der Aufwand, der für eine Analyse benötigt wird, zu quantifizieren ist. Das heißt, dass wir ein Maß benötigen, um zu bestimmen, ob ein Analysealgorithmus

effizient ausführbar ist. Zur Angabe eines solchen Maßes bedient man sich der Techniken der Komplexitätstheorie (u.a. [148]).

Effiziente und nicht effiziente Algorithmen

Komplexitäts-theorie

Den Rechenaufwand von Algorithmen bewertet man in der Komplexitätstheorie als eine Funktion in der Länge der Eingabedaten. Um Klassen von Algorithmen im Hinblick auf ihren Aufwand klassifizieren zu können, ohne Realisierungsdetails wie die Eigenschaften der zugrunde liegenden Hardware beachten zu müssen, werden Abschätzungen durchgeführt. Für den Bereich der Kryptografie ist wesentlich, dass es sich dabei häufig um Worst-Case Abschätzungen handelt. Wir werden auf diesen Aspekt noch zurückkommen. Zur Beschreibung solcher Abschätzungen wird in der Regel die Groß-O-Notation mit dem Symbol \mathcal{O} verwendet, die von Edmund Landau (1877 – 1938) eingeführt wurde, weshalb das Symbol \mathcal{O} auch als das Landau-Symbol bezeichnet wird.

Definition 7.4 (Landau-Symbol)

O-Notation

Gegeben seien zwei Funktionen f und g . Wir sagen $f(n) = \mathcal{O}(g(n))$ für $n \in \mathbb{N}$ gegen unendlich, wenn es eine Konstante $c > 0$ und ein $n_0 \in \mathbb{N}$ gibt, so dass gilt:

$$\forall n > n_0 \quad |f(n)| \leq c |g(n)|.$$

□

Definition 7.4 besagt, dass f bis auf einen konstanten Faktor nicht schneller als g wächst. Mit der \mathcal{O} -Notation wird eine Abschätzung für große Werte n der Problemgröße gegeben, bei der man davon ausgeht, dass konstante Faktoren nur noch einen geringen Einfluss auf das Gesamt-Laufzeitverhalten des Algorithmus f besitzen und deshalb ignoriert werden.

Wachstumverhalten

Tabelle 7.1 gibt einen Einblick in das Wachstumsverhalten von Funktionen, die häufig zur Beschreibung der Komplexität von Algorithmen verwendet werden.

Klasse P

Ein Algorithmus wird als effizient bezeichnet, wenn seine Laufzeit durch ein Polynom in der Eingabegröße beschränkt wird. Diese Eigenschaft charakterisiert die Klasse P der mit polynomiellem Aufwand lösbar Probleme. Eine Funktion f wächst polynomiell, wenn es eine natürliche Zahl k gibt, so dass die Laufzeit von f durch $\mathcal{O}(n^k)$ charakterisierbar ist. Zur Klasse P gehören somit Algorithmen mit konstanter, logarithmischer, linearer, quadratischer etc. Laufzeit.

$f(n)$	$n = 10$	$n = 20$	$n = 30$	$n = 50$	$n = 100$
n	10	20	30	50	100
n^2	100	400	900	2500	10 000
n^3	1000	8000	27 000	125 000	1 000 000
n^4	10 000	160 000	810 000	6 250 000	100 000 000
2^n	1024	1 048 576	$\approx 10^9$	$\approx 10^{15}$	$\approx 10^{30}$

Tabelle 7.1: Wachstumsverhalten von f **Beispiel 7.5 (Komplexität von Algorithmen)**

Einen konstanten Aufwand, nämlich $\mathcal{O}(1)$, benötigen beispielsweise Verfahren zum Einfügen bzw. Löschen von Hashtabelleneinträgen, da jeweils nur der Hashwert zu berechnen ist. Eine logarithmische Laufzeit $\mathcal{O}(\log n)$ oder $\mathcal{O}(n \log n)$, die mit wachsendem n nur noch langsam wächst, ist charakteristisch für Algorithmen, die ein Problem durch eine Divide-et-Impera-Strategie lösen. Die Teilprobleme werden unabhängig bearbeitet und danach kombiniert, wie beispielsweise bei Sortierverfahren. Algorithmen besitzen eine lineare Laufzeit, $\mathcal{O}(n)$, wenn ein kleiner Verarbeitungsanteil auf jedes Datum einer Eingabe entfällt. Beispiele sind Worterkennungsverfahren für deterministische kontextfreie Sprachen wie man sie im Übersetzerbau zur Syntaxanalyse verwendet. Eine quadratische Komplexität ist typisch für Algorithmen, die aus zweifach verschachtelten Schleifen aufgebaut sind und paarweise Kombinationen von Datenelementen verarbeiten, wie zum Beispiel der bekannte Dijkstra-Algorithmus zur Lösung des Kürzeste-Wege-Problems für einen gegebenen Startknoten.



Beispiele

Mit NP wird die Klasse der nicht effizient lösbar Probleme bezeichnet. Ein Algorithmus gehört zur Klasse NP, wenn eine korrekt geratene Lösung des Problems durch eine nicht-deterministische Turingmaschine [81] in polynomieller Zeit als korrekt erkannt wird und die Maschine bei falschen

Klasse NP

NP-hart

Lösungen unter Umständen eine exponentielle Rechenzeit benötigt. Ein Problem f heißt NP-hart, wenn sich jedes Verfahren der Klasse NP in polynomieller Zeit auf f reduzieren lässt. f heißt NP-vollständig, wenn das Problem NP-hart ist und selber in NP liegt. Beispiele für NP-vollständige Probleme sind das Erfüllbarkeitsproblem für aussagenlogische Ausdrücke (SAT) oder das Problem des Handlungsreisenden (engl. *travelling salesman*).

Komplexität und Kryptografie

Für den Bereich der Kryptoanalyse sind diese Komplexitätsbetrachtungen und Komplexitätsklassen aber nur eingeschränkt nützlich. Im Kontext kryptografischer Verfahren ist man natürlich daran interessiert, dass einem Angreifer kein effizientes Kryptoanalyseverfahren zur Verfügung steht, um ein gegebenes kryptografisches Verfahren effizient zu brechen. Aufgrund der Worst-Case Vorgehensweise fällt aber ein Algorithmus bereits dann in die Klasse NP, wenn seine Berechnung für *ein* Eingabedatum einen nicht-polynomiellen Aufwand erfordert. Ist aber der Aufwand für andere Eingaben polynomiell, so kann das zugrunde liegende Kryptosystem gebrochen werden; die Charakterisierung der praktischen Sicherheit eines kryptografischen Verfahrens anhand der Einteilung P und NP ist somit mit großer Vorsicht zu betrachten. Dennoch basieren in der modernen Kryptografie Sicherheitsaussagen auf solchen komplexitätstheoretischen Aussagen über Eigenschaften kryptografischer Verfahren.

 $P \neq NP?$

Anzumerken ist ferner, dass bis heute die Frage $P \neq NP$ noch nicht beantwortet werden konnte. Es wird zwar die Ungleichheit vermutet, aber einen Beweis dafür gibt es nicht. Wird ein effizientes Verfahren gefunden, auf das sich alle NP-vollständigen Probleme reduzieren lassen, dann ist auch die Sicherheit kryptografischer Verfahren, für die zurzeit nur Analysemethoden mit der Komplexität der Klasse NP bekannt sind, in Frage gestellt.

7.5 Symmetrische Verfahren

geheimer Schlüssel

Symmetrische Verfahren verwenden gemeinsame, geheime Schlüssel zur vertraulichen Kommunikation. Sie sind von einer großen praktischen Relevanz, da die verwendeten Ver- und Entschlüsselungsalgorithmen in der Regel auf sehr einfachen Operationen (u.a. Shifts, XOR) basieren, die effizient in Hard- oder Software implementierbar sind.

7.5.1 Permutation und Substitution

Klassische, in symmetrischen Verfahren eingesetzte Verschlüsselungsprinzipien sind die Permutation und die Substitution.

Definition 7.5 (Permutation)

Sei A eine endliche Menge. Eine Permutation von A ist eine bijektive Abbildung $f : A \rightarrow A$.

Eine spezielle Klasse von Permutationen sind die Bitpermutationen. Gegeben sei $A = \{0, 1\}^n$, also die Menge aller Binärworte der Länge n . Sei π eine Permutation der Menge $\{1, \dots, n\}$. Eine Bitpermutation ist eine bijektive Abbildung

$$f : \{0, 1\}^n \rightarrow \{0, 1\}^n, \text{ mit} \\ b_1, \dots, b_n \mapsto b_{\pi(1)}, \dots, b_{\pi(n)}.$$

□

Bei einer Permutation, auch Transposition genannt, wird die Anordnung der Klartextzeichen vertauscht. Permutationen werden häufig durch Tabellen beschrieben, die die Vertauschung der Zeichen festlegen.

Beispiel 7.6 (Bitpermutation)

Tabelle 7.2 zeigt ein Beispiel für eine Bitpermutation. Die Einträge sind wie folgt zu interpretieren: Ausgangspunkt ist ein 64-Bit Klartext, dessen Bits permutiert werden. Das 58te Bit wird auf die erste Bit-Position des Chiffretextes permutiert, das 50te auf die zweite Position usw. Diese Permutation wird übrigens im DES-Verfahren als initiale Permutation verwendet.

58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

Tabelle 7.2: Beispiel einer Bitpermutation

▲

Permutationen operieren auf dem gleichen Klartext- und Chiffretextalphabet und führen eine Zeichenweise Ersetzung durch. Verallgemeinert man diese Vorgehensweise, so erhält man allgemeine Substitutionen.

Definition 7.6 (Substitution)*Substitution*

Gegeben seien zwei Alphabete A_1, A_2 . Eine Substitution ist eine Abbildung

$$f : A_1^n \longrightarrow A_2^m.$$

□

Eine Substitutionschiffre ersetzt systematisch Zeichen des Klartextes durch Chiffertextzeichen. Die substituierten Zeichen können auch aus einem anderen Alphabet als dem der Klartextnachrichten entnommen werden.

Produktchiffre

Durch die Kombination der Permutations- und der Substitutionstechnik erhält man die Produktchiffre. Das wohl bekannteste Beispiel einer Produktchiffre ist der DES-Algorithmus. Eine ausführlichere Einführung in klassische Chiffriertechniken ist u.a. in [12] zu finden.

7.5.2 Block- und Stromchiffren

Die zu verschlüsselnden Klartexte besitzen in der Praxis unterschiedliche Längen, während die Verschlüsselungsverfahren in der Regel eine Eingabe fester Länge erwarten. Das führt dazu, dass der Klartext vor der Verschlüsselung in Einheiten (Blöcke oder Zeichen) fester Länge aufgeteilt werden muss. Man unterscheidet Block- und Stromchiffren-Verarbeitung.

Blockchiffren*Blockverarbeitung*

Blockchiffren sind dadurch charakterisierbar, dass sie Eingabeblocks fester Länge verarbeiten, wobei jeder Block mit der gleichen Funktion verschlüsselt wird. Eine typische Blockgröße beträgt 64-Bit bzw. 128 Bit. Abbildung 7.2 veranschaulicht die prinzipielle Arbeitsweise einer Blockchiffre. Ein zu verschlüsselnder Klartext M der Länge m wird in r Blöcke von jeweils der Länge n zerlegt, M_1, M_2, \dots, M_r . Der letzte Block hat dabei die Länge k mit $1 \leq k \leq n$. Um auch für diesen die Blocklänge von n -Bits zu erhalten, wird er mit einem Füllmuster auf n -Bits aufgefüllt. Dieses Auffüllen bezeichnet man auch als Padding (s.u.). Damit bei der Entschlüsselung das Ende des ursprünglichen Klartextes im aufgefüllten Block erkennbar ist, muss die Länge des Füllbereichs oder die Länge des Klartextbereichs in diesem Block kodiert werden.

Padding

Zur Verschlüsselung benötigt man einen Schlüssel K aus dem Schlüsselraum EK und jeder Klartextblock M_i wird separat mit diesem Schlüssel K verschlüsselt, also $C_i = E(M_i, K)$ für $i \in \{1, \dots, r\}$. Der Chiffertext C ergibt sich durch Konkatenation dieser Chiffertextblöcke, also

$$C = C_1 | C_2 | \dots | C_r.$$

Ver- und Entschlüsselung

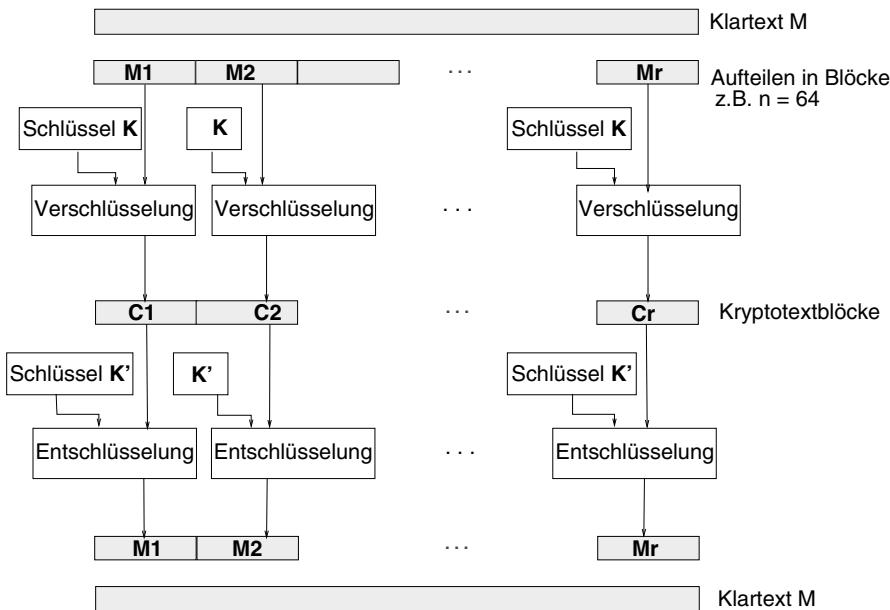


Abbildung 7.2: Funktionsweise einer Blockchiffre

Für die Entschlüsselung werden die Chiffretextblöcke C_1, \dots, C_r wiederum separat mit dem Entschlüsselungsschlüssel K' , für den $f(K) = K'$ gilt, entschlüsselt.

Padding

Blockchiffren liegt die Annahme zugrunde, dass die n Eingabeblocks M_0, M_1, \dots, M_{n-1} vollständig sind. Das heißt, dass jeder Block r Bit lang ist, falls r die Blockgröße der zugrunde liegenden Blockchiffre ist. Beim Zerlegen einer Eingabe M entsteht aber im Allgemeinen ein unvollständiger letzter Block M_{n-1} . Dieser muss dann vor der Anwendung der Blockchiffre aufgefüllt werden. Dieser Vorgang wird als Padding bezeichnet. Abbildung 7.3 fasst die wichtigsten Paddingvorschriften zusammen.

Padding

Im einfachsten Fall (Abbildung 7.3 (a)) wird die Eingabe um eine Zeichenfolge $\dots c$ verlängert (c und d bezeichnen verschiedene Zeichen). Der Beginn des Paddingbereichs ist dabei durch das letzte Vorkommen des Zeichens d eindeutig gekennzeichnet. In der Regel wird dabei eine Bitfolge der Form $100 \dots 0$ verwendet.

Padding-Vorschriften

Eine andere Möglichkeit (Abbildung 7.3 (b)) besteht darin, die Eingabe mit einer konstanten Zeichenfolge $cc \dots c$ aufzufüllen, wobei zusätzlich die Anzahl m der angeführten Zeichen oder die Länge der ursprünglichen Nachricht angegeben werden muss. Die beiden Verfahren können auch kombiniert werden (Abbildung 7.3 (c)).

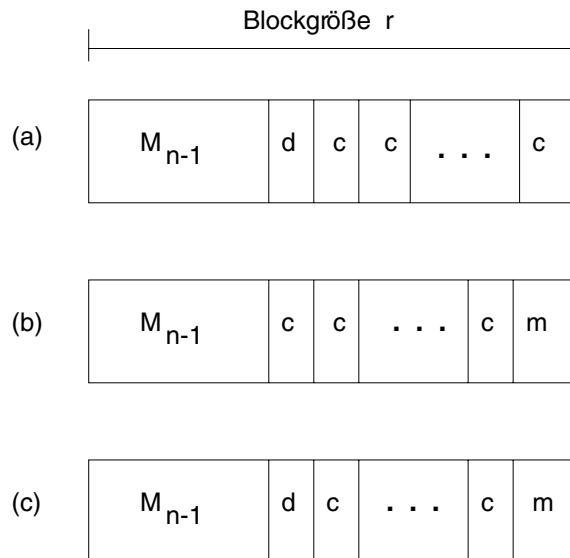


Abbildung 7.3: Paddingvorschriften

Das Padding umfasst jeweils eine bestimmte Mindestlänge (das Zeichen d bzw. den Wert m). Kann diese Länge nicht mehr an den Block angefügt werden, muss ein neuer Block für die Paddinginformation erzeugt werden.

Blockchiffren verwendet man meist dann, wenn bereits vor Beginn der Verschlüsselung der Klartext vollständig vorliegt. Beispiele hierfür sind das Verschlüsseln von Dateien auf der Festplatte und die Verschlüsselung von Nachrichten wie E-Mails. In Anwendungsbereichen, in denen es nicht praktikabel ist, mit der Übertragung so lange zu warten, bis genug Zeichen für einen ganzen Block vorhanden sind, setzt man Stromchiffren ein. Beispiele hierfür sind die verschlüsselte Sprachübertragung oder das Verschlüsseln eines Stroms von Tastatur-Eingaben.

Stromchiffre

Stromverarbeitung

Stromchiffren sind sequentielle (lineare) Chiffren, die eine Folge von kleinen Klartexteinheiten mit einer variierenden Funktion verschlüsseln. In Abgrenzung zu den i.d.R. größeren Einheiten der Blöcke bezeichnet man die Klartexteinheiten einer Stromchiffre meist als Zeichen. Gängige Zeichengrößen eingesetzter Stromchiffren sind 1 Bit oder 1 Byte.

Pseudozufallsfolge

Bei der Untersuchung der Sicherheitseigenschaften kryptografischer Verfahren haben wir bereits gesehen, dass die größte Sicherheit (absolut sicher) erreicht werden kann (vgl. Abschnitt 7.4.3), wenn die Schlüsselzeichen zufällig gewählt sind und der Schlüssel die gleiche Länge wie der Klartext besitzt. Wir hatten aber weiterhin auch schon gesehen, dass das einzige Ver-

fahren, das diese Eigenschaften erfüllt, nämlich das One-time-pad, in der Praxis ungeeignet ist. Es liegt somit nahe, Mechanismen einzusetzen, um die Eigenschaften des One-time-pads zu approximieren, ohne dessen Aufwand für den Schlüsselaustausch betreiben zu müssen.

Die Lösung besteht in der Verwendung so genannter Pseudozufallsfolgen anstelle der zufälligen Zeichenfolgen des One-time-pad Schlüssels. Eine Pseudozufallsfolge wird durch einen deterministischen Prozess unter Verwendung eines Initialisierungswertes generiert und muss Eigenschaften einer echt zufälligen Folge aufweisen. Benötigt werden Zufallszahlengeneratoren (vgl. [28]), die Folgen generieren, so dass es keinen polynomiellen Algorithmus gibt, der ohne Kenntnis des Initialwertes aus einem Abschnitt der Folge das nächste Zeichen mit einer Trefferwahrscheinlichkeit größer 0.5 vorhersagen kann.

Zur Realisierung von Generatoren für Schlüsselströme für Stromchiffren werden in der Praxis in der Regel linear rückgekoppelte Schieberegister (engl.: *Linear Feedback Shift Registers, LFSR*) eingesetzt. LFSR sind hierfür sehr geeignet, da sie Pseudozufallsfolgen mit guten statistischen Eigenschaften und mit großer Periode erzeugen können und sehr effizient in Hardware implementierbar sind. Ein solches linear rückgekoppeltes Schieberegister der Länge n -Bit besteht aus n 1-Bit Speicherzellen. Über einen Taktgeber wird der Datenfluss kontrolliert, indem bei jedem Takt in den Folgezustand gewechselt wird. Dabei wird der Inhalt der Speicherzelle 0 ausgegeben, der Inhalt der i -ten Speicherzelle in die $(i - 1)$ -te Zelle weiter geschoben, mit $(i = 1, 2, \dots, n - 1)$ wird. Die $n - 1$ -te Speicherzelle speichert das so genannte Rückkopplungsbit, das eine festgelegte Menge von Speicherzellen mittels XOR verknüpft. Da ein LFSR linear ist, wird in kryptografischen Anwendungen in der Regel eine Kombination von LFSR bzw. eine Kombination eines LFSR mit nicht-linearen Funktionen verwendet.

Bei der Verwendung von Zufallszahlengeneratoren müssen die Kommunikationspartner dann nur noch über den gleichen Generator zur Erzeugung der Pseudozufallsfolgen verfügen und sich über den gleichen Initialwert verständigen. Auf dieser Basis sind anschließend beide Partner unabhängig voneinander in der Lage, die Schlüsselfolge zur Ver- und Entschlüsselung zu erzeugen.

Abbildung 7.4 verdeutlicht die allgemeine Arbeitsweise einer Stromchiffre. Über die Eigenschaften des Zufallszahlengenerators wird sichergestellt, dass ein Angreifer – wie bei der absoluten Sicherheit gefordert – aus der Beobachtung von Chiffretexten nicht darauf schließen kann, welches Zeichen als Nächstes generiert wird. Der für den Schlüsselaustausch zu leistende Aufwand beschränkt sich auf den sicheren Austausch eines relativ kurzen Initialwertes.

Zufallszahlen-
generator

Schieberegister

Initialwert

Arbeitsweise

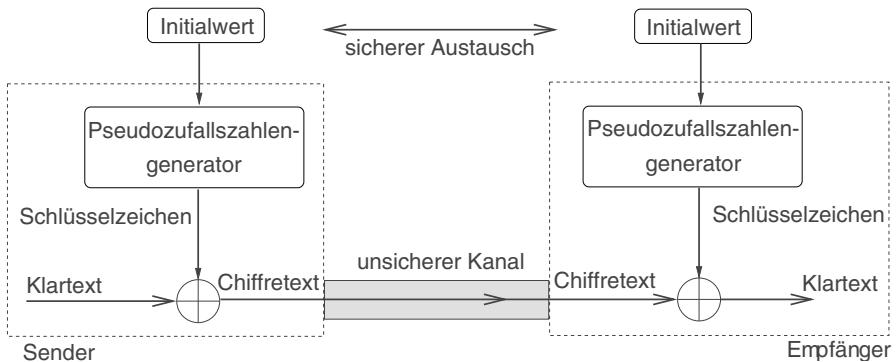


Abbildung 7.4: Funktionsweise einer Stromchiffre

XOR

Im Gegensatz zu Blockchiffren werden bei Stromchiffren sehr einfache Verknüpfungsoperationen zur Ver- und Entschlüsselung verwendet. Sind Klartext und Schlüssel Folgen von Binärzeichen, so ist die Verschlüsselungsoperation das XOR, also die Addition modulo 2, und die Entschlüsselung erfolgt ebenfalls mittels XOR. Für die kryptografische Sicherheit des Systems ist der Generator ausschlaggebend. Ein bekanntes Beispiel für eine Stromchiffre ist der A5-Algorithmus [71], der im GSM-Standard für die Verschlüsselung der Kommunikation auf der Luftschnittstelle genutzt wird (vgl. Kapitel 15.1).

Das von Vernam 1917 entwickelte One-Time Pad (siehe auch S. 296) ist eine beweisbar sichere Stromchiffre, die für jede Klartextfolge einen eigenen Schlüsselstrom unter Nutzung eines echten Zufallszahlengenerators erzeugt. Die einmalige Verwendung eines Schlüsselstroms ist eine zentrale Bedingung für die perfekte Sicherheit dieser und generell von Stromchiffren. Die Verwendung von Stromchiffren ist deshalb trotz ihrer Einfachheit in der Praxis häufig problematisch und führt (siehe Kapitel 15) immer wieder zu ganz erheblichen Sicherheitsproblemen. Da bei Stromchiffren die Klartexte mittels XOR mit einem Schlüsselstrom *Key* verknüpft werden, $C = M \text{ XOR } Key$, muss beim Einsatz der Chiffre sichergestellt werden, dass für jede Nachricht eine neuer Schlüsselstrom *Key* generiert und verwendet wird. Andernfalls kann ein Angreifer auch ohne Kenntnis des Schlüssels *Key* Klartexte erhalten.

Probleme

Angriff

Betrachten wir dazu zwei Klartexte M_1 und M_2 , die mit dem gleichen Schlüssel *Key* mittels einer Stromchiffre verschlüsselt werden. Es gilt also: $C_1 = M_1 \text{ XOR } Key$ und $C_2 = M_2 \text{ XOR } Key$.

Aufgrund der XOR-Eigenschaften gilt aber auch:

$$C_1 \text{ XOR } C_2 = (M_1 \text{ XOR } Key) \text{ XOR } (M_2 \text{ XOR } Key) = M_1 \text{ XOR } M_2.$$

Das heißt, dass ein Angreifer allein aus dem Abhören von transferierten

Chiffretexten C1 und C2 das XOR der zugehörigen Klartexte M1 und M2 gewinnen kann. Falls er nun Teile eines der Klartexte kennt, so kennt er unmittelbar auch den entsprechenden Teil des Klartextes der zweiten Nachricht. Das heißt natürlich insbesondere, dass ein Angreifer, dem es gelingt, eine eigene Nachricht M2 mit der Stromchiffre und dem gleichen Schlüssel verschlüsseln zu lassen, direkt in der Lage ist, sämtliche anderen Klartexte M, die auch mit dem Schlüssel verschlüsselt wurden, zu entschlüsseln. Diese Art von Angriffen heißen Known-Plaintext Angriffe (siehe Seite 359). Sie sind in heutigen Systemen relativ einfach durchzuführen. Ein Angreifer kann beispielsweise über eine Ping-Nachricht an ein Opfersystem schicken und dieses dazu bringen, eine Nachricht mit bekanntem Inhalt verschlüsselt zu versenden (z.B. die Ping-Antwort). Mit einem frei verfügbaren Paket-Sniffer Programm kann dann der Angreifer die verschlüsselten Daten abfangen und für seine Angriffsziele nutzen. Noch einfacher wird das Sammeln von Klartext/Kryptotext-Paaren für einen Angreifer, wenn er z.B. drahtlose oder mobile Funknetze abhört, bei denen zur Nutzerauthentisierung ein so genanntes Challenge-Response Protokoll (vgl. Abschnitt 10.2.3) durchgeführt wird. Dies ist sowohl bei der Mobilkommunikation via GSM/GPRS oder UMTS der Fall, als auch beim Zugang zu einem WLAN.

ChaCha20

Die symmetrische Blockchiffre AES (s.u.) ist als Basis für eine Stromchiffre weit verbreitet. Sie benötigt jedoch dedizierte Hardware-Beschleuniger, um effizient zu arbeiten. Zudem ist der AES schwierig korrekt zu implementieren, so dass Implementierungen häufig anfällig gegen Seitenkanalattacken, wie Timing-Attacken, sind. Benötigt werden effiziente Stromchiffren, die keine dedizierte Hardware benötigen, einfach umzusetzen sind und eine hohe Sicherheit bieten.

Auf D. J. Bernstein geht die Entwicklung einer ganzen Familie von solchen Stromchiffren zurück. Die Basis bildet eine Chiffre namens Salsa [21]. Die ChaCha20-Chiffre [22] definiert eine leicht modifizierte Variante von Salsa, wobei die Zahl 20 angibt, dass der ChaCha insgesamt 20 Runden durchläuft. Der ChaCha20 ist eine 256-Bit-Stromchiffre, die einen pseudozufälligen Bitstrom, die Schlüsselbits, basierend auf einem Zählerwert, der sukzessive hochgezählt wird, generiert. ChaCha20 ist sehr effizient konzipiert und nutzt lediglich die drei Operationen Add-Rotate-XOR (ARX), um aus einem 256-Bit Schlüssel, einer 64-Bit Nonce und einem 64-Bit Zählerwert einen 512-Bit Block des Schlüsselstroms zu erzeugen. Durch die ausschließliche Nutzung von Add-Rotate-XOR Operationen werden Timing-Angriffe auf die Chiffre verhindert. Wie bei Stromchiffren üblich, erfolgt der eigentliche Verschlüsselungsschritt durch eine xor-Verknüpfung des Stroms der Schlüsselbits mit den Bits des Klartextes. Das ChaCha20-Verfahren ist

ChaCha20

sehr effizient, benötigt keinen speziellen Hardware-Beschleuniger und ist insbesondere für eingebettete Systeme oder auch mobile Endgeräte sehr interessant. Im Vergleich zu AES arbeitet ChaCha20 ungefähr dreimal schneller und kommt zudem ohne Hardwarebeschleuniger aus.

Poly1305

Der ChaCha20 wird häufig in Kombination mit einem ebenfalls von D. Bernstein entwickelten, sehr effizient arbeitenden MAC-Verfahren (siehe Abschnitt 8.1.4), dem Poly1305 [23], verwendet. Poly1305 generiert ein Authentisierungs-Token, das von einem Schlüssel abhängt (keyed token). Die integrierte Variante ChaCha20-Poly1305 ist ein AEAD-Verfahren (vgl. Seite 315), das sowohl Verschlüsselung als auch Nachrichtenauthentizität bereit stellt.

Durchgeführte Sicherheitsanalysen des ChaCha20-Poly1305 (vgl. [90])⁶ haben gezeigt, dass die ChaCha20-Chiffre resistent gegen die bekannten Angriffe auf Verschlüsselungsverfahren, wie differentielle Analyse, lineare Kryptoanalyse, oder auch algebraische Angriffe ist. Weiterhin haben die Untersuchungen gezeigt, dass Seitenkanalangriffe zwar möglich sind, diesen jedoch mit bekannten Abwehrmaßnahmen begegnet werden kann. Die Analysen attestieren, dass in ChaCha20-Poly1305 keine Schwachstellen aufgedeckt wurden.

Mit dem RFC 7539 (vgl. <https://tools.ietf.org/pdf/rfc7539.pdf>) wurde der ChaCha-Algorithmus in 2015 von der IETF standardisiert. Dieser Standard wurde in 2016 in dem RFC 7905 (vgl. <https://tools.ietf.org/html/rfc7905>) übernommen. Der RFC7905 beschreibt die Nutzung von ChaCha20 und Poly1305 für das Transport Layer Security-Protokoll (TLS) (vgl. Abschnitt 14.4). ChaCha20-Poly1305 wird ab TLS 1.2 unterstützt und ist vielfältig im Einsatz, wie beispielsweise in OpenSSL, in OpenBSD als Ersatz für den RC4-Algorithmus, oder auch als Verfahren im IKE und IPsec-Protokoll.

7.5.3 Betriebsmodi von Blockchiffren

Betriebsmodus

Setzt man zur Erzeugung des Schlüsselstroms, der bei einer Stromchiffre benötigt wird, eine Blockchiffre als Pseudozufallsfolgengenerator ein, so erkennt man unmittelbar, dass der Übergang zwischen Block- und Stromchiffre fließend ist. Man spricht in diesem Zusammenhang von einer stromorientierten Betriebsart der Blockchiffre. Die Betriebsart einer Blockchiffre legt fest, wie die Verschlüsselung mehrerer Klartextblöcke erfolgt. In diesem Abschnitt werden die wichtigsten Betriebsmodi von Blockchiffren kurz vorgestellt. In der Praxis werden heutzutage meist der Counter-Modus (CTR) bzw. der AEAD-Modus für eine authentisierte Verschlüsselung genutzt. Eine häufig eingesetzte Implementierung des AEAD

⁶ Ein Update aus 2017 findet man unter <https://www.cryptrec.go.jp/estimation/cryptrec-ex-2601-2016.pdf>

ist der Galois-Counter Modus (GCM), auf den wir auf Seite 7.5.3 noch eingehen werden. Aber auch der Cipher Block Chaining Modus (CBC) wird noch vielfach bei Blockchiffren verwendet, obwohl dieser Modus anfällig gegen Padding-Oracle-Angriffe ist.

Electronic Code Book (ECB)

In der einfachsten Betriebsart wird jeder Klartextblock einzeln mit demselben Schlüssel verschlüsselt; dies entspricht dem bereits eingeführten Schema für Blockchiffren (vgl. Abbildung 7.2). Der Modus hat den Vorteil, dass die Entschlüsselung der Chiffretextblöcke in beliebiger Reihenfolge stattfinden kann. Tritt beim Transfer eines Chiffretextblocks ein Übertragungsfehler auf, so kann dieser vom Empfänger nicht korrekt entschlüsselt werden. Die Störung beschränkt sich aber auf diesen einen Block, d.h. sie pflanzt sich nicht fort.

ECB

Sicherheitsprobleme des ECB-Modus ergeben sich daraus, dass gleiche Klartextblöcke stets auf gleiche Schlüsseltextblöcke abgebildet werden. Es ist damit für einen Angreifer relativ einfach möglich, Informationen über Teile einer verschlüsselten Nachricht durch Vergleiche mit den ihm bereits bekannten Klartext-/Schlüsseltext-Paaren zu gewinnen. Eine Klartextnachricht M wird zunächst in eine Folge von Blöcken, $M = M_1, \dots, M_r$, aufgeteilt und die zugehörigen Chiffretextblöcke $C = C_1, \dots, C_r$ werden unabhängig voneinander erstellt und zum Empfänger übertragen. Ein Angreifer kann einen Chiffretextblock C_j aus dieser Folge entfernen, einen Block C_j modifizieren, die Reihenfolge der Blöcke verändern oder auch einen eigenen Block C'_j in die Folge einspielen, ohne dass dies bei der Entschlüsselung durch den Empfänger bemerkbar ist. Das Einfügen eines Blockes mittels einer Wiedereinspielung ist dann besonders einfach möglich, wenn über einen langen Zeitraum der gleiche Schlüssel zur Verschlüsselung von Klartexten verwendet wurde. Zeichnet ein Angreifer einen dieser Chiffretextblöcke auf und spielt ihn zu einem späteren Zeitpunkt wieder ein, so präsentiert er dem Empfänger einen Chiffretext, der nach seiner Entschlüsselung einen sinnvollen Klartext enthält (z.B. die Überweisung eines Geldbetrags).

Sicherheitsprobleme

Aufgrund der angesprochenen Probleme sollte der ECB-Modus nur für kurze Nachrichten verwendet werden und es sollte durch zusätzliche Maßnahmen wie Zeitstempeln oder Sequenznummern sichergestellt werden, dass eine Wiedereinspielung eines bereits übertragenen Blockes erkennbar ist.

Anwendung

Cipher Block Chaining (CBC)

Angriffe auf den ECB-Modus basieren hauptsächlich auf der unabhängigen Verarbeitung einzelner Blöcke. Hier setzt der CBC-Modus an, der eine Verkettung der Blöcke durchführt, indem jeder Klartextblock vor der

CBC

Verschlüsselung mit dem vorhergehenden Kryptotextblock XOR-verknüpft wird. Abbildung 7.5 verdeutlicht diese Verkettung. Für die Eigentliche Ver- und Entschlüsselung wird zusätzlich der Verschlüsselungsschlüssel K für jeden Verschlüsselungsschritt E bzw. Entschlüsselungsschritt D als Parameter benötigt. Die Verschlüsselung ist festgelegt durch

$$C_i = E(M_i \oplus C_{i-1}, K), i \in \{1, \dots, r\}.$$

Um auch den ersten Klartextblock mit einem anderen Block zu verketten, benötigt man noch die Vereinbarung eines Startwertes C_0 . Dieser wird als Initialisierungsvektor (IV) bezeichnet. Entschlüsselt wird mit

$$M_i = D(C_i, K) \oplus C_{i-1}, i \in \{1, \dots, r\}$$

unter Verwendung des gleichen Initialisierungsvektors.

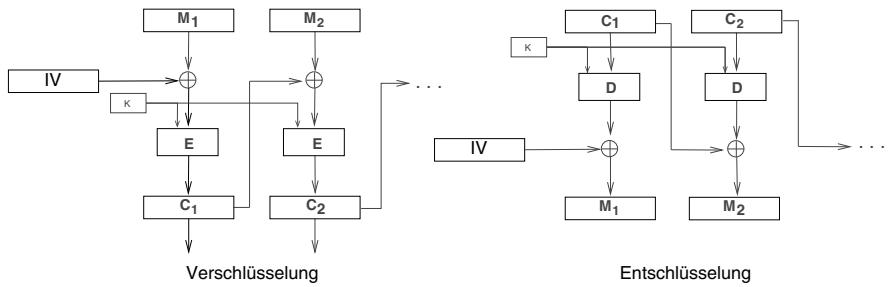


Abbildung 7.5: Der CBC-Modus

Unter dem CBC-Modus ist jeder Schlüsseltextblock vom dazugehörigen Klartextblock und dem vorangegangenen Schlüsseltextblock abhängig. Das Auftreten gleicher Schlüsseltextblöcke $C_i = C_j$, mit

$$C_i = E(M_i \oplus C_{i-1}, K) \text{ und } C_j = E(M_j \oplus C_{j-1}, K),$$

bei zugrunde liegenden gleichen Klartextblöcken M_i und M_j , $M_i = M_j$ einer zu verschlüsselnden Nachricht M , ist dadurch sehr selten. Für die Verschlüsselung unterschiedlicher Nachrichten M und M' muss jeweils ein neuer Initialisierungsvektor verwendet werden, so dass auch bei der Beibehaltung des Verschlüsselungsschlüssels K unterschiedliche Schlüsseltextblöcke erstellt werden.

Ein Nachteil der Verkettung besteht in der Fehlerfortpflanzung beim Auftreten von Übertragungsfehlern. Wird ein Schlüsseltextblock C_i fehlerhaft übertragen, so wird nicht nur dieser fehlerhaft entschlüsselt, sondern auch der nachfolgende Klartextblock M_{i+1} übernimmt alle fehlerhaften Bits von

C_i , da gilt: $M_{i+1} = D(C_{i+1}, K) \oplus C_i$. Eine weitere Fortpflanzung des Fehlers findet jedoch nicht statt, weil die korrekte Entschlüsselung von C_{i+2} nur von der Korrektheit von C_{i+2} und C_{i+1} abhängt. Da durch die Verkettung und der damit verbundenen Fehlerfortpflanzung aber auch das unbemerkte Einspielen von falschen Chiffretextblöcken weitgehend unmöglich gemacht wird, entpuppt sich somit der Nachteil für viele Anwendungsbereiche sogar als ein Vorteil.

Der CBC-Modus bietet eine höhere Sicherheit als der ECB-Modus, falls der zu verschlüsselnde Klartext aus mehr als einem Block besteht. Der CBC-Modus wird häufig zum Verschlüsseln von Dateien verwendet, da diese meist als Ganzes ver- und entschlüsselt werden, so dass der wahlfreie Zugriffe auf dedizierte Blöcke nicht benötigt wird.

Anwendung

Output Feedback und Cipher Feedback

Jede Blockchiffre kann mithilfe des Output Feedback-Modus (OFB) bzw. des Cipher Feedback-Modus (CFB) auch als Stromchiffre genutzt werden. In beiden Betriebsmodi wird die Blockchiffre als Pseudozufallsfolgengenerator verwendet. Sowohl beim Sender als auch beim Empfänger arbeitet die Chiffre nur generierend und erzeugt denselben Schlüsselstrom, der mit den zu übertragenden Klartextdaten XOR-verknüpft wird. Die Schlüssel werden durch die Verschlüsselung eines Eingabeblocks mittels der Blockchiffre und eines gemeinsamen Schlüssels erzeugt. Der Empfänger muss diese generierte Zeichenfolge nicht entschlüsseln, sondern generiert unter Einsatz des vorab ausgetauschten, gemeinsamen Schlüssels für die Blockchiffre seinerseits die gleiche Folge von Schlüsselzeichen. In beiden Modi ist der erste verschlüsselte Eingabeblock ein Initialisierungsvektor, der zunächst zum Empfänger übertragen werden muss. Der Austausch des Initialisierungsvektors kann unverschlüsselt erfolgen.

generierende
Blockchiffre

Der OFB- und der CFB-Modus unterscheiden sich im Wesentlichen nur darin, wie die Eingabeblocke erzeugt werden, die die Blockchiffre zur Schlüsselgenerierung benötigt. Sollen nun bei Verwendung des OFB-Modus (siehe Abbildung 7.6) jeweils k Bit des Klartextes verschlüsselt und zum Empfänger übertragen werden, so wird zunächst eine Verschlüsselungsoperation der Blockchiffre durchgeführt, die n -Bit Blöcke eines Datums (in Abbildung 7.6 als Eingabeblock beschrieben) verschlüsselt. D.h. die Blockchiffre wird nicht auf den eigentlich zu verschlüsselnden Klartext angewandt, sondern dient nur zur Generierung von Schlüsselbits. k Bits des resultierenden Ausgabeblocks werden als Schlüsselbits verwendet und vor der Übertragung mit den k Klartextbits XOR-verknüpft. Gleichzeitig ersetzen sie auch k Bits des Eingabeblocks (Schieberegister) für die Blockchiffre, der zu Beginn der Verschlüsselung den Initialisierungsvektor enthält. Abbil-

OFB

dung 7.6 verdeutlicht noch einmal, dass die eingesetzte Blockchiffre sowohl vom Sender als auch vom Empfänger zum Verschlüsseln, aber nie zum Entschlüsseln verwendet wird.

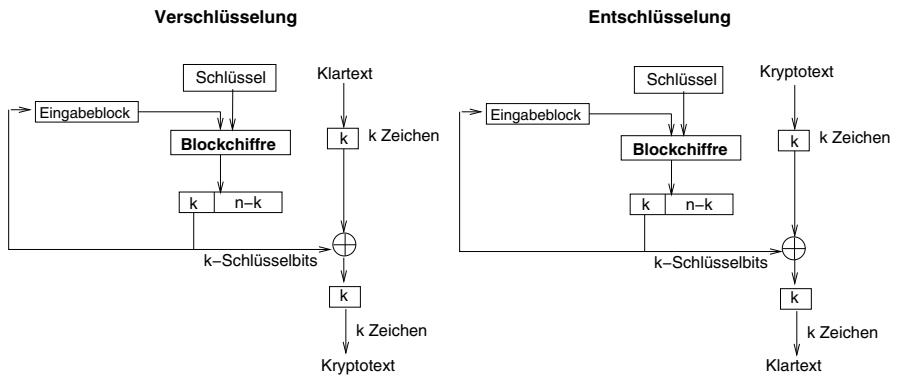


Abbildung 7.6: Der OFB-Modus

Bei Synchronisationsfehlern kann der Empfänger keines der nachfolgenden Zeichen korrekt entschlüsseln. Übertragungsfehler, die den Wert einzelner Bits ändern, wirken sich dagegen nur auf die Stelle aus, an der sie auftreten. Deswegen bietet diese Betriebsart keinen Schutz vor Manipulationen einzelner Chiffretextzeichen.

CFB

Rückkopplung

Der CFB-Modus (vgl. Abbildung 7.7) arbeitet ähnlich wie der OFB-Modus mit dem Unterschied, dass für das Ersetzen von k Bits des Eingabeblocks der Blockchiffre nicht auf einen Teil des Ausgabeblocks, sondern auf die erzeugten Chiffretextzeichen zurückgegriffen wird. Aufgrund der Rückkopplung beeinflusst eine Änderung eines Chiffretextzeichens alle folgenden Zeichen, so dass Übertragungsfehler und Manipulationen erkannt werden können. Bei

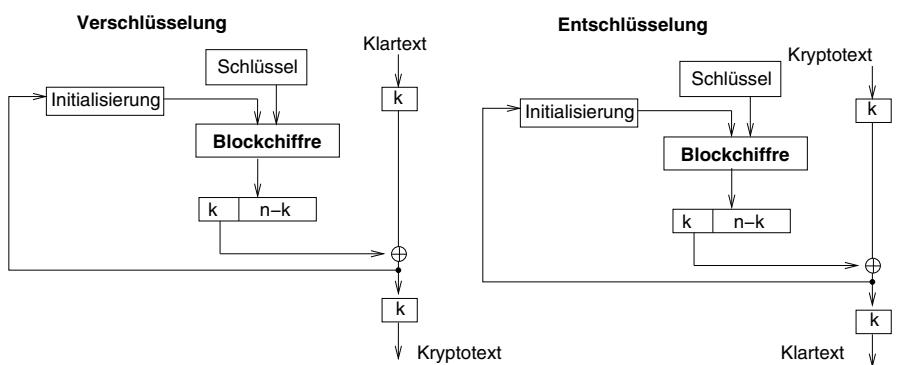


Abbildung 7.7: Der CFB-Modus

Synchronisationsfehlern kann auch mit dieser Betriebsart keines der folgenden Zeichen entschlüsselt werden, außer für $k = 1$. In diesem Fall wird der Bitstrom wieder korrekt entschlüsselt, sobald das fehlende oder zusätzliche Bit nicht mehr den Eingabeblock der Empfängerseite verfälscht.

Counter Modus (CTR)

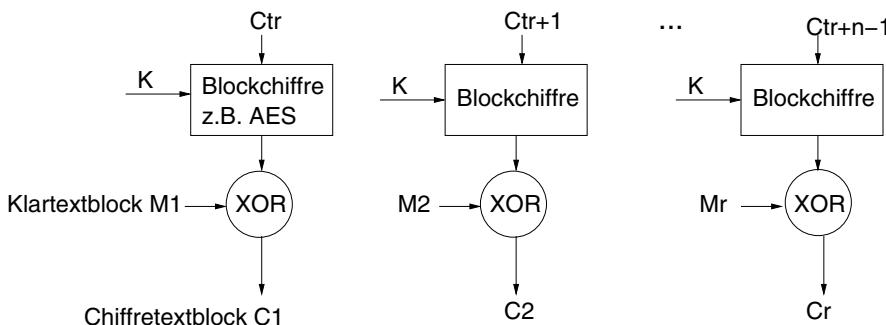
Der Counter Modus dient ebenso wie OFB und CFB dazu, eine Blockchiffre als Stromchiffre einsetzen zu können. Der Modus wurde bereits 1979 von Diffie und Hellman in [54] vorgestellt, hat aber erst in den letzten Jahren stark an Bedeutung gewonnen. Er wird heute insbesondere zusammen mit der Blockchiffre AES vielfach eingesetzt und wird zum Beispiel auch in dem aktuellen Standard 802.11i zur sicheren WLAN Kommunikation verwendet.

Abbildung 7.8 veranschaulicht die Arbeitsweise des CTR-Modus.

CTR

Arbeitsweise

Verschlüsselung von Blöcken M_1, \dots, M_r



Entschlüsselung von Blöcken C_1, \dots, C_r



Abbildung 7.8: Der CTR-Modus

Die dem Modus zugrunde liegende Idee besteht darin, den Klartext in r Blöcke mit jeweils der Länge n Bit (z.B. bei AES in 128-Bit Blöcke) aufzuteilen. Jeder Block wird in zwei Phasen verschlüsselt. In der ersten Phase wird eine Zufallszahl (Nonce) mit einem n -Bit Zähler-Wert Ctr verknüpft, z. B. mittels xor . Dies dient als Eingabe für die Blockchiffre, wie zum Beispiel AES, und wird unter Nutzung eines geheimen Schlüssels K verschlüsselt. In der zweiten Phase wird das Ergebnis dieses Verschlüsselungsschrittes, wie bei Stromchiffren üblich, mittels xor mit dem Klartextblock verknüpft. Der Zähler Ctr startet mit einem beliebigen Anfangszustand und wird für jeden zu verschlüsselnden Klartextblock verändert. Dies kann durch ein einfaches Hochzählen des Zählers realisiert werden. Wichtig ist, dass sichergestellt wird, dass die Zählerwerte sich über einen langen Zeitraum hinweg nicht wiederholen. Dazu dient die Verknüpfung des Zählerwertes mit der Zufallszahl. Zur Entschlüsselung wird ebenfalls die Zufallszahl, die kein Geheimnis ist, und der Zähler Ctr benötigt. Zusammengefasst ergibt sich für die Verschlüsselung mittels CTR folgender Ablauf:

- Initialisierung des Zählers Ctr_1 durch eine Verknüpfung des Zählers mit einer Zufallszahl.
- Folge der Zählerstände des Zählers Ctr :

$$Ctr_i = Ctr_{i-1} + 1 \text{ für } i = 2, \dots, n;$$
- Verschlüsselung des i -ten Klartextblockes M_i :

$$C_i = M_i \ xor \ E(Ctr_i, K)$$
- Entschlüsselung des i -ten Chiffertextblockes C_i :

$$M_i = C_i \ xor \ E(Ctr_i, K)$$

Da die Erzeugung des Schlüsselstroms nur von der Nonce und dem Zähler abhängt, kann die Berechnung der Phase 1 vorab erfolgen. Die eigentliche Verschlüsselung des Klartextstroms ist dann in der Phase 2 sehr effizient möglich, wenn die Schlüsselbits bereits generiert sind.

Die Sicherheit des CTR-Modus ist bereits vielfältig analysiert worden, so hat Bellare beispielsweise bereits 1997 und in einer Überarbeitung im Jahre 2000 die beweisbare Sicherheit des CTR⁷ gezeigt.

Der Vorteil des CTR-Modus gegenüber dem OFB-Modus besteht darin, dass man einen wahlfreien Zugriff (random access) auf jeden Klartext bzw. Chiffertextblock hat, so dass die einzelnen Ver- und Entschlüsselungsoperationen für die Blöcke auch parallel durchgeführt werden können. Der Modus bietet damit eine sehr effiziente Verschlüsselungs-Option. Dieser wahlfreie Zugriff ist auch hilfreich für die Verschlüsselung von Massen-Daten-Volumina,

⁷ Siehe <http://www-cse.ucsd.edu/users/mihir/papers/sym-enc.html>

wie z.B. die Verschlüsselung ganzer Festplatten-Laufwerke oder von ZIP-Archiven.

Authenticated Encryption with Associated Data (AEAD)

Die dem AEAD-Konzept zugrundeliegende Fragestellung, Nachrichtenvertraulichkeit sowie Authentizität zu gewährleisten, ist altbekannt und kann durch die Komposition von Verschlüsselungs- und MAC-Verfahren gelöst werden: Seien M eine Klartextnachricht, E ein symmetrisches Verschlüsselungsverfahren mit dem Schlüssel K_c und HMAC ein Message Authentication Code (vgl. Seite 8.1.4) mit dem Schlüssel K_{mac} , sowie AD Daten, die authentisiert, aber nicht verschlüsselt werden sollen, wie beispielsweise Header-Daten eines Nachrichtenpakets. Eine authentisierte Verschlüsselung kann klassisch durch Komposition wie folgt erzeugt werden:

$$C = E(M, K_c), \text{AUT} = \text{HMAC}(C \mid AD, K_{mac}), \\ \text{authenticated-message} = AD, \text{AUT}, C.$$

Ziel ist es, mit AEAD-Kryptoprimitiven eine solche Komposition als ein effizientes, integriertes Verfahren umzusetzen.

AEAD (vgl. u.a. [154], RFC 5116) beschreibt einen Betriebsmodus bei Blockchiffren, bei dem Vertraulichkeit der Nachricht sowie Integrität und Authentizität der Nachricht und der assoziierten Daten sichergestellt wird. Im Unterschied zu einem komponierten Verfahren erfolgen die Berechnung des Kryptotextes und die Berechnung eines Authentifizierungs-Tags in einem Schritt. Ein AEAD-Verfahren bietet eine authentisierte Verschlüsselung (Authenticated Encryption (AE)) und eine authentisierte Entschlüsselung (Authenticated Decryption).

AEAD

Eine authentisierte Verschlüsselung basiert auf einem symmetrischen Verschlüsselungsverfahren, wie AES, verwendet vier Eingabeparameter und berechnet zwei Ausgabewerte $(C, T) = AE(M, N, K, AD)$.

Dabei sind M der zu verschlüsselnde und zu authentisierende Klartext, K der symmetrische Schlüssel, N eine Nonce (auch häufig als IV bezeichnet) und AD die assoziierten Daten, die zu authentisieren, aber nicht zu verschlüsseln sind. Die AE-Verschlüsselung liefert als Ausgabe den Kryptotext C der Nachricht sowie einen Authentifikations-Tag T . Es wird dringend empfohlen, bei der authentisierten Verschlüsselung die Reihenfolge Encrypt-then-MAC einzuhalten.

Encrypt-then-MAC

Die authentisierte Entschlüsselung $AD(C, N, K, AD, T)$ liefert den Klartext M oder eine Fehlermeldung, falls die Eingaben als nicht authentisch erkannt werden.

AES-GCM-Verschlüsselung

Der AEAD kann mit Standardverschlüsselungsverfahren umgesetzt werden, beispielsweise mit ChaCha20-Poly1305 oder mit AES im Galois-Counter-Modus (GCM) mit 128- oder 256-Bit-Schlüsseln (vgl. Abbildung 7.9).

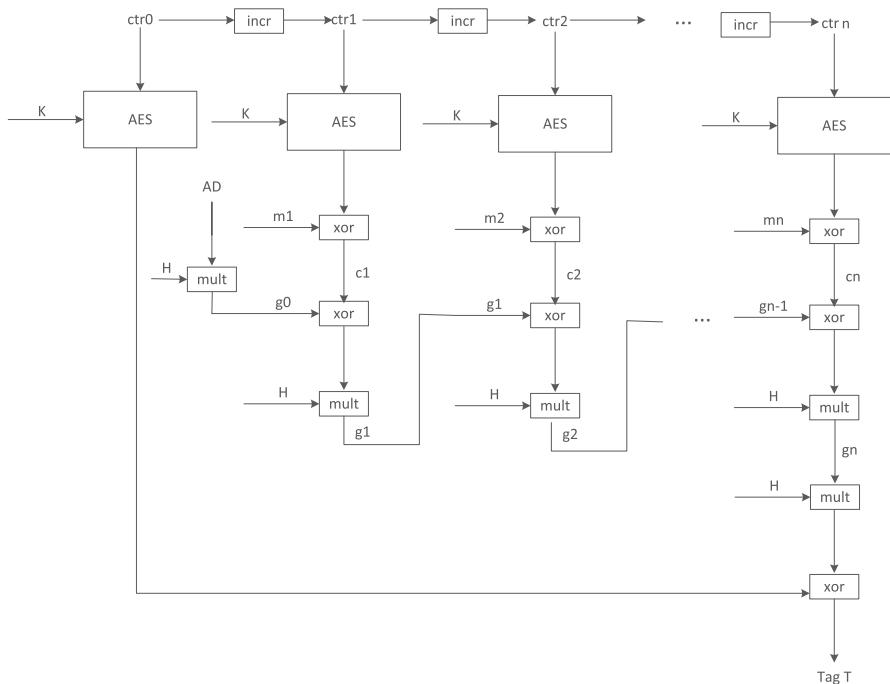


Abbildung 7.9: AEAD-Verschlüsselung im AES-GCM-Modus

Gegeben sei ein Klartext M , ein Initialisierungsvektor IV , ein Schlüssel K und Daten AD sowie der authentisierte Kryptotext:

$$(C, T) = \text{AES-GCM}(M, IV, K, AD)$$

Der GCM-Modus verarbeitet den Klartext $M = m_1, m_2, \dots, m_n$ in Eingabeblocks der Länge 128 Bit und führt eine blockweise Verschlüsselung im Counter-Modus (CTR) durch. Der Initialisierungsvektor IV entspricht der Nonce im allgemeinen AEAD-Schema. Aus dem IV wird ein initialer Counterwert ctr_0 abgeleitet. Abbildung 7.9 zeigt die Verschlüsselung der Blöcke m_i . Die Verschlüsselung von m_i erfolgt im Countermodus, wobei gilt:

- $ctr_0 = f(IV)$, Initialisierung des Zählers,
- $ctr_i = ctr_{i-1} + 1, i \in \{1, \dots, n\}$,
- $c_i = AES(ctr_i, K)$.

Die Authentizität des Kryptotextes und der assoziierten Daten AD erfolgt unter Nutzung einer verketteten Galois-Feld-Multiplikation wie folgt: MAC

- Berechne Konstante H mit $H = AES((0, \dots, 0), K)$, d. h. es wird ein Block bestehend aus Nullen verschlüsselt. H wird auch als Hash-Subkey bezeichnet.
- $g_0 = AD \times H$.
- Für $i \in \{1, \dots, n\}$ berechne $g_i = (g_{i-1} \text{ xor } c_i) \times H$. Multiplikationen werden im Galois-Feld $GF(2^{128})$ durchgeführt unter Nutzung des Polynoms $P(x) = x^{128} + x^7 + x^2 + x + 1$.
- Berechne Authentifikations-Tag T , mit $T = (g_n \times H) \text{ xor } AES(CTR_0, K)$.
- Ausgabe von AES-GSM: C und Tag T .

AES-GCM-Entschlüsselung: $\text{AES-GCM}(C, T, IV, K, AD)$

Bei der Entschlüsselung wird die Authentizität, also das Tag T geprüft. Dazu wird mit den Eingabewerten C, IV, K, AD ein Authentisierungs-Tag \tilde{T} berechnet, indem mit dem Kryptotext C und den Daten IV, AD die gleichen Schritte wie bei der Verschlüsselung durchgeführt werden. Falls gilt: $T = \tilde{T}$, ist die Authentizität erfolgreich festgestellt und der Klartext M wird ausgegeben.

7.5.4 Data Encryption Standard

Obwohl der DES aufgrund seiner zu kurzen Schlüssellänge bereits seit vielen Jahren als zu unsicher eingestuft ist, wird er im Folgenden vorgestellt, da er nach wie vor in vielen Systemen und Produkten im Einsatz ist. Meist wird heutzutage jedoch nicht mehr der reine DES eingesetzt, sondern der DES wird in abgewandelter Form, meist als 3DES mit mehreren Schlüsseln verwendet, um das Problem des zu kleinen Verschlüsselungsschlüssels des DES abzumildern (vgl. Seite 325).

Der Data Encryption Standard (DES) ist eine Blockchiffre, die einen Eingabeblock von 64 Bit mit einem 64-Bit Schlüssel, von dem aber nur 56 Bit relevant sind, zu einem Ausgabeblock von 64 Bit verschlüsselt. Der DES ist ein symmetrisches Verfahren, das sowohl zur Ver- als auch zur Entschlüsselung verwendet wird. Als Verschlüsselungstechniken werden Bitpermutationen (Transpositionen), Substitutionen und die bitweise Addition modulo 2 vermischt. Der DES ist also eine Produktchiffre, die die von Shannon geforderte Konfusion und Diffusion erzielt.

DES

Techniken

Die Geschichte des DES

Entwicklung

Der Anfang der Entwicklungsgeschichte geht auf das 1972 vom National Bureau of Standards (NBS), heute National Institute of Standards (NIST), gestartete Programm zur sicheren Datenspeicherung und -übermittlung zurück. Gesucht wurde ein einheitlicher Algorithmus, der kompatibel zu verschiedenen Geräten ist, eine effiziente und preiswerte Implementierung ermöglicht sowie garantiert sicher und allgemein verfügbar ist.

1973 erfolgte die Ausschreibung im Federal Register, aber die eingereichten Vorschläge waren unzureichend. Nach der zweiten Ausschreibung 1974 reichte IBM einen Vorschlag ein, den 128-Bit LUCIFER Algorithmus. Der modifizierte LUCIFER-Algorithmus wurde 1975 von der National Security Agency (NSA) veröffentlicht. Eine wesentliche Modifikation betraf die Reduktion der Schlüssellänge von 128 auf 56 Bits. Trotz vielfacher Kritik, die sich insbesondere an der geringen Schlüssellänge und an der Geheimhaltung der Designentscheidungen, die hinter der Entwicklung des Algorithmus standen, entzündete, wurde dieser Algorithmus als Data Encryption Standard 1977 durch das National Bureau of Standards (NBS) zum US-Verschlüsselungsstandard genormt.

Standardisierung

Der DES floss später auch in einige ISO-Standards ein, wurde aber als Algorithmus an sich in keinem anderen Land zum Standard erklärt. Jedoch übernahmen ihn viele Institutionen und passten ihn an spezifische Bereiche an. So standardisierte das American National Standard Institute (ANSI) 1981 den DES für den privaten Sektor als DEA (ANSI X3.92) [88]. Eine sehr breite Akzeptanz fand der DES im Bankenumfeld, in dem unter anderem spezielle Standards für die Verwaltung von PINs (vgl. ANSI X9.8 [89]) veröffentlicht wurden. Gerade in diesem sicherheitskritischen Umfeld ist der DES auch bis heute noch in sehr vielen Produkten im Einsatz. Deshalb ist es wichtig, sich mit seiner Funktionsweise und seinen Sicherheitseigenschaften vertraut zu machen, um in der Lage zu sein, die möglichen Bedrohungen, die aus seiner Verwendung resultieren können, zu erfassen und korrekt einzuführen.

Bankenbereich

1983 erfolgte die erste Neuzertifizierung des Standards, die ab dann alle fünf Jahre durchgeführt werden sollte. Zu diesem Zeitpunkt wurde der DES weiterhin als sicher eingestuft. Ab 1987 gab es zunehmende Anzeichen für Mängel (u.a. [26, 179, 114]), aber der Algorithmus wurde dennoch erneut für weitere fünf Jahre zertifiziert. Da 1993 noch immer keine Alternative zur Verfügung stand, erfolgte wiederum eine Zertifizierung, obwohl die Unsicherheit des Algorithmus aufgrund des zu kleinen Schlüsselraumes offensichtlich war. Deutlich wurde diese Schwäche durch Angriffe, die den Schlüsselraum systematisch durchsuchten, wie 1994, als ein DES-Schlüssel auf 12 HP 9735 Workstations in 50 Tagen ermittelt werden konnte.

Re-Zertifizierung

Angriffe

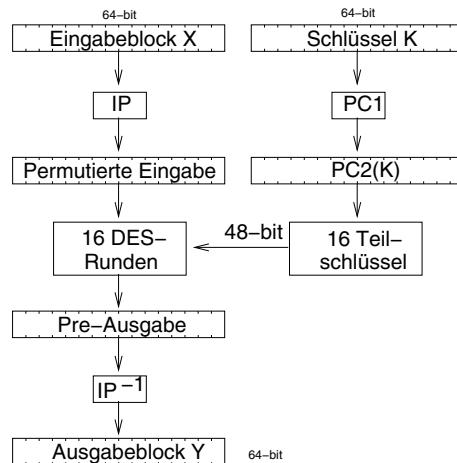


Abbildung 7.10: Das Grundschema des DES

(durchschnittlich 2^{43} benötigte Klartexte). Am 15. Juli 1998 wurde ein weniger als 250 000 US-Dollar teurer Spezialcomputer vorgestellt [65], der einen mit einem DES-Schlüssel verschlüsselten Klartext in 56 Stunden entschlüsseln konnte.

1998 lief die letzte Zertifizierungsperiode des DES aus; eine erneute Zertifizierung fand nicht mehr statt. Vielmehr erfolgte 1997 die Ausschreibung für den AES (Advanced Encryption Standard) als dem Nachfolger des DES. Als Sieger der Ausschreibung wurde im Herbst 2000 der belgische Algorithmus Rijndael erklärt; er bildet die Basis des AES-Standards. Auf den AES gehen wir in Abschnitt 7.5.5 etwas genauer ein.

AES

Funktionsweise des Standard-DES

Der DES ist eine Feistel-Chiffre (u.a. [36]). Feistel-Chiffren sind symmetrische Blockchiffren, deren Charakteristikum darin besteht, dass jeder Eingabeblock in eine linke und eine rechte Hälfte geteilt wird und die Blöcke in mehreren Runden verarbeitet werden. Die Rundenfunktion wird dabei nur auf eine der beiden Hälften angewandt und das Ergebnis mit der anderen Hälfte mittels XOR verknüpft. Danach werden die beiden Hälften vertauscht und die nächste Runde wird ausgeführt. Der DES ist die bekannteste Blockchiffre, die nach diesem Prinzip arbeitet.

Feistel-Chiffre

Der DES verschlüsselt einen 64-Bit Eingabeblock mit einem Schlüssel von 64 Bit, bei dem 56 Bit frei wählbar sind, die restlichen acht Bits sind Paritätsbits. Der DES besitzt damit einen Schlüsselraum von $2^{56} \sim 7,2 \cdot 10^{16}$ unterschiedlichen Schlüsseln. Das Grundschema des DES ist in Abbildung 7.10 zu sehen.

Schlüsselraum

58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

Tabelle 7.3: Die Permutation IP

Gemäß dieses Schemas durchläuft der Eingabeblock zunächst die Initialpermutation IP , die die permutierte Eingabe erzeugt. Die Permutationstabelle hierzu ist in Tabelle 7.3 zu sehen. Sie ist wie folgt zu interpretieren: Liest man die Einträge einer Zeile von links nach rechts und die Zeilen von oben nach unten, so geben die Einträge die Nummer desjenigen Bits im Eingabeblock an, das an diese Stelle gesetzt wird. Der Eingabeblock $X = (x_1, x_2, \dots, x_{64})$ ergibt einen permutierten Eingabeblock $IP(X) = (x_{58}, x_{50}, \dots, x_7)$.

Runden

Der permutierte Eingabeblock durchläuft sechzehn identische Chiffrierschritte, die als Runden bezeichnet werden. In jeder dieser Runden wird ein bestimmter Teilschlüssel verwendet. Diese Teilschlüssel werden aus einem 56-Bit Wert erzeugt, der seinerseits aus dem ursprünglichen 64-Bit Schlüssel durch eine permutierende Auswahl ($PC\ 1$) entsteht. Die Ableitung der Teilschlüssel wird weiter unten genau erläutert. Das Ergebnis der sechzehn Runden durchläuft abschließend die inverse Initialpermutation IP^{-1} .

Die Verschlüsselung

Der permutierte Eingabeblock wird in die Hälften L_0 und R_0 mit je 32 Bit zerlegt. Es gilt für $i = 1, \dots, 16$:

$$L_i = R_{i-1} \text{ und } R_i = L_{i-1} \oplus f(R_{i-1}, K_i).$$

K_i ist dabei der i -te Teilschlüssel; \oplus steht für die bitweise Addition modulo 2, also einem XOR. Die beiden Hälften $L_{16}R_{16}$ des Ergebnisses der letzten Runde werden getauscht zu $R_{16}L_{16}$ und ergeben die Pre-Ausgabe. Der genauere Aufbau der Runden ist in Abbildung 7.11 zu sehen.

Funktion f

Die Funktion f , die in Abbildung 7.12 beschrieben ist, bildet den Kern des DES. Ihre wesentlichen Elemente sind die acht Substitutionsboxen (S-Boxen). Mit diesen wird eine schlüsselabhängige Substitution einzelner Teile des Eingangswertes durchgeführt.

Expansion

Bevor der 32-Bit Wert R_{i-1} mit dem 48-Bit Teilschlüssel K_i verknüpft wird, muss R_i durch die Expansionsabbildung E (siehe Tabelle 7.4) auf einen 48-

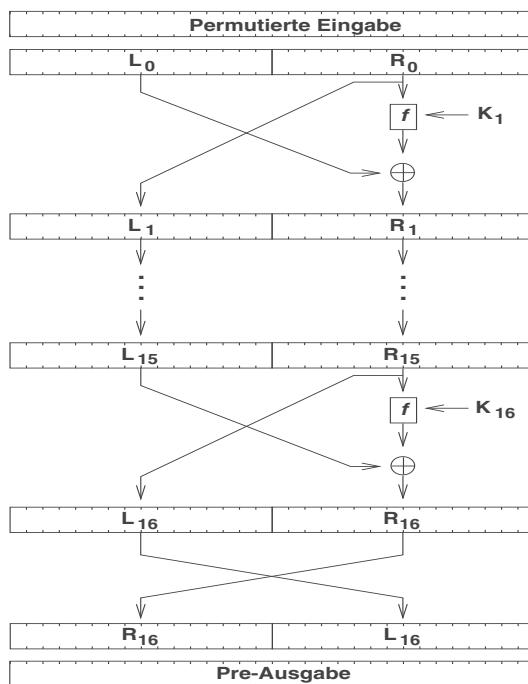
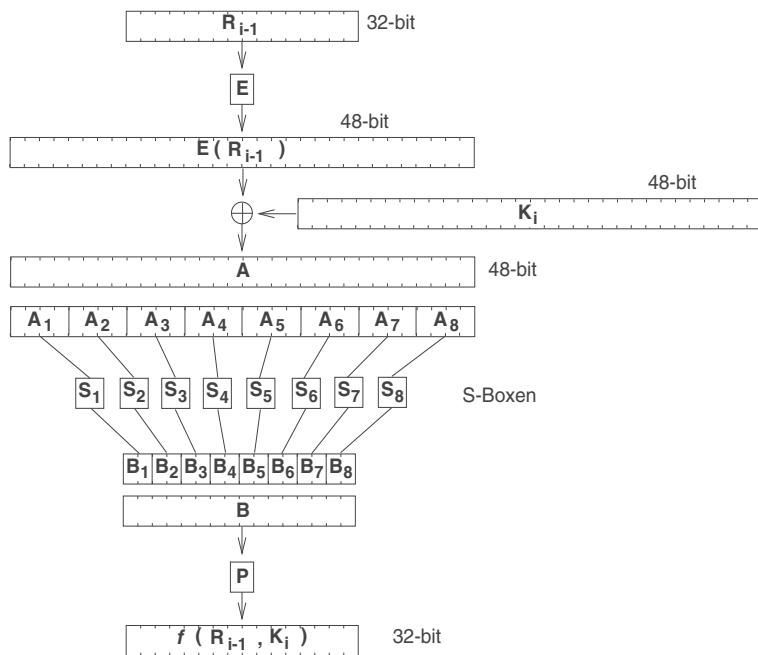


Abbildung 7.11: Der Aufbau der DES-Runden

Abbildung 7.12: Die Funktion f

Bit Wert $E(R_{i-1})$ erweitert werden. Diese Abbildung verdoppelt einfach einige ausgewählte Bits.

32	1	2	3	4	5		16	7	20	21	
4	5	6	7	8	9		29	12	28	17	
8	9	10	11	12	13		1	15	23	26	
E:	12	13	14	15	16	17	P:	5	18	31	10
	16	17	18	19	20	21		2	8	24	14
	20	21	22	23	24	25		32	27	3	9
	24	25	26	27	28	29		19	13	30	6
	28	29	30	31	32	1		22	11	4	25

Tabelle 7.4: Die Expansion E und die Permutation P

Die XOR-Verknüpfung von K_i und $E(R_{i-1})$ ergibt einen 48-Bit Wert A , der in acht 6-Bit Blöcke A_1, \dots, A_8 zerlegt wird. Sei j eine der acht Blocknummern, $j \in \{1, \dots, 8\}$. Jeder Block A_j dient als Eingabe für die entsprechende S-Box S_j , die ihrerseits einen 4-Bit Ausgabeblock $B_j = S_j(A_j)$ erzeugt. Die Konkatenation der Blöcke B_1, \dots, B_8 wird von der Permutation P (siehe Tabelle 7.4) permutiert und liefert den Ausgabewert $f(R_{i-1}, K_i)$.

S-Box

Eine S-Box ist eine Matrix bestehend aus 4 Zeilen und 16 Spalten. Die Substitution wählt aus der S-Box S_j ein Element aus, indem der Spalten- und der Zeilenindex aus dem Eingabeblock A_j berechnet wird. Dazu bilden das linke und das rechte Bit des Eingabeblocks A_j von S_j den Zeilenindex k und die mittleren vier Bits bilden den Spaltenindex l . Der 6-Bit Block A_j wird durch den Matrix-Eintrag $S_j(k, l)$ ersetzt. Das Ergebnis dieser Substitution ist der 4-Bit Ausgabeblock B_j .

Beispiel 7.7 (S-Box)

In Tabelle 7.5 ist als Beispiel auszugsweise die S-Box S_3 angegeben. Betrachten wir nun die Substitution eines Blocks A_3 für diese S-Box S_3 . Der 6-Bit Block A_3 habe den Wert $(111010)_2$. Dann ergibt sich als Zeilenindex $k = (10)_2 = 2$ und als Spaltenindex $l = (1101)_2 = 13$ (vgl. eingerahmte Werte in Tabelle 7.5). Der Substitutionsschritt liefert somit die Ausgabe $B_3 = 10 = (1010)_2$.

▲

Berechnung der Teilschlüssel

Teilschlüssel

Ein 64-Bit Schlüssel K durchläuft zuerst eine permutierende Auswahl $PC\ 1$ (Permuted Choice). Diese entfernt jedes achte Bit und permutiert die

	0	1	2	3	4	...	9	10	11	12	13	14	15
$S_3^0:$	10	0	9	14	6	...	13	12	7	11	4	2	8
$S_3^1:$	13	7	0	9	3	...	8	5	14	12	11	15	1
$S_3^2:$	13	6	4	9	8	...	1	2	12	5	10	14	7
$S_3^3:$	1	10	13	0	6	...	15	14	3	11	5	2	12

Tabelle 7.5: Die Substitutionsbox S_3 (Auszug)

verbleibenden 56 Bits. Die ausgelassenen Bits sind als Paritätsbits gedacht, haben aber auf das Verfahren keinen Einfluss. Diese Reduktion hat zur Folge, dass nur 2^{56} verschiedene Schlüssel existieren. Der entstandene 56-Bit Wert wird in die Hälften C_0 und D_0 mit je 28 Bits aufgespalten. Diese durchlaufen nun sechzehn Runden, in denen C_i und D_i mittels eines zyklischen Links-Shifts um ein oder zwei Bits aus C_{i-1} und D_{i-1} erzeugt werden. Die Anzahl der Positionen, um die in der jeweiligen Runde geshiftet wird, ist festgelegt. Aus jedem 56-Bit Wert $C_i D_i$ wird mittels einer weiteren Permutation, $PC\ 2$, ein 48-Bit Teilschlüssel K_i erzeugt. Die Reduktion erfolgt einfach durch das Weglassen festgelegter Bits.

Entschlüsselung

Wie eingangs bereits erwähnt, wird der DES sowohl zum Ver- als auch zum Entschlüsseln verwendet, also $E = D = DES$ (vgl. Definition 7.1). Für die Schlüssel gilt ebenfalls $K_E = K_D$, jedoch werden die zur Entschlüsselung benötigten Teilschlüssel in umgekehrter Reihenfolge eingesetzt. Während bei der Verschlüsselung die Teilschlüssel in der Reihenfolge K_1, K_2, \dots, K_{16} angewandt werden, ist die Reihenfolge bei der Entschlüsselung $K_{16}, K_{15}, \dots, K_1$. Es wird also in der Runde i der Teilschlüssel K_{17-i} verwendet.

Entschlüsselung

Sicherheit des DES

Die Sicherheit des DES ist schon seit seiner Einführung ein viel diskutiertes Thema, da zunächst nicht alle seine Entwurfskriterien oder Ergebnisse von Sicherheitsanalysen veröffentlicht worden sind. Deshalb hielten sich hartnäckig Gerüchte darüber, dass durch die NSA in das Verfahren Hintertüren integriert wurden, die es staatlichen Stellen ermöglichen sollen, verschlüsselte Texte einfach zu entschlüsseln, ohne im Besitz des verwendeten, geheimen Schlüssels zu sein. Bis heute konnten jedoch Kryptoanalyse-Experten die-

DES-Sicherheit

se Gerüchte nicht bestätigen. Beim DES ist allein die Geheimhaltung des Schlüssels von Bedeutung. Ist dieser bekannt, so kann sowohl ver- als auch entschlüsselt werden, da es sich um ein symmetrisches Verfahren handelt. Da der gesamte Schlüsselraum nur eine Größe von 2^{56} besitzt, ist er beim Stand heutiger Technologie nicht mehr als ausreichend zu betrachten.

Stärke des Algorithmus

Avalanche-Effekt Durch die beschriebene Kombination von Permutation und Substitution erfüllt der DES die Forderung nach größtmöglicher Streuung. Das Design der S-Boxen zusammen mit der sich jeder Substitution anschließenden Permutation P sorgt dafür, dass sich kleine Differenzen zwischen zwei Klartextblöcken, die in den ersten beiden Runden nur eine S-Box betreffen, schnell auf die Eingaben der anderen S-Boxen ausbreiten. Eine Änderung eines Bits in den Eingabewerten verursacht eine Änderung von durchschnittlich der Hälfte der Bits in den Ausgabewerten. Dies wird als Lawineneffekt (engl. *avalanche*) bezeichnet. Bereits nach fünf Runden hängt jedes Bit des Zwischenergebnisses von jedem Bit der Eingabe und jedem der 56 Schlüsselbits ab.

Im Gegensatz zu der Permutation P haben die Permutationen IP und IP^{-1} keine Bedeutung für die kryptografische Sicherheit des DES. Man vermutet, dass sie nur deshalb eingeführt wurden, um Softwarerealisierungen des DES zu verlangsamen, da die Permutationen in einer Hardwarerealisierung praktisch „kostenlos“ durchführbar sind.

kryptografisch stark Der DES ist sehr stark gegen analytische Angriffe (vgl. Abschnitt 7.8.3). Die S-Boxen sind die einzigen nicht-linearen Bestandteile des DES. Dies ist wichtig, da man dadurch aus der Differenz zwischen den Bits zweier Eingabeblocks nicht auf deren Differenz in den erzeugten Ausgabeblocks schließen kann. Der DES ist damit stark gegen Angriffe der differentiellen Kryptoanalyse. Etwas weniger gut schneidet er bei einer linearen Kryptoanalyse ab. Doch auch der beste bekannte Angriff erfordert immerhin noch 2^{43} Klartext/Kryptotextpaare.

Mehrfachverschlüsselung

Mehrzahlver- schlüsselung Der DES ist in viele kommerzielle Produkte integriert und insbesondere auch im Bankenumfeld nach wie vor sehr stark im Einsatz. Es stellt sich die Frage, ob man den kryptografisch starken Algorithmus beibehalten, aber die Schlüssellänge vergrößern kann, so dass DES-basierte Produkte weiterhin eingesetzt werden können. Die Antwort darauf ist ein eingeschränktes Ja. Die Lösung basiert auf einer Mehrfachverschlüsselung eines Klartextes mittels des DES unter Verwendung mehrerer Schlüssel. Da die

DES-Verschlüsselungen keine Gruppe bilden⁸, ist es möglich, dass sich der Schlüsselraum bei der Verwendung verschiedener Schlüssel tatsächlich vergrößert. Die nächste Frage, die sich stellt, ist die, ob zum Beispiel bei einer zweifachen Verschlüsselung mit zwei Schlüsseln, dem so genannten 2DES, eine Verdoppelung des Schlüsselraums erzielbar ist, so dass ein möglicher Angreifer einen Schlüsselraum von 2^{112} unterschiedlichen Schlüsseln durchsuchen muss. Die überraschende Antwort darauf ist leider Nein, da man mit einem so genannten Meet-in-the-middle Angriff (vgl. [120]) nur der Suchraum wesentlich verkleinert werden kann. Durch diesen Angriff verkleinert sich die Schlüssellänge und die NIST hat im Mai 2000 die effektive Schlüssellänge von 2DES auf 80 Bit festgelegt⁹.

Meet-in-the-middle Angriff:

Bei einem solchen Angriff geht man von einem Klartext/Kryptotextpaar (M, C) aus, mit

$$C = DES(DES(M, K^1), K^2),$$

und versucht, die verwendeten Schlüssel K^1, K^2 zu brechen. Im ersten Schritt wird M mit allen möglichen Schlüsseln $K_i \in \{0, \dots, 2^{56} - 1\}$ verschlüsselt,

$$M_i = DES(M, K_i),$$

und die Paare (M_i, K_i) werden in einer Liste lexikografisch geordnet abgelegt. Im nächsten Schritt wird C mit allen möglichen Schlüsseln $K_j \in \{0, \dots, 2^{56} - 1\}$ entschlüsselt,

$$M_j = DES^{-1}(C, K_j),$$

und es wird geprüft, ob M_j in der Liste bereits vorhanden ist. Ist dies der Fall, so stellt man die Hypothese auf, dass K_i und K_j die gesuchten Schlüssel sind. Diese Hypothese wird anschließend mit weiteren Klartext/Kryptotextblöcken überprüft.

Triple-DES (3DES)

Fortschritte lassen sich durch die Technik des Triple-DES (auch als 3DES bezeichnet) mit drei Schlüsseln erzielen. Seien K^1, K^2, K^3 drei DES-Schlüssel. Die Verschlüsselung mit dem Tripel-DES erfolgt durch

$$DES(DES^{-1}(DES(X, K^1), K^2), K^3)$$

und die Entschlüsselung durch

$$DES^{-1}(DES(DES^{-1}(X, K^3), K^2), K^1).$$

2DES

Angriff

Triple-DES

⁸ D.h. die DES-Verschlüsselungen sind nicht abgeschlossen unter Hintereinanderausführung.

⁹ Vgl. http://csrc.nist.gov/publications/nistpubs/800-57/sp800-57-Part1-revised2_Mar08-2007.pdf

Falls $K^1 = K^2$ oder $K^3 = K^2$, dann ist der Triple-DES kompatibel zum einfachen DES. Zu beachten ist aber, dass sich der zu leistende Verschlüsselungsaufwand verdreifacht. Der beschrieben Modus zur Verwendung von DES wird auch als 3DES-EDE-Modus¹⁰ bezeichnet.

Schlüssellänge

Die effektive Schlüssellänge beträgt beim Triple-DES aufgrund möglicher Meet-in-the-middle Angriffen anstelle von 168 Bit nur 112 Bit, wodurch sich ein Schlüsselraum der Größe 2^{112} ergibt¹¹. Dies ist zwar deutlich mehr als beim einfachen DES, aber auf längere Sicht dennoch nicht ausreichend, so dass in sicherheitskritischen Softwareprodukten wie insbesondere im Umfeld elektronischer Banktransaktionen der Einsatz kryptografischer Verfahren mit einem größeren Schlüsselraum mittel- bis langfristig unausweichlich ist.

7.5.5 AES

AES

Seit Mai 2002 ist der AES der offizielle NIST-Standard für symmetrische Blockchiffren.

Mathematische Basis des AES

Galois-Feld

Der AES arbeitet auf der mathematischen Struktur eines Galois-Feldes (GF) bzw. Galois-Körpers. Ein Galois Feld ist eine endliche Menge von Elementen. Für die Elemente der Menge sind die Operationen Addition, Subtraktion, Multiplikation und das Berechnen der Inversen definiert. Berechnungen in Galois-Feldern haben den Vorteil, dass alle Zahlen von endlicher Größe sind und es bei der Division nicht zu Rundungsfehlern kommt. Viele kryptografische Verfahren basieren auf einem Galois-Feld $GF(q)$, wobei q eine große Primzahl ist. Die Elemente der Trägermenge des durch q erzeugten Galois-Feldes nennt man die Einheitswurzeln.

Der AES arbeitet im Galois-Feld $GF(2^8)$ von 256 Elementen. Jedes Element kann damit durch 1 Byte repräsentiert werden. Die Elemente von $GF(2^8)$ werden als Polynome vom Grad 7 repräsentiert mit Koeffizienten aus GF(2). Jedes Element $A(x)$ aus $GF(2^8)$ wird somit repräsentiert durch: $A(x) = a_7x^7 + \dots + a_1x^1 + a_0x^0$ mit $a_i \in GF(2) = \{0, 1\}$. Jedes Element kann als 8-Bit Vektor $A = (a_7, a_6, \dots, a_0)$ seiner Koeffizientenwerte beschrieben und abgelegt werden.

¹⁰ Verschlüsseln, Entschlüsseln, Verschlüsseln

¹¹ Vgl. <http://csrc.nist.gov/publications/nistpubs/800-57/SP800-57-Part2.pdf>

Operationen auf $GF(2^8)$:

- Die Additionen und Subtraktion auf $GF(2^8)$ ist wie folgt definiert:

$$C(x) = A(x) + B(x) = c_7x^7 + \cdots + c_1x^1 + c_0x^0 \quad \text{mit}$$

$$c_i = a_i + b_i \bmod 2 = a_i - b_i \bmod 2.$$

Im AES wird diese Operation bei der Verknüpfung von Klartextbits mit Schlüsselbits in der Operation *AddRoundKey* verwendet.

- Die Multiplikation auf Elementen des Feldes $GF(2^8)$ entspricht einer Standard-Multiplikation auf Polynomen. Da das Ergebnis einer solchen Multiplikation in der Regel einen höheren Grad als den Grad 7 der Multiplikanten haben wird, muss der Ergebniswert reduziert werden. Dazu ist ein irreduzibles Polynom P vom Grad 8 erforderlich. Zur Reduktion wird dann das Ergebnis der Multiplikation durch das Polynom P dividiert.

In der Spezifikation des AES ist das zu verwendende Polynom P wie folgt festgelegt: $P(x) = x^8 + x^4 + x^3 + x + 1$, d.h. P(x) kann durch 100011011 repräsentiert werden. Im AES wird die Multiplikations-Operation in der Transformation *MixColumn* verwendet.

- Die Berechnung des Inversen für ein Element A(x) in $GF(2^8)$ ist mit gegebenem Polynom P(x) wie folgt definiert:

$$A^{-1}(x) * A(x) = 1 \bmod P(x).$$

Zur effizienten Verschlüsselung werden im AES Tabellen mit vorab berechneten multiplikativen Inversen verwendet. Im AES wird die Berechnung des Inversen für ein Element in der *Byte-Substitution (S-Box)* verwendet.

Funktionsweise des AES

Der AES ist eine symmetrische Blockchiffre mit fester Blocklänge von 128 Bit und variabler Schlüssellänge von 128, 192, und 256 Bit. Jeder Klartextblock wird in mehreren Runden mit einer sich wiederholenden Abfolge von Funktionen bearbeitet. Die Anzahl der durchzuführenden Runden ist von der Schlüssellänge abhängig.

Der AES ist keine Feistel-Chiffre, da bei jeder Iteration stets der gesamte Block verschlüsselt wird. Der AES ist aus drei Schichten aufgebaut, die nacheinander auf den jeweiligen Klartextblock einwirken. Die drei Schichten Key Addition, Substitutions- und Diffusions-Schicht manipulieren jeweils

AES-Schichten

jedes Bit der Eingabe. Jeder Klartextblock $B = b_0, b_1, \dots, b_{15}$ wird für die

$$\begin{matrix} b_0 & b_4 & b_8 & b_{12} \\ b_1 & b_5 & b_9 & b_{13} \end{matrix}$$

Bearbeitung in einer 4x4-Matrix B gespeichert: B=

$$\begin{matrix} b_0 & b_4 & b_8 & b_{12} \\ b_1 & b_5 & b_9 & b_{13} \\ b_2 & b_6 & b_{10} & b_{14} \\ b_3 & b_7 & b_{11} & b_{15} \end{matrix}$$

Jeder Eintrag b_i der Matrix B beschreibt 1 Byte. Diese Matrix wird intern als 2-dimensionales Feld genannt `state` abgelegt und der AES arbeitet auf Zeilen und Spalten der Matrix.

Die Verschlüsselung eines Klartextblocks erfolgt in Runden und in jeder Runde werden vier groben Transformationsschritte durchlaufen, die weiter unten etwas genauer beschrieben sind. Jede Runde $\text{Round}(\text{State}, \text{RoundKey})$ nutzt den aktuellen Zustand der Klartextblocks, der in dem Feld `state` abgelegt ist, sowie einen Runden-Schlüssel, und führt die folgenden Transformationen durch:

1. SubBytes(State): Substitutionschiffre, die eine Byte-weise Substitution durchführt.
2. ShiftRows(State): zyklischer Links-Shift.
3. MixColumns(State): Spaltenweise Multiplikation mit Polynom P(x).
4. AddRoundKey(State, RoundKey).

Jeder Block wird den Transformationen unterworfen, wodurch erreicht wird, dass verschiedene Teile des Verschlüsselungsschlüssels nacheinander auf den Klartext-Block angewandt werden; also der Schlüssel nicht nur einmal zur Verschlüsselung des Blocks verwendet wird. Die Anzahl der insgesamt für eine Verschlüsselung zu durchlaufenden Runden ist von der Schlüssellänge abhängig. Bei einer Schlüssellänge von 128 Bit sind 10 Runden, bei einer Länge von 192-Bit sind 12 und bei einer Länge von 256 Bit sind 14 Runden erforderlich. In der nachfolgenden Beschreibung bezeichnet NR die Anzahl der Runden.

Transformationen

1. Der erste Rundenschlüssel wird mittels XOR mit dem Block verknüpft.
2. Es werden NR-1 Runden durchlaufen, wobei jede Runde aus vier Einzelschritten besteht.
 - (a) **SubBytes:** Jedes Byte des Blocks wird durch einen S-Box Eintrag ersetzt (monoalphabetische Substitutionschiffre). Die S-Box des AES ist wie auch schon beim DES der nicht-lineare Anteil des AES und ist ebenfalls stark gegen differentielle und lineare Krypto-Analyse. Aber anders als beim DES wird beim AES für jedes der 16 Byte die gleiche S-Box verwendet, die als 4x4 Matrix angeordnet ist. Für eine effiziente Verarbeitung werden die 256

Eingaben der S-Box in der Regel in einer vorab berechneten Lookup-Tabelle abgelegt.

- (b) **ShiftRow:** Die Bytes in den Zeilen des 2-dimensionalen Feldes state werden zyklisch nach links verschoben, wobei die erste Zeile unverändert bleibt, die Elemente der 2-ten Zeile um eine Position nach links, die der 3-ten Zeile um zwei Positionen und die der 4-ten Zeile um drei Positionen nach links geshiftet werden.
- (c) **MixColumn:** Jede Spalte der 4x4 Matrix wird mit der Matrix C im $GF(2^8)$ multipliziert, wobei C(x) als Polynom in der AES Spezifikation festgelegt ist und die Koeffizienten des Polynoms so gewählt wurden, dass die Berechnung sehr performant mittels Shift und xor Operationen erfolgen kann. C ist festgelegt durch

$$C = \begin{matrix} 02 & 03 & 02 & 02 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{matrix}$$

Also $C(x) = 02x^{15} + 03x^{14} + \dots + 03x^3 + 01x^2 + 01x + 02$. Diese spaltenweise Transformation ist linear und invertierbar. Sie bewirkt eine Diffusion, indem Änderungen in der Eingabe sich auf alle 4 Ausgabe-Byte ausbreiten.

- (d) **Schlüsselverknüpfung:** Der Block wird mit dem jeweiligen Rundenschlüssel mittels XOR verknüpft; diese Verknüpfung wird als KeyAddition bezeichnet.
3. In der abschließenden Runde wird die MixColumn-Transformation der regulären Runden ausgelassen.

Für die Ver- und Entschlüsselung sind jeweils NR Rundenschlüssel zu erzeugen. Dies erfolgt durch das Expandieren des geheimen Schlüssels, indem rekursiv abgeleitete 4-Byte Worte an den geheimen Schlüssel angehängt werden. Die Schlüssel-Expansion erfolgt durch die Funktion KeyExpansion¹². Die Schlüsselerzeugung ist nichtlinear und nicht invertierbar. Durch die Expansion ergibt sich eine sehr starke Diffusion auf den geheimen Schlüssel. Im Gegensatz zum DES sind für den AES keine schwachen Schlüssel bekannt.

Rundenschlüssel

Die Wirkungen der verschiedenen Transformationen auf den Klartextblock lassen sich wie folgt zusammenfassen. Durch die Verknüpfung des Blockes mit dem jeweiligen Rundenschlüssel, sowohl vor der ersten Runde als auch als letzter Schritt innerhalb jeder Runde, wirkt sich dieser Schlüssel auf jedes Bit der Ergebnisse der Runde aus (starker Avalanche Effekt). Im Verlauf der Verschlüsselung eines Blocks gibt es keinen Schritt, dessen Ergebnis

Wirkung

¹² Vgl. u.a. <http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf>

nicht in jedem Bit vom Schlüssel abhängt. Wie auch beim DES sind die Substitutionen der S-Boxen nichtlineare Operationen, wobei die S-Boxen so konstruiert wurden, dass sie sehr stark gegen lineare und differentielle Kryptoanalyse (vgl. Abschnitt 7.8) sind. Mit den ShiftRow und MixColumn-Transformationen erhält man eine sehr gute Durchmischung der Bits eines Blocks.

Entschlüsselung

Da AES keine Feistel-Chiffre ist, müssen für die Entschlüsselung alle Transformationen invertiert werden und in umgekehrter Reihenfolge zur Entschlüsselung angewandt werden. Dabei ist die inverse MixColumn Transformation relativ komplex, so dass die Entschlüsselung eine längere Rechenzeit benötigen kann als die Verschlüsselungsoperation. Durch die Verwendung von Tabellen mit vorab berechneten Ergebnissen der Transformationen kann diese Diskrepanz jedoch überwunden werden.

Sicherheit des AES

Der AES ist in der Vergangenheit eingehend analysiert worden. Der AES wurde im Juni 2003 von der US Regierung zum Einsatz für vertrauliche Verschlusssachen freigegeben: *The design and strength of all key lengths of the AES algorithm (i.e., 128, 192 and 256) are sufficient to protect classified information up to the SECRET level. TOP SECRET information will require use of either the 192 or 256 key lengths.*

Der AES gehört auch zu den vom Projekt NESSIE¹³ (New European Schemes for Signatures, Integrity and Encryption) empfohlenen kryptografischen Algorithmen.

Praxisrelevante Angriffe, die bislang bekannt geworden sind, gehören in die Klasse der Seitenkanalangriffe, wie der durch Bernstein im Jahr 2005 bekannt gemachte Angriff¹⁴, oder Timing-Angriffe, wie der Angriff von Shamir¹⁵. Einer der dort erklärten Angriffe war in der Lage, den AES-Schlüssel bereits nach 65 Millisekunden zu brechen, wobei der Angriff jedoch voraussetzt, dass das Programm auf dem gleichen Rechner läuft, wie der, auf dem die AES-Verschlüsselung durchgeführt wird.

Angriffe auf Blockchiffren richten sich häufig gegen die Anzahl der durchzuführenden Runden. Bei AES benötigt man, abhängig von der Schlüssellänge 10, 12 oder 14 Runden bei der Verschlüsselung. Die bislang besten Attacken auf den AES mit reduzierter Rundenzahl von 7, 8 bzw. 9 Runden wurden 2000 von B. Schneier¹⁶ vorgestellt. In 2011 veröffentlichten A. Bogdanov, D. Khovratovich und C. Rechberger einen Angriff, der viermal schneller

¹³ <https://www.cosic.esat.kuleuven.be/nessie/>

¹⁴ vgl. <http://cr.yp.to/antiforgery/cachetiming-20050414.pdf>

¹⁵ Vgl. <http://people.csail.mit.edu/tromer/papers/cache.pdf>

¹⁶ Vgl. <http://www.schneier.com/paper rijndael.pdf>

als Brute Force ist [29]. Der Angriff erfordert ca. $2^{126.1}$ Operationen für AES-128 Bit, bzw. ca. $2^{189.7}$ Operationen für AES-192 und ca. $2^{254.4}$ für AES-256. Auch dieser Angriff ist jedoch in der Praxis (noch) nicht relevant, so dass der AES auch weiterhin als sehr sicher gilt. Verbesserung könnte durch eine Erhöhung der Rundenzahl erreicht werden. Auch wenn Kryptografen Bedenken bezüglich der dem AES zugrunde liegenden algebraischen Struktur hegen, sind bislang keine erfolgreiche Angriffe auf die algebraische Struktur, die nicht nur eine theoretische Bedeutung haben, bekannt.

Der AES ist ein starkes Verschlüsselungsverfahren, das frei verfügbar ist und sowohl in Software als auch in Hardware heute bereits vielfach eingesetzt wird. Beispiele für Anwendungen des AES sind der Standard 802.11i für Wireless LAN (WLAN), SSH, IPsec, PGP bzw. GnuPG. Auch der Messenger-Dienst WhatsApp nutzt AES zur Ende-zu-Endeverschlüsselung von Nachrichten wie Chats, Sprach-Nachrichten, Videos oder auch Dateien. Für die Nachrichten-Verschlüsselung verwendet WhatsApp den AES256 im CBC-Modus sowie einen HMAC-SHA256 für die Nachrichtenauthentizität.

Fazit

7.6 Asymmetrische Verfahren

Der Abschnitt führt zunächst in 7.6.1 die charakteristischen Eigenschaften asymmetrischer Verfahren ein, bevor mit dem RSA-Verfahren in 7.6.2 das in der Praxis am häufigsten eingesetzte asymmetrische Kryptosystem vorgestellt wird.

7.6.1 Eigenschaften

Das Konzept der asymmetrischen Kryptografie wurde Mitte der 70er Jahre zeitgleich von Diffie und Hellman [55] sowie Ralph Merkle [119] entwickelt. Die zentrale Idee asymmetrischer Kryptosysteme besteht in der Verwendung eines Schlüsselpaares für jeden beteiligten Kommunikationspartner. Jedes Schlüsselpaar besteht aus einem geheimen und einem öffentlichen Schlüssel, wobei der öffentliche Schlüssel in allgemein zugänglichen Datenbanken abgelegt sein kann. Ein Klartext wird vom Sender mit dem öffentlichen Schlüssel des Empfängers verschlüsselt und der Empfänger entschlüsselt den Kryptotext mit seinem geheimen Schlüssel. Es ist also kein geheimer Schlüssel zwischen den Kommunikationspartnern auszutauschen. Da die Verschlüsselung unter Verwendung des öffentlichen Schlüssels des Partners erfolgt, muss die Authentizität dieses Schlüssels gewährleistet sein, damit ein Angreifer nicht seinen eigenen Schlüssel als den öffentlichen Schlüssel eines anderen ausgeben und ein vertrauensseliges Opfer dazu verleiten kann, sensible Daten damit zu verschlüsseln. Zur Sicherstellung und Kontrolle der Schlüsselauthentizität werden Zertifikate (vgl. Kapitel 9.1) eingesetzt.

zentrale Idee

Ausgehend von den bereits in Definition 7.1 angegebenen allgemeinen Eigenschaften eines kryptografischen Systems präzisiert Definition 7.7 die Eigenschaften eines asymmetrischen Kryptosystems (vgl. [55]).

Definition 7.7 (Asymmetrisches Kryptosystem)

asymmetrisches System

Gegeben sei ein kryptografisches System $\mathcal{KS} = (\mathcal{M}, \mathcal{C}, EK, DK, E, D)$ gemäß Definition 7.1. Das System besitzt die Eigenschaften eines asymmetrischen Kryptosystems, wenn gilt:

Schlüsselpaar

1. Schlüsselpaare (K_E, K_D) müssen leicht (effizient) zu erzeugen sein, wobei für solche Paare gelten muss:
 - (a) $K_D = f(K_E)$, $K_E \in EK$, $K_D \in DK$ und
 - (b) $\forall M \in \mathcal{M} : D(E(M, K_E), K_D) = M$ und
 - (c) K_E kann öffentlich bekannt gegeben werden.
2. Die Verschlüsselungsfunktion E und die Entschlüsselungsfunktion D sind effizient zu berechnen.
3. K_D ist aus der Kenntnis von K_E nicht mit vertretbarem Aufwand berechenbar.
4. Gilt $\mathcal{M} = \mathcal{C}$ und zusätzlich die optionale Eigenschaft

$$\forall M \in \mathcal{M} \quad E(D(M, K_D), K_E) = D(E(M, K_E), K_D) = M,$$

so ist das Kryptosystem auch zur Erstellung digitaler Unterschriften geeignet.

□

Der Blick auf die Definition 7.7 verdeutlicht, dass das zentrale Problem bei der Entwicklung asymmetrischer Kryptosysteme darin besteht, solche Funktionen zu finden, dass aus der Kenntnis des öffentlichen Schlüssels der zugehörige private Schlüssel nicht effizient berechnet werden kann. Die Basis zur Lösung dieses Problems liefern Klassen mathematischer Funktionen, die man als Einweg-Funktionen bezeichnet.

Definition 7.8 (Einweg-Funktion)

Einweg-Funktion

Eine injektive Funktion $f : X \longrightarrow Y$ heißt Einweg-Funktion (engl. *one way function*), wenn

1. für alle $x \in X$ der Funktionswert $f(x)$ effizient berechenbar ist und
2. wenn es kein effizientes Verfahren gibt, um aus einem Bild $y = f(x)$ das Urbild x zu berechnen.

□

Die Einweg-Eigenschaft einer Funktion basiert wesentlich auf Aussagen über die Effizienz entwickelter Algorithmen zur Berechnung der Funktionswerte und deren Umkehrabbildung. In Abschnitt 7.4.3 haben wir bereits auf die Problematik komplexitätstheoretischer Aussagen als Basis der Klassifikation von Algorithmen in effiziente und nicht effiziente hingewiesen. Da bei den bekannten Verfahren untere Schranken für Komplexitätsaussagen fehlen, ist die Existenz von Einweg-Funktionen schwer zu beweisen. Man begnügt sich meist mit Kandidaten, für die man die Eigenschaft zwar nicht formal bewiesen hat, für die es aber zurzeit noch keine effizienten Verfahren zur Berechnung der Umkehrfunktion gibt.

Existenz

Beispiel 7.8 (Faktorisierungsproblem)

Gegeben seien zwei große Primzahlen (z.B. 200 Dezimalstellen) p und q . Die Berechnung des Produkts von p und q , $n = p \cdot q$, ist auch für große natürliche Zahlen effizient durchführbar. Für ein gegebenes n ist es jedoch sehr schwierig, dessen Primfaktorzerlegung, also die Zahlen p und q , zu ermitteln.

Faktorisierung

Das Faktorisierungsproblem zählt zu den ältesten Problemen der Zahlentheorie, zu dessen Lösung bereits einige sehr gute Verfahren existieren. Deren Berechnungsaufwand ist aber für große Zahlen n immer noch sehr hoch und mit heutiger Technologie nicht effizient zu leisten. Das bekannteste Verfahren ist das des quadratischen Siebs [144], das mit seinen unterschiedlichen Varianten zu dem schnellsten Verfahren zur Faktorisierung von Zahlen zählt, die weniger als 110 Dezimalstellen lang sind.



Beispiel 7.9 (Diskreter Logarithmus)

Gegeben seien eine Basis a , eine Zahl $n \in \mathbb{N}$ und die Funktion

Logarithmus-
Problem

$$f_a : \{0, \dots, n-1\} \longrightarrow \{1, \dots, n-1\} \text{ mit}$$

$$f_a(x) = a^x \bmod n = y.$$

Die Exponentiation modulo p ist effizient berechenbar. Das diskrete Logarithmusproblem ist die Aufgabe, zu der gegebenen Zahl y den diskreten Logarithmus

$$x = \log_a y \bmod n$$

zu berechnen, also das x zu bestimmen, für das gilt, $y \equiv a^x \bmod n$.

Die Berechnung des diskreten Logarithmus ist für große Zahlen sehr aufwändig und es sind bis heute keine effizienten Verfahren bekannt. In kryptografischen Systemen wählt man häufig eine große Primzahl q und arbeitet mit dem Galois-Feld $GF(q)$. Die Komplexität von Verfahren zur Berechnung des diskreten Logarithmus in $GF(q)$ ist ungefähr die gleiche

wie die Faktorisierung einer natürlichen Zahl n , mit $|q| \sim |n|$, unter der Bedingung, dass n das Produkt zweier ungefähr gleich langer Primzahlen ist.



Da eine Einweg-Funktion aber auch für denjenigen, der zur Entschlüsselung autorisiert ist, eine fast unüberwindliche Hürde darstellt, ist sie in dieser reinen Form für den Einsatz in asymmetrischen Verfahren ungeeignet. Gesucht sind somit Funktionen, die für Unbefugte die Einweg-Eigenschaft besitzen, also sehr schwer umkehrbar sind, jedoch für autorisierte Benutzer, die ein spezifisches Geheimnis, nämlich einen privaten Schlüssel, kennen, leicht umzukehren sind. Man spricht hierbei von Einweg-Funktionen mit Falltür oder mit Geheimtür.

Definition 7.9 (Einweg-Funktion mit Falltür)

Eine injektive Funktion $f : X \rightarrow Y$ heißt Einweg-Funktion mit Falltür (engl. *trapdoor one way function*), wenn

1. für alle $x \in X$ der Funktionswert $f(x)$ effizient berechenbar ist und
2. wenn es ein effizientes Verfahren zur Berechnung der Umkehrfunktion $f^{-1}(y)$ für alle $y \in f[X]$ gibt, jedoch das Urbild x von y nicht allein aus dem Zusammenhang $y = f(x)$ bestimmbar ist, sondern wenn dafür zusätzliche Informationen erforderlich sind.



Beispiel 7.10 (h-te Potenz modulo n)

Falls n das Produkt zweier großer Primzahlen und die Zerlegung von n in diese Primfaktoren bekannt ist, so besitzt die Funktion

$$f_h(x) = x^h \bmod n = y$$

eine Falltür und ist mit der Kenntnis der Primfaktoren effizient umkehrbar (vgl. das RSA-Verfahren in 7.6.2). Jedoch ist für ein genügend großes n ($n > 10^{160}$) kein effizientes Verfahren bekannt, das allein mit der Kenntnis von h und n den Wert $x = f_h^{-1}(y)$ berechnet.



Beispiel 7.11 (Zusammengesetzter Modul n)

Gegeben ist das Produkt $n = pq$, wobei p und q verschiedene Primzahlen sind. In diesem Fall hat die Funktion $f_g(x) = g^x \bmod n$ eine Falltür, da der Chinesische Restsatz anwendbar ist. Dieser Satz lautet wie folgt. Gegeben

seien zwei Zahlen $i, j \in \mathbb{N}$ mit $\text{ggT}(i, j) = 1$. Dann gibt es für beliebige, natürliche Zahlen $a, b \in \mathbb{N}$, mit $a < i$ und $b < j$ eine eindeutige natürliche Zahl, $t \in \mathbb{N}$, so dass

$$a \bmod i = t \quad \text{und} \quad b \bmod j = t.$$



Chinesischer
Restsatz

Die gängigen, in der Praxis eingesetzten, asymmetrischen Verfahren basieren auf den in den Beispielen eingeführten Kandidaten für Einweg-Funktionen.

7.6.2 Das RSA-Verfahren

Das RSA-Verfahren [152] wurde 1978 von Ronald Rivest, Adi Shamir und Leonard Adleman entwickelt. Es basiert auf dem Faktorisierungsproblem (Einweg-Funktion) großer, natürlicher Zahlen und erfüllt alle vier Bedingungen von Definition 7.7. In der Praxis kommt das Verfahren sowohl zum Verschlüsseln als auch zum sicheren Austausch von Kommunikations-schlüsseln (z.B. von AES-Schlüsseln) sowie zur Erstellung und Überprüfung digitaler Unterschriften zum Einsatz.

Obwohl das RSA-Verfahren Bestandteil vieler offizieller Standards ist, wurde es selber nicht als offizieller Standard normiert. Aufgrund seiner großen Verbreitung und Akzeptanz unter anderem als Bestandteil von TLS-fähigen WWW-Browsern bezeichnet man aber das RSA-Verfahren häufig als einen de-facto Standard für asymmetrische Verschlüsselung.

de facto Standard

Mathematische Grundlagen

Für das Verständnis des RSA-Verfahrens sind einige mathematische Grundlagen erforderlich, die wir im Folgenden einführen. Für eine vertiefende Behandlung der Thematik sei auf die vielfältige Literatur zum Bereich der Kryptografie verwiesen (u.a. [166, 118]).

Definition 7.10 (Restklassenarithmetik)

Seien $m \in \mathbb{N} \setminus \{1\}$, $a, b \in \mathbb{Z}$ und $+_m, \cdot_m$ zwei Verknüpfungen. Dann gelten folgende arithmetische Rechenregeln:

Arithmetik

$$\begin{aligned} a \bmod m +_m b \bmod m &= (a + b) \bmod m \quad \text{und} \\ a \bmod m \cdot_m b \bmod m &= (a \cdot b) \bmod m. \end{aligned}$$

□

Mit $(\{\overline{0}, \dots, \overline{m-1}\}, +_m, \cdot_m)$ erhält man einen kommutativen Ring, den Restklassenring, den wir mit \mathbb{Z}_m bezeichnen, wobei \overline{i} die Restklasse i des Rings bezeichnet.

Beispiel 7.12 (Restklassenring modulo 4)

Gegeben sei $m = 4$ und die Menge $M = \{\bar{0}, \bar{1}, \bar{2}, \bar{3}\}$ der Restklassen modulo 4. $(M, +_4, \cdot_4)$ ist ein Restklassenring, wobei die Verknüpfungen wie folgt definiert sind:

$+_4$	$\bar{0}$	$\bar{1}$	$\bar{2}$	$\bar{3}$	\cdot_4	$\bar{0}$	$\bar{1}$	$\bar{2}$	$\bar{3}$
$\bar{0}$	$\bar{0}$	$\bar{1}$	$\bar{2}$	$\bar{3}$	$\bar{0}$	$\bar{0}$	$\bar{0}$	$\bar{0}$	$\bar{0}$
$\bar{1}$	$\bar{1}$	$\bar{2}$	$\bar{3}$	$\bar{0}$	$\bar{1}$	$\bar{0}$	$\bar{1}$	$\bar{2}$	$\bar{3}$
$\bar{2}$	$\bar{2}$	$\bar{3}$	$\bar{0}$	$\bar{1}$	$\bar{2}$	$\bar{0}$	$\bar{2}$	$\bar{0}$	$\bar{2}$
$\bar{3}$	$\bar{3}$	$\bar{0}$	$\bar{1}$	$\bar{2}$	$\bar{3}$	$\bar{0}$	$\bar{3}$	$\bar{2}$	$\bar{1}$

Für die Restklassen gilt:

$$\bar{0} = \{x \mid x = 4n, n \in \mathbb{N}\}$$

$$\bar{1} = \{x \mid x = 4n + 1, n \in \mathbb{N}\}$$

$$\bar{2} = \{x \mid x = 4n + 2, n \in \mathbb{N}\}$$

$$\bar{3} = \{x \mid x = 4n + 3, n \in \mathbb{N}\}$$

▲

Definition 7.11 (Multiplikative Inverse)

Inverse

Seien $a, b \in \mathbb{Z}_m$. Dann nennt man ein b mit $a \cdot b = 1 \pmod{m}$ das multiplikative Inverse modulo m von a .

□

Definition 7.12 (Euler'sche φ -Funktion)

φ -Funktion

Die Funktion φ mit der Eigenschaft¹⁷:

$$\varphi(m) := |\{a \in \mathbb{Z}_m : \text{ggT}(a, m) = 1\}|$$

heißt die Euler'sche φ -Funktion.

□

Für die Funktion φ gelten folgende Eigenschaften:

$$\text{Für alle Primzahlen } p : \quad \varphi(p) = p - 1$$

$$\text{Für alle Primzahlen } p \text{ und } r \in \mathbb{N} : \quad \varphi(p^r) = (p - 1) \cdot p^{r-1}$$

$$\text{Für alle Primzahlen } p, q \text{ mit } p \neq q : \quad \varphi(p \cdot q) = (p - 1) \cdot (q - 1)$$

Funktionsweise des RSA-Verfahrens

RSA-Algorithmus

- Wähle zwei große Primzahlen¹⁸ p und q und berechne $n = pq$.

¹⁷ Der ggt bezeichnet den größten gemeinsamen Teiler zweier natürlicher Zahlen.

Den Wert n bezeichnet man auch als RSA-Modul.

2. Wähle öffentlichen Exponenten $e \in \{0, \dots, \varphi(n) - 1\}$, so dass gilt:

e ist relativ prim zu $\varphi(n) = (p - 1) \cdot (q - 1)$,

d. h. $ggT(\varphi(n), e) = 1$,

wobei $\varphi(n)$ die Euler'sche Funktion bezeichnet.

Wähle dazu zum Beispiel eine Primzahl e , für die gilt:

$$\max(p, q) < e < \varphi(n) - 1.$$

Beispiele für e sind: $e = 65537 = 2^{16} + 1$, $e = 3$, $e = 17$.

3. Berechne den privaten Exponenten d so, dass gilt $e \cdot d = 1 \pmod{\varphi(n)}$, d.h. d ist multiplikative Inverse modulo $\varphi(n)$ zu e . Da e so gewählt wird, dass $ggT(\varphi(n), e) = 1$, ist gewährleistet, dass es immer ein Inverses zu e modulo $\varphi(n)$ gibt, d.h. dass es stets einen zu e passenden privaten Schlüssel d gibt.
4. (e, n) ist der öffentliche Schlüssel.
5. (d, n) ist der geheime, private Schlüssel.
6. Verschlüsseln eines Klartextes $M \in [0, n - 1]$:

$$E(M) = M^e \pmod{n}.$$

7. Entschlüsseln eines Kryptotextes $C \in [0, n - 1]$:

$$D(C) = C^d \pmod{n}.$$

Die Sicherheit des RSA-Verfahrens basiert auf der Geheimhaltung der Werte p, q und $\varphi(n)$.

Satz

Für Schlüsselpaare (e, n) und (d, n) , die die in den Schritten 2 und 3 des RSA-Algorithmus festgelegten Eigenschaften besitzen, gilt:

$$M^{e \cdot d} \pmod{n} = M.$$

Da das RSA-Verfahren auch zur Erstellung digitaler Signaturen einsetzbar ist, gilt darüber hinaus:

$$M^{d \cdot e} \pmod{n} = M.$$

¹⁸ Für einen für heutigen Zeit angemessen großen Modul von 1024 Bit benötigt man zwei Primzahlen der Länge 512 Bit.

Beweis

Zu zeigen: (1) $M^{e \cdot d} \bmod n = M^{d \cdot e} \bmod n$
 (2) $M^{e \cdot d} \bmod n = M$

Satz von Euler

Der Beweis basiert auf dem Satz von Euler, der besagt, dass für zwei natürliche Zahlen n und M , die relativ prim sind¹⁹, gilt:

$$M^{\varphi(n)} \bmod n = 1.$$

Satz von Fermat

Weiterhin wird auf den Satz von Fermat zurückgegriffen, der besagt, dass für eine Primzahl p und eine natürliche Zahl M mit $\text{ggT}(M, p) = 1$ gilt:

$$M^{p-1} = 1 \bmod p.$$

- ad (1) 1. Mit $e \cdot d = 1 \bmod \varphi(n)$, d.h. $e \cdot d = k \cdot \varphi(n) + 1$, gilt:

$$M^{ed} \bmod n = M^{k\varphi(n)+1} \bmod n = M^{de} \bmod n.$$

- ad (2) 2. Es bleibt also zu zeigen: $M^{e \cdot d} \bmod n = M$.

Man unterscheidet nun die Fälle, dass $p \mid M$ teilt und dass $p \nmid M$ nicht teilt.
 Falls p Teiler von M ist, gilt (mit einem passenden $i \in \mathbb{N}$):

$$\begin{aligned} M^{ed} \bmod p &= M^{k\varphi(n)+1} \bmod p \\ &= \underbrace{(ip)}_{=0 \bmod p}^{k\varphi(n)+1} \bmod p = 0 = M \bmod p. \end{aligned}$$

Ist p nicht Teiler von M , dann gilt: $\text{ggT}(p, M) = 1$, da p nach Konstruktion (siehe Schritt 1 des Algorithmus) eine Primzahl ist.
 Die Sätze von Euler und Fermat sind anwendbar und es gilt:

$$M^{p-1} = 1 \bmod p.$$

Da $(p-1) \mid \varphi(n)$ teilt, gilt weiterhin:

$$M^{k \cdot (p-1) \cdot (q-1)} \bmod p = 1^{k \cdot (q-1)} \bmod p = 1 \bmod p,$$

also auch: $M^{k \cdot (p-1) \cdot (q-1)+1} \bmod p = 1 \cdot M \bmod p$

und damit: $M^{k\varphi(n)+1} \bmod p = M \bmod p$;

also: $M^{ed} \bmod p = M^{k\varphi(n)+1} \bmod p = M \bmod p$.

Analog gilt für q : $M^{ed} \bmod q = M \bmod q$.

Es existieren weiterhin $i, j \in \mathbb{N}$ mit

$$i \cdot p + M = M^{ed} = j \cdot q + M \text{ und damit } ip = jq.$$

¹⁹ D.h. $\text{ggT}(M, n) = 1$.

Da p und q Primzahlen sind, gilt: $\exists r \in \mathbb{N} : j = rp$ und $i = rq$.

Damit gilt: $M^{ed} = jq + M = rpq + M = rn + M$, also

$$M^{ed} \bmod n = rn + M \bmod n = M.$$

•

Beispiel 7.13 (ASCII-Codierung)

Bemerkung: Die im Beispiel verwendeten Primzahlen wurden aus Gründen der Übersichtlichkeit klein gewählt; in der Praxis werden mindestens 200-stellige Dezimalzahlen benötigt.

Gegeben sei der ASCII Zeichensatz als Klartextalphabet, so dass jedes Klartextzeichen durch acht Bits repräsentiert wird.

1. Sei $p = 251$ und $q = 269$, $n = p \cdot q = 67519$, $\varphi(n) = 67000$.
2. Wir wählen $e = 50253$ und $d = 27917$.
3. Mit dem Modul $n = 67519$ können Werte $M \in \{0, \dots, 67518\}$ verschlüsselt werden. Das heißt, dass jeweils zwei Klartextzeichen zu einem Block zusammengefasst werden können. Zwei Zeichen benötigen 16 Bits, womit maximal der Wert 65536 repräsentierbar ist; dieser Wert ist mit dem festgelegten Modul noch verschlüsselbar.
4. Die Klartextnachricht M sei: $M = \text{RSA works!}$

Zunächst erfolgt die Codierung des Klartextes in Dezimalzahlen und anschließend eine blockweise Chiffrierung:

ASCII	Dezimalwert	Kryptotext
RS	$M_1 = 21075$	$C_1 = 21075^{50253} \bmod 67519 = 48467$
A	$M_2 = 16672$	$C_2 = 16672^{50253} \bmod 67519 = 14579$
wo	$M_3 = 30575$	$C_3 = 26195$
rk	$M_4 = 29291$	$C_4 = 58004$
s!	$M_5 = 29473$	$C_5 = 30141$

5. Dechiffrierung des Kryptotextes:

$$M_1 = 48467^{279917} \bmod 67519, \dots,$$

$$M_5 = 30141^{279917} \bmod 67519.$$

▲

Implementierung

RSA-Implementierung

Aus dem Algorithmus ist unmittelbar abzuleiten, dass bei der Verwendung des RSA-Verfahrens Klartexte zunächst in eine numerische Repräsentation transformiert werden müssen, da die Ver- und Entschlüsselungsoperationen Potenzberechnungen auf natürlichen Zahlen sind. Weiterhin ist abzuleiten, dass eine Implementierung des RSA-Verfahrens Algorithmen zur effizienten Berechnung großer Primzahlen (siehe Schritt 1), zur Berechnung der multiplikativen Inversen (siehe Schritt 3) sowie zur effizienten Potenzierung großer Zahlen (siehe Schritte 6 und 7) erfordert. Beispiele für solche Verfahren sind der Rabin-Miller-Primzahltest, der erweiterte Euklid'sche Algorithmus zur Berechnung der multiplikativen Inversen sowie das schnelle Potenzieren mittels wiederholtem Quadrieren und Multiplizieren. Auf diese Algorithmen gehen wir hier nicht weiter ein, sondern verweisen auf einschlägige Literatur zu dieser Thematik, wie beispielsweise [101].

Eigenschaften des RSA-Verfahrens

Aufwand

Die Ver- und Entschlüsselungen sind Potenzberechnungen mit dem festgelegten Modul n . In der Praxis wird für den öffentlichen Schlüssel (e, n) in der Regel ein kleiner Wert e gewählt, so dass das Verschlüsseln und das Verifizieren digitaler Unterschriften (siehe Abschnitt 8.2) schneller durchführbar ist als das Entschlüsseln bzw. Signieren. Sei k die Anzahl der Bits in dem verwendeten, binär repräsentierten Modul n . Für die Operationen, die den öffentlichen Schlüssel benutzen, werden $\mathcal{O}(k^2)$ und für die, die den geheimen Schlüssel verwenden, $\mathcal{O}(k^3)$ Schritte benötigt. Die Schlüsselerzeugung erfordert $\mathcal{O}(k^4)$ Schritte; sie wird aber nur selten durchgeführt.

In der Praxis häufig verwendete Werte für den öffentlichen Schlüssel e sind 3 (wird im PEM-Standard empfohlen [10]), 17 oder 65537 ($= 2^{16} + 1$) (wird vom X.509-Standard empfohlen [39]).

asymmetrisch
versus symmetrisch

Durch die im RSA-Verfahren zu verwendenden Operationen sowie die großen Zahlen, den die Klartextblöcke repräsentieren, auf die die Operationen anzuwenden sind (vgl. Beispiel 7.13), ist eine Ver- oder Entschlüsselung erheblich langsamer als mit einem symmetrischen Verfahren. Aus diesem Grund wird der RSA-Algorithmus normalerweise nicht zur Verschlüsselung großer Datenmengen eingesetzt. Dagegen verwendet man ihn häufig zum Austausch eines symmetrischen Schlüssels, der ja meist nicht länger als 128 Bits ist.

Padding

Padding

Würde man das RSA-Verfahren nach Schulbuch (Textbook RSA) implementieren, würde man ein angreifbares Verschlüsselungssystem erhalten. Beispiele für Angriffe auf Textbook RSA findet man u.a. unter [htt](#)

ps://fachhacks.cr.yp.to. Hierbei spielt die Wahl der Primzahlen eine wichtige Rolle, aber eine Hauptschwäche liegt darin, dass RSA ein deterministisches Verfahren ist, so dass bei der Nutzung des gleichen öffentlichen Schlüssels, gleiche Klartexte auf gleiche Chiffretexte abgebildet werden. Aus der Analyse des Chiffretextes kann ein Angreifer statistische Informationen über den Klartext ableiten. Wie beim ECB-Modus bei symmetrischen Blockchiffren bleiben bei der Verschlüsselung des Klartextes Muster erhalten. Durch das Einbetten von zufälligen Padding-Mustern in den Klartext bevor dieser verschlüsselt wird, ist das Problem zu lösen. Mit dem RSA-OAEP (Optimal Asymmetric Encryption Padding) Verfahren (vgl. RFC 3447) steht ein standardisiertes Padding-Verfahren zur Verfügung.

Wahl der Modul-Größe

Die Größe des Moduls n hat unmittelbare Auswirkungen auf die Sicherheit des Verfahrens, aber auch auf die Performanz der Ver- und Entschlüsselungsoperationen. Je größer der Modul gewählt wird, desto schwieriger ist es, den Schlüssel zu brechen und desto länger ist die Berechnungszeit, die die Operationen erfordern, aber desto größer können dafür auch die zu verschlüsselnden Werte sein. Eine Verdopplung der Modul-Größe führt im Durchschnitt dazu, dass die Zeit für die Verschlüsselung um einen Faktor vier und die für die Entschlüsselung um einen Faktor acht wächst. Das liegt daran, dass bei einer Veränderung des Moduls der Wert e des öffentlichen Schlüssels (e, n) fest bleiben kann, während der Wert d des privaten Schlüssels exponentiell wächst.

Modul-Größe

Größe der Primzahlen

Um eine ausreichende Sicherheit gegen einen Faktorisierungsversuch des Moduls n zu gewährleisten, sollte man p und q mindestens in der Größenordnung von 100-stelligen Dezimalzahlen wählen, so dass der Modul n eine mindestens 200-stellige Dezimalzahl ist. Die Primzahlen sollten sich aber um einige Ziffern in der Länge unterscheiden, da gleich lange Primzahlen Faktorisierungsangriffe erleichtern (s.u.).

Primzahlen

Sicherheit des RSA-Verfahrens

Man geht gemeinhin davon aus, dass die Sicherheit des RSA-Verfahrens auf dem Faktorisierungsproblem für große Zahlen basiert. Dies ist aber nur eine Vermutung, die jedoch bis zum heutigen Tag nicht widerlegt werden konnte. Das heißt, dass noch kein mathematischer Beweis erbracht wurde, dass zur Entschlüsselung eines Kryptotextes C unter Kenntnis des öffentlichen Schlüssels e die Faktorisierung des Moduls n wirklich notwendig ist.

Faktorisierung**Faktorisierungsangriff**

Bei diesem Angriff werden Algorithmen eingesetzt, die die Primfaktzerlegung einer natürlichen Zahl berechnen. Eines dieser Verfahren ist das der Fermat'schen Faktorisierung. Es führt besonders schnell zum Ziel, wenn sich die beiden Primzahlen p und q , deren Produkt n ist, nur wenig von \sqrt{n} unterscheiden. Der Algorithmus beruht auf dem Satz, dass es für eine natürliche Zahl n , die Produkt zweier Primzahlen p und q ist, zwei natürliche Zahlen a und b gibt, so dass gilt:

$$n = a^2 - b^2 = \underbrace{(a+b)}_{=p} \cdot \underbrace{(a-b)}_{=q} = p \cdot q.$$

Die Idee des Algorithmus besteht nun darin, nach Zahlen a und b zu suchen, die die obige Gleichung erfüllen. Man beginnt mit $a = \lfloor \sqrt{n} + 1 \rfloor$ und erhöht a schrittweise um 1 so lange, bis $a^2 - n$ Quadratzahl ist. Mit den Werten von a und b sind dann auch die Werte p und q gefunden.

Zu dieser einfachen Vorgehensweise gibt es noch schnellere Varianten. Wählt man zum Beispiel zu Beginn $a = \lfloor \sqrt{k \cdot n} \rfloor + 1$ mit einem kleinen k und erhöht a schrittweise um 1 so lange, bis $a^2 - kn (= b^2)$ Quadratzahl ist, dann gilt: $a^2 - b^2 = kn$ und man erhält die Primzahlen p und q wie folgt:

$$ggT(a+b, n) = p \quad \text{und} \quad q = \frac{n}{p}.$$

Dieser Algorithmus terminiert sehr schnell, wenn k nahe bei $\frac{p}{q}$ liegt (o.B.d.A. sei $p > q$).

Als Konsequenz der Aussagen über die Effizienz des skizzierten Faktorisierungsverfahren resultiert, dass bei der Generierung eines RSA-Schlüsselpaares sichergestellt werden sollte, dass sich die Primzahlen p und q um einige Ziffern in ihrer Länge unterscheiden.

Konsequenzen eines erfolgreichen Faktorisierungsangriffs

Ein Angreifer kennt das Modul n , den öffentlichen Schlüssel e sowie einen Kryptotext C . Er kennt aber nicht $\varphi(n)$. Gelingt es dem Angreifer, das Modul n zu faktorisieren, so ist er in der Lage, $\varphi(n) = (p-1)(q-1)$ zu berechnen. Damit kann er in einem nächsten Schritt $d^{-1} = e \bmod \varphi(n)$ und damit den Klartext M mit $M = C^d \bmod n$ berechnen.

Durchgeführte Faktorisierungs-Angriffe auf RSA-Module**512-Bit Modul**

Analysen der Sicherheit eines Moduls festgelegter Länge (u.a. in [151]) verdeutlichen, dass bereits beim heutigen Stand der Technik ein 512-Bit langer Modul — wie er von einigen kommerziellen Produkten immer noch verwendet wird — keine ausreichende Sicherheit mehr gewährleistet. Zwar erfordert das Faktorisieren eines 512-Bit RSA-Moduls (512 Bits entsprechen

einer Dezimalzahl mit 155 Stellen) noch einen gewissen Aufwand, aber im August 1999 gelang es einer Forschergruppe den RSA-155 Modul, der dazu von der Firma RSA Security Inc. als so genannte RSA Factoring Challenge im Internet verbreitet wurde, zu faktorisieren. Die Wissenschaftler setzten einen Cray C916 Supercomputer (mit 224 Stunden verbrauchter CPU-Zeit und 3.2 GByte benötigtem Hauptspeicher) sowie über 300 Workstations und PCs (mit einer verbrauchten Rechenzeit von insgesamt 35.7 CPU-Jahren) zur Berechnung ein. Die Faktorisierung hat insgesamt 7.4 Monate in Anspruch genommen.

Im Vergleich dazu benötigte die Faktorisierung des RSA-140 Moduls lediglich 9 Wochen. Auch wenn eine Faktorisierung eines 512-Bit Moduls sicherlich nicht von einem „Gelegenheitshacker“ zu leisten ist, zeigt der erfolgreiche Angriff, dass er mit einem für größere Institutionen durchaus zu leistenden Aufwand durchführbar ist.

Von den RSA Laboratories wurden nach dem erfolgreichen Brechen der RSA-155 Challenge weitere, so genannte RSA Factoring Challenges²⁰ gestellt. Die Primfaktorzerlegung des RSA-576-Moduls, das einer 174-stelligen Dezimalzahl entspricht, gelang im Jahr 2003 Wissenschaftlern des Bonner Max Planck Instituts für Mathematik. Im Jahr 2005 gelang die Faktorisierung des RSA-640 Moduls (eine Zahl mit 193 Dezimalstellen). Im Mai haben die RSA Laboratories überraschend die Challenge zur Faktorisierung von RSA-1024 (309 Dezimalstellen) und RSA-2048 (617 Dezimalstellen) eingestellt, nachdem ebenfalls im Mai 2007 die Bonner Wissenschaftler erfolgreich eine 1039-Bit Zahl (312 Dezimalstellen) faktoriert hatten²¹.

RSA-Challenge

Klar ist, dass durch die zunehmende Rechenleistung und die Möglichkeiten der verteilten Berechnung auf einem Grid von vernetzten Rechnern, kleine RSA-Module keinen ausreichenden Schutz mehr bieten. Empfohlen wird deshalb (vgl. Seite 291) ein Modul von mindestens 2048 Bit und für längerfristige Sicherheit oder auch für sicherheitskritische Anwendungsgebiete ein Modul von 4096 Bit.

Im Folgenden werden drei Schwachstellen und Angriffspunkte beschrieben, die sich aus dem Einsatz des RSA-Verfahrens zum Signieren ergeben. Diese Angriffe haben zum Teil eine große praktische Relevanz, so dass die empfohlenen Abwehrmaßnahmen beachtet werden sollten.

Angriff mit gewähltem Kryptotext²²

Gegeben seien die Kommunikationspartner Alice (A) und Bob (B) mit ihren jeweiligen Schlüsselpaaren (K_E^A, K_D^A) bzw. (K_E^B, K_D^B) sowie der

Chosen-Ciphertext
Angriff

²⁰ <http://www.rsa.com/rsalabs/node.asp?id=2092>

²¹ Siehe <http://www.crypto-world.com/announcements/m1039.txt>

²² Vgl. Abschnitt 7.8.

Angreifer X. Dieser hört die Kommunikation zwischen Alice und Bob ab und versucht, den Klartext einer verschlüsselt übertragenen Nachricht zu bestimmen. Der Angriff durchläuft folgende Schritte:

1. Alice sendet an Bob eine verschlüsselte Nachricht

$$C = RSA(M, K_E^B) = M^{K_E^B} \text{ mod } n,$$

die vom Angreifer X aufgezeichnet wird.

2. X bestimmt eine Zufallszahl $r < n$ und berechnet unter Verwendung des öffentlichen Schlüssels K_E^B von Bob:

$$s = RSA(r, K_E^B) = r^{K_E^B} \text{ mod } n.$$

$$\text{Es gilt: } r = s^{K_D^B} \text{ mod } n.$$

X berechnet weiterhin: $y = s \cdot C \text{ mod } n$ und $t = r^{-1} \text{ mod } n$.

3. X veranlasst Bob dazu, die Nachricht y zu signieren, so dass gilt:

$$RSA(y, K_D^B) = y^{K_D^B} \text{ mod } n = u.$$

4. X berechnet:

$$\begin{aligned} t \cdot u \text{ mod } n &= r^{-1} \cdot y^{K_D^B} \text{ mod } n \\ &= r^{-1} \cdot s^{K_D^B} \cdot C^{K_D^B} \text{ mod } n \\ &= r^{-1} \cdot r \cdot C^{K_D^B} \text{ mod } n = M \end{aligned}$$

Der Angreifer X ist damit im Besitz des Klartextes M , der von Alice zu Bob gesendet wurde.

Der Angriff lehrt, dass man beim Einsatz von RSA zum Signieren und Verschlüssen unterschiedliche Schlüsselpaare für diese beiden Aufgaben verwenden sollte. Ferner ist darauf zu achten, dass man keine unbekannten Dokumente signiert. Dies ist besonders bei der Konfigurierung von Werkzeugen wie Mail-Programmen zu beachten, die diese Schritte dann ggf. automatisch und unbemerkt für den Anwender ausführen.

Existentielle Fälschung

Fälschung

Ein sehr einfach durchzuführender Angriff, der darauf abzielt, die Signatur einer Teilnehmerin — nennen wir sie wieder Alice — zu fälschen, ist die sogenannte existentielle Fälschung (engl. *existential forgery*). Hierbei wählt der Angreifer X einfach eine Zahl $r \in \{0, \dots, n-1\}$, wobei n der öffentlich bekannte RSA-Modul ist. Dann behauptet er, dass r ein von Alice signiertes Dokument sei, z.B. ein Geldbetrag, der am Geldautomat abgehoben werden soll. Zur Verifikation der Signatur berechnet der Empfänger, also z.B. der

Geldautomat, mit dem öffentlichen Schlüssel K_E^A von Alice den Wert $M = r^{K_E^A} \bmod n$ und glaubt, dass Alice diesen signiert hat. Wenn M ein sinnvoller Text ist, wie zum Beispiel ein Geldbetrag, wurde auf sehr einfache Weise die Signatur von Alice gefälscht, ohne dass Alice in der Lage ist nachzuweisen, dass sie nicht die Urheberin des signierten M war.

Solche Angriffe können abgewehrt werden, wenn anstelle des eigentlichen Dokuments nur dessen digitaler Hashwert (vgl. Kapitel 8.1) signiert wird und bei der Vorlage einer Signatur die Kenntnis über das Originaldokument nachgewiesen werden muss. Da es bei kryptografisch sicheren Hashfunktionen für den Angreifer praktisch nicht möglich ist, zu einem gewählten Klartext dessen RSA-signierten Hashwert zu erraten, werden Fälschungsversuche erkannt und abgewehrt. Eine weitere Abwehrmaßnahme besteht darin, den zu signierenden Text zu formatieren, das heißt, ein Padding-Muster zu integrieren. Seien eine Formatierungsfunktion f sowie ein Klartext M gegeben, wobei gelte $f(M) = 00 \mid M$. Für eine erfolgreiche existentielle Fälschung ist es nunmehr notwendig, dass der Angreifer eine Nachricht M' konstruieren müsste, für die gilt, $M' = f(M)$. Dies ist aber bei der gewählten Konstruktion nicht möglich. In der Praxis verwendet man zur Formatierung häufig PKCS#1, ISO 9796-1 oder ISO 9796-2 (u.a. [105]).

Abwehr

Multiplikativität von RSA

Eine weitere Gefahrenquelle beim Einsatz von RSA zur Signaturerstellung besteht in der Eigenschaft der Multiplikativität des RSA-Verfahrens. Es ist somit zu klären, welche Konsequenzen diese Eigenschaft für die Signursicherheit haben. Zur Klärung dieser Frage betrachten wird zwei Klartexte $M_1, M_2 \in \{0, \dots, n-1\}$ sowie deren Signaturen

$$\text{sig}_1 = M_1^{K_D^A} \bmod n \text{ bzw. } \text{sig}_2 = M_2^{K_D^A} \bmod n,$$

wobei K_D^A der RSA-Signaturschlüssel einer beliebigen Teilnehmerin Alice sei.

multiplikativ

Aus den zwei Signaturen sig_1 und sig_2 kann man nun eine dritte, gültige Signatur erzeugen, ohne im Besitz des Schlüssels K_D^A von Alice zu sein. Aufgrund der Multiplikativität von RSA gilt nämlich:

$$(M_1 \cdot M_2)^{K_D^A} \bmod n = s = \text{sig}_1 \cdot \text{sig}_2 \bmod n.$$

Auch dieser Angriff kann durch die Verwendung von Hashfunktionen abgewehrt werden, da es für einen Angreifer praktisch unmöglich ist, zu einem Hashwert, der das Produkt der beiden Einzelwerte M_1 und M_2 ist, einen Text zu finden, der diesen Hashwert erzeugt. Dies stellt sicher, dass der Angreifer nicht in der Lage ist, zu der vermeintlich korrekten Signatur auch das dazugehörige Originaldokument zu präsentieren. Zur Abwehr kann man

Abwehr

auch wieder eine Formatierung, ein Padding einsetzen. Sei wieder f eine Formatierungsfunktion mit $f(M) = 00 \mid M$, wobei M der Klartext ist. Die Formatierungsfunktion f besitzt nicht die Multiplikativitätseigenschaft, d.h. es gilt, $f(M1) * f(M2) \neq f(M1 * M2)$.

Fazit

Fazit

Zusammenfassend ist festzuhalten, dass das RSA-Verfahren nach wie vor als praktisch sicheres Verschlüsselungsverfahren gilt, da die Vermutung, dass die Sicherheit des Verfahrens auf das Faktorisierungsproblem zurückführbar ist, noch nicht widerlegt werden konnte. Voraussetzung für die praktische Sicherheit ist aber, dass man sich bei der Festlegung der Modulgröße und bei der Wahl der dafür notwendigen Primzahlen nach den angegebenen Empfehlungen richtet. Wie auf Seite 343 erklärt, lassen sich RSA-Module mit 512-Bit, aber auch bereits solche mit 576 und 640 Bit faktorisieren. Die Faktorisierung eines 1024-Moduls ist nur noch eine Frage der Zeit, besonders nachdem bereits Anfang 2007 erfolgreich eine 1039 Bit Zahl faktorisiert werden konnte. Setzt man RSA zur Erstellung digitaler Signaturen ein, so sollte man dedizierte Signierschlüssel verwenden, keine unbekannten Texte signieren und, soweit möglich, Signaturen in Kombination mit Hashfunktionen verwenden.

Public-Key Cryptography Standards (PKCS)

PKCS

Das RSA-Verfahren ist Bestandteil der Public-Key Cryptography Standards (PKCS) [98]. Dabei handelt es sich um eine Menge von Standards für den Bereich der asymmetrischen Verschlüsselung. Diese Standards wurden von den RSA Laboratories in Kooperation mit einem Konsortium entwickelt, dem ursprünglich Apple, Microsoft, DEC, Lotus, Sun und das MIT angehörten. Neben dem RSA-Verfahren umfasst PKCS auch das Diffie-Hellman Verfahren zur Schlüsselvereinbarung (vgl. Kapitel 9.3.6) sowie algorithmunabhängige Syntaxfestlegungen für digitale Signaturen und Zertifikate. Wichtige veröffentlichte PKCS Standards sind PKCS #1, #3, #5, #6, #7 und #8. PKCS #1 spezifiziert den Einsatz von RSA zur Verschlüsselung und zum Signieren von Daten. PKCS #3 definiert ein Diffie-Hellman Protokoll und PKCS #5 beschreibt, wie eine Zeichenkette mittels eines geheimen Schlüssels verschlüsselt werden kann, wobei der Schlüssel von einem Passwort abgeleitet wird. PKCS #6 erfasst erweiterte X.509 Zertifikate (siehe Kapitel 9.1.1) und PKCS #7 definiert eine allgemeine Syntax für Nachrichten, die um kryptografische Zusätze wie Verschlüsselungen oder Signaturen angereichert sind. PKCS #8 beschreibt das Format der Informationen (u.a. Schlüssel, Initialisierungsvektor), die bei der Verwendung hybrider Kryptosysteme zu verwenden sind.

7.7 Elliptische Kurven Kryptografie (ECC)

Klassische asymmetrische Verfahren wie RSA erfordern aufwändige Exponentiationsoperationen in Ganzzahl-Ringen und Galois-Feldern, wobei Parameter verarbeitet werden müssen, die länger als 2048 oder 4096 Bit sind. Diese Parametergrößen führen bei heutigen Rechnerarchitekturen mit einer 32- oder 64-bit Arithmetik zu einem sehr hohen Berechnungsaufwand. Zugleich sind diese Parameter auch sehr specheraufwändig, so dass die Verfahren in ressourcenbeschränkten Umgebungen, wie eingebetteten Systemen, nur eingeschränkt eingesetzt werden können.

Bereits frühzeitig wurde deshalb nach neuen asymmetrischen Ansätzen gesucht, die mit einer deutlich geringeren Parametergröße auskommen, ohne das Sicherheitsniveau zu senken. Mit kryptografischen Verfahren basierend auf der mathematischen Struktur einer elliptischen Kurve wurde eine solche Alternative bereits in den 80er Jahren gefunden. Die Anwendung elliptischer Kurven für die Kryptografie wurde erstmals 1985 unabhängig von N. Koblitz [102] und V. Miller [122] vorgeschlagen.

Die elliptische Kurven-Kryptografie (engl. *Elliptic Curve Cryptography* ECC) arbeitet auf einer Menge von Punkten auf einer Kurve, wobei die Punktmenge eine zyklische Gruppe definiert. Eine präzise Definition elliptischer Kurven ist in Definition 7.13 angegeben. Verfahren basierend auf elliptischen Kurven kommen mit Parametern der Länge von 160-256 Bit aus, was zu einer signifikanten Reduktion des Berechnungs- und auch Specheraufwandes führt. Tabelle 7.6 auf Seite 354 enthält einen Vergleich der Schlüssellängen.

ECC

Elliptische Kurven-Kryptografie basiert auf dem schwer zu lösenden Problem des Diskreten Logarithmus und ist damit eine Erweiterung der Verfahren von beispielsweise ElGamal [59]. Das der elliptischen Kurven-Kryptografie zugrundeliegende Problem wird als Elliptic Curve Discrete Logarithm Problem (ECDLP) bezeichnet und gilt derzeit als noch schwerer zu lösen als das Faktorisierungsproblem für ganze Zahlen, das die Grundlage von RSA bildet. Das ECDLP-Problem gilt zudem auch als schwerer als die Berechnung des Diskreten Logarithmus in einer multiplikativen, endlichen Gruppe, wie sie dem klassischen Diffie-Hellmann Verfahren (vgl. Abschnitt 9.3.6) zugrunde liegt.

ECC Verfahren sind bereits vielfältig im Einsatz, wie beispielsweise in SSL/TLS, OpenSSL zur Schlüsselvereinbarung in Kombination mit dem Diffie-Hellman-Verfahren (DH) als ECDH-Verfahren, in Smartcards insbesondere zur Erstellung von Signaturen u.a. als ECDSA mit dem digitalen Signatur-Standard DSA, oder aber auch in hoheitlichen Ausweisdokumenten, wie dem digitalen Personalausweis (nPA). Auch das Trusted Platform

Anwendungen

Modul als Basis des Trusted Computing (vgl. Kapitel 11.4) wird ab Version TPM 2.0 die elliptische Kurven-Kryptografie unterstützen.

Im Folgenden werden wir die wichtigsten Grundlagen der EC-Kryptografie kurz einführen. Eine ausführliche Behandlung der Thematik ECC sprengt jedoch den Rahmen dieses Buches. Interessierte Leser seien u.a. auf [136] für eine tiefer gehende Behandlung der Thematik verwiesen.

7.7.1 Grundlagen

Eine elliptische Kurve ist charakterisiert durch eine Menge von Punkten, die eine Polynom-Gleichung lösen. Die grundlegende Idee besteht darin, dass man auf einer elliptischen Kurve eine zyklische Gruppe definieren kann, so dass in dieser Gruppe das Problem des diskreten Logarithmus sehr schwer zu lösen ist. Die zyklische Gruppe ist eine Menge von Elementen mit einer Verknüpfungsoperation, für die die bekannten Rechenregeln der Multiplikation und Addition gelten. Abbildung 7.13 zeigt ein Beispiel einer elliptischen Kurve über reellen Zahlen \mathbb{R} : $y^2 = x^3 - 3x + 3$.

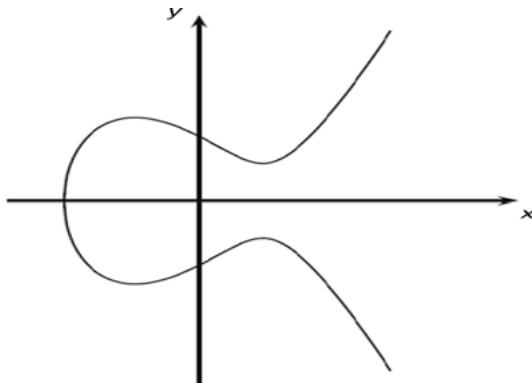


Abbildung 7.13: Elliptische Kurve $y^2 = x^3 - 3x + 3$ über \mathbb{R} .

Eine elliptische Kurve ist wie folgt definiert.

Definition 7.13 (Elliptische Kurve)

Gegeben sei \mathbb{Z}_p , mit $p > 3$, mit $\mathbb{Z}_p = \{0, \dots, p-1\} \text{ mod } p$. Eine elliptische Kurve E über \mathbb{Z}_p ist die Menge der Punkte (x,y) aus \mathbb{Z}_p , für die gilt:

$$y^2 = x^3 + ax + b \text{ mod } p$$

und einem Punkt Θ im Unendlichen, wobei für $a, b \in \mathbb{Z}_p$ gilt: $4a^3 + 27b^2 \neq 0 \text{ mod } p$.

Die Parameter a, b definieren die Form der jeweiligen Kurve.

Die Bedingung $4a^3 + 27b^2 \neq 0 \bmod p$ stellt sicher, dass die Kurve nicht-singulär ist, d.h. dass man für jeden Punkt auf der Kurve die Tangente bestimmen kann²³.

□

Eine graphische Darstellung einer elliptischen Kurve über \mathbb{Z}_p entspricht jedoch nicht einer intuitiven Vorstellung einer Kurve, sondern es ergibt sich eine Punktwolke in einem 2-dimensionalen Gitter (vgl. Abbildung 7.14). Wählt man als Basis die reellen Zahlen, so erhält man eine intuitiv eingängige Kurvenform, wie beispielsweise in Abbildung 7.13 verdeutlicht.

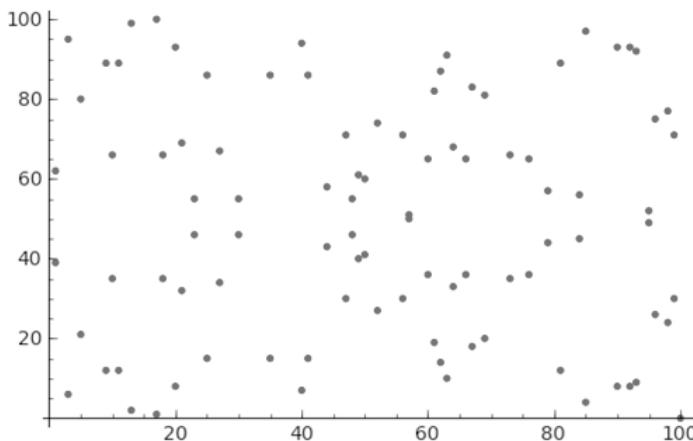


Abbildung 7.14: Elliptische Kurve über einem Ganzzahlkörper graphisch als Punktwolke repräsentiert.

Für die Anwendung von elliptischen Kurven in der Kryptografie sind die Primzahlkörper $GF(p)$ über einer großen Primzahl p sehr gut geeignet. Eine elliptische Kurve basierend auf einem Primzahlkörper $GF(p)$ kann dann auch durch folgende Kurvengleichung vereinfacht charakterisiert werden: $y^2 = x^3 + ax + b$. Die Kurve ist achsensymmetrisch zur x-Achse. Alternativ verwendet man in der Praxis anstelle eines Primzahlkörpers $GF(p)$ auch das Galoisfeld $GF(2^m)$.

Die Menge der Punkte, die die Polynom-Gleichung aus Definition 7.13 erfüllen, bilden eine zyklische Gruppe²⁴; dies ist für das diskrete Logarithmus-Problem essentiell. Eine elliptische Kurve basiert somit auf zwei unter-

²³ Diese Eigenschaft ist wichtig, da die Operationen auf den Kurvenpunkten, wie die Punktaddition, auf Tangentenberechnungen basieren.

²⁴ Eine zyklische Gruppe ist eine Gruppe, deren Elemente als Potenz eines ihrer Elemente a , der primitiven Einheitswurzel, dargestellt werden können, also a^1, a^2, a^3, \dots .

schiedlichen algebraischen Strukturen, zum einen auf dem Galoisfeld $GF(p)$ über einer großen Primzahl p und zum anderen auf der zyklischen Gruppe der Punkte der Kurve E . Auf die Gruppe-Eigenschaften, wie der Existenz eines neutralen Elements, der Existenz von Inversen oder aber auch der Operationen auf Punkten der Gruppe, nämlich die Punkt-Addition und die Punkt-Verdopplung, wird im Folgenden eingegangen.

Punkt-Addition

Punkt-Addition

Berechnungen auf Punkten elliptischer Kurven sind die Punkt-Addition oder auch die Multiplikation mit einem skalaren Wert. Da die Punkte der Kurve E , die die Eigenschaften aus der Definition 7.13 erfüllen, eine zyklische Gruppe bilden, müssen sich beliebige Punkte addieren lassen, hierfür dient der Punkt im Unendlichen Θ .

Eine Punktaddition kann auf einer elliptischen Kurve E über den reellen Zahlen \mathbb{R} graphisch gut veranschaulicht werden. Um zwei Punkte P und Q zu addieren, legt man eine Gerade durch die beiden Punkte (vgl. Abbildung 7.15). Diese Gerade schneidet die Kurve E an einem Punkt $-(P+Q)$, den man an der x-Achse spiegelt und auf diese Weise den Punkt $P+Q$ erhält.

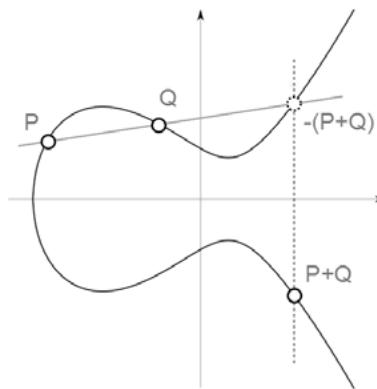


Abbildung 7.15: Punkt-Addition auf einer Kurve über \mathbb{R} .

Diese geometrische Interpretation der Punkt-Addition über den reellen Zahlen ist für die elliptischen Kurven über Primzahlkörpern nicht möglich. Hierfür wird eine mathematische Beschreibung benötigt.

Für eine elliptische Kurve E über dem Primzahlkörper $GF(p)$ gilt:

- Der Punkt im Unendlichen Θ ist das neutrale Element, es gilt: $P + \Theta = P$, für alle Punkte P auf der Kurve.
- Für jeden Punkt P auf der Kurve E existiert ein inverses Element: $-P$, so dass gilt: $P + (-P) = \Theta$.

Für elliptische Kurven über reellen Zahlen gilt: sei $P = (x, y)$ dann ist das inverse Element definiert durch $-P = (x, -y)$, also ergibt sich das inverse Element einfach durch Berechnung des negativen Wertes der y-Koordinate des ursprünglichen Punktes P . Dies lässt sich direkt auch auf Punkte von elliptischen Kurven über Primzahlkörpern $\text{GF}(p)$ übertragen. Hierfür gilt $-P = (x, p - y)$.

- Die Addition von zwei Punkten P und Q auf der Kurve E , wobei E durch $y^2 = x^3 + ax + b \bmod p$ definiert ist, $P + Q = R$, mit $(x_1, y_1) + (x_2, y_2) = (x_3, y_3)$, ist wie folgt definiert:

$$\lambda = \begin{cases} \frac{y_2 - y_1}{x_2 - x_1} & \text{falls } P_1 \neq P_2 \text{ Punkt-Multiplikation} \\ \frac{3x_1^2 + a}{2y_1} & \text{falls } P_1 = P_2 \text{ Punkt-Verdopplung (doubling)} \end{cases}$$

$$\text{und } x_3 = \lambda^2 - x_1 - x_2 \text{ und } y_3 = \lambda(x_1 - x_3) - y_1$$

Graphisch werden die Punkte einer solchen Kurve, wie oben bereits erwähnt, als eine Wolke von Punkten dargestellt (vgl. Abbildung 7.14). Mit der Punkt-Addition springt man scheinbar zufällig in dem Punktraum. Es ist sehr schwer, für eine Folge von Additionsschritten die zugehörige Folge von Sprüngen zu bestimmen, wenn man lediglich den Ausgangspunkt und den Endpunkt der Additionsfolge kennt. Dies macht die Schwierigkeit aus, den diskreten Logarithmus zu berechnen. Das wird nachfolgend noch etwas genauer erläutert.

Problem des Diskreten Logarithmus für EC (ECDLP)

Wir formulieren zunächst das Problem des diskreten Logarithmus auf abelschen Gruppen. Das Problem des diskreten Logarithmus ist für eine abelsche Gruppe G und zwei Gruppenelemente $a, b \in G$ wie folgt definiert: Bestimme eine ganze Zahl $x \in \mathbb{Z}$, so dass $a^x = b$, d.h. $x = \log_a(b)$. Bei genügend großer Gruppenordnung von G ist das zu lösende Logarithmusproblem sehr schwer. Die Ordnung einer Gruppe ist bei endlichen Mengen die Kardinalität der Menge. Also ist bei elliptischen Kurven die Gruppenordnung $\text{ord}(E)$ gegeben durch die Anzahl der Punkte der Kurve. Damit eine elliptische Kurve für einen Einsatz in der Kryptografie nutzbar ist, fordert man eine große Ordnung $\text{ord}(E)$, zum Beispiel $\text{ord}(E) > 2^{160}$ und dass die Gruppenordnung einen großen Primfaktor besitzt, also $\text{ord}(E) = rp$, wobei r eine kleine Zahl und p eine große Primzahl ist.

Logarithmus-
Problem

Wie bereits betont, bildet die Menge der Punkte einer elliptischen Kurve E eine zyklische Gruppe. Bei elliptischen Kurven verwendet man statt der multiplikativen Schreibweise üblicherweise eine additive Schreibweise, um das zu lösende Problem zu charakterisieren.

ECDLP

Definition 7.14 (Diskretes-Logarithmus-Problem auf EC)

Gegeben sei eine elliptische Kurve E über $\text{GF}(p)$. Gegeben seien ein primitives Element P und ein Element Q auf der Kurve E . Das ECDL-Problem besteht darin, einen skalaren Wert d zu finden, für den gilt: $1 \leq d \leq |E|$ und $Q = dP = \underbrace{P + P + \dots + P}_{d \text{ mal}}$

□

Die Multiplikation eines Punktes P einer elliptischen Kurve E mit einem skalaren Wert d wird damit auf die d -malige Punktaddition zurückgeführt. Die Schwierigkeit des zu lösenden diskreten Logarithmusproblems besteht darin, die Anzahl d der Sprünge auf der Kurve E zu bestimmen, die benötigt werden, um von einem Startwert P , der öffentlich bekannt sein kann, zu einem ebenfalls öffentlich bekannten Endpunkt Q auf der Kurve zu gelangen.

Die Punkt-Multiplikation ist das Analogon zur Exponentiation in multiplikativen Gruppen. Dies kann man ebenfalls effizient unter Rückgriff auf einen angepassten Square-and-Multiply-Algorithmus implementieren, bei dem das Quadrieren durch die Punkt-Verdoppelung und das Multiplizieren durch die Punkt-Addition umgesetzt wird. Um die Punkt-Multiplikation dP zu berechnen, wird zunächst der skalare Wert d als Binärzahl $(d_nd_{n-1}\dots d_0)$ dargestellt. Der Algorithmus arbeitet den Bitstring von links nach rechts bis zum niedrigwertigsten Bit d_0 ab:

- Mit dem ersten Bit wird P erzeugt: $(x_n, y_n) = P$.
- Verarbeitung des Bit d_i : Punktverdoppelung und anschließende Punkt-addition, falls $d_i = 1$:

$$(x_i, y_i) = \begin{cases} 2(x_{i+1}, y_{i+1}) + P, & \text{falls } d_i = 1 \\ 2(x_{i+1}, y_{i+1}), & \text{sonst} \end{cases}$$

Performance

ECC ist besonders für den Einsatz in Ressourcen-schwachen Geräten, wie eingebetteten Systemen, geeignet. Die schnellsten bekannten Software-Implementierungen berechnen eine Punkt-Multiplikation in ca. $40\mu s$. Eine hoch-optimierte ECC-Implementierung kann mit ca 10.000 Gattern auskommen. Die Berechnungskomplexität von ECC wächst kubisch in der Länge der zugrundeliegenden Primzahl p . Dies kann man an dem Square-and-Multiply-Algorithmus zur Berechnung der Punkt-Multiplikation unmittelbar ableiten: Punkte werden verdoppelt, was einen quadratischen Aufwand bezogen auf die Bitlänge erfordert. Zusätzlich wird - abhängig vom Wert des jeweiligen Bits - noch eine Punkt-Addition erforderlich, wodurch sich der

insgesamt kubische Aufwand ergibt. Diese Komplexität hat zur Folge, dass eine Verdoppelung der Schlüsselbit-Länge eine Performance-Reduktion um den Faktor $8 = 2^3$ nach sich zieht. Auch das RSA Verfahren und das diskrete Logarithmus-Problem erfordern kubischen Aufwand. Der Vorteil von ECC besteht darin, dass die Schlüssellänge deutlich langsamer als bei den klassischen asymmetrischen Verfahren erhöht werden muss, um eine signifikante Erhöhung des Sicherheitslevels zu erlangen (vgl. Tabelle 9.3).

7.7.2 Einsatz elliptischer Kurven

Elliptische Kurven können als asymmetrische Kryptoverfahren verwendet werden. Dabei bezeichnet der skalare Wert d aus Definition 7.13 den privaten Schlüssel und der Punkt $Q = (x_Q, y_Q)$ den öffentlichen Schlüssel.

Mit dieser Interpretation ergibt sich, wie asymmetrische Schlüsselpaare mittels EC zu erzeugen sind.

Asymmetrische
Kryptografie

Schlüssel-
generierung

- Gegeben sei eine große Primzahl p . Typische Größen für p sind $2^{160} \leq p \leq 2^{320}$. Sei $GF(p)$ der zugehörige Primzahlkörper.
- Gegeben sei eine elliptische Kurve E , mit $E = \{(x, y) \mid y^2 = x^3 + ax + b \text{ mod } p; x, y \in GF(p)\} \cup \{0\}$.
- Wähle P auf E und einen privaten Schlüssel d , mit $d \in \{1, \dots, n - 1\}$.
- Berechne öffentlichen Schlüssel Q , mit $Q = dP$.

Die für eine Ver- und Entschlüsselung benötigten öffentlich bekannten Werte sind der Primzahlkörper $GF(p)$, die Kurve E über $GF(p)$, die Parameter a, b , der Basispunkt P sowie der öffentliche Schlüssel Q , mit $Q = dP$.

Die asymmetrische Verschlüsselung mittels ECC über der Kurve E wird wie folgt durchgeführt.

Verschlüsselung

- Der zu verschlüsselnde Klartext m sei als Punkt M auf der Kurve E repräsentiert.
- Wähle $k \in \{0, \dots, n - 1\}$.
- Berechne: $C1, C2$ mit $C1 = kP, C2 = M + kQ$.

Eine Verschlüsselung wird somit bei ECC auf eine Punktaddition auf der zugrunde liegenden Kurve E zurückgeführt. Ein Angreifer müsste für eine Entschlüsselung entweder den Wert k oder den geheimen Schlüssel d kennen. Dies entspricht aber der Schwierigkeit, das Diskrete Logarithmus-Problem für EC zu lösen.

Entschlüsselung

Zur Entschlüsselung wird der skalare Wert d , also der geheime Schlüssel benötigt, für den gilt, $Q = dP$.

- Gegeben sei der zu entschlüsselnde Kryptotext $C1, C2$, mit $C1 = kP, C2 = M + kQ$.
- In einem ersten Schritt wird kQ berechnet:

$$dC1 = d(kP) = k(dP) = kQ.$$
- Mit dieser Kenntnis kann der Klartext-Punkt M entschlüsselt werden:

$$C2 - kQ = M.$$

Schlüssellängen

Der Vorteil von elliptischer Kurven-Kryptografie besteht darin, dass deutlich geringere Schlüssellänge ausreichen, um ein vergleichbares Sicherheitsniveau wie z.B. bei RSA, zu erreichen werden. So kann mit einem 224 Bit ECC-Schlüssel ein vergleichbares Sicherheitsniveau wie mit einem 2048 Bit RSA-Schlüssel erreicht werden. Bei EC-basierten Verfahren über GF(p) wird die Schlüssellänge durch die Länge der Primzahl p definiert. Sowohl die NIST, als auch die ENISA und das deutsche BSI geben regelmäßig Empfehlungen zu Schlüssellängen heraus. Basierend auf den Empfehlungen der NIST aus dem Jahr 2011 zu Schlüssellängen und den damit verbundenen Sicherheitsniveaus, stellt Tabelle 7.6 die verschiedenen Schlüssellängen symmetrischer Verfahren und asymmetrischer, hier RSA, EC-basierter Verfahren gegenüber, wobei die erste Spalte der Tabelle das erreichte Sicherheitsniveau beschreibt.

Sicherheitsniveau	Symm. Verfahren	RSA, DH	elliptische Kurve
80	3DES (2 Schlüssel)	1024	160
112	3DES (3 Schlüssel)	2048	224
128	AES-128	3072	256
192	AES-192	7680	384
256	AES-256	15360	521

Tabelle 7.6: Vergleich der Schlüssellängen

Für jede der Schlüsselgrößen aus Tabelle 7.6 spezifiziert unter anderem die NIST eine elliptische Kurve über einem Primzahlkörper GF(p). Dazu werden neben der gewählten, ausreichend großen Primzahl p , die Parameter a, b und der öffentlich bekannte Basispunkt P spezifiziert.

So wird beispielsweise für ein 256-Bit EC-Verfahren die NIST Kurve P-256 spezifiziert, mit

- den Koeffizienten: $a = -3$ und $b = 41058363725152142129326129780047268409114441015993725554835256314039467401291$.
- Der zugrundeliegende Primzahlkörper ist definiert als $GF(p_{256})$, wobei $p_{256} = 2^{256} - 2^{224} + 2^{192} + 2^{96} - 1$.
- Der Basispunkt $P = (x_P, y_P)$ wird wie folgt definiert: $x_P = 48439561293906451759052585252797914202762949526041747995844080717082404635286$
 $y_P = 3613425095674979579858512791958788195661106672985015071877198253568414405109$

Gemäß Tabelle 7.6 ist klar, dass ein RSA-Schlüssel, der ein äquivalentes Sicherheitsniveau erreicht, einen 3072 Bit Modul erfordert; der öffentliche Schlüssel bei RSA ist mehr als 12 mal größer als der EC-Schlüssel.

Elliptische Kurven-Kryptografie ist jedoch wie alle asymmetrischen Verfahren deutlich ineffizienter als symmetrische Verschlüsselung, so dass ECC in der Regel zur Berechnung von digitalen Signaturen, meist in Kombination mit dem DSA (vgl. Abschnitt 8.2.3) als ECDSA, sowie zum Schlüsselaustausch in Kombination mit dem Diffie-Hellman-Verfahren als ECDH verwendet wird. Darauf wird in Abschnitt 9.3.6 auf Seite 434 näher eingegangen.

Elliptische Kurven können zudem als Basis für Zufallszahlengeneratoren dienen.

Zufallszahlengenerator Dual_EC_DRBG

Die NIST hat in seinen Special Publications 800-90 vier Pseudozufallszahlen-Generatoren spezifiziert. Einer davon basiert auf ECC und wurde zusammen mit der NSA standardisiert. Das Verfahren ist unter dem Namen Dual_EC_DRBG (Dual Elliptic Curve Deterministic Random Bit Generator) bekannt und wurde aufgrund erheblicher Bedenken, auf die nachfolgend kurz eingegangen wird, am 21.4. 2014 als Standard zurückgezogen. Die generelle Vorgehensweise dieses Verfahrens ist wie folgt: das Verfahren generiert iterierend 16-Bit Blöcke mit Zufallsbits, wobei in jedem Iterationsschritt ein interner Zustand s , der geheim ist, in die Erzeugung der Bits eingeht. Der Standard definiert die zugrundeliegende elliptische Kurve E, sowie zwei Punkte P und Q auf der Kurve.

Dual_EC_DRBG

1. Der interne Startzustand wird durch einen zufällig generierten, skalaren Startwert s_0 definiert.

2. Der Zustandswert s_i in der i -ten Iteration wird mit dem im Standard festgelegten Punkt P multipliziert und liefert als Ergebnis einen Punkt R_i auf der Kurve. Die x-Koordinate x_R^i des Punktes R_i definiert einen neuen skalaren Wert. Aus x_R^i werden in der i -ten Iteration 16 Ausgabebits generiert. Mit dem Wert x_R^i wird der interne Zustand s_{i+1} des nächsten Iterationsschritts $i + 1$ wie folgt berechnet.
- Der skalare Wert x_R^i wird mit dem ebenfalls im Standard festgelegten Basispunkt Q der Kurve E multipliziert und bestimmt damit einen neuen Punkt T_i auf der Kurve E. Die 16 niedrigwertigsten Bit der x-Koordinate x_T^i des Punktes T_i liefern einen Ausgabeblock an Zufallsbits. Nach jedem so erzeugten Ausgabeblock wird der interne Zustand des Generators verändert.
 - Für den Übergang vom internen Zustand s_i auf den neuen Zustand s_{i+1} wird der skalare Wert x_R^i wiederum mit dem im Standard festgelegten Punkt P multipliziert. Daraus resultiert ein neuer Punkt Z_i auf der Kurve E. Der Wert der x-Koordinate x_Z^i des Punktes Z_i definiert den neuen internen Zustand s_{i+1} des Generators.
3. Mit dem neuen internen Zustand s_{i+1} kann ein weiterer Generierungsschritt, beginnend bei Schritt 2, gestartet werden.

Abbildung 7.16 fasst die Schritte zur Generierung von Ausgabebits noch einmal zusammen.

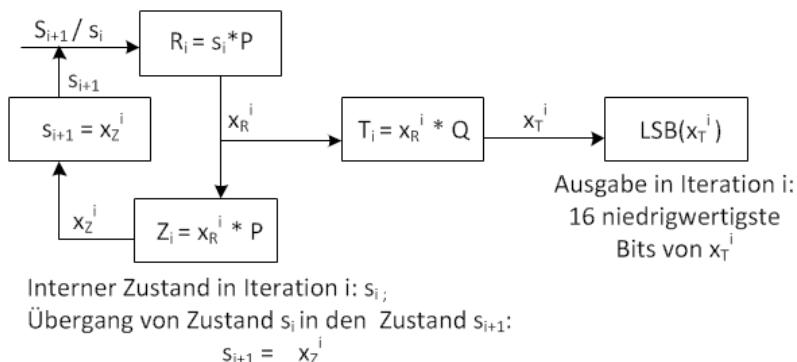


Abbildung 7.16: Funktionsprinzip des Dual_EC_DRBG

Bereits kurz nach der Publikation des Standards im Jahr 2006 wurden im August 2007 von Dan Shumow und Niels Ferguson Schwächen in der spezifizierten elliptischen Kurve veröffentlicht²⁵. Der NIST-Standard legt

²⁵ <http://rump2007.cr.yp.to/15-shumow.pdf>

lediglich die Werte der zu verwendenden Konstanten P und Q für die Kurve fest, aber stellt nicht dar, auf welche Weise diese Werte berechnet wurden. Da P und Q Punkte auf der Kurve E und damit Elemente der damit definierten zyklischen Gruppe sind, gilt: $P = dQ$ für einen Wert d .

Shuman und Ferguson haben gezeigt, dass mit Kenntnis von d der interne Zustand des Generators und damit sein Ausgabeverhalten bestimmbar wäre. Dazu würde es ausreichen, wenn man in der Lage ist, 32 Byte an Ausgabe des Generators zu beobachten. Die kryptografische Sicherheit des Verfahrens beruht darauf, dass das Berechnen des Wertes d mit der Kenntnis von P und Q der Schwierigkeit des Lösen des diskreten Logarithmus-Problems entspricht.

Die Problematik besteht nun darin, dass ein d gewählt und damit die im Standard festgelegten Konstanten P und Q berechnet werden, so dass diejenigen, die in diesem Prozess Kenntnis von d erlangen, in der Lage sind, die mit dem Verfahren erzeugten Zufallszahlen zu kontrollieren. Die Kenntnis von d ist damit eine Hintertür in dem Verfahren.

Da Zufallszahlen die Basis vieler Sicherheitsfunktionen sind, z.B. zur Erzeugung von Schlüsselbits, ist dies eine gravierende Sicherheitsschwäche. Würde beispielsweise der Standard-Dual_EC_DRBG verwendet, um beim Start einer SSL/TLS-Verbindung die Zufallszahl *Client Random* zu erzeugen, könnte mit Kenntnis der Hintertür das in nachfolgenden Schritten berechnete Pre-Master-Secret bestimmt werden.

Der NIST-Standard sah zwar auch eine Option vor, mit der unter Verwendung eines anderen Zufallszahlengenerators neue Werte für die Konstanten generiert werden konnten, so dass die darauf aufbauende Dual_EC_DRBG-Implementierung dann nicht mehr diese mögliche Hintertür besitzen würde. Aber da dies nur eine Option war, konnte man davon ausgehen, dass normalerweise die vordefinierten Werte beim Einsatz des Verfahrens verwendet wurden.

Die NIST hat auf die bereits seit 2007 diskutierten Bedenken erst im Zuge der NSA Affäre im Jahr 2013 umfassend reagiert, da aus den bekannt gewordenen, vertraulichen NSA Unterlagen die Vermutung erhärtet wurde, dass die NSA Kenntnis der Hintertürinformation d besitzt. Eine überarbeitete Version des Standards SP 800-90A ohne Dual_EC_DRBG wurde als Draft am 21. April 2014 veröffentlicht.

Anzumerken ist, dass die oben beschriebene Schwachstelle keine Schwachstelle der zugrundeliegenden elliptischen Kurve ist, sondern sich auf den Prozess der Bestimmung der Parametergrößen bezieht.

7.8 Kryptoanalyse

Die Kryptoanalyse ist die Wissenschaft von den Methoden zur Entschlüsselung von Nachrichten ohne dafür Zugriff auf den Schlüssel zu haben. Mit einer erfolgreichen Kryptoanalyse kann man entweder den Klartext eines verschlüsselten Textes oder den verwendeten Schlüssel bestimmen. Kryptanalytische Methoden setzt man darüber hinaus ein, um Schwachstellen in entwickelten bzw. im Einsatz befindlichen Kryptoverfahren zu ermitteln.

Der Abschnitt gibt einen knappen Einblick in das Gebiet der Kryptoanalyse. Dazu stellt der Abschnitt 7.8.1 zunächst eine Klassifikation von Angriffen vor, bevor in 7.8.2 auf einfache Kryptoanalysemethoden eingegangen wird. In 7.8.3 und 7.8.4 werden dann mit der linearen und der differenziellen Kryptoanalyse moderne Techniken kurz erläutert. Eine ausführlichere Einführung in die Kryptoanalyse liefert u.a. [12].

7.8.1 Klassen kryptografischer Angriffe

Exhaustive Search

Ein einfacher Weg, den Klartext eines gegebenen Kryptotextes zu ermitteln, besteht darin, den Kryptotext mit allen möglichen Schlüsseln des Schlüsselraumes zu entschlüsseln und zu prüfen, ob der resultierende Text einen sinnvollen Klartext ergibt. Diese Art des Angriffs nennt man ein erschöpfendes Durchsuchen des Schlüsselraumes (*exhaustive search* oder auch *brute force*), wozu keine besonderen Analysetechniken eingesetzt werden müssen. Die Anfälligkeit eines kryptografischen Verfahrens gegenüber dedizierten Kryptoanalysen misst man häufig daran, inwieweit der Aufwand für eine erfolgreiche Analyse hinter der einer erschöpfenden Suche zurückbleibt. Das heißt, man misst, inwieweit Schlüssel effizienter gebrochen werden können als mit einer Brute-force-Attacke.

Angriffsklassen

Angriffe auf ein kryptografisches System kann man allgemein danach klassifizieren, welche Informationen dem Kryptoanalytiker über das zur Verschlüsselung eingesetzte System zur Verfügung stehen, wobei stets vorausgesetzt wird, dass der Analytiker vollständige Informationen über das verwendete Verschlüsselungsverfahren besitzt.

Klartext

Beim Klartextangriff (engl. *ciphertext-only*) stehen dem Analytiker mehrere Kryptotexte zur Verfügung, die alle mit demselben Verschlüsselungsverfahren verschlüsselt wurden. Mithilfe der verschlüsselten Texte versucht der Kryptoanalytiker nun, den Klartext oder aber den verwendeten Schlüssel zu bestimmen. Weitere Informationen, die zur Kryptoanalyse herangezogen werden können, betreffen statistische Eigenschaften des zugrunde liegenden Nachrichtenraums wie zum Beispiel die Häufigkeit des Auftretens bestimmter Buchstaben. Falls das verwendete Verschlüsselungsverfahren diese Häufigkeitsverteilung beibehält, so kann der Analytiker daraus Informatio-

nen über den verwendeten Schlüssel ableiten. Dies trifft zum Beispiel auf einfache Substitutionsverfahren zu, die die Häufigkeitsverteilung der Klartextbuchstaben auf den Kryptotext übertragen.

Der Angriff mit bekanntem Klartext (engl. *known-plaintext*) Attacke geht über die Ciphertext-only-Attacke hinaus, da dem Kryptoanalytiker Klartext/Kryptotext-Paare zur Verfügung stehen, die mit dem gleichen Schlüssel chiffriert wurden. Der Analytiker versucht entweder den Schlüssel zu bestimmen oder aber einen Algorithmus zu entwickeln, der jede mit diesem Schlüssel chiffrierte Nachricht entschlüsseln kann. Known-Plaintext-Angriffe treten sehr häufig auf und sind leicht durchzuführen, da der Analytiker dazu meist nicht direkt Kenntnisse über den zu einem verschlüsselten Text gehörenden Klartext besitzen muss. Die erforderlichen Hinweise über den mutmaßlichen Klartext lassen sich nämlich häufig aus dem Verwendungskontext der verschlüsselten Nachricht herleiten. Betrachten wir als Beispiel die Verschlüsselung von Programmen. Deren Quellcode besitzt meist eine einheitliche Struktur bestehend aus Headern oder Präambeln. Deshalb sind die verwendeten Schlüsselworte wie z.B. `include` gute Kandidaten für gesuchte Klartexte. Eine einheitliche Struktur weisen beispielsweise auch Briefe auf, die in einem Standardformat versendet werden und mit standardisierten Floskeln beginnen bzw. enden.

Beim Angriff mit gewähltem Klartext (engl. *chosen-plaintext*) verfügt der Analytiker nicht nur über Klartext/Kryptotext-Paare, sondern er kann auch wählen, welche unverschlüsselten Nachrichten er sich verschlüsseln lässt. Diese Angriffsvariante wird häufig genutzt, um Passwörte zu knacken, indem der Angreifer versucht, das korrekte Passwort des Opfers zu erraten. Dazu wählt er sich ein Passwort – das ist hier der gewählte Klartext – und lässt dieses mit dem Passwortverschlüsselungsverfahren des Systems (z.B. die `crypt(3)`-Funktion in Unix, vgl. Kapitel 10.2.2) verschlüsseln und vergleicht den Kryptotext mit dem verschlüsselt abgelegten Passwort seines Opfers.

Beim Angriff mit gewähltem Kryptotext (engl. *chosen-ciphertext*) kann sich der Analytiker zu ausgewählten Kryptotexten den assoziierten Klartext angeben lassen. Seine Aufgabe hierbei ist es, den verwendeten Schlüssel zu ermitteln. Diese Klasse von Angriffen ist nur auf asymmetrische Verfahren sinnvoll anwendbar (vgl. Seite 343).

bekannter Klartext

gewählter Klartext

gewählter
Kryptotext

7.8.2 Substitutionschiffren

Substitutionschiffren sind dadurch gekennzeichnet, dass einzelne Buchstaben oder Buchstabengruppen des Klartextes durch andere Zeichen oder Zeichengruppen substituiert werden

Monoalphabetische Substitutionschiffren

Eine monoalphabetische Substitution ersetzt jedes Klartextzeichen durch ein fest zugeordnetes Kryptotextzeichen. Für monoalphabetische Substitutionen gilt der erste Invarianzsatz, der besagt, dass die Wiederholungsmuster, die sich in der natürlichen Sprache aufgrund ihrer Redundanz (siehe Abschnitt 7.4.2) ergeben, auch im substituierten Kryptotext erhalten bleiben. Da jede natürliche Sprache redundant ist, ist die Kryptoanalyse solcher Substitutionschiffren relativ einfach, sobald man einen Einstiegspunkt gefunden hat.

häufige Zeichen

Bei einer monoalphabetischen Substitution bleiben die Charakteristika und Häufigkeiten einzelner Zeichen sowie die der Bi- oder Trigramme des Klartextes erhalten. Im Kryptotext kann man daher nach häufig vorkommenden Zeichen suchen und diese auf die in der jeweiligen Sprache häufig vorkommenden Zeichen abbilden. So ist zum Beispiel das e in der deutschen und englischen Sprache mit Abstand der am häufigsten auftretende Buchstabe.

Muster

Als Ansatzpunkt sehr hilfreich ist zudem noch die Suche nach spezifischen Mustern, mit deren Hilfe sich der Arbeitsaufwand zum Teil stark reduzieren lässt. Unter einem Muster versteht man eine Ziffernfolge in Normalform. Um ein solches Muster zu erhalten, ordnet man jedem Buchstaben eine Zahl so zu, dass jede neu vorkommende Zahl in der entstehenden Zahlenfolge links von sich keine größeren Zahlen besitzt. Für die Mustersuche gibt es spezielle Bücher, in denen man sämtliche Muster der jeweiligen Sprache finden kann.

wahrscheinlichstes Wort

Eine Sonderform der Mustersuche ist die Methode des wahrscheinlichen Wortes. Erfolgreich angewandt erleichtert sie den Einstieg erheblich. Das Prinzip ist einfach. Man nimmt den chiffrierten Text und sucht nach einem Wort oder Wortteil, das eventuell in dem Kryptotext vorkommen könnte. Anschließend durchsucht man das Textstück nach Mustern, die mit denen des vermuteten Wortes übereinstimmen. Diese Art der Kryptoanalyse setzt natürlich voraus, dass der Angreifer zumindest den Kontext der Nachricht kennt, bzw. die Hintergründe, aufgrund derer die Nachricht geschrieben worden sein könnte. So konnten z.B. die Alliierten im Zweiten Weltkrieg Nachrichten entschlüsseln, da in den meisten Fällen das Muster 1234135426 = heilhitler nicht fehlen durfte. Auch führten vermutete Datumsangaben innerhalb militärischer Kryptotexte zu erfolgreichen Analysen.

Polyalphabetische Substitutionschiffren

Um statistische Muster zu verschleiern, verzichtet man bei polyalphabetischen Substitutionschiffren auf eine starre Abbildung und verwendet mehrere, voneinander unabhängige Substitutionschiffren. Diese Chiffren wählt man nun abhängig vom jeweils zu verschlüsselnden Klartextzeichen aus. Dazu verwendet man eine Schlüsselphrase, die zum Beispiel ein Wort, ein Satz oder der Inhalt eines Buches sein kann. Die Auswahl der verwendeten Substitutionschiffren ist in der Regel (abhängig von der Schlüsselphrase) periodisch. Bis in die Mitte des 19. Jahrhunderts galt diese Chiffrierung bei der Wahl einer langen Schlüsselphrase ohne mehrfach auftretende Zeichen als sehr sicher. Erst dem deutschen Kryptoanalytiker F.W. Kasiski gelang es 1863 mit einem Verfahren, dem Kasiski-Test, polyalphabetische Substitutionschiffren zu brechen.

Schlüsselphrase

Kasiski-Test

Die Analyse eines chiffrierten Textes erfolgt in zwei Phasen: Zunächst untersucht man den Kryptotext hinsichtlich Auffälligkeiten, mit deren Hilfe man Rückschlüsse auf die Periode p der Schlüsselphrase ziehen kann. Hat man die Periode p ermittelt, so sind anschließend p einfache Substitutionschiffren zu lösen. Damit hat man das Gesamtproblem auf überschaubare Teilprobleme reduziert.

Kasiski-Test

In der ersten Phase untersucht man zunächst, ob im Kryptotext bestimmte Zeichenketten mehrfach vorkommen. Hat man derartige Zeichenketten gefunden, so ist man in der Lage, Rückschlüsse auf den Schlüssel zu ziehen, da man davon ausgehen kann, dass der Abstand zwischen diesen Ketten ein Vielfaches der Periode ist. Dies folgt aus der begründeten Annahme, dass wenn der Abstand gewisser Sequenzen im Klartext ein Vielfaches der Periode der Chiffre ist, auch die zugehörigen Abschnitte des Chiffrats übereinstimmen. Der Kasiski-Test sucht also nach gleichen Sequenzen im Kryptotext und untersucht deren Abstände. Natürlich kann es dabei zu zufälligen Übereinstimmungen kommen, da identische Teilstücke im Kryptotext ja auch anders zustande kommen können. Diese so genannten unpassenden Abstände können jedoch dann bemerkt und ausgesondert werden, wenn ein günstiges Verhältnis zwischen Periode und Kryptotext besteht, also der Kryptotext mehrere Perioden enthält. Als Periode kann man nun alle gemeinsamen Teiler der einzelnen Abstände verwenden, wobei bei mehreren verschiedenen Abständen die einfache Mehrheitsentscheidung am günstigsten ist, falls p häufig genug im Chiffrat vorkommt.

unpassende
Abstände

Dieser Test ist sehr wirkungsvoll, aber die erfolgreiche Anwendung kann durch das Variieren des Schlüssels oder der Periode stark erschwert (z.B. Beale-Chiffre) oder sogar unmöglich gemacht werden (One-time-pad), da ohne die Periodenbestimmung keine weitere Analyse möglich ist. Eine ausführliche Behandlung des Kasiski-Tests findet man u.a. in [177].

7.8.3 Differentielle Kryptoanalyse

differentielle Analyse

Die differentielle Kryptoanalyse wurde 1990 von den beiden Kryptoanalytikern Adi Shamir und Eli Biham [27] veröffentlicht. Es handelt sich dabei um eine Chosen-plaintext-Attacke, die zunächst hauptsächlich gegen den DES gerichtet war. Das Prinzip dieser Attacke basiert darauf, dass Unterschiede zweier Klartexte nach der Verschlüsselung signifikante Unterschiede im Kryptotext hervorrufen, wobei diese Unterschiede mit einer bestimmten Wahrscheinlichkeit auftreten. Variiert man nun Klartexte durch das Vertauschen einzelner Bits und beobachtet die Abstände in den zugehörigen Kryptotexten, so lassen sich, wenn genügend Klartexte zur Verfügung stehen, Rückschlüsse auf den verwendeten Schlüssel ziehen. Das Vorgehen bei der differentiellen Kryptoanalyse wird am Beispiel des DES erklärt.

Beispiel 7.14 (Differentielle Kryptoanalyse des DES)

Abbildung 7.17 zeigt eine Runde des DES-Algorithmus zusammen mit den zu betrachtenden Differenzen ($\Delta X, \Delta A, \Delta B, \Delta C, \Delta Y$). Die Eingangs- und Ausgangspermutationen können ignoriert werden, da sie für die Analyse ohne Bedeutung sind. Gegeben seien zwei Klartexte X und X' , die sich nur in bestimmten Bits unterscheiden; d.h. die Differenz ΔX zwischen X und X' ist bekannt. Auch die Ausgaben Y und Y' sind bekannt und somit die Differenz ΔY . Da die Expansionsfunktion $E(X)$ bekannt ist, kennt man auch die Differenz ΔA zwischen A und A' , wobei gilt: $A = E(X), A' = E(X')$.

Die Verknüpfung von A und A' mit dem Teilschlüssel K_i mittels XOR ändert nichts an ihrer Differenz, da beide denselben Schlüssel verwenden. Zwar sind weder $B = A \text{ XOR } K_i$ noch $B' = A' \text{ XOR } K_i$ bekannt, wohl aber ΔB , das gleich ΔA ist. Kombiniert man nun ΔA mit ΔC , das ebenfalls durch die feste Belegung der S-Boxen bekannt ist, so ergeben sich aufgrund der Tatsache, dass je nach S-Box 16%–31,5% der möglichen Bitkombinationen nicht vorkommen, einige Bitwerte für $A \text{ XOR } K_i$ und $A' \text{ XOR } K_i$.

Wenn nun A und A' bekannt sind, so erhält man Informationen über Teile des Schlüssels. Man beginnt die Analyse mit der letzten, also der 16ten Runde und führt sie mit einer Anzahl von Klartexten durch, die sich immer geringfügig durch einen festen Abstand unterscheiden. Als Resultat dieses Schrittes erhält man dann den 16ten Unterschlüssel K_{16} , also einen 48-Bit-Schlüssel. Nun können weitere Runden nach diesem Verfahren durchlaufen werden, um die restlichen 8 Bits des Schlüssels zu erhalten oder man gewinnt die fehlenden Bits mittels eines Brute-force-Angriffs.

Differenzen

Charakteristika

Wird das Verfahren mehrmals durchgeführt, so erkennt man, dass es Klartextpaare mit gewissen Abständen gibt, die mit einer hohen Wahrscheinlichkeit bestimmte Abstände in den Ergebnischiffertexten verursachen, dass also gewisse Eingabedifferenzen bestimmte Ausgabedifferenzen hervorru-

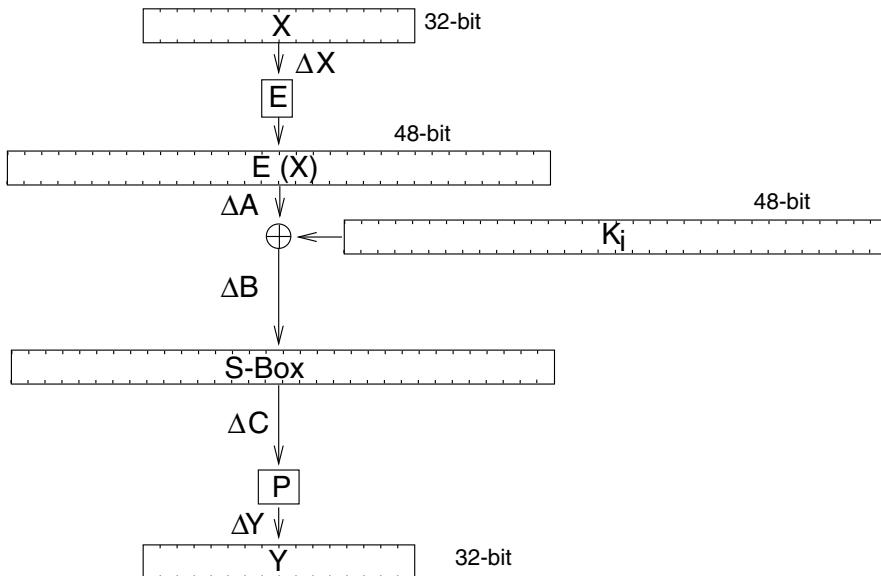


Abbildung 7.17: Differenzen in den DES-Runden

fen. Diese Unterschiede nennt man Charakteristika, die für jede S-Box zu ermitteln sind. Klartextpaare, die die Charakteristika erfüllen, werden als korrekte Paare bezeichnet, da mit deren Hilfe der richtige Schlüssel herleitbar ist. Demgegenüber lassen Paare, die die Charakteristika nicht erfüllen, nur auf zufällige Rundenschlüssel schließen. Das Finden von Charakteristika ist sehr mühsam, langwierig und die Hauptaufgabe der differentiellen Kryptoanalyse.

Insgesamt erweist sich der DES als sehr stark gegen Angriffe mittels der differentiellen Kryptoanalyse. Dies ist darauf zurückzuführen, dass beim Entwurf des DES das Prinzip dieser Analysemethode den DES-Entwicklern bereits bekannt war, obwohl es der breiten Öffentlichkeit erst etwa 1990 zugänglich gemacht wurde.



Auch der aktuelle Standard AES gilt als stark gegen Differentielle Kryptoanalyse. AES

7.8.4 Lineare Kryptoanalyse

Die lineare Kryptoanalyse wurde 1994 von Mitsuru Matsui [114] entwickelt. Sie basiert auf statistischen, linearen Relationen zwischen Klartext, Kryptotext sowie Schlüssel und versucht, das Verhalten eines Verschlüsselungsverfahrens ohne Kenntnis des verwendeten Schlüssels durch lineare Approximationen zu beschreiben.

lineare Analyse

Approximationen

Das Verfahren arbeitet mit einer Known-plaintext-Attacke, wobei der Analytiker beliebig viele Klartexte und die zugehörigen Kryptotexte erhält. Das Vorgehen lässt sich grob wie folgt zusammenfassen. Man verknüpft sowohl einige Klartextbits als auch einige Kryptotextbits mittels XOR miteinander und führt die Ergebnisse mit XOR zu einer 1-Bit Ausgabe zusammen. Dieses Bit repräsentiert die XOR-Verknüpfung einiger Bits des unbekannten Schlüssels. Auf die beschriebene Weise hat man mit einer gewissen Wahrscheinlichkeit eine lineare Approximation des Verhaltens des Kryptosystems mit dem unbekannten Schlüssel entwickelt. Ist diese Wahrscheinlichkeit größer als 0.5, so kann man daraus eine positive Tendenz ableiten. Das Ganze wiederholt man mit vielen Klartexten und den zugehörigen Kryptotexten; je mehr Daten man der Analyse zugrunde legt, desto verlässlicher sind die ermittelten Tendenzen. Wir skizzieren wiederum die Anwendung des Verfahrens anhand der Analyse des DES.

Beispiel 7.15 (Lineare Kryptoanalyse des DES)

Die Anfangs- und Endpermutationen sind wieder zu vernachlässigen. Zunächst sucht man nach einer linearen Approximation für die erste DES-Runde. Da die S-Boxen die einzigen nicht-linearen Bestandteile des DES sind, ist deren Verhalten zu approximieren.

**S-Box
Approximation**

Jede S-Box besitzt sechs Eingabebits und erzeugt vier Ausgabebits, so dass es pro S-Box $2^6 = 64$ mögliche XOR-Kombinationen für die Eingabe und $2^4 = 16$ mögliche XOR-Kombinationen für die Ausgabe gibt. Nun berechnet man die Wahrscheinlichkeit, dass es für eine willkürliche Eingabe-XOR-Kombination eine Ausgabe-XOR-Kombination gibt, die dieser Eingabe-XOR-Kombination entspricht. Man bemüht sich, möglichst gute Approximationen für die verschiedenen DES-Runden zu finden, um so zu einer guten Gesamtapproximation zu kommen.

Ergebnis

Unter Anwendung der linearen Kryptoanalyse benötigt man zum Brechen eines Schlüssels eines vollständigen 16-Runden DES im Durchschnitt 2^{43} bekannte Klartext/Kryptotextpaare und liegt damit deutlich unter dem Aufwand einer Brute-force-Analyse. Die lineare Analyse ist das zurzeit effektivste Verfahren, um den DES zu analysieren.

**AES**

Auch der aktuelle Standard AES gilt als stark gegen Lineare Kryptoanalyse.

8 Hashfunktionen und elektronische Signaturen

Der erste Teil dieses Kapitels behandelt in Abschnitt 8.1 Verfahren zur Berechnung kryptografisch sicherer Hashwerte für digitale Dokumente. 8.1.1 führt zunächst die Anforderungen ein, die an eine sichere Hashfunktion zu stellen sind. Die Abschnitte 8.1.2 und 8.1.3 präsentieren mit blockchiffrenbasierten Techniken und dem dedizierten Hashverfahren SHA gängige Techniken. Zum Nachweis des authentischen Dokumentenursprungs werden Message Authentication Codes verwendet. Abschnitt 8.1.4 stellt die entsprechenden Verfahren vor. Der zweite Teil dieses Kapitels beschäftigt sich dann mit elektronischen Signaturen¹. Abschnitt 8.2.1 erläutert zunächst die allgemeinen Anforderungen, die an digitale Signaturverfahren gestellt werden. Wichtige Verfahren zur Erstellung elektronischer Unterschriften werden in Abschnitt 8.2.2 erklärt und in Bezug auf die Erfüllung der allgemeinen Anforderungen analysiert. Als konkretes Verfahren wird in 8.2.3 der digitale Signaturstandard (DSS) vorgestellt.

8.1 Hashfunktionen

Sichere Hashfunktionen bilden wichtige Bestandteile heutiger Verschlüsselungsinfrastrukturen. Mittels kryptografisch sicherer Hashfunktionen werden eindeutige, so genannte digitale Fingerabdrücke von Datenobjekten berechnet und zusammen mit dem Objekt versandt bzw. gespeichert, so dass der Empfänger bzw. Objektnutzer anhand dieses Fingerabdrucks die Integrität des Objekts prüfen kann. Das heißt, dass damit unautorisierte Modifikationen aufdeckbar sind. In zunehmendem Maß ist es darüber hinaus erwünscht bzw. erforderlich, neben der Integrität auch die Authentizität des Datenursprungs überprüfen zu können. Hierfür werden so genannte Message Authentication Codes verwendet.

¹ Die Begriffe elektronische und digitale Signatur werden im Folgenden synonym verwendet

8.1.1 Grundlagen

allgemeine
Hashfunktion

Hashfunktionen werden in vielen Teilgebieten der Informatik eingesetzt, um den Zugriff auf Objekte effizient zu realisieren bzw. ungeordnete Objektmen gen zu verwalten. Eine Hashfunktion definiert einen endlichen Bildbereich, der häufig als Adressbereich bezeichnet wird. Der Adressbereich ist in der Regel erheblich kleiner als der Urbildbereich, der das Universum genannt wird. Eine Hashfunktion ist eine *nicht injektive*² Abbildung, die jedes Objekt des Universums auf eine Hashadresse abbildet. Wichtige Anwendungsbereiche für Hashfunktionen sind schnelle Such- und Zugriffsverfahren im Datenbankumfeld oder auch Adressierungsfunktionen in Betriebssystemen zur effizienten Berechnung realer Adressen, ausgehend von virtuellen. Dort werden in der Regel einfache Funktionen wie die Modulo-Rechnung bezüglich einer Primzahl p zur Berechnung von Hashwerten verwendet.

Kollision

Da Hashfunktionen nicht injektiv sind und ihr Bildbereich meist erheblich kleiner ist als das abzubildende Universum, können Kollisionen auftreten. Das heißt, dass zwei unterschiedliche Objekte u_1, u_2 des Universums auf den gleichen Hashwert abgebildet werden, $h(u_1) = h(u_2), u_1 \neq u_2$. In den genannten Anwendungsbereichen ist dies allein ein Verwaltungsproblem, das man mit bekannten Techniken zur Kollisionsauflösung wie Kollisionslisten oder mehrfaches Hashing löst. Soll jedoch mit einer Hashfunktion ein Wert berechnet werden, der ein Objekt eindeutig charakterisiert und damit die Überprüfung der Integrität von Daten ermöglicht, so werden diese Eigenschaften durch das Auftreten von Kollisionen in Frage gestellt. Das bedeutet, dass man im Kontext sicherer Systeme nicht an Techniken zur Kollisionsbehandlung, sondern an Maßnahmen zu deren Vermeidung, bzw. präziser an Maßnahmen zur Verringerung der Wahrscheinlichkeit des Auftretens von Kollisionen interessiert ist. Im Folgenden werden schrittweise die Anforderungen an Hashfunktionen eingeführt, die zu Funktionen mit den gewünschten Eigenschaften führen.

Eindeutigkeit

Kryptografisch sichere Hashfunktionen H erzeugen zu einem beliebig langen Wort M aus dem Universum von H einen Wert $H(M)$, den Hashwert (engl. *message digest*) fester Länge. Die Eigenschaften einer schwach kollisionsresistenten Hashfunktion sind mit Definition 8.1 erfasst.

Hashwert

schwach
kollisionsresistent

Definition 8.1 (Schwach kollisionsresistente Hashfunktion)

Gegeben seien zwei endliche Alphabete A_1 und A_2 .

Eine schwach kollisionsresistente Hashfunktion ist eine nicht injektive Funktion

$$H : A_1^* \longrightarrow A_2^k,$$

² $f : A \longrightarrow B$ heißt *injektiv*, genau dann, wenn $\forall x, y \in A$ gilt: $x \neq y \Rightarrow f(x) \neq f(y)$.

die folgende Eigenschaften erfüllt:

1. H besitzt die Eigenschaften einer Einweg-Funktion (vgl. Definition 7.8). Man nennt diese Eigenschaft im Zusammenhang mit kryptographischen Hashfunktionen auch *Preimage Resistance*.
 - Der Hashwert $H(M) = h$, mit $|h| = k$, ist bei gegebener Eingabe M leicht zu berechnen.
 - Gegeben sei $h = H(M)$. Das Bestimmen des Wertes M , mit $M = H^{-1}(y)$ ist nicht effizient möglich.
2. Bei gegebenem Hashwert $h = H(M)$ für ein $M \in A_1^*$ ist es nicht effizient möglich, eine dazu passende Nachricht $M' \neq M$, $M' \in A_1^*$ zu bestimmen, die denselben Hashwert liefert, also $H(M') = h$.

Eine schwach kollisionsresistente Hashfunktion wird auch häufig mit dem Begriff *Second Preimage* charakterisiert.

□

Da wir uns im vorliegenden Buch mit der digitalen Verarbeitung von Daten beschäftigen, werden wir im Folgenden von binären Zeichenvorräten ausgehen, also $A_1 = A_2 = \{0, 1\}$.

Es stellt sich nun zunächst die Frage, ob die Anforderungen von Definition 8.1 hinreichend scharf sind, um damit das Ziel zu erreichen, die Wahrscheinlichkeit für das Auftreten von Kollisionen möglichst klein zu halten. Dazu müssen wir uns die Eigenschaft (3.) näher ansehen. Sie besagt, dass es kein effizientes Verfahren gibt, um zu einer gegebenen Nachricht M eine Nachricht M' so zu konstruieren, dass diese denselben Hashwert liefert, also $H(M) = H(M')$. Sie besagt aber nicht, dass es praktisch ausgeschlossen ist, Paare (M, M') mit $H(M) = H(M')$ zu finden. Das heißt, dass Paare gefunden werden können, deren Hashwerte kollidieren.

Kollisionspaare

Es bleibt somit zu klären, wie hoch die Wahrscheinlichkeit für das Auftreten solcher Kollisionspaare ist und ob sich daraus überhaupt Sicherheitsprobleme ergeben können. Zur Beantwortung der zweiten Frage betrachte man ein Szenario, in dem digitale Signaturen in Kombination mit Hashwerten eingesetzt werden, um die Urheberschaft von Dokumenten zu beweisen. Wir gehen weiterhin davon aus, dass ein Angreifer ein kollidierendes Nachrichtenpaar M und M' mit $H(M) = H(M')$ konstruiert. Nun signiere er den Hashwert $H(M)$ für das zugehörige Dokument M , z.B. einen Kaufvertrag, behaupte aber später, dass sich seine Unterschrift nicht auf das Dokument M sondern auf M' beziehe, das möglicherweise einen ganz anderen Kaufpreis enthält. Die Beweiskraft der digitalen Unterschrift ist somit nicht mehr

Sicherheitsproblem

gegeben, so dass sich aus der Möglichkeit, Kollisionspaare konstruieren zu können, für derartige Szenarien erhebliche Sicherheitsprobleme ergeben.

Zur Beantwortung der ersten Frage greifen wir auf ein Ergebnis aus dem Bereich der Wahrscheinlichkeitstheorie zurück, nämlich auf das bekannte Geburtstags-Paradoxon.

Das Geburtstags-Paradoxon

Geburtstags-
Paradoxon

Das Geburtstags-Paradoxon behandelt die Frage, wie viele Personen in einem Raum sein müssen, damit man eine gute Chance hat, nämlich eine Wahrscheinlichkeit größer als 0.5, dass wenigstens zwei von diesen am gleichen Tag Geburtstag haben. Dazu betrachten wir eine Gruppe von n Personen. Für je zwei Personen dieser Gruppe ist zu prüfen, ob sie am gleichen Tag Geburtstag haben. Es sind somit alle möglichen Personenpaare zu überprüfen, also $(n(n - 1))/2$. Sei k die Anzahl der möglichen Geburtstage, also $k = 365$, dann kann man berechnen, dass für³

$$n \geq (1 + \sqrt{1 + (8 \ln 2)k})/2$$

die Wahrscheinlichkeit größer als 0.5 ist, dass zwei Personen am gleichen Tag Geburtstag haben. Das heißtt, es genügen bereits 23 Personen, um diese Wahrscheinlichkeit zu erzielen.

Test-Anzahl

Interpretiert man k , also die Anzahl der möglichen Geburtstage, als die Menge der möglichen Hashergebnisse und die Anzahl n der erforderlichen Personen als die Anzahl der Tests, die durchzuführen sind, um die Wahrscheinlichkeit so signifikant zu steigern, dass ein Kollisionspaar gefunden wird, so kann man als allgemeines Ergebnis festhalten, dass etwas mehr als \sqrt{k} viele Tests ausreichen, also $n > \sqrt{k}$.

Demgegenüber sind mindestens 183 Personen erforderlich, um mit einer Wahrscheinlichkeit größer als 0.5 eine Person zu finden, deren Geburtstag mit einem vorgegebenen Geburtstag übereinstimmt (vgl. Anforderung an schwach kollisionsresistente Hashfunktionen aus Definition 8.1).

Das Ergebnis des Geburtstags-Paradoxons ist wie folgt auf Hashfunktionen zu übertragen: Sei n die Anzahl von Eingaben (Personen), also Worte $w \in \{0, 1\}^*$, und k die möglichen Ausgaben der Hashfunktion (Geburtstage), wobei die Verteilung der Hashwerte die Gleichverteilung sei. Aus dem Ergebnis des Geburtstags-Paradoxons ergibt sich unmittelbar: Falls

$$n \geq (1 + \sqrt{1 + (8 \ln 2)k})/2$$

Kollisions-
Wahrscheinlichkeit

ist, dann ist die Wahrscheinlichkeit dafür, dass zwei der Eingaben den gleichen Hashwert besitzen, größer als 0.5. Das heißtt, wenn man mindestens

³ Beachte: bei einem allgemeinen Alphabet A gilt: $n \geq (1 + \sqrt{1 + (8 \ln 2) |A|^r})/2$.

$2^{\frac{k}{2}}$ Hashwerte berechnet, so wird mit einer Wahrscheinlichkeit größer als 0.5 eine Kollision gefunden. Dadurch kann die Komplexität eines Angriffs von 2^k auf die Größenordnung von $k \cdot 2^{\frac{k}{2}}$ reduziert werden.

Beispiel 8.1 (Geburtstagsangriff)

Das Geburtstags-Paradoxon zeigt, dass „nur“ ein Aufwand in der Größenordnung von 2^{32} statt 2^{64} erforderlich ist, um zu einem 64-Bit Hashwert eine Kollision zu finden. Ein Angriff könnte wie folgt ablaufen. Der Angreifer erzeugt 2^{32} Nachrichten M sowie 2^{32} Nachrichten M' und berechnet alle zugehörigen Hashwerte. Er hat damit eine relativ gute Chance, dass eine Kollision auftritt, d.h., dass die Nachrichtenmengen ein Paar (M, M') mit $H(M) = H(M')$ enthalten. Speichert man die beiden Nachrichtenmengen jeweils in einer sortierten Tabelle ab⁴, so kann das gesuchte Paar verhältnismäßig einfach gefunden werden. Dieses Verfahren ist natürlich nur dann anwendbar, wenn der Angreifer die ursprüngliche Nachricht selbst bestimmen kann. In Anlehnung an das zugrunde liegende Problem, nennt man solche Angriffe auch Geburtstagsangriff (engl. *birthday attack*).

▲

Geburtstagsangriff

Das Geburtstags-Paradoxon und der Geburtstagsangriff lehren, dass kryptografisch sichere Hashfunktionen auch diesen Angriffen standhalten müssen. Das heißt, dass k groß genug gewählt werden muss, so dass es praktisch unmöglich ist, $2^{\frac{k}{2}}$ Hashwerte zu berechnen und zu speichern. Diese Forderung wird durch eine Verschärfung der Eigenschaft (3.) von Definition 8.1 erfasst und führt zur Definition der stark kollisionsresistenten Hashfunktion.

Definition 8.2 (Stark kollisionsresistente Hashfunktion)

Gegeben sei eine Funktion $H : A_1^* \longrightarrow A_2^k$.

H heißt stark kollisionsresistente Hashfunktion, wenn sie eine schwach kollisionsresistente Hashfunktion ist und es darüber hinaus nicht effizient möglich ist, zwei verschiedene Eingabewerte M und M' , $M, M' \in A_1^*$ zu finden, deren Hashwerte übereinstimmen, also $H(M) = H(M')$.

□

stark
kollisionsresistent

Stark kollisionsresistente Hashfunktionen werden häufig kurz als kollisionsresistent (manchmal auch in irreführender Weise als kollisionsfrei) bezeichnet.

Beim heutigen Stand der Technik gilt beispielsweise SHA-1 (vgl. dazu auch Abschnitt 8.1.3) mit einem 160 Bit Hashwert nicht mehr als ausreichend

SHA-Verfahren

⁴ Der Sortieraufwand beträgt $\mathcal{O}(2^{\frac{k}{2}} \cdot 2^{\frac{k}{2}})$.

sicher. Als Konsequenz wird dringend von der Nutzung des SHA-1 abgeraten. Microsoft hatte bereits im November 2013 angekündigt, dass in Windows-Systemen ab 2017 keine TLS-Zertifikate mehr akzeptiert werden, die SHA-1 nutzen. Google zog im September 2014 Google nach mit der eigenen Ankündigung, ab 2017 kein SHA-1 basierten TLS-Zertifikate mehr im Chrome-Browser zu akzeptieren.

Als Alternative steht der SHA-2, mit Hashwerten der Länge 256 Bit, also SHA-256 Bit, oder mit Hashwerten der Länge 512 Bit, also SHA-512 Bit zur Verfügung. Seit 2012 ist das Verfahren Keccak⁵ als Alternative zu den herkömmlichen SHA-Verfahren als SHA-3 Standard ebenfalls verfügbar und gilt als starkes Hashverfahren. Der Keccak-Algorithmus wurde nach 5 jähriger Prüfung als SHA-3 Standard ausgewählt. Er erlaubt sowohl Hashwerte fester Länge mit 224, 256, 384 und 512 Bit Länge als auch Hashwerte beliebiger Länge. Dies ist für die Verarbeitung in Stromchiffren relevant. SHA-3 ist sehr effizient in Hardware implementierbar. Die Kernidee des SHA-3 basiert auf dem so genannten Schwamm-Prinzip (Sponge-Funktion). Dabei werden Eingabeblocke verarbeitet (aufsaugen) und Ausgaben der konfigurierbaren Länge (ausdrücken) berechnet. Wichtig ist, dass anders als bei den MD5, SHA-1 und auch SHA-2 Verfahren, ein Weiterhashen nicht einfach möglich ist, da der Hashwert von SHA-3 nicht als interner Zustand genutzt wird, von dem ausgehend ein nächster Hashwert berechnet werden kann. SHA-3 verbirgt seinen internen Zustand nach außen. Durch die Verwendung von HMAC-Wrappern wird dieses Verbergen des internen Zustandes auch für MD5 und SHA-1 bzw. SHA-2 erreicht, wodurch bekannt Angriffe abgewehrt werden. Da SHA-3 auf einem anderen Ansatz als SHA-2 basiert, gefährden Angriffe, die in Zukunft möglicherweise erfolgreich gegen SHA-2 durchgeführt werden, nicht unbedingt auch den SHA-3.

Einsatz von Hashfunktionen

Wie bereits eingangs erwähnt, dienen schwache und starke Hashfunktionen zur Kontrolle der Integrität gespeicherter oder über ein unsicheres Medium übertragener Daten.

Der allgemeine Ablauf einer Integritäts-Kontrolle ist wie folgt:

1. Der Urheber einer Nachricht bzw. eines Dokumentes M berechnet den Hashwert $h = H(M)$ und hinterlegt diesen Wert zusammen mit M . Für den Fall einer Nachrichtenübertragung bedeutet das also, dass sowohl die Nachricht M als auch ihr Hashwert h übertragen werden.

⁵ vgl. <http://keccak.noekeon.org/>

2. Zur Kontrolle der Integrität eines Dokumentes M' wird zunächst dessen Hashwert $h' = H(M')$ berechnet und dieses Ergebnis mit dem dem Dokument assoziierten Hashwert h verglichen.
3. Falls gilt: $h = h'$, wird aufgrund der Kollisionseigenschaften kryptografisch sicherer Hashfunktionen davon ausgegangen, dass auch $M = M'$ gilt. Das heißt, dass es sich bei M' um das unmodifizierte Originaldokument handelt.

In der Praxis werden Hashfunktionen häufig in Kombination mit Signaturverfahren eingesetzt, um nicht nur die Integrität eines Dokumentes zu prüfen, sondern um auch eine eindeutige Urheberschaft nachweisen zu können. Dazu berechnet man zunächst den Hashwert eines Dokumentes und signiert diesen anschließend digital. Die Kombination aus Hashfunktion und Signatur hat den Vorteil, dass nur ein relativ kleiner Hashwert zu signieren ist, während das Dokument, für das die Signatur eigentlich gilt, beliebig groß sein kann. Da zur Signaturerstellung in der Regel die zu signierenden Dokumente mittels asymmetrischer Verfahren (vgl. Kapitel 7.6) verschlüsselt werden, bedeutet dies eine erhebliche Reduktion des Berechnungsaufwandes.

signierter Hash

Für Signaturverfahren werden aus den oben bereits genannten Gründen üblicherweise starke Hashfunktionen verwendet. Schwache Hashfunktionen sind nur dann verwendbar, wenn das zu signierende Dokument vor jedem Signaturvorgang noch verändert wird, zum Beispiel durch das Voranstellen eines zufällig gewählten Präfixes.

Konstruktion sicherer Hashfunktionen

Hashfunktionen werden in der Regel durch eine Folge gleichartiger Kompressionsfunktionen realisiert, durch welche die Eingabe M blockweise zu einem Hashwert verarbeitet wird (siehe Abbildung 8.1). Um Eingaben variierender Länge zu komprimieren, wendet man die Hashfunktion iterierend

Kompression

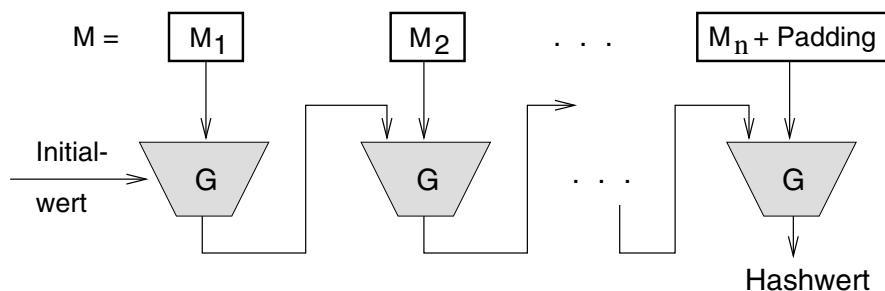


Abbildung 8.1: Allgemeine Arbeitsweise von Hashfunktionen

an. Die Berechnung des Hashwertes für M startet mit einem festgelegten Initialisierungswert IV , der Bestandteil der Spezifikation von Hashalgorithmen ist. Seien G die verwendete Kompressionsfunktion der Hashfunktion H und f eine Funktion $A_1^k \rightarrow A_2^k$.

Dann gilt:

$$\begin{aligned} f(0) &:= IV; \\ f(i) &:= G(f(i-1), M_i), \text{ mit } M = M_1, \dots, M_n, i = 1, \dots, n. \\ H(M) &:= f(n) = h \text{ ist der Hashwert von } M. \end{aligned}$$

Abhängig vom Design der Kompressionsfunktion G unterscheidet man zwei Klassen kryptografischer Hashfunktionen:

1. Hashfunktionen auf der Basis symmetrischer Blockchiffren und
2. dedizierte Hashfunktionen.

Dedizierte Hashfunktionen sind in der Regel deutlich effizienter zu berechnen als symmetrische Blockchiffren und sie unterliegen keinen Exportrestriktionen bzw. Kryptoregulierungen. Im Folgenden werden beide Klassen von Hashfunktionen erklärt.

8.1.2 Blockchiffren-basierte Hashfunktionen

Zur Konstruktion von Hashfunktionen können symmetrische Blockchiffren als Kompressionsfunktion G genutzt werden. Ein Beispiel für eine Hashfunktion, basierend auf einer symmetrischen Blockchiffre, ist die Meyer-Hashfunktion (vgl. Abbildung 8.2). Diese Hashfunktion verwendet den DES (vgl. Kapitel 7.5.4); sie lässt sich aber auf eine beliebige Blockchiffre verallgemeinern.

Im Fall des DES wird die zu komprimierende Eingabe M in n 56-Bit Blöcke M_0, M_1, \dots, M_{n-1} zerlegt, die in den einzelnen Berechnungsschritten als DES-Schlüssel benutzt werden. Der 64-Bit Hashwert wird aus einem vorgegebenen 64-Bit Initialisierungswert IV und den n Eingabeblocks errechnet, wobei die Ausgabe der i -ten DES-Verschlüsselung mit der Eingabe für diesen Verschlüsselungsschritt mittels XOR verknüpft wird und als Eingabe in die $(i+1)$ -te Verschlüsselung einfließt. Damit ergibt sich:

$$\begin{aligned} C_0 &:= IV \\ C_{i+1} &:= \text{DES}(M_i, C_i) \oplus C_i \text{ (für } i = 0, \dots, n-1\text{)} \\ H(M) &:= C_n \end{aligned}$$

Bei dieser Konstruktion handelt es sich um eine schwache Hashfunktion, bei der die Länge des Hashwertes der Blockgröße der zugrunde liegenden Blockchiffre (beim DES 64 Bit) entspricht.

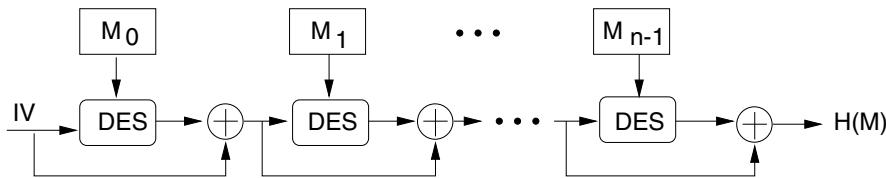


Abbildung 8.2: Meyer-Hashfunktion

Mithilfe von symmetrischen Blockchiffren lassen sich auch starke Hashfunktionen konstruieren, die jedoch einen Hashwert erfordern, dessen Länge mindestens die doppelte Blockgröße erreicht. Die Konstruktion solcher starker Hashfunktionen ist aufwändig, so dass man in der Praxis eher dedizierte Hashfunktionen verwendet.

starke
Hashfunktion

8.1.3 Dedizierte Hashfunktionen

Neben den auf symmetrischen Blockchiffren basierenden Hashfunktionen finden heutzutage vor allem Hashfunktionen Verwendung, bei denen die Kompressionsfunktionen speziell für die Erzeugung von Hashwerten konstruiert wurden. Die Verfahren der SHA-Familie gehören zu den am häufigsten verwendeten Techniken, weshalb das generelle Prinzip des SHA-Verfahrens nachfolgend kurz erläutert wird. In der Praxis wird nach wie vor auch noch der MD5 als Hashfunktion verwendet, obwohl schon seit langem bekannt ist, dass er ohne eine Wrapping-Funktion, z.B. einem HMAC, nicht sicher ist und nicht mehr genutzt werden sollte.

dedizierte H.

Secure Hash Algorithm (SHA)

Der Secure Hash Algorithm SHA ist Bestandteil des Secure Hash Standards und wurde 1993 vom amerikanischen National Institute of Standards and Technology (NIST) als Federal Information Processing Standard [133] veröffentlicht. Nachfolgend erläutern wir die Variante SHA-1, um das Prinzip darzustellen. Der SHA-1 erzeugt 160-Bit Hashwerte und verwendet eine Blockgröße von 512 Bits, wobei eine Paddingvorschrift, gemäß Abbildung 7.3 (c) verwendet wird. Wie weiter oben bereits ausgeführt, sollten in der Praxis SHA-3, als völlig anderes Verfahren, oder SHA-256 oder SHA-512 verwendet werden, um mögliche Kollisionsangriffe aufgrund des zu kleinen Hashwertes bei SHA-1 zu vermeiden.

SHA

Jeder 512-Bit Nachrichtenblock wird in sechzehn 32-Bit Worte W_i zerlegt. Der SHA-1 benötigt insgesamt 80 Verarbeitungsschritte pro 512-Bit Nachrichtenblock. Abhängig vom jeweiligen Verarbeitungsschritt ist eine von vier verschiedenen Grundoperationen anzuwenden. Diese Operationen werden durch eine Funktion f_i beschrieben, mit

$$f_i : \{0, 1\}^{32} \times \{0, 1\}^{32} \times \{0, 1\}^{32} \longrightarrow \{0, 1\}^{32},$$

Algorithmus

wobei gilt⁶:

$$f_i(x, y, z) = \begin{cases} 0x5a827999 + ((x \wedge y) \vee (\neg(x) \wedge z)) & 0 \leq i \leq 19 \\ 0x6ed9eba + (x \oplus y \oplus z) & 20 \leq i \leq 39 \\ 0x8f1bbcde + ((x \wedge y) \vee (x \wedge z) \vee (y \wedge z)) & 40 \leq i \leq 59 \\ 0xca62c1d6 + (x \oplus y \oplus z) & 60 \leq i \leq 79 \end{cases}$$

Abbildung 8.3 skizziert die Kompressionsfunktion des SHA-1, wobei S_x einen zyklischen Linksshift um x Stellen bezeichnet.

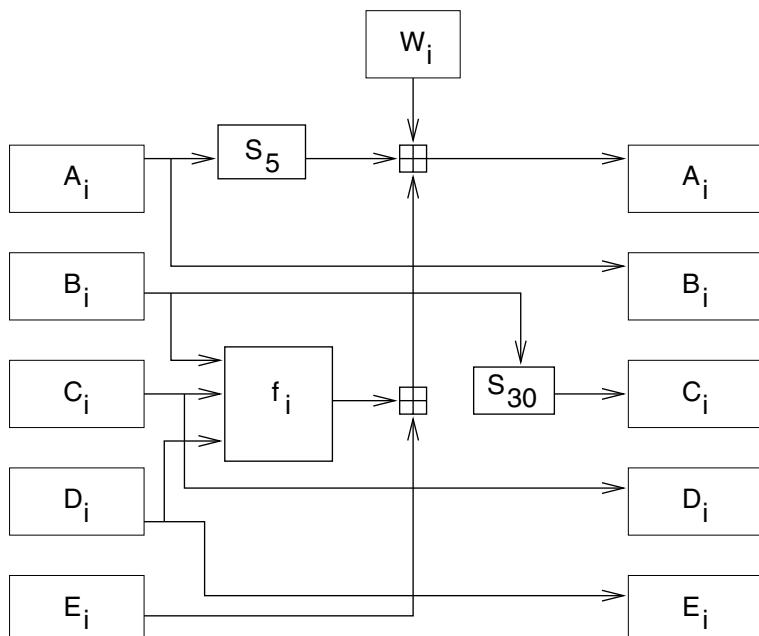


Abbildung 8.3: Kompressionsfunktion des SHA-1

Zur Berechnung des Hashwertes benötigt man zwei Puffer, die jeweils fünf 32-Bit Worte h_0, h_1, h_2, h_3, h_4 bzw. A, B, C, D, E speichern. Diese Puffer werden bei der Initialisierung des Verfahrens mit vorgegebenen Initialisierungswerten belegt.

Die Verarbeitung der 512-Bit Eingabeblocks M_0, M_1, \dots, M_{n-1} erfolgt sequentiell, wobei jeder Block M_i zunächst in sechzehn 32-Bit Blöcke W_0, W_1, \dots, W_{15} zerlegt wird. Die 16 Worte W_i werden dann zu achtzig 32-Bit Wörtern expandiert und zu einem Zwischenergebnis verarbeitet. Nach

⁶ \oplus bezeichnet die XOR-Verknüpfung.

der Verarbeitung des letzten Nachrichtenblocks M_{n-1} ergibt sich der 160-Bit Hashwert aus der Konkatenation der Worte h_0, h_1, \dots, h_4 .

SHA-1-Hashwert

Die blockweise Verarbeitung geschieht wie folgt:

Expansion der W_i FOR t := 16 TO 79 DO $W_t := W_{t-3} \oplus W_{t-8} \oplus W_{t-14} \oplus W_{t-16};$ OD; $A := h_0; B := h_1; C := h_2; D := h_3; E := h_4;$ Verarbeitungsschritte FOR t := 0 TO 79 DO $TEMP := S_5(A) + f_t(B, C, D) + E + W_t;$ $E := D; D := C; C := S_{30}(B); B := A; A := TEMP;$ OD; $h_0 := h_0 + A; h_1 := h_1 + B; h_2 := h_2 + C;$ $h_3 := h_3 + D; h_4 := h_4 + E;$	<i>blockweise Verarbeitung</i>
--	------------------------------------

Sicherheit von SHA-1 und die SHA-2 Familie

Nachdem im August 2004 und im Februar 2005 Angriffe gegen SHA-1 bekannt geworden sind, musste an der langfristigen Sicherheit von SHA-1 ernsthaft gezweifelt werden. Die von Prof. Xiaoyun Wang 2005 beschriebene differentielle Attacke auf den SHA-1 ermöglicht es, eine Kollision bereits in 2^{69} Schritten zu berechnen⁷, statt mit 2^{80} , wie dies für eine starke 160-Bit Hashfunktion normalerweise der Fall wäre. Obwohl dieser Angriff im Jahr 2005 nicht nachgewiesen war, hat das NIST beschlossen darauf zu reagieren. Im Juni 2007 hat das NIST mit dem FIPS 180-3, Secure Hash Standard (SHS)⁸ eine Spezifikation für starke Hashverfahren, der Familie der SHA-2-Verfahren, veröffentlicht und zur Kommentierung freigegeben. Bei den Verfahren handelt es sich um SHA-224, SHA-256, SHA-384 und SHA-512, die alle einen deutlich längeren Hashwert als SHA-1-160 verwenden. Wie bereits weiter vorne erwähnt, wurde im Oktober 2012 von der NIST das Verfahren Keccak⁹ zum SHA-3 Standard erklärt, um eine Alternative zum SHA-2 Verfahren zu besitzen. Im Februar 2017 hat ein Forscherteam der CWI Amsterdam und von Google erfolgreich eine SHA-1 Kollision konstruiert (vgl. <https://security.googleblog.com/2017/02/announcing-first.html>)

SHA-2

⁷ vgl. http://cryptome.org/wang_sha1_v2.zip

⁸ Vgl. http://csrc.nist.gov/publications/fips/fips180-3/fips180-3_final.pdf

⁹ vgl. <http://keccak.noekeon.org/>

sha1-collision.html) und damit auch einen in der Praxis real durchführbaren Angriff gezeigt. Das SHA-1 Verfahren gilt als unsicher und sollte, wie das MD5-Verfahren auch, gar nicht mehr oder nur mit der Wrapping-Funktion als HMAC-Verfahren verwendet werden.

8.1.4 Message Authentication Code

Hashfunktionen dienen zur Überprüfung der Unverfälschtheit von Daten. In der bislang vorgestellten Form kann eine allgemein bekannte Hashfunktion wie SHA-3 von Jedermann auf ein Datum angewandt werden. D.h. ein derart erzeugter Hashwert dient nicht dem Nachweis der Authentizität des Erzeugers. Wollen zwei Kommunikationspartner Alice und Bob überprüfen können, dass die zwischen ihnen ausgetauschten Dokumente einen authentischen Ursprung besitzen, so können sie dazu einen Message Authentication Code (MAC) verwenden.

Definition 8.3 (Message Authentication Code)

Gegeben sei eine Familie $\{H_K \mid K \in \mathcal{K}\}$ von Hashfunktionen, wobei \mathcal{K} ein Schlüsselraum und H_K eine starke bzw. schwache Hashfunktion¹⁰ gemäß Definition 8.2 bzw. 8.1 ist. Die durch diese Familie definierte, parametrisierte Hashfunktion heißt Message Authentication Code (MAC).

□

MAC

MAC-Geheimnis

Ein Message Authentication Code ist somit eine Hashfunktion mit Einweg-Eigenschaften, die zusätzlich noch einen geheimen Schlüssel K verwendet. Der Schlüssel K ist nur den beiden Kommunikationspartnern bekannt, die authentifizierte Dokumente austauschen möchten. Man nennt Authentication Codes deshalb auch Hashfunktion mit Schlüssel (engl. *keyed one-way function*) und der gemeinsame Schlüssel wird als MAC-Geheimnis bezeichnet.

schwache
Authentizität

Im Zusammenhang mit MACs wird häufig irreführender Weise behauptet, dass mittels eines MACs die Authentizität von Daten nachgewiesen werden kann. Ein MAC ermöglicht zwar Aussagen über die Authentizität des Urhebers der Daten, jedoch lassen sich keinerlei Aussagen über die Daten selbst daraus ableiten. Handelt es sich bei diesen zum Beispiel um mobilen Code, so sind aus dem MAC der Daten keine Informationen über dessen Funktionalität (z.B. ob es sich um Schadsoftware handelt) ableitbar. Es handelt sich hierbei also um eine sehr schwache Ausprägung des in Definition 1.4 eingeführten Authentizitätsbegriffs.

¹⁰ Anstelle von $H_K(M)$ schreiben wir im Folgenden auch $H(M, K)$.

Einsatz eines MACs

Unter Nutzung eines MACs kann eine Kommunikation zwischen den Partnern Alice (A) und Bob (B) beispielsweise wie folgt ablaufen:

*authentifizierter
Datenursprung*

1. Alice errechnet den Message Authentication Code mac eines Dokuments M unter Verwendung des vorab mit dem Partner Bob vereinbarten geheimen Schlüssels $K_{A,B}$.

$$MAC(M, K_{A,B}) = mac$$

2. Alice sendet das Dokument M zusammen mit dem Wert mac an Bob.
3. Bob überprüft den Message Authentication Code von der empfangenen Nachricht M' , indem er ihn ebenfalls aus dem gerade empfangenen Dokument M' mit dem ihm bekannten Schlüssel $K_{A,B}$ berechnet. Bei Wertegleichheit ist das Dokument authentisch.

$$MAC(M', K_{A,B}) = mac' \stackrel{?}{=} mac = MAC(M, K_{A,B}).$$

Analog zu den beiden Klassen von Hashfunktionen unterscheidet man auch bei den Message Authentication Codes solche, die auf Blockchiffren basieren, sowie MACs, die auf dedizierten Hashfunktionen aufsetzen.

Blockchiffren-basierte MACs

Weit verbreitet ist die Verwendung des AES-Verfahrens zur MAC-Erzeugung.

AES-basiert

Im einfachsten Fall werden dazu die zu authentifizierenden Daten mit einem AES-Schlüssel unter Verwendung des CBC-Modus (vgl. Seite 309) verschlüsselt. Der MAC der Nachricht ist der letzte Kryptotextblock, also ein 128-Bit Wert.

MAC basierend auf dedizierter Hashfunktion

Mit dem Anwachsen der Menge der Daten, die authentifiziert über ein unsicheres Transportmedium wie das Internet versandt werden soll, wächst der Bedarf an MAC-Verfahren, die einen MAC-Wert deutlich effizienter berechnen, als dies mit herkömmlichen kryptografischen Verfahren möglich ist. Da in vielen TCP/IP-basierten Protokollen und Diensten der Anwendungsebene (ISO/OSI-Schicht 7) bereits kryptografische Hashfunktionen wie SHA-2 oder SHA-3 eingesetzt werden, ist es nahe liegend, MAC-Verfahren zu konzipieren, die auf diesen Funktionen aufsetzen.

Zur Berechnung von MACs kann (u.a. in TLS oder in IPsec, vgl. Kapitel 14.2) eine dedizierte Hashfunktion eingesetzt und um die Verwendung eines Schlüssels ergänzt werden. Dieses auch als Keyed Hash bezeichnete Ver-

Keyed HAsh

fahren setzt den sicheren Austausch eines geheimen Schlüssels zwischen den Kommunikationspartnern voraus. Sei $K_{A,B}$ der entsprechende Schlüssel und M das Dokument, das authentifiziert werden soll. Die übliche Vorgehensweise beim Einsatz von Schlüsseln zusammen mit Hashfunktionen ist, den Schlüssel als einen Bestandteil der zu hashenden Daten zu verwenden. Das heißt, dass ein Dokument M' als eine Konkatenation aus Originaltext und Schlüssel konstruiert wird, $M' = M | K_{A,B}$. Der Keyed-Hash ergibt sich dann durch Anwendung eines dedizierten Hashverfahrens, z.B. SHA-2, auf das so konstruierte Dokument M' , also $\text{SHA-2}(M')$. Da es das Anfügen des MAC-Schlüssels vor oder nach der der Nachricht zu Angriffen führen kann, sollte wenn möglich ein Keyed-MAC unter Nutzung des HMAC-Verfahrens (s.u.) zum Einsatz kommen. HMAC-Verfahren verschleiern den internen Zustand der genutzten Hashfunktion. Dies ist für MD5, aber auch SHA-1 und SHA-2 sehr wichtig gerade wenn eine Keyed-Variante genutzt wird. Ohne die HMAC-Verschleierung geben die genannten Hashverfahren ihren internen Zustand bekannt, der genutzt werden kann, um einen bereits berechneten Hashwert aufzugreifen und darauf aufbauend weitere Hashwerte zu generieren, also weiter zu hashen. Dies ist bei den Keyed-Varianten problematisch. Hier wird unter Pre- oder Suffix-Konkatenation eines geheimen Schlüssels K an eine Nachricht M ein Hashwert mac berechnet. Dieser Wert kann als Initialisierungsvektor aufgegriffen und damit können weitere Klartextblöcke zusätzlich zur ursprünglichen Nachricht M gehasht und ein gültiger MAC-Wert berechnet werden, der an den geheimen Schlüssel K gebunden ist, ohne dass jedoch bei dem weiterhashen dieser Schlüssel bekannt sein muss. Das heißt, ein Angreifer nutzt einen korrekten MAC-Wert mac , verknüpft ihn mit eigenen Daten und der Empfänger akzeptiert die Daten als authentisch.

HMAC

HMAC

Die zentrale Idee des HMAC-Verfahrens (siehe [16] und RFC 2104) besteht darin, den Schlüssel dazu zu verwenden, den Initialwert, mit dem die Anfangswerte der Kompressionsfunktion festgelegt werden, zu beeinflussen. Wesentlich dabei ist weiterhin, dass die zugrunde liegende Hashfunktion als eine Black Box betrachtet wird, also bei ihrer Verwendung nicht modifiziert werden muss.

Gegeben seien eine kryptografische Hashfunktion H , die Eingabeblocks der Länge r Bytes verarbeitet, und ein Schlüssel K . Der Schlüssel K muss ebenfalls die Länge r -Byte besitzen. Dazu muss der Schlüssel ggf. wie folgt bearbeitet werden.

1. Falls gilt: $|K| < r$, dann muss der Schlüssel mit einer Nullsequenz auf die erforderliche Länge aufgefüllt werden.

2. Falls gilt: $|K| > r$, so muss zunächst $H(K) = h$ berechnet werden, wobei H eine kryptografische Hashfunktion ist. Falls danach gilt, $|h| < r$, ist wie unter (1.) zu verfahren, also Auffüllen (padding) von h auf die erforderliche Länge von r -Byte.

Durch das HMAC-Verfahren werden zwei spezielle Strings ipad und opad ¹¹ festgelegt.

$$\begin{aligned}\text{ipad} &= r\text{-malige Konkatenation von } (36)_{16} = (00110110)_2; \\ \text{opad} &= r\text{-malige Konkatenation von } (5C)_{16} = (01011100)_2.\end{aligned}$$

Der HMAC einer Nachricht M unter Verwendung des geheimen Schlüssels K ergibt sich dann wie folgt:

$$\text{HMAC}(M, K) = H(K \oplus \text{opad}, H(K \oplus \text{ipad}, M)).$$

Abbildung 8.4 veranschaulicht die Funktionsweise des allgemeinen HMAC-Verfahrens, wobei H eine beliebige Hashfunktion ist und K_{AB} der gemeinsame geheime MAC-Schlüssel. Die HMAC-Konstruktion umfasst zwei Anwendungen der Hash-Funktion, den inneren und den äußeren Hash, wodurch die Kollisionsresistenz deutlich erhöht wird.

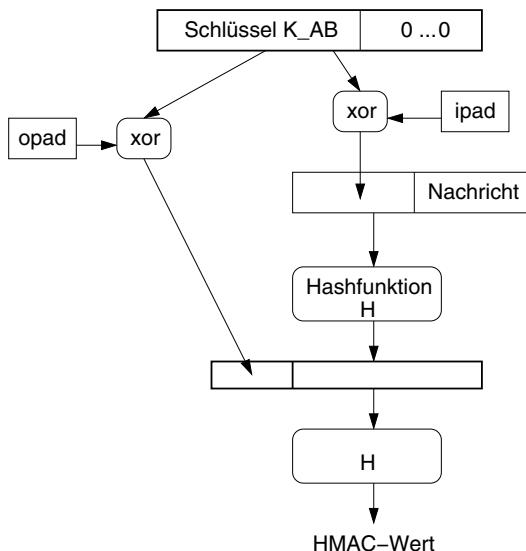


Abbildung 8.4: Funktionsweise des HMAC-Verfahrens

Die erfolgreiche Durchführung eines Geburtstagsangriffs auf einen HMAC, zum Beispiel einen HMAC-MD5, erfordert, dass ein Angreifer mindestens

Sicherheit

¹¹ Das i und das o stehen für inner und outer Padding Daten.

2^{64} Blöcke unter der HMAC-Funktion komprimiert, um die Wahrscheinlichkeit des Auftretens einer Kollision signifikant zu steigern. Geht man von einer Blockgröße von 64 Byte und von einer Verarbeitungsgeschwindigkeit von 1GBit/sec aus, so würde die Berechnung dieser 2^{64} Hashwerte ca. 250 000 Jahre erfordern, vorausgesetzt, dass der Schlüssel über diesen langen Zeitraum hinweg nicht verändert wird. Auch aus den Ergebnissen neuerer Analysen über die Sicherheit des HMAC-Ansatzes¹² haben sich bislang keine Anhaltspunkte für ernsthafte Angriffe auf das HMAC-Verfahren ergeben.

keine Beweiskraft

Zu beachten ist, dass ein MAC bzw. ein HMAC keine digitale Unterschrift für eine Nachricht darstellt, da beide (oder ggf. sogar noch mehr) Partner, die das MAC-Geheimnis kennen, einen MAC berechnen können. Es ist nicht möglich, die gehashte Nachricht einem Dritten, der die Funktion einer Notariats-Instanz übernimmt, vorzulegen und mittels des MAC-Wertes z.B. einen Vertragsbruch geltend zu machen. Einem MAC fehlt die dafür benötigte Beweiskraft, da dessen Berechnung nicht eindeutig einem Urheber zuordenbar ist.

8.2 Elektronische Signaturen

Wie wir im vorherigen Abschnitt gesehen haben, kann durch den Einsatz von sicheren Hashfunktionen oder auch Message Authentication Codes die Urheberschaft einer Nachricht bzw. eines Dokumentes im juristischen Sinn nicht zweifelsfrei bestimmt werden. Dies gewinnt aber gerade im Zusammenhang mit der Nutzung des Internets zur Abwicklung elektronischer Geschäfte, dem E-Business, stark an Bedeutung. Um digitale Dokumente zweifelsfrei einer natürlichen oder juristischen Person zuordnen zu können, benötigt man ein digitales Gegenstück zur handschriftlichen Unterschrift, die digitale bzw. elektronische Signatur¹³. Im Wortlaut des Global and National Commerce Act heißt es über die elektronische Signatur:

The term *electronic signature* means an electronic sound symbol, or process, attached to or logically associated with a contract or other record and executed or adopted by a person with the intent to sign the record.

¹² Vgl. <https://www.cosic.esat.kuleuven.be/publications/article-797.pdf>

¹³ Man beachte, dass damit keineswegs eingescannte handschriftliche Unterschriften gemeint sind.

8.2.1 Anforderungen

Falls elektronische Signaturen als elektronisches Äquivalent zu handschriftlichen Unterschrift eingesetzt werden sollen, ergeben sich die an eine solche elektronische Unterschrift zu stellenden Anforderungen direkt aus den Funktionen einer herkömmlichen (handschriftlichen) Unterschrift. Diese umfassen die Identitäts-, Echtheits-, Abschluss- und Warnfunktion.

Eigenschaften

Identifikation: Die Unterschrift gibt Auskunft über die Person des Unterzeichners.

Echtheit: Die Unterschrift bezeugt, dass das Dokument dem Aussteller vorgelegen hat und von ihm anerkannt wurde.

Abschluss: Die Unterschrift erklärt den Text für inhaltlich richtig und vollständig.

Warnung: Durch die Notwendigkeit, dass eine Unterschrift geleistet werden muss, wird dem Verfasser die rechtliche Bedeutung des Dokuments aufgezeigt.

Aus diesen Funktionen lassen sich Anforderungen ableiten, welche sichere, elektronische Signaturen erfüllen müssen. Die Signatur muss die Identität des Unterzeichners zweifelsfrei bestätigen. Sie darf nicht wieder verwendbar und nur zusammen mit dem Originaldokument gültig sein. Ein signiertes Dokument darf nicht mehr veränderbar sein, bzw. eine nachträgliche Veränderung muss erkennbar sein. Eine Signatur darf nicht zurückgewiesen werden können, d.h. der Unterzeichner eines Dokuments darf das Unterzeichnen des Dokuments nicht im Nachhinein erfolgreich abstreiten können.

Anforderungen

Die gestellten Anforderungen sind nicht leicht zu erfüllen, da durch die spezifischen Eigenschaften digital repräsentierter Informationen Angriffe auf Signaturen, wie beispielsweise das Herausfiltern der Unterschrift und deren Verbindung mit einem anderen Dokument, wesentlich einfacher durchzuführen sind, als bei handschriftlichen Unterschriften auf Papier. Der Einsatz digitaler Techniken zur Unterschriftenherstellung eröffnet andererseits aber auch Vorteile gegenüber handschriftlichen Signaturen. So kann durch die Verwendung von Verschlüsselungstechniken zusätzlich der Inhalt eines Dokuments geheim gehalten werden und durch Message Authentication Codes bzw. kryptografische Hashfunktionen ist ein höheres Maß an Sicherheit vor unbemerkt nachträglicher Manipulation erreichbar. Durch die Verwendung von Zeitstempeln ist es weiterhin einfach möglich, die Gültigkeit einer Signatur auf einen festgelegten Zeitraum zu begrenzen. Da digitale Signaturen mithilfe kryptografischer Verfahren erstellt werden, können die verwendeten Schlüssel bei einer vertrauenswürdigen Instanz hinterlegt werden, so dass der Urheber einer Unterschrift durch eine einfache Anfrage bei dieser Instanz

Probleme

Vorteile

Zeitstempel

ermittelbar ist. Bei handschriftlichen Unterschriften muss demgegenüber im Zweifelsfall ein aufwändig zu erstellendes, graphologisches Gutachten eingeholt werden.

8.2.2 Erstellung elektronischer Signaturen

Im Folgenden werden Techniken zur Erstellung elektronischer Signaturen erklärt. Da nationale und internationale Signaturgesetze den Einsatz asymmetrischer Verfahren vorsehen, beschränken wir uns im Folgenden auf die Verwendung asymmetrischer Verfahren.

In der Praxis werden dedizierte Signaturverfahren oder asymmetrische Verfahren zur Signaturerstellung und -verifikation verwendet, wobei sicherzustellen ist, dass das gewählte asymmetrische Verfahren die Bedingung vier von Definition 7.7 erfüllt. Das heißt, man muss eine Nachricht zunächst mit einem privaten Schlüssel verschlüsseln und sie danach mit dem öffentlichen Schlüssel wieder herstellen können. Einige asymmetrische Algorithmen wie das RSA-Verfahren können sowohl zum Verschlüsseln als auch zum Signieren eingesetzt werden, während dedizierte Verfahren, wie der DSA (Digital Signature Algorithm) ausschließlich zum Signieren verwendbar sind.

Protokoll

1. Sei (S_A, V_A) das Schlüsselpaar von Alice, bestehend aus dem privaten Signaturschlüssel S_A und dem öffentlichen Verifikationsschlüssel V_A .
2. Alice hinterlegt V_A in einer öffentlichen Datenbank.
3. Sie signiert ein Dokument M durch Verschlüsseln mit ihrem privaten Schlüssel, $D(M, S_A) = sig$, und sendet das signierte Dokument sig an Bob.
4. Bob ruft den benötigten Verifikationsschlüssel V_A aus der Datenbank ab
5. und verifiziert die Signatur sig , $M = E(sig, V_A)$.

Der Ablauf eines Protokolls mit asymmetrischem Verfahren ist in Abbildung 8.5 zusammengefasst.

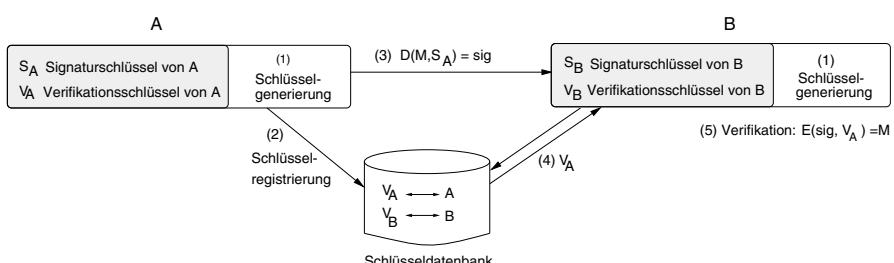


Abbildung 8.5: Signaturen mit asymmetrischen Verfahren

Das Protokoll benötigt keinen Vermittler, sondern jeder Beteiligte kann durch die Nutzung der öffentlich bekannten Verifikationsschlüssel die Gültigkeit der Signatur nachprüfen. Es bleibt zu klären, ob das Protokoll die gestellten Anforderungen an digitale Signaturen erfüllt.

Zweifelsfreie Identität: Unter der Voraussetzung, dass der öffentliche Verifikationsschlüssel eindeutig einer natürlichen oder juristischen Person zuordenbar ist, bezeugt die Signatur die Identität des Unterzeichners. Durch die Verifikation wird Alice als Urheberin bestätigt, da nur sie den zum Verifikationsschlüssel passenden Signaturschlüssel kennt.

Erfüllung der Anforderungen

Keine Wiederverwendbarkeit: Die Signatur kann nicht unautorisiert wieder verwendet werden, da sie das Ergebnis einer Verschlüsselungsoperation, also ein Funktionswert abhängig vom jeweiligen Dokument ist.

Unveränderbarkeit: Das signierte Dokument ist nicht mehr veränderbar bzw. eine Änderung ist erkennbar, da dadurch keine korrekte Verifikation mehr möglich ist.

Verbindlichkeit: Unter der Voraussetzung, dass der private Signaturschlüssel nicht kompromittiert ist, kann Alice die Signatur nicht zurückweisen, da ausschließlich sie selbst über ihren privaten Signaturschlüssel verfügt.

Das auf asymmetrischen Verfahren basierende Protokoll geht von hohen Voraussetzungen aus, um die geforderten Eigenschaften gewährleisten zu können. Zur Überprüfung der zweifelsfreien Identität sind zusätzliche Maßnahmen erforderlich, die die Authentizität des öffentlichen Verifikationschlüssels des Unterzeichners bestätigen. Der Empfänger eines signierten Dokuments muss sicher sein können, dass der von ihm zur Verifikation der Unterschrift eingesetzte öffentliche Schlüssel zum Signaturschlüssel des Absenders gehört. Eine solche Sicherheit bzw. auch ein Vertrauen (engl. *trust*) in die Authentizität des öffentlichen Schlüssels besteht sicherlich nicht, wenn man sich den Schlüssel einfach vom Absender zusenden lässt oder ihn von einem allgemein zugänglichen Schlüsselserver bezieht. In diesen Fällen muss der Empfänger des öffentlichen Schlüssels den Schlüsselanbieter vertrauen. Dies führt zum Beispiel zu einem Vertrauensnetz (Web of Trust), wie es im Kontext des E-Mailverschlüsselungsdienstes PGP (vgl. Abschnitt 14.6) bekannt ist.

Voraussetzungen

Ein solches Vertrauensmodell, das im Wesentlichen auf sozialen Vertrauensstrukturen basiert (z.B. ich vertraue meinem Freund Bob, der vertraut Alice, also vertraue ich Alice auch, ggf. mit Vorbehalten), ist im geschäftlichen, unternehmerischen oder auch behördlichen Umfeld meist nicht ausreichend, um das erforderliche Vertrauen in die Echtheit und Glaubwürdigkeit in die Gültigkeit öffentlicher Schlüssel (und den dazu gehörenden privaten Schlüs-

Authentizität

Trust

PKI

sel) zu etablieren. In solchen Szenarien ist es unerlässlich, die benötigten öffentlichen Schlüssel von einer vertrauenswürdigen Quelle zu beziehen oder sich die Authentizität des Schlüssels durch eine vertrauenswürdige Stelle bescheinigen zu lassen. Dies erfordert den Aufbau und sicheren Betrieb einer Public-Key Infrastruktur (PKI) (vgl. Abschnitt 9.1).

Verbindlichkeit

Um die Verbindlichkeit zu garantieren, ist sicherzustellen, dass ein Kommunikationsteilnehmer seinen privaten Signaturschlüssel nicht absichtlich oder unabsichtlich offen legt. Dies flächendeckend sicherzustellen ist aber in der Praxis nicht möglich, so dass zusätzliche Maßnahmen zur Begrenzung der damit einhergehenden Gefährdungen oder zur Erschwerung der Schlüsselweitergabe erforderlich sind. Im einfachsten Fall führen Trust Center schwarze Listen, um Benutzer, deren Schlüssel offen gelegt wurden, zu identifizieren und um die weitere Verwendung dieser Schlüssel zu unterbinden. Unter Verwendung von Sicherheitsmodulen wie Smartcards (vgl. Kapitel 11) kann der direkte Zugriff auf die Schlüssel und damit deren absichtliche oder unabsichtliche Preisgabe erheblich erschwert werden. Eine rechtliche Gleichstellung elektronischer und handschriftlicher Signaturen ist in Deutschland nach dem SigG nur für qualifizierte elektronische Signaturen gegeben (vgl. Abschnitt 8.2.4).

Zeitstempel

Durch die Verwendung von Zeitstempeln lassen sich Signaturen datieren, so dass solche Signaturen zurückgewiesen werden können, die nach der Bekanntgabe der Veröffentlichung des privaten Signaturschlüssels ausgestellt wurden. Zeitstempel können daneben auch verwendet werden, um eine Mehrfachvorlage eines signierten Dokumentes, zum Beispiel eines Schecks, zu erkennen. Erforderlich ist die Ausstellung eines vertrauenswürdigen Zeitstempels, der auf einem nicht manipulierbaren Zeitgeber basiert. Um dies zu erreichen, kann beispielsweise das DCF-77 Funkzeitsignal von einer geschützten Sicherheitsbox empfangen werden, welche der Signatur die genaue Zeit hinzufügt.

Signaturen und Verschlüsselung

Verschlüsselung

Signaturverfahren können mit Verschlüsselungsverfahren kombiniert werden, so dass sowohl die Vertraulichkeit der Daten als auch deren Zuordnbarkeit gewährleistet wird. Das Vorgehen ist das digitale Gegenstück zum Versenden eines unterschriebenen Briefes in einem Umschlag: Die Unterschrift bestätigt den Autor, der Umschlag gewährleistet die Privatsphäre. Beim Einsatz beider Techniken sollte jedoch darauf geachtet werden, dass die zu transferierenden Daten zuerst signiert und dann verschlüsselt werden. Wählt man die andere Reihenfolge, so bestehen dagegen rechtliche Bedenken, da ein Dokument zum Zeitpunkt der Unterzeichnung für den Unterzeichner klar lesbar sein muss.

Zu beachten ist weiterhin, dass es beim Einsatz asymmetrischer Kryptosysteme zu Sicherheitsproblemen kommen kann, wenn man das gleiche Schlüsselpaar sowohl zum Verschlüsseln als auch zum Signieren verwendet. Im Folgenden erklären wir einen Angriff, der eine solche doppelte Verwendung von Schlüsseln ausnutzt.

Sicherheitsproblem

Angriff auf Signaturverfahren

Das Schlüsselpaar eines Kommunikationspartners Alice sei durch (K_E^A, K_D^A) gegeben, wobei K_E^A der öffentliche Verifikations- und gleichzeitig Verschlüsselungsschlüssel und K_D^A der private Signatur- und gleichzeitig Entschlüsselungsschlüssel ist. Die dem Angriff zugrunde liegende Idee ist, dass der Angreifer den Empfänger Bob einer signierten und verschlüsselten Nachricht dazu bringt, den Verschlüsselungsumschlag von der Nachricht zu entfernen und dem Angreifer das signierte Dokument zu senden. Mit dem öffentlichen Schlüssel des Dokumenterstellers Alice ist der Angreifer dann im Besitz der vermeintlich vertraulich übermittelten Information. Der Ablauf des Angriffs ist wie folgt:

Angriff

1. Die Kommunikationspartnerin Alice (A) sendet eine signierte,

$$D(M, K_D^A) = sig,$$
 und verschlüsselte Nachricht M an Bob (B),

$$E(sig, K_E^B) = C.$$
2. Ein Angreifer X fängt die Nachricht C ab und sendet sie zu einem späteren Zeitpunkt seinerseits an Bob.
3. Bob entschlüsselt das von X vorgelegte C mit seinem privaten Schlüssel,

$$D(C, K_D^B) = D(M, K_D^A) = sig,$$
 und verifiziert sig mit dem Verifikationsschlüssel des Angreifers X, da er diesen ja für den rechtmäßigen Absender hält,

$$E(sig, K_E^X) = M'.$$
 Er erhält so eine i.d.R. unsinnige Nachricht M' .
4. Bob sendet nun M' signiert und verschlüsselt als Empfangsbestätigung an X zurück,

$$E(D(M', K_D^B), K_E^X) = C'.$$

Dies ist ein durchaus realistisches Szenario, wenn man davon ausgeht, dass Empfangsbestätigungen beispielsweise durch eine Mail-Software automatisch zurückgesendet werden, ohne dass die Nachricht M' analysiert wird.

5. X entschlüsselt C' mit seinem privaten Schlüssel,

$$D(C', K_D^X) = sig',$$

verifiziert die Signatur sig' mit dem öffentlichen Schlüssel von Bob,

$$E(sig', K_E^B) = M' = E(sig, K_E^X),$$

wendet erneut seinen eigenen privaten Schlüssel an,

$$D(M', K_D^X) = sig,$$

und verifiziert schließlich sig mit dem Verifikationsschlüssel von Alice,

$$E(sig, K_E^A) = M.$$

X erhält dadurch die ursprüngliche Nachricht M.

Solche Angriffe können auf einfache Weise wirkungslos gemacht werden, wenn man anstelle der Originaldokumente nur deren Hashwerte signiert. Kenntnisse des Hashwertes liefern dem Angreifer aufgrund der Einweg-Eigenschaft der Hashfunktion keine Auskunft über das Originaldokument.

Auf Angriffsmöglichkeiten bei der Verwendung des RSA-Verfahrens zur Signaturerstellung wurde bereits in Kapitel 7.6.2 (vgl. Seite 343 ff) eingegangen, so dass wir an dieser Stelle nur noch einmal darauf hinweisen.

Es existiert eine Vielzahl von Varianten, um elektronische Signaturen zu erstellen. Eine ausführliche Behandlung dieser Thematik findet sich unter anderem in [142]. Abschließend wird als Beispiel für ein dediziertes Signaturverfahren der Digital Signature Standard erklärt.

8.2.3 Digitaler Signaturstandard (DSS)

DSS

Der Digital Signature Standard (DSS) [133] wurde vom National Institute of Standards der USA 1994 als Signaturstandard festgelegt. Herzstück des Standards ist der Digital Signature Algorithm (DSA), der von der NSA entwickelt wurde und auf dem Problem des diskreten Logarithmus basiert.

Kritik

Nachdem das Verfahren 1991 erstmalig einer breiten Öffentlichkeit vorgestellt worden war, wurden viele kritische Stimmen laut, die meist aber weniger die Funktionalität und Sicherheit des Algorithmus in Zweifel zogen, sondern eher politische und kommerzielle Gründe gegen den Vorschlag ins Feld führten. Hauptkritikpunkt war dabei, dass sich das RSA-Verfahren in der Industrie als Quasi-Standard bereits etabliert habe und somit keine Notwendigkeit für einen neuen Standard mit einem neuen Algorithmus gesehen wurde. Substantielle Kritik betraf die im Vorschlag von 1991 verwendete Schlüsselgröße von 512 Bit. Da in den letzten Jahren erhebliche Leistungssteigerungen bei der Berechnung des diskreten Logarithmus erzielt werden konnten, forderten Kritiker einen wesentlich größeren Modul; dies wurde bei der Verabschiedung des endgültigen Standards dann auch berücksichtigt.

Der Digital Signature Algorithm (DSA)

Der DSA basiert auf dem Problem des diskreten Logarithmus (vgl. Beispiel 7.9) und ist eine Erweiterung der Schnorr und ElGamal Signaturverfahren [59]. Im Folgenden beschreiben wir das DSA Verfahren mit einem Sicherheitsniveau, das vergleichbar einer Schlüssellänge von 512 Bit bis zu 1024 Bit ist; der Standard unterstützt aber auch ein höheres Sicherheitsniveau. Der Vorteil von DSA gegenüber RSA ist, dass die Signaturen kürzer sind, als bei RSA-Signaturen mit vergleichbarem Sicherheitsniveau. So sind DSA Signaturen bei einem Sicherheitsniveau von 1024 Bit lediglich 320 Bit lang. DSA ist somit für ressourcenbeschränkte Anwendungsgebiete, wie beispielsweise Smartcards oder RFID Chips, gut geeignet. So wird DSA unter Nutzung der Elliptic Curve Cryptography als DSA-ECC auch im neuen Personalausweis (nPA) eingesetzt. Nachteilig ist jedoch, dass die Signatur-Verifikation bei DSA einen höheren Berechnungsaufwand als RSA benötigt. Der DSA basiert auf der Nutzung von zwei Gruppen, die auf Primzahlen p und q basieren.

Im Folgenden wird beschrieben, wie die erforderlichen Signier- und Verifikationsschlüssel bei DSA erzeugt und wie diese zur Erstellung und Validierung von digitalen Signaturen eingesetzt werden. Abschließend skizzieren wir einen Beweis für die Korrektheit des Signaturschemas und gehen auf die Sicherheit des Verfahrens ein.

Schlüsselgenerierung

- Generiere eine Primzahl p der Länge L Bits, wobei $512 \leq L \leq 1024$ Gruppen und L ein Vielfaches von 64 ist, also

$$2^{511+64t} < p < 2^{512+64t} \text{ mit } t \in \{0, \dots, 8\}.$$
- Berechne einen Primfaktor q von $p - 1$, mit $2^{159} < q < 2^{160}$, d.h. q ist 160 Bit lang.
- Bestimme einen Wert g , mit $g = j^{(p-1)/q} \bmod p$, wobei gilt

$$0 < j < p - 1 \text{ und } j^{(p-1)/q} \bmod p > 1.$$
- Wähle einen Wert x , mit $0 < x < q$.
 x ist der geheime Schlüssel, also der Signierschlüssel. Schlüssel
- Berechne y , mit $y = g^x \bmod p$.
 y ist der zu x gehörende öffentliche Schlüssel, also der Verifikations-schlüssel.
- Wähle eine sichere Hashfunktion H.

Die Werte p, q und g sind öffentlich bekannt und können von einer Gruppe von Benutzern gemeinsam verwendet werden.

Durch die Nutzung von zwei Gruppen, basierend einerseits auf einer großen Primzahl p und andererseits auf einer deutlich kleineren Primzahl q ist es möglich, den hohen Berechnungsaufwand, der für Operationen in der großen Gruppe erforderlich ist, auf wenige Operationen zu reduzieren.

Signatur-Erstellung

Wir nehmen an, dass die Kommunikationspartnerin Alice das Dokument M signieren möchte und x_A ihr privater und y_A ihr öffentlicher Schlüssel ist.

Signieren

1. Alice generiert eine Zufallszahl $0 < k < q$. k wird auch als flüchtiger Schlüssel, ephemeral Key, bezeichnet.
2. Alice berechnet:

$$\begin{aligned} r &= (g^k \bmod p) \bmod q \text{ und} \\ s &= (k^{-1}(H(M) + x_A r)) \bmod q. \end{aligned}$$

k^{-1} ist das multiplikative Inverse von k modulo q .

Die Parameter r und s bilden die Signatur (r, s) , die Alice dem Empfänger (Bob) zusendet.

Signatur-Verifikation

Verifikation

Gegeben sei der Klartext M die Signatur (r, s) und der Verifikationsschlüssel y_A von Alice. Bob verifiziert die Signatur (r, s) , indem er zunächst deren generelle Gültigkeit überprüft. Falls eine der beiden Bedingungen

$$1 \leq r \leq q - 1 \text{ und } 1 \leq s \leq q - 1$$

verletzt ist, verwirft er die Signatur sofort als ungültig. Andernfalls führt er folgende Berechnungsschritte durch:

1. Berechne Hilfswert w , mit $w = s^{-1} \bmod q$.
 s^{-1} ist wiederum das multiplikative Inverse von s modulo q .
2. Berechne Hilfswert u_1 , mit $u_1 = (H(M)w) \bmod q$.
3. Berechne Hilfswert u_2 , mit $u_2 = (rw) \bmod q$.
4. Berechne v , mit $v = ((g^{u_1} \cdot y_A^{u_2}) \bmod p) \bmod q$.

Falls $r = v \bmod q$, dann ist die Signatur verifiziert.

Korrektheit des Verfahrens

Im Folgenden skizzieren wir einen Beweis für die Korrektheit des DSA-Signaturverfahrens. Zu zeigen ist, dass bei einer korrekt konstruierten

Signatur (r, s) mit einem korrekt berechneten Wert v gilt: $r = v \bmod q$.

Nach Konstruktion von s gilt:

$$s = (k^{-1}(H(M) + x_A \cdot r)) \bmod q.$$

Dies kann man äquivalent umformen zu:

$$\begin{aligned} k &= (H(M) + x_A \cdot r) \cdot s^{-1} \bmod q \\ &= H(M) \cdot s^{-1} + (x_A \cdot r) \cdot s^{-1} \bmod q \\ &= u_1 + (x_A \cdot u_2) \bmod q \end{aligned}$$

Eine äquivalente Umformung ergibt:

$$g^k \bmod p = (g^{u_1} \cdot g^{x_A \cdot u_2} \bmod p) \bmod q.$$

Da weiterhin per Konstruktion gilt:

$$y_A = g^{x_A} \bmod p, \text{ gilt}$$

$$g^k \bmod p = (g^{u_1} \cdot y_A^{u_2} \bmod p) \bmod q.$$

Eine beidseitige Reduktion $\bmod q$ ergibt:

$$\begin{aligned} g^k \bmod p \bmod q &= ((g^{u_1} \cdot y_A^{u_2} \bmod p) \bmod q) \bmod q \\ r &= v \bmod q \end{aligned}$$

Damit ist gezeigt, dass bei korrekt konstruierten Werten r, s, v die Signatur (r, s) mittels v verifiziert werden kann.

•

Sicherheit des DSA-Verfahrens

Ein Angreifer, der die Signatur von Alice fälschen möchte, benötigt dazu gemäß des beschriebenen Verfahrens den privaten Schlüssel x_A von Alice. Da ihm aber nur y_A, p, q und g bekannt sind, muss er x_A berechnen, also $x_A = \log_g y_A \bmod p$. Das heißt, er muss den diskreten Logarithmus von y_A zur Basis g berechnen. Die besten bekannten Verfahren benötigen dazu aber mehr als \sqrt{q} Operationen. Da $q > 2^{159}$ ist, sind dies mehr als 2^{79} Operationen, was einen Aufwand bedeutet, der beim heutigen Stand der Technik nicht zu leisten ist.

diskreter
Logarithmus

Ein möglicher Angriffspunkt, um x_A einfacher zu bestimmen, ist die Zufallszahl k , die bei jeder Signaturerstellung neu zu generieren ist. Falls es einem Angreifer gelingt, k zu erraten, so kann mit dieser Kenntnis x_A berechnet werden. Für die Sicherheit des DSA ist somit die Verwendung eines guten Zufallszahlengenerators, der keine Vorhersagen über die generierten Werte ermöglicht, unabdingbar.

Zufallszahl k

Nach heutigem Stand der Technik sollte die Primzahl p mindestens 1024 Bit groß sein. Die NIST hat folgende Empfehlungen für die Längen der Primzahlen p und q heraus gegeben:

p	q	Hashwert	Sicherheitslevel
1024	160	160	80
2048	224	224	112
3072	256	256	128

Im Folgenden werden die rechtlichen Rahmenbedingungen für den Umgang mit digitalen Signaturen, die in Deutschland mit dem Signaturgesetz festgelegt sind, erklärt.

8.2.4 Rechtliche Rahmen

In vielen Anwendungsbereichen ist es gar nicht erforderlich, von der elektronischen Signatur die Beweiskraft einer handschriftlichen zu fordern. Man denke hier beispielsweise an den E-Mailaustausch zwischen Geschäftspartnern. Hierbei ist wichtig, dass die Partner auf glaubwürdige und nachprüfbare Weise ihre jeweilige Identität nachweisen, ohne dass jede signierte E-Mail aber gleich rechtsverbindlich sein muss. In Deutschland und auch anderen europäischen und außer-europäischen Ländern wurden gesetzliche Rahmenbedingungen festgelegt, um die Anforderungen, die an derart unterschiedliche Arten von Signaturen gestellt werden, präzise zu fassen.

In Deutschland war jahrelang Vorreiter in der Datenschutzgesetzgebung. Bereits am 13. Juni 1997 wurde das erste Gesetz zur digitalen Signatur (Signaturgesetz – SigG) als Artikel 3 des Gesetzes zur Regelung der Rahmenbedingungen für Informations- und Kommunikationsdienste (IuKDG) vom deutschen Bundestag beschlossen und ist am 1. August 1997 in Kraft getreten. Der Zweck des Gesetzes war es, Rahmenbedingungen für digitale Signaturen zu schaffen, unter denen diese als sicher gelten und Fälschungen digitaler Signaturen oder Verfälschungen von signierten Daten zuverlässig festgestellt werden können.

EU-Richtlinien

Das hohe Sicherheitsniveau der deutschen Richtlinien von 1997 für die Gültigkeit elektronischer Signaturen konnte sich europaweit nicht durchsetzen. Die im April 1999 verabschiedeten EU-Richtlinien sahen weit weniger restriktive Regelungen vor. Das Ziel der Richtlinien war es, eine grenzüberschreitende Anerkennung von digitalen Signaturen¹⁴ und Zertifikaten sicherzustellen und einen harmonisierten rechtlichen Rahmen für die Europäische Gemeinschaft zu schaffen. Dazu wurden Kriterien festgelegt, die als Grundlage für die rechtliche Anerkennung digitaler Signaturen dienen.

Die Richtlinien enthalten lediglich allgemeine Anforderungen an Zertifizierungsdienste. Sie besagen, dass solche Dienste ohne vorherige Genehmigung

erstes Gesetz

EU-Richtlinien

Ziel

schwaches Sicherheitsniveau

¹⁴ Die Richtlinien sprechen durchgehend von elektronischen Signaturen.

angeboten werden können, dass also insbesondere auch keine Zertifizierung von Dienstleistern, wie sie im deutschen Gesetz vorgeschrieben ist, erforderlich ist. Den Mitgliedsstaaten war es freigestellt, durch Akkreditierungsregelungen auf freiwilliger Basis ein höheres Sicherheitsniveau anzustreben. Die abgeschwächten Anforderungen dieser Richtlinien erleichterten somit die Einrichtung von Zertifizierungsstellen (Trust Centern) (vgl. Seite 393) erheblich, da kein Antrag bei der Regulierungsbehörde (Reg TP) mehr gestellt werden muss. Eine Überprüfung der Dienste war lediglich in Form von unangemeldeten Kontrollen vorgesehen.

Die europäischen Signatur-Richtlinien zielten in erster Linie auf ein Haftungskonzept, um die Vertrauensbildung zwischen Verbrauchern und Unternehmen zu fördern. So war eine Haftungsregelung vorgeschlagen, die vorsieht, dass der Dienstanbieter für die inhaltliche Richtigkeit des Zertifikats sowie für die korrekte Funktionalität seiner Signaturverfahren haftet. Voraussetzung dafür, dass die Rechtsfolgen beim Einsatz digitaler Unterschriften geklärt sind, ist, dass digitale Signaturen mit herkömmlichen Unterschriften rechtlich gleichgestellt sind. Hierfür war es notwendig, existierender Paragraphen und Verordnungen anzupassen.

Haftungskonzept

Signaturgesetz bzw. Vertrauensdienstegesetz

Mit dem Gesetz über die Rahmenbedingungen für elektronische Signaturen und zur Änderung weiterer Vorschriften [37] hat der deutsche Gesetzgeber den vereinfachten Rahmenbedingungen der EU Rechnung getragen. Das Gesetz trat am 22.5.2001 in Kraft und löste das Signaturgesetz von 1997 ab. Das Gesetz soll Rahmenbedingungen schaffen, durch deren Einhaltung qualifizierte elektronische Signaturen als gleichwertig zu handschriftlichen Signaturen vor Gericht anerkannt werden. Mit dem Formanpassungsgesetz traten 2001 Regelungen in Kraft, die Festlegungen darüber treffen, wann eine elektronische Signatur der handschriftlichen gleichzustellen ist.

Das Signaturgesetz wurde mit Wirkung vom 29. Juli 2017 durch das Vertrauensdienstegesetz (VDG) abgelöst, welches die Verordnung (EU) Nr. 910/2014 (eIDAS-Verordnung) ergänzt. eIDAS (electronic IDentification, Authentication and trust Services) ist die Verordnung des Europäischen Parlaments und des Rates über elektronische Identifizierung und Vertrauensdienste für elektronische Transaktionen im EU-Binnenmarkt. Der Durchführungsbeschluss (EU) 2015/1506 legt Formate fortgeschrittener elektronischer Signaturen und der Durchführungsbeschluss (EU) 2016/650 legt Normen für die Sicherheitsbewertung qualifizierter Signaturerstellungseinheiten fest.

eIDAS

Das Vertrauensdienstegesetz unterscheidet drei Arten elektronischer Signaturen. Aufsteigend nach den Sicherheitsanforderungen, die für diese gestellt sind, handelt es sich dabei um

- die einfache elektronische Signatur,
- die fortgeschrittene elektronische Signatur
- und die qualifizierte elektronische Signatur.

einfache Signatur

Nach §2 Nr. 1 SigG ist eine einfache elektronische Signatur ein Datum in elektronischer Form, das anderen Daten beigefügt ist und zur Authentifizierung dient. Aufgrund der geringen Anforderungen an diese Klasse von Signaturen, muss ein Signaturanbieter auch nicht (wie bei der fortgeschrittenen Signatur) für die Richtigkeit und Vollständigkeit der Zertifikatangaben haften. Bei der Verwendung einfacher Signaturen muss den Nutzern klar sein, dass ein potentiell Geschädigter den entstandenen Schaden selber nachweisen muss. Dieser Nachweis ist in der Praxis sicherlich nicht einfach zu führen.

fortgeschrittene Signaturen

Fortgeschrittene Signaturen sind nach §2 Nr. 1 SigG einfache Signaturen, die

- ausschließlich dem Signaturschlüssel-Inhaber zugeordnet sind und
- seine Identifizierung ermöglichen,
- mit Mitteln erzeugt werden, die der Signaturschlüssel-Inhaber unter seiner alleinigen Kontrolle halten kann, und
- mit den Daten, auf die sie sich beziehen, so verknüpft sind, dass eine nachträgliche Veränderung der Daten erkannt werden kann.

Die einfache sowie die fortgeschrittene Signatur ist vollständig unreguliert. Als Beweismittel vor Gericht und als Ersatz für die eigenhändige Unterschrift wird nur die qualifizierte Signatur zugelassen, so dass der Gesetzgeber den Umgang damit reguliert.

qualifizierte Signatur

Im Sinne dieses Gesetzes ist eine qualifizierte elektronische Signatur eine fortgeschrittene Signatur, die auf einem zum Zeitpunkt ihrer Erzeugung gültigen qualifizierten Zertifikat beruht und mit einer sicheren Signaturerstellungseinheit erzeugt wurde. Die qualifizierte elektronische Signatur ist somit ein mit einem privaten Signaturschlüssel erzeugtes Siegel, das digitalen Daten assoziiert ist und das über den zugehörigen öffentlichen Schlüssel, der mit einem Zertifikat einer Zertifizierungsstelle versehen ist, den Inhaber des Signaturschlüssels und die Unverfälschtheit der Daten erkennen lässt. Ein Zertifikat ist eine mit einer qualifizierten elektronischen Signatur versehene, digitale Bescheinigung über die Zuordnung eines öffentlichen Signaturschlüssels zu einer natürlichen Person. Das Zertifikat muss den Namen des Signaturschlüsselhabers oder ein dem Signaturschlüsselhaber zugeordnetes, unverwechselbares Pseudonym, den zugeordneten öffentlichen Signaturschlüssel, die Algorithmen, mit denen der öffentliche Schlüssel des

Zertifikat

Schlüssel-Inhabers sowie der öffentliche Schlüssel der Zertifizierungsstelle verwendet werden kann, den Gültigkeitszeitraum des Zertifikates, den Namen der Zertifizierungsstelle und Angaben, ob die Nutzung des Signaturschlüssels auf bestimmte Anwendungen nach Art und Umfang beschränkt ist, enthalten.

Eine Zertifizierungsstelle ist im Sinne dieses Gesetzes eine natürliche oder juristische Person, die die Zuordnung von öffentlichen Signaturschlüsseln zu natürlichen Personen bescheinigt, wozu sie ihrerseits ein Signaturverfahren basierend auf asymmetrischen Verfahren und den ihr assoziierten privaten Schlüssel verwendet. Generell ist der Betrieb einer Zertifizierungsstelle im Rahmen des Gesetzes genehmigungsfrei. Die Aufnahme eines solchen Dienstes ist jedoch der zuständigen Behörde im Geschäftsbereich des Bundesministeriums für Wirtschaft, nämlich der Bundesnetzagentur¹⁵ anzugeben. Mit dieser Anzeige muss der Anbieter ein Sicherheitskonzept vorlegen, mit dem er darlegt, wie die gesetzlichen Voraussetzungen erfüllt und wie sie praktisch umgesetzt werden. Zertifizierungsdienste-Anbieter können sich auf eigenen Antrag bei der Bundesnetzagentur akkreditieren lassen. Ein akkreditierter Anbieter darf für seine Zertifizierungstätigkeit nur geprüfte Produkte für qualifizierte elektronische Signaturen einsetzen. Ferner darf er Zertifikate nur für Personen ausstellen, die nachweislich geprüfte und bestätigte sichere Einheiten zur Signaturerstellung besitzen. Mit dem Gütezeichen der Bundesnetzagentur wird bestätigt, dass der akkreditierte Anbieter den Nachweis erbracht hat, dass seine technischen und administrativen Sicherheitsmaßnahmen den gesetzlichen Anforderungen entsprechen. Der Anbieter ist berechtigt, qualifizierte elektronische Signaturen mit Anbieter-Akkreditierung zu erstellen.

Das Gesetz schreibt vor, dass neben den Teilnehmerdaten das Zertifikat des öffentlichen Schlüssels sowie der öffentliche Schlüssel der Zertifizierungsstelle auf einem geeigneten Trägermedium für den Teilnehmer (z.B. Chipkarte) gespeichert werden sollten. Dies wird als Personalisierung bezeichnet.

In manchen Fällen ist es notwendig, digitale Daten authentisch mit einem bestimmten Zeitpunkt zu verknüpfen. Solche Daten (oder ihr Hashwert) werden dazu mit der vom Zeitstempeldienst der Zertifizierungsstelle anzubietenden, vertrauenswürdigen Zeit digital verknüpft und das Ergebnis wird anschließend von der Zertifizierungsstelle digital signiert. Stattdessen kann auch der mit dem Zeitstempel versehene Hashwert zusammen mit einem Verweis auf die entsprechenden Daten in einem öffentlich zugänglichen Verzeichnis bekannt gegeben werden.

Trust Center

Personalisierung

Zeitstempel

¹⁵ früher Regulierungsbehörde für Telekommunikation und Post (RegTP)

Technische Komponenten

Das Gesetz dient vor allem dazu, den Entwicklern von Signaturverfahren die entsprechenden rechtlichen Rahmenbedingungen aufzuzeigen und insbesondere die Zulassung und den Betrieb von Zertifizierungsstellen (Trust Center) sowie die Anforderungen an die erforderlichen technischen Komponenten festzulegen. Detaillierte Vorgaben zu den Anforderungen an die technischen Komponenten finden sich in der Signaturverordnung und vor allem im Maßnahmenkatalog des Bundesamts für Sicherheit in der Informationstechnik.

Anforderungen

Zur Erstellung qualifizierter elektronischer Signaturen werden spezielle technische Komponenten und Produkte benötigt, die je nach Zweck, Einsatz und Anwendung entweder durch eine Bestätigungsstelle als signaturgesetzkonform eingestuft wurden oder für die eine Herstellererklärung vorliegt. Die Prüfung der technischen Komponenten zur Erstellung qualifizierter elektronischer Signaturen hat nach den Kriterien für die Bewertung der Sicherheit von Systemen der Informationstechnik (Common Criteria vgl. Abschnitte 5.3 und 5.4) zu erfolgen. Die Prüfung muss bei Komponenten zur Erzeugung von Signaturschlüsseln oder zur Speicherung oder Anwendung privater Signaturschlüssel und bei Komponenten, die geschäftsmäßig Dritten zur Nutzung angeboten werden, mindestens die EAL4 nach Common Criteria umfassen¹⁶. Die Stärke der Sicherheitsmechanismen muss mit hoch und die Algorithmen und zugehörigen Parameter müssen als geeignet bewertet sein. Diese Eigenschaften müssen durch eine akkreditierte Prüfstelle geprüft und durch eine Bestätigungsstelle bestätigt werden.

Bei den Komponenten, für die eine hohe Sicherheitsstufe gefordert wird, handelt es sich um Spezialkomponenten, so dass die aufwändige Evaluierung (z.B. mit Erstellung eines formalen Sicherheitsmodells) mit vertretbarem Aufwand durchgeführt werden könnte. Die für die übrigen Komponenten geforderte Standardsicherheitsstufe EAL3 (z.B. mit einer Überprüfung der Implementierung der Mechanismen, einer Schwachstellenanalyse und Tests zur Fehlersuche) sollte mit einem vertretbaren Aufwand realisierbar sein und erscheint für deren Funktionalität angemessen. Dies gilt auch für die Komponenten zur Prüfung einer qualifizierten elektronischen Signatur, da dafür nur der öffentliche Schlüssel eingesetzt wird.

Signaturkonforme Verfahren

Das Bundesamt für die Sicherheit in der Informationstechnik (BSI) erstellt jährlich eine Liste, die die Kryptoverfahren enthält, die beim aktuellen Stand der Technik als signaturgesetzkonform eingestuft sind. Zurzeit zählen dazu neben den bekannten Verfahren wie RSA und DSS auch Verfahren, die auf elliptischen Kurven basieren. Maßgeblich für die Sicherheit der Verfahren ist, wie bereits an verschiedenen Stelle betont, eine den technischen Ge-

¹⁶ Zur Erinnerung: CC umfasst die Stufen EAL1-EAL7.

benheiten angemessene Schlüssellänge, aber auch die Verwendung sicherer Hashfunktionen. Die Bundesnetzagentur gibt jährlich die gültigen Verfahren und Schlüssellängen bekannt (siehe Seite 291).

Langzeitarchivierung

Beim alltäglichen Einsatz elektronischer Signaturen in Geschäftsprozessen ergeben sich auch neue Probleme, die bislang noch nicht zufriedenstellend gelöst sind. Dazu zählt das Präsentationsproblem, aber auch die Problematik der Langzeitarchivierung von Dokumenten. Beim Präsentationsproblem handelt es sich um die Frage, wie ein digital signiertes Dokument so in unterschiedliche Darstellungsformate transformiert werden kann, dass die Gültigkeit der elektronischen Signatur erhalten bleibt. Man denke hier beispielsweise an die Transformation von signierten Word-Dokumenten in pdf-Formate und umgekehrt. Vorhandene Standard-Konverter berücksichtigen die Problematik der Aufrechterhaltung der Gültigkeit elektronischer Signaturen in keiner Weise.

Präsentations-
problem

Noch deutlich weit reichender ist die Problematik, die mit der langfristigen sicheren Archivierung von digital signierten Dokumenten verbunden ist. So sehen beispielsweise Auflagen im medizinischen Umfeld vor, dass Arztbriefe und Patientenakten über 30 Jahre sicher zu verwahren sind, während für andere Bereiche wie z.B. dem Grundbuchamt, noch deutlich längere Aufbewahrungsfristen (100 Jahre und mehr) gelten. Im Laufe der Zeit ändern sich Signatur- und Datenformate und die jeweils verwendeten Hard- und Software-Komponenten sind einer sehr hohen Dynamik in Bezug auf den Alterungsprozess ausgesetzt. Dies alles wird dazu führen, dass alte Formate und Programme nach einer gewissen Zeit nicht mehr verfügbar sind, so dass eine Signaturüberprüfung nicht mehr möglich sein könnte. Konvertierungen signierter Dokumente würden auch hierbei dazu führen, dass die Signaturen ungültig werden. Neue Verfahren zur rechtssicheren Transformation elektronisch signierter Dokumente werden benötigt, um die Rechtssicherheit derartiger Dokumente auch über lange Zeiträume hinweg bzw. unter verschiedenen Transformationen nachhaltig zu gewährleisten.

Langzeit-
archivierung

9 Schlüsselmanagement

Die Hauptaufgaben des Schlüsselmanagements bestehen in der sicheren Erzeugung, Verteilung, Zertifizierung, Speicherung bzw. Archivierung sowie Vernichtung von Schlüsseln. Wir beschäftigen uns zunächst in Abschnitt 9.1 mit den Eigenschaften von Zertifikaten, klären die Anforderungen, die auch aus rechtlicher Sicht an eine Zertifizierungsstelle zu stellen sind, und erklären die Komponenten einer Public-Key Infrastruktur (PKI). Abschnitt 9.2 konzentriert sich dann auf allgemeine Aspekte im Zusammenhang mit der Schlüsselverwaltung wie der Schlüsselerzeugung und -speicherung. Protokolle zum sicheren Austausch von Schlüsseln zwischen entfernten Kommunikationspartnern sind wichtige Managementdienste in vernetzten Systemen. Abschnitt 9.3 führt grundlegende derartige Protokolle ein. Als Beispiel eines Schüsselaustauschprotokolls wird das Diffie-Hellman-Verfahren vorgestellt, das von großer praktischer Bedeutung ist. Abschnitt 9.4 beschäftigt sich schließlich mit Techniken und Problemen der Schlüssel-Rückgewinnung.

9.1 Zertifizierung

Mit einem Zertifikat wird eine digitale Bescheinigung über die Zuordnung eines öffentlichen Signierschlüssels zu einer natürlichen oder juristischen Person ausgestellt. Es ist somit zu klären, welche Struktur ein solches Zertifikat besitzt, wie es ausgestellt wird und insbesondere, auf welche Weise Zertifikate geprüft – man nennt das auch verifiziert – werden können. Zu beachten ist, dass ein Zertifikat selbstverständlich keine Aussage über den Inhalt signierter Dokumente oder über die Vertrauenswürdigkeit der signierenden Person machen kann. Die Vorlage eines Zertifikates trägt somit zwar zur Vertrauensbildung (trust) bei¹, ersetzt aber keinesfalls weitere Kontrollen, die besonders dann notwendig sind, wenn es sich um signierten ausführbaren Code handelt.

digitale
Bescheinigung

¹ Man beachte, dass eine Person oder allgemein ein Objekt, dem man vertraut (trusted), keineswegs auch vertrauenswürdig (trustworthy) sein muss.

9.1.1 Zertifikate

Die Struktur und der Aufbau heutiger, in der Praxis eingesetzter Zertifikate wird durch den X.509 Standard als Bestandteil des X.500 Authentifikations-Frameworks [83] festgelegt. Das heißt, dass ein Zertifikat den Namen des Signierschlüsselhabers, den zugeordneten öffentlichen Signierschlüssel, die verwendeten Algorithmen, den Gültigkeitszeitraum des Zertifikates und den Namen der Zertifizierungsstelle enthalten muss.

X.509

Ein X.509 Zertifikat besitzt die in Tabelle 9.1 festgehaltene allgemeine Struktur und ist mit dem privaten Signierschlüssel der ausstellenden Instanz signiert.

Inhalt	Erläuterung
Versionsnummer	beschreibt verwendetes Zertifikatformat
Seriennummer	eindeutiger Identifikator
Signatur	verwendete Algorithmen und Parameter
Zertifikataussteller	Name der ausstellenden Instanz
Gültigkeitsdauer	Angabe eines Zeitintervalls
Benutzername	eindeutiger Name des Benutzers
Schlüsselinformationen	Schlüssel des Benutzers und Algorithmen
eindeutiger Identifikator	in Version v2, v3
Erweiterungen	in Version v2, v3

Tabelle 9.1: Struktur eines X.509 bzw. X.509v3 Zertifikats

X.509v3

Über die Versionsnummer können Änderungen der Formatfestlegungen nachgehalten werden. Die erste Version von 1988 trägt die Versionsnummer v1 und wird mit dem Versionswert 0 identifiziert. Seit 1996 liegt die Version v3 vor (X.509v3), die Erweiterungen der ursprünglichen Struktur (vgl. Tabelle 9.1) beinhaltet. Die Angabe des eindeutigen Namens des Benutzers oder des Ausstellers im Zertifikat ermöglicht es, diesen Namen mehrfach zu verwenden. Das Erweiterungsfeld definiert eine Folge von einer oder mehreren Erweiterungen. Beispiele für solche Erweiterungen sind *key usage*, *private key usage period* oder *certificate policies*. Die *key usage* Erweiterung definiert, für welchen Zweck (Verschlüsseln, Signieren, Zertifizieren) der Schlüssel, der im Zertifikat enthalten ist, eingesetzt werden darf. Die *private key usage period* Erweiterung ermöglicht es, unterschiedliche Gültigkeitsdauern für den privaten Signierschlüssel des Benutzers sowie für das

Zertifikat festzulegen. Über *certificate policy* Erweiterungen kann spezifiziert werden, unter welchen Bedingungen das Zertifikat ausgestellt wurde und zu welchem Zweck es zu verwenden ist. Eine ausführliche Darstellung aller Erweiterungen findet man im RFC2459.

Die ausstellende Instanz ist dafür verantwortlich, dass keine zwei Zertifikate mit gleichen Seriennummern ausgestellt werden, so dass bei Zertifikatrückrufen eine eindeutige Zuordnung möglich ist. Damit der Empfänger eines Zertifikats dieses verifizieren kann, benötigt er Informationen über die verwendeten Hash- und Signierverfahren, sowie über den öffentlichen Schlüssel der Zertifizierungsinstanz. Diese Informationen sind im Zertifikat enthalten und werden zusätzlich repliziert den signierten Daten des Zertifikats assoziiert. Der Name der ausstellenden Instanz identifiziert eindeutig denjenigen Dienst, der die Korrektheit der Bescheinigung bestätigt. Der Zertifikataussteller kann für jedes Zertifikat eine Gültigkeitsdauer festlegen, die durch einen Anfangs- und einen Endzeitpunkt definiert wird. Der Eintrag für den Benutzernamen spezifiziert den Benutzer², für den die Bescheinigung ausgestellt wird, und der Eintrag über Schlüsselinformationen enthält Informationen über dessen öffentlichen Schlüssel und die Signaturalgorithmen, die mit diesem Schlüssel zu verwenden sind. Diese können sich von den Algorithmen, die die ausstellende Instanz verwendet, unterscheiden. Tabelle 9.2 zeigt einen Ausschnitt der ASN.1-Spezifikation eines X.509v3 Zertifikats unter Verzicht auf die Angabe der benötigten Datentypen.

versionsunabhängige Felder

ASN.1

9.1.2 Zertifizierungsstelle

Eine Zertifizierungsstelle (CA) (engl. *Trust Center* oder *Certification Authority*) bietet Dienste zur Ausstellung von Zertifikaten an, wozu sie ihrerseits ein Signaturverfahren, in der Regel basierend auf asymmetrischen Verfahren, und den ihr assoziierten privaten Schlüssel verwendet.

Trust Center

Es sind Schlüsselpaare sowohl für den Trust Center als auch gegebenenfalls für Teilnehmer zu generieren. Das Schlüsselpaar der Zertifizierungsstelle wird für die Zertifizierung der öffentlichen Schlüssel der Teilnehmer benötigt. Die Schlüsselerzeugung hat in einer geeigneten und sicheren Umgebung innerhalb des Trust Centers stattzufinden. Darauf hinaus muss sichergestellt werden, dass kein unautorisierte Zugriff auf den privaten Schlüssel der Zertifizierungsstelle möglich ist.

Schlüsselgenerierung

Die Teilnehmer an einem Verfahren für digitale Signaturen müssen sich gegenüber der Zertifizierungsstelle ausweisen. Bei erfolgreicher Identifizierung wird dem Teilnehmer ein eindeutiger Name zugewiesen, unter dem dieser digitale Signaturen erzeugen kann, oder es kann ein Pseudonym zuge-

Identifizierung

² Für Server, insbesondere Web-Server können auch Server-Zertifikate ausgestellt werden.

```

Certificate ::= SEQUENCE {
    tbsCertificate      TBSCertificate,
    signatureAlgorithm AlgorithmIdentifier,
    signatureValue      BIT STRING  }

TBSCertificate ::= SEQUENCE {
    version           [0] EXPLICIT Version DEFAULT v1,
    serialNumber      CertificateSerialNumber,
    signature          AlgorithmIdentifier,
    issuer             Name,
    validity           Validity,
    subject            Name,
    subjectPublicKeyInfo SubjectPublicKeyInfo,
    issuerUniqueID     [1] IMPLICIT UniqueIdentifier
                           OPTIONAL,
                           -- nur Version v2 oder v3
    subjectUniqueID   [2] IMPLICIT UniqueIdentifier
                           OPTIONAL,
                           -- nur Version v2 oder v3
    extensions         [3] EXPLICIT Extensions OPTIONAL
                           -- nur Version v2 oder v3  }

```

Tabelle 9.2: ASN.1-Spezifikation eines X.509v3 Zertifikats

teilt werden, so dass die Identität des Teilnehmers gegenüber anderen nicht unmittelbar erkennbar ist.

Zertifizierung

Für jeden Teilnehmer ist von der Zertifizierungsstelle ein Zertifikat, z.B. ein X.509 bzw. X.509v3 Zertifikat, zu erzeugen und mit dem privaten Schlüssel der Zertifizierungsstelle zu signieren.

Personalisierung

Das Zertifikat und ggf. der öffentliche und private Schlüssel werden auf eine Signaturkomponente, z.B. eine Chipkarte, übertragen. Die Teilnehmerdaten, das Zertifikat des öffentlichen Schlüssels sowie der öffentliche Schlüssel der Zertifizierungsstelle sollten auf einem geeigneten Trägermedium für den Teilnehmer (z.B. Chipkarte) gespeichert werden. Auf demselben Medium wird in der Regel bereits der private Schlüssel des Teilnehmers sicher abgelegt sein.

Verzeichnis

Über einen Verzeichnisdienst (z.B. einen LDAP-Server) muss für jeden Teilnehmer Auskunft darüber gegeben werden, ob ein bestimmtes, ihm zugeordnetes Zertifikat noch gültig ist oder nicht. Ungültige Zertifikate sind durch einen Sperrvermerk zu kennzeichnen, der Auskunft über den Zeitpunkt des Eintretens der Ungültigkeit gibt. Diese Informationen sind so zu verwalten, dass sie von jedermann abgerufen werden können. Demgegenüber sind die

Zertifikate selbst bzw. einzelne Informationen daraus (z.B. Teilnehmernname, öffentlicher Schlüssel) nur mit Erlaubnis des Teilnehmers für Dritte zugänglich.

Für bestimmte Daten kann es notwendig sein, diese mit einem vertrauenswürdigen Zeitpunkt zu verknüpfen. Solche Daten (oder ihr Hashwert) werden dazu mit der vom Zeitstempeldienst der Zertifizierungsstelle anzubietenden, vertrauenswürdigen Zeit digital verknüpft und das Ergebnis anschließend digital signiert und an den Benutzer zurückgesandt. Alternativ kann auch der mit dem Zeitstempel versehene Hashwert mit einem Verweis auf die entsprechenden Daten in einem öffentlich zugänglichen Verzeichnis bekannt gegeben werden. Wird die Zertifizierungsstelle konform zum deutschen Signaturgesetz betrieben, so ist der Zeitstempeldienst obligatorisch.

Zeitstempel

Zertifizierungsstellen können darüber hinaus einen Dienst zur Schlüsselaufbewahrung (vgl. Abschnitt 9.4) anbieten, so dass bei einem Schlüsselverlust der Schlüsselbesitzer eine Kopie des Schlüssels erhält, die in der Zertifizierungsstelle geschützt aufbewahrt wird.

Recovery

Der allgemeine Ablauf zur Nutzung der vorgestellten Dienste eines Trust Centers lässt sich wie folgt zusammenfassen. Zunächst wird vom Teilnehmer selbst oder in der Zertifizierungsstelle ein Schlüsselpaar generiert. Der Teilnehmer identifiziert sich, lässt sich registrieren und beantragt ein Zertifikat. Dieses Zertifikat wird anhand der Teilnehmer- und Schlüsseldaten von der Zertifizierungsstelle erstellt und sowohl an den Personalisierungsdienst als auch an den Verzeichnisdienst übermittelt. Der Personalisierungsdienst überträgt die für den Teilnehmer relevanten Daten auf die Signaturkomponente, welche nun dem Teilnehmer ausgehändigt werden kann.

Dienste-Nutzung

SigG-konforme CA

Die Einrichtung von Zertifizierungsstellen, die entsprechende Dienste anbieten und konform zum Signaturgesetz arbeiten, wird zurzeit in Unternehmen und Behörden verstärkt vorangetrieben. Als problematisch und aufwändig erwies sich die gesetzeskonforme Umsetzung der hohen Sicherheitsanforderungen, die bis zum Inkrafttreten des Gesetzes über die Rahmenbedingungen elektronischer Signaturen im Jahre 2001 in Deutschland galten. Mit den neuen, an das EU-Recht angeglichenen Regelungen kann der zu leistende Aufwand an die Art der Signatur, also einfach, fortgeschritten oder qualifiziert, angepasst werden. Ein sehr hoher Aufwand ist nunmehr nur noch für qualifizierte Signaturen zu leisten, die nach dem Gesetz der handschriftlichen Unterschrift gleichgestellt sind. Die Akkreditierung einer CA als gesetzeskonforme CA ist nach der gesetzlichen Neuregelung nicht mehr verpflichtend, dennoch haben in Deutschland nahezu alle Trust Center den

SigG-konform

Akkreditierung

Akkreditierungsprozess durchlaufen, obwohl damit sehr hohe Kosten verbunden sind. An den Aufbau und den Betrieb einer gesetzeskonformen CA werden in Deutschland Anforderungen sowohl in Bezug auf die baulichen Maßnahmen als auch in Bezug auf die Sicherheit der verwendeten IT-Systeme gestellt. So wird beispielsweise das Vorhandensein einer durchbruchshemmenden Bauweise, abstrahlsicherer Wände oder elektronischer Zugangsschutzsicherungen gefordert. Zu dem zu leistenden Aufwand zählen auch die Etablierung eines fälschungssicheren Zeitstempeldienstes oder der Rückruf bzw. das Sperren von Zertifikaten, falls der darin beglaubigte Schlüssel kompromittiert wurde.

PKI

Die Zertifizierungsstellen sind Bestandteil von Public-Key Infrastrukturen (PKI) (vgl. Abschnitt 9.1.3), die noch immer häufig als proprietäre Lösungen etabliert werden.

Beispiel 9.1 (Authenticode und VeriSign)

Authenticode

Die von Microsoft entwickelte Technologie des *Authenticodes*³ ist eine spezielle Ausprägung einer Zertifizierungsinfrastruktur zum Signieren von Code, der von Webservern auf Webclients geladen wird. Von dem zu signierenden Code (u.a. .dll, .exe, .ct1 Dateien) wird zunächst ein Hashwert berechnet, der mit dem privaten Schlüssel des Codeherstellers signiert wird. Gemeinsam mit dem Code und der Signatur wird ein X.509 v3 konformes Zertifikat an den WebClient übertragen, wobei die Echtheit des Herstellerzertifikates wiederum durch eine digitale Signatur der ausstellenden Instanz (Zertifizierungsstelle) nachgewiesen werden kann.

Microsofts Internet Explorer (IE) verwendet die Authenticode-Technik, um die Signatur von Code, bevor er über den Browser auf den Rechner heruntergeladen wird, zu überprüfen. Abhängig von den Sicherheitseinstellungen des Browsers, wird signierter Code automatisch ausgeführt.

Zertifikate

Ein Dienstleister, der diese Technologie nutzen bzw. anbieten will, muss ein Zertifikat bei einer Zertifizierungsstelle beantragen, woraufhin er einen privaten Schlüssel und das Zertifikat zugewiesen bekommt. Microsoft arbeitet mit VeriSign als Zertifizierungsstelle zusammen und vergibt zwei unterschiedliche Zertifikate, nämlich Individualzertifikate und kommerzielle, für die unterschiedlich aufwändige Identitätsnachweise zu führen sind. Bei Beantragung eines Individualzertifikats genügt die Angabe des Namens und der Adresse des Antragstellers. Diese Angaben werden nur mit Angaben in einer öffentlichen Datenbank abgeglichen: „*Individual applicants must submit their name, address, and other material that will be checked against an independent consumer database to verify their credentials*“. Bei kommerziellen

³ <http://msdn.microsoft.com/en-us/library/ms537359%28v=vs.85%29.aspx>

Zertifikaten wird vom Antragsteller eine spezielle Bescheinigung, das so genannte Dun & Bradstreet Rating, gefordert. In beiden Fällen verpflichten sich die Antragsteller, keinen bösartigen Code oder Viren wissentlich oder fahrlässig herzustellen oder zu verteilen: „*they cannot and will not distribute software that they know, or should have known, contains viruses or would otherwise maliciously harm the user's computer or code*“.

Die Ausführungen verdeutlichen hinreichend klar, dass ein mittels der Authenticode Technologie erstellter Ursprungsnachweis von Code nur eine sehr beschränkte Aussagekraft besitzt. Zu beachten ist, dass Authenticodes keine Nachweise über die Korrektheit oder Zuverlässigkeit des assoziierten Codes liefern. Insbesondere dann, wenn keine Maßnahmen zur Beweissicherung vorhanden sind, ist beim Auftreten von Sicherheitsproblemen, zum Beispiel infolge eines Virenbefalls, kaum noch nachvollziehbar, wer der Verursacher war. Problematisch ist darüber hinaus, dass das Hinzufügen von Authenticodes zu aktiven Inhalten einem in Sicherheitsfragen nicht versierten Benutzer eine trügerische Sicherheit vorgaukelt und ihn eventuell sogar erst zu einem zu leichtfertigen Umgang mit den zertifizierten Daten verleitet.

Probleme



9.1.3 Public-Key Infrastruktur

Zur Erzeugung und Verwaltung von Zertifikaten werden spezielle Infrastrukturen eingesetzt. Im Bereich der asymmetrischen Kryptosysteme hat sich für die Gesamtheit der Komponenten, die hierfür benötigt werden, der Begriff der Public-Key Infrastruktur (PKI) eingebürgert. Abbildung 9.1 gibt einen Überblick über die Komponenten einer PKI.

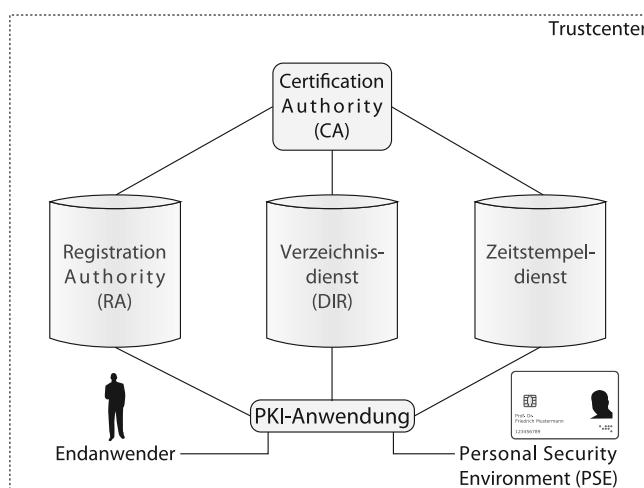


Abbildung 9.1: Komponenten einer PKI

CA

Komponenten einer PKI

Eine Public-Key Infrastruktur umfasst eine Zertifizierungsinstanz (CA), die Zertifikate herausgibt und ggf. wieder zurückruft (revozieren).

RA

Die Registrierungsinstanz (Registration Authority) RA bürgt für die Verbindung zwischen öffentlichem Schlüssel und Identitäten sowie Attributen der Zertifikatsinhaber. Die Komponenten CA und RA sind häufig in einem Trust Center zusammengefasst. Die Regeln, nach denen das Trust Center seine Zertifikate ausstellt und verwaltet, sind in einer Zertifikatspolicy, einem Certification Practice Statement (CPS), festgehalten. Eine solche Policy enthält u.a. rechtliche und finanzielle Bedingungen, beschreibt den Ablauf der Authentifikation von Zertifikatsinhabern im Trust Center, oder auch die Kriterien zum Schutz des Trust Centers.

CRL

Zurückgezogene Zertifikate werden in einer Sperrliste (Certificate Revocation List CRL) verwaltet. Die CRL beinhaltet die Nummern derjenigen Zertifikate, die vor dem Ablauf ihrer Gültigkeit zurückgezogen wurden. Als Alternative zur Sperrliste wird auch häufig ein Protokoll zur Zertifikats-Status-Abfrage eingesetzt. Ein Standardprotokoll hierfür ist das OCSP, das Online Certificate Status Protocol.

Verzeichnis

Ausgestellte Zertifikate und Sperrlisten werden in einem Verzeichnis verwaltet und zur Verfügung gestellt.

Standardisierung

Standardisierung

Im Zuge des Aufbaus von PKIs wurden bereits einige Standards entwickelt. Hier sind insbesondere die Aktivitäten der PKIX Working Group der IETF (Internet Engineering Task Force) zu nennen, die seit Herbst 1995 Internet Standards für eine X.509 unterstützte PKI entwickeln und deren Ergebnisse in einer Reihe von RFCs (u.a. RFC2459, RFC2510) festgehalten sind. Der RFC 2459 spezifiziert beispielsweise das Internet PKI Profile, das im Wesentlichen eine Instanziierung des X.509 Frameworks für Zertifikate darstellt.

Standards, die Datenformate spezifizieren, um eine Interoperabilität verschiedener Komponenten in einer PKI zu ermöglichen, werden durch die Public Key Cryptography Standards⁴ (PKCS) 1 – 15 festgelegt. In einer PKI werden verschiedene Verzeichnisdienste benötigt, um z.B. auf Zertifikate zugreifen zu können, oder um Zertifikat-Rückruflisten zu verwalten. Verzeichnisse sowie ein Protokoll, das Directory Access Protocol (DAP), zum Zugriff auf Verzeichnisse werden im X.500 Standard spezifiziert. DAP setzt auf dem OSI-Modell auf und ist relativ komplex. Eine vereinfachte Struktur bietet das Lightweight Directory Access Protocol (LDAP), das auf TCP/IP Protokollen basiert. LDAP-Server sind heutzutage sehr weit verbreitete Ver-

⁴ Vgl. Seite 346.

zeichnisdienste in Unternehmen, die auch für die PKI-Aufgaben verwendet werden können.

SigG-konforme PKI

Im deutschen Signaturgesetz wird eine zweistufige Zertifizierungsinfrastruktur festgelegt. Die Wurzelinstanz bildet die Bundesnetzagentur (BNetzA)⁵, die den Betrieb von Zertifizierungsstellen genehmigt (vgl. Abbildung 9.2). Sie stellt die Zertifikate für deren Signaturschlüssel aus und verwaltet sie in einem Verzeichnisdienst. Die genehmigten Zertifizierungsstellen wie zum Beispiel die Deutsche Telekom stellen die Zertifikate für die Teilnehmer aus und verwalten eine Liste sämtlicher gültiger und gesperrter Zertifikate. Das Signaturgesetz unterscheidet zwischen Verzeichnissen, die den Abruf von Zertifikaten ermöglichen, und Verzeichnissen, die lediglich das Nachprüfen der Zertifikate erlauben. Die vom Gesetzgeber anerkannten Prüf-

Signaturgesetz-
konforme PKI

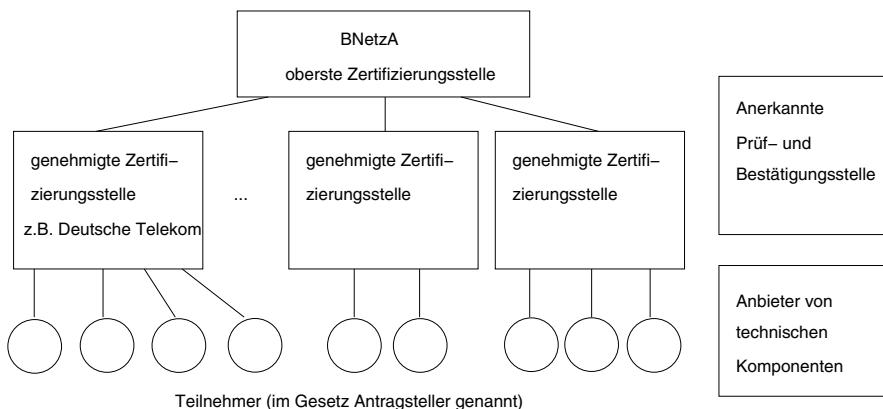


Abbildung 9.2: Deutsche Zertifizierungs-Infrastruktur

und Bestätigungsstellen haben die Aufgabe, Aussagen darüber zu treffen, ob Zertifizierungsstellen und bzw. oder technische Komponenten den Anforderungen des Signaturgesetzes und der Signaturverordnung entsprechen. Anbieter von gesetzeskonformen technischen Komponenten müssen den Nachweis der Konformität durch eine Bestätigung einer der anerkannten Prüfstellen führen. Zum Schutz vor Missbrauch veröffentlicht die BNetzA regelmäßig eine Liste von bestätigten Anbietern im Bundesanzeiger.

Mehrstufige Hierarchien

Im Folgenden wollen wir ein allgemeineres Architekturmodell betrachten, so wie es in seinen Grundzügen im Kontext des Internet-Standards für elektronische Nachrichten, dem PEM-Standard [100] (vgl. Kapitel 14.6),

Hierarchie

⁵ Früher unter dem Namen RegTP geführt.

spezifiziert wurde. Ausgangspunkt ist ebenfalls eine einzelne Wurzelinstanz, die hier in Anlehnung an das PEM-Umfeld als IPRA (Internet Policy Registration Authority) bezeichnet wird. Die nächste Stufe bilden PCAs (Policy Certification Authorities). Jede PCA kann eine eigene Strategie zur Registrierung von Benutzern und zur Beglaubigung von Schlüsseln definieren. Jede PCA-Instanz wird von der Wurzelinstanz zertifiziert. Unterhalb der PCAs sind die CAs (Certification Authorities) angesiedelt, die ihrerseits noch beliebig hierarchisch untergliedert sein können. Eine CA signiert direkt Benutzerschlüssel oder den Schlüssel hierarchisch nachgeordneter CAs, die zum Beispiel die Zertifizierungsstrategie von Unterorganisationen, wie einzelne Abteilungen, Tochterfirmen, Institute oder Behörden, festlegen.

Problem

Die hierarchische Organisation von Zertifizierungsstellen ermöglicht eine flexible und skalierbare Verwaltung. Die Aufgaben können dezentral durch verschiedene Institutionen durchgeführt werden, so dass sich Engpässe vermeiden lassen. Problematisch ist jedoch, dass sich mit der Anzahl dieser Instanzen auch die Zahl der Angriffspunkte erhöht. Dies ist eine typische Situation bei der Konstruktion sicherer verteilter Systeme. Es ist somit darauf zu achten, dass nicht durch das schwächste Glied der Konfiguration das gesamte System infrage gestellt wird. Zusätzliche Kontrollen sind notwendig, um zu verhindern, dass sich ein erfolgreicher Angriff auf das gesamte System auswirkt.

Trust Anchor

Eine einzige, weltweit operierende CA-Hierarchie ist weder praktikabel noch stößt sie auf die Akzeptanz der Benutzer, da alle Nutzer einer einzigen Wurzel-Instanz vertrauen müssten. Im Laufe der Jahre haben sich in Unternehmen, Verwaltungen und sonstigen Institutionen jeweils eigene „PKI-Inseln“ etabliert (z.B. eine FhG-PKI für alle Institute der Fraunhofer-Gesellschaft, die PKI an der TU München, eine BMW-interne PKI etc.). Die jeweiligen Wurzel-Instanzen dieser unterschiedlichen PKIs bilden die so genannten Vertrauensanker (engl. *trust anchor*) ihrer jeweiligen Zertifikats-Hierarchie. Das heißt, dass sie ihren eigenen Signatur-Schlüssel selbst zertifizieren und die an der Hierarchie Beteiligten der Glaubwürdigkeit dieses Schlüssels vertrauen.

Zertifikat-Überprüfung**gleiche CA**

Möchten zwei Teilnehmer A und B, die der gleichen CA unmittelbar zugeordnet sind, miteinander kommunizieren und ihre jeweiligen Zertifikate verifizieren, so ist dies in diesem Fall sehr einfach. Da beide im Besitz des öffentlichen Schlüssels ihrer gemeinsamen Zertifizierungsstelle CA sind, können sie das Zertifikat des Partners, das von der CA ja signiert wurde, unmittelbar überprüfen.

Pfad

Auch etwas längere Ketten lassen sich nach diesem Schema bilden und validieren. Abbildung 9.3 veranschaulicht ein Beispiel für eine Zertifikatskette. Betrachtet man den Pfad von links nach rechts, so ergibt sich folgende Zer-

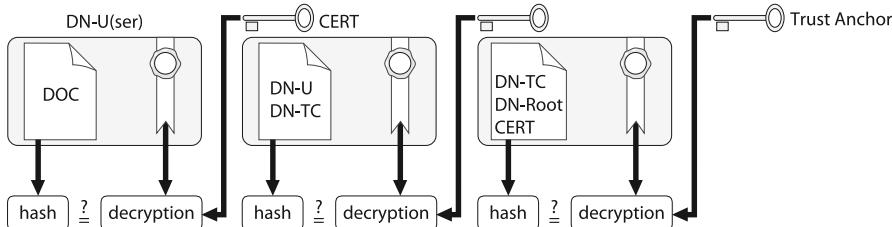


Abbildung 9.3: Zertifizierungspfad

tifizierungskette: das Dokument (doc) wird mit dem Signaturschlüssel des Benutzers signiert und der zum Validieren erforderliche Schlüssel befindet sich im Zertifikat des Nutzer, im DN-U-Zertifikat. Das DN-U-Zertifikat ist seinerseits signiert von einer Instanz TC, dessen Zertifikat von der Instanz DN-Root-CA ausgestellt (signiert) wurde. Diese Instanz ist die Wurzel und bildet den Vertrauensanker. Der Empfänger des signierten Dokumentes doc benötigt zur Prüfung der Signatur des Nutzer DN-U dessen Zertifikat, das Zertifikat von dessen CA, also von DN-TC, und den Vertrauensanker.

Problematischer wird die Situation, wenn die Kommunikationspartner unterschiedlichen CAs zugeordnet sind. Die Validierung von Zertifikaten ist ein rekursiver Prozess. Betrachten wir dazu das in Abbildung 9.4 dargestellte Szenario, in dem zwei Partner Alice und Bob miteinander kommunizieren möchten.

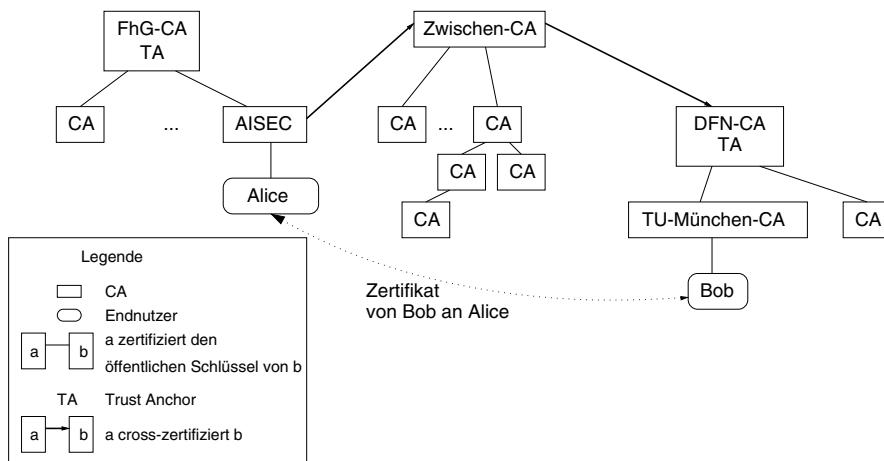
unterschiedliche
CAs

Abbildung 9.4: Zertifikatvalidierung

Cross-Zertifikat

Bob schickt Alice sein Zertifikat, das von seiner CA, im Beispiel der CA der TU München zertifiziert wurde. Da sich die beteiligten Kommunikationspartner in unterschiedlichen PKI-Vertrauensbereichen befinden, ist eine Zertifikatvalidierung nur möglich, wenn über Cross-Zertifikate ein Validierungspfad aufgebaut werden kann. Mit einem Cross-Zertifikat zertifiziert eine CA_i den öffentlichen Schlüssel einer CA_j , so dass alle Beteiligten, die der CA_i vertrauen, über diese Cross-Zertifizierung auch dem Schlüssel der CA_j vertrauen können.

In unserem Beispiel-Szenario existiere eine solche Cross-Zertifizierung durch die AISEC-CA für die im Bild angegebene Zwischen-CA, die ihrerseits die DFN⁶-CA zertifiziere. Um aus der Sicht der Nutzerin Alice das Zertifikat von Bob validieren zu können, benötigt sie einen Vertrauensanker sowie die Kette der Zertifikate, die Alice mit Bob „verbindet“. Man nennt diese Kette auch den Validierungspfad. Im Beispiel sei der Vertrauensanker das Zertifikat der FhG-CA. Der Validierungspfad besteht somit aus den Zertifikaten der FhG-CA, AISEC-CA, Zwischen-CA, DFN-CA, TU-München-CA und dem Zertifikat-Bob.

Da das Zertifikat für die AISEC-CA von der FhG-CA ausgestellt wurde, kann es mit dem öffentlichen Schlüssel der FhG-CA validiert werden (das ist ja in diesem Beispiel der Trust Anchor) und man erhält den gültigen öffentlichen Schlüssel der AISEC-CA. Mit diesem kann man das Zertifikat der Zwischen-CA überprüfen, da es von der AISEC-CA Cross-zertifiziert wurde. Durch diesen Schritt erhält man den gültigen öffentlichen Schlüssel der Zwischen-CA, mit dem das Zertifikat der DFN-CA überprüft und im vorletzten Schritt dann das Zertifikat der TU-München überprüft werden kann. Damit steht der öffentliche Schlüssel zur Verfügung, um schlussendlich das Zertifikat von Bob, um das es ja eigentlich ging, zu validieren. Wie man leicht sieht, kann das Durchlaufen der Validierungskette sehr schnell zu einer sehr aufwändigen Prozedur werden. Sobald ein Zertifikat in der Kette abgelaufen ist oder als ungültig erklärt wird, aber natürlich auch wenn gar nicht erst ein solcher Pfad aufgebaut werden kann, scheitert die Zertifikat-Überprüfung und liefert eine Fehlermeldung.

Validierungspfad

Wie man leicht sieht, kann die Überprüfung von Zertifikaten, die von CAs aus unterschiedlichen administrativen Domänen ausgestellt wurden, sehr aufwändig sein. Diese Beobachtung bekommt im Zusammenhang mit dem Übergang auf elektronische Reisepässe eine besondere Bedeutung zu (vgl. Abschnitt 11.2.1). Diese Pässe besitzen einen integrierten RFID-Chip, auf dem die Authentisierungsdaten des Passinhabers abgelegt sind. Besonders sensitive biometrischen Daten des Passinhabers, wie der Fingerabdruck, müssen besonders vor unberechtigten Zugriffen geschützt werden. Mit der

⁶ Deutsches Forschungs Netz

Reisepass

Spezifikation der Extended Access Control (EAC) für elektronische Reisedokumente wird festgelegt, dass zum Auslesen dieser sensiven Daten, die Lesegeräte sich mittels einer Zertifikat-basierte Authentisierung ausweisen müssen. Hierbei bestimmt die Passausgebende Instanz, welches ausländische Lesegerät, welche Leseberechtigungen auf die Passdaten erhält. Zur Absicherung der Daten wird eine spezielle PKI verwendet (vgl. <http://www.bsi.de/fachthem/elekausweise/epass/>).

Zertifikate besitzen nur eine begrenzte Gültigkeitsdauer. Erste Probleme sind dadurch bereits Anfang 2004 aufgetreten, als zwei wichtige Verisign-Zertifikate abgelaufen sind. Bei den betroffenen Zertifikaten handelte es sich um Intermediate-Zertifikate, die unter anderem im Internet Explorer und Mozilla dafür sorgen, die Vertrauenswürdigkeit und Herkunft von SSL-Server-Zertifikaten zu überprüfen. Abgelaufen ist also nicht das Root-Zertifikat von Verisign, sondern ein Zwischen-Zertifikat in einer Zertifizierungskette. Da dieses ungültig und die Kette damit unterbrochen ist, zeigt der Browser beim Aufbau einer mit SSL zu sichernden Verbindung eine Fehlermeldung an. Allerdings ist es weiterhin möglich, verschlüsselt mit dem Server zu kommunizieren. Um den Fehler zu beseitigen, genügt es, auf der Serverseite ein neues Intermediate-Zertifikat zu installieren. Da dies mit einem weiterhin gültigen Root-Zertifikat unterschrieben ist, können Browser wieder ohne Fehlermeldung auf den Server zugreifen. Betroffen sind auch andere Anwendungen, wie beispielsweise Java-Applikationen und Browser-Applets, die die alten Zertifikate verwenden. Sie zeigen ihren Benutzern Warnhinweise oder verweigern den Aufbau von SSL-Verbindungen.

Ablauf der Gültigkeit

Fazit

Nachdem in den letzten Jahren in Unternehmen und Behörden eine Vielzahl von PKI-Projekten gestartet wurde, ist mittlerweile eine gewisse Ernüchterung eingetreten, da der Aufbau derartiger Sicherheitsinfrastrukturen mit erheblichen Kosten und großen organisatorischen sowie Akzeptanzproblemen seitens der Benutzer verbunden ist. Nach wie vor problematisch ist auch die Heterogenität der PKI-Lösungen mit fehlender Interoperabilität. Dadurch wird eine unternehmens- oder gar länderübergreifende sichere Kommunikation stark beeinträchtigt. Der zur Zeit noch häufigste Einsatz der PKI-Technologie besteht in der Verschlüsselung von E-Mails (vgl. Abschnitt 14.6).

Probleme

Mit der zunehmenden Integration elektronischer Abläufe in unternehmerische und behördliche Geschäftsprozesse wird jedoch der Bedarf an digitalen Signaturen und insbesondere auch an qualifizierten elektronischen Signaturen als rechtsverbindliches Äquivalent zur handschriftlichen Signatur stark steigen. Mit elektronischen Signaturen lassen sich Vorgänge erheblich ver-

Entwicklung

einfachen, die bislang eine handschriftliche Unterschrift verbunden mit einer persönlichen Anwesenheit erforderlich gemacht haben. Anwendungsfelder für den Einsatz der elektronischen Signatur ergeben sich unter anderem im Bereich E-Procurement, also der elektronischen Beschaffung und Rechnungsstellung, oder auch der elektronischen Dokumentenverwaltung. Auch in der Verwaltung könnten durch den Austausch und die Bearbeitung elektronisch signierter Akten viele Prozesse erheblich beschleunigt und Kosten eingespart werden.

Ein weiterer wichtiger Einsatzbereich für PKIs ergibt sich aus der zunehmenden Mobilität von Benutzern. Über mobile, drahtlose Netze lassen sich auch spontan Kommunikationsbeziehungen zu vorab nicht bekannten Partnern aufbauen. PKIs mit den entsprechenden Zertifikatinfrastrukturen können auch hierfür die benötigten Vertrauensanker liefern.

9.2 Schlüsselerzeugung und -aufbewahrung

Wir haben wiederholt darauf hingewiesen, dass die Sicherheit von kryptografisch starken Systemen im Wesentlichen auf der sicheren Verwaltung des geheimen Schlüssels beruht. Sind solche Schlüssel leicht zu erraten oder dem Angreifer einfach zugänglich, so helfen auch keine noch so starken Verschlüsselungsverfahren mehr, die Vertraulichkeit der Daten zu gewährleisten. Den Managementdiensten zur Schlüsselerzeugung und -aufbewahrung kommt deshalb eine große Bedeutung zu.

9.2.1 Schlüsselerzeugung

Für die Schlüsselerzeugung in symmetrischen Kryptosystemen benötigt man gute Zufallszahlengeneratoren (engl. *Random Number Generator* (RNG)).

Zufallszahlengeneratoren können deterministisch oder nicht-deterministisch sein. Ein deterministischer Zufallszahlengenerator liefert bei gleichen Ausgangsbedingungen immer die gleiche Folge von Zufallszahlen, man spricht von Pseudozufallszahlengeneratoren. Ein nicht-deterministischer Zufallszahlengenerator erfordert die Einbeziehung von externen Vorgängen. Beispiele sind physikalische Vorgänge, wie z.B. ein Geigerzähler, der die Anzahl der radioaktiven Zerfälle in einer bestimmten Zeitspanne misst.

Pseudozufallszahlengeneratoren (PNG) werden durch einen Algorithmus beschrieben und man fordert, dass die generierte Zufallszahl so erzeugt wird, dass der Wert für einen Dritten, z.B. einen Angreifer, nicht vorhersagbar ist. Wichtig ist also, dass alle erzeugten Zufallszahlen mit der gleichen Wahrscheinlichkeit auftreten und dass ein Fremdeinwirken ausgeschlossen ist. Ein Pseudozufallsgenerator erzeugt somit keine Zufallszahl

Zufallszahlen-
generator

PNG

im eigentlichen Sinn, sondern berechnet aus einem Startwert eine Zahl, deren Entstehung für einen Außenstehenden nicht nachvollziehbar ist. Diese Zahl kann für die meisten Anwendungen als „zufällig genug“ angesehen werden. Allerdings besteht hier nach wie vor die Notwendigkeit, einen geeigneten Startwert zu finden, der den Generator initialisiert. Unter Linux wird deshalb beispielsweise mit dem `/dev/random` eine Mischung aus Pseudozufallszahlengenerator und echten Zufallswerten, die sich u.a. aus dem Rauschen von Gerätetreibern ergeben, verwendet. Kryptografisch starke Pseudozufallszahlengeneratoren werden in der Praxis häufig durch nichtlinear rückgekoppelte Schieberegister realisiert.

Deterministische Zufallszahlengeneratoren können durch Hardware-basierte Generatoren, wie zum Beispiel der RNG im TPM-Chip, oder durch Software-basierte Generatoren realisiert werden.

Nachfolgend sind einige Methoden genannt, wie man in der Praxis Zufallswerte gewinnen kann. Diese können dann entweder direkt als Schlüssel eingesetzt werden oder sie dienen als die benötigten Startwerte für die Generatoren.

Die Schlüsselerzeugung in asymmetrischen Verfahren erfordert im Allgemeinen einen höheren Berechnungsaufwand als die Generierung von Pseudozufallszahlen in symmetrischen Systemen. Für die meisten der in der Praxis eingesetzten Verfahren wie das RSA- oder DSS-Verfahren werden große Primzahlen benötigt, für die aufwändige Primzahltests durchzuführen sind. Man behilft sich hier in der Regel mit Pseudoprimalzahlen, für die mit einer gegebenen Wahrscheinlichkeit sichergestellt ist, dass es sich um eine Primzahl handelt. Zu den bekanntesten, effizienten probabilistischen Primzahltest-Algorithmen gehört der Rabin-Miller-Test [101].

asymmetrische
Verfahren

Techniken zur Schlüsselerzeugung

Obwohl manuelle Verfahren wie das Werfen einer Münze oder das Würfeln bei den meisten Anwendungen nicht in Betracht kommen, sind sie doch aufgrund ihrer einfachen Handhabung für die Klasse von Schlüsseln anwendbar, die nur sehr selten ausgetauscht werden müssen und bei denen sonstige automatische Prozesse der Schlüsselerzeugung nicht zur Verfügung stehen. Beispiele hierfür sind so genannte Master-Schlüssel, die auf externem Weg, wie dem Postweg, ausgetauscht oder vom Hersteller technischer Geräte, z.B. Chipkarten, direkt in einen geschützten Speicherbereich geschrieben werden. Weiterhin sind diese manuellen Techniken einsetzbar, um das Problem der Initialisierung von Zufallszahlengeneratoren zu lösen.

manuelle Verfahren

Eine sehr zuverlässige, aber auch sehr aufwändige Methode ist das Ausnutzen von natürlichen (in manchen Fällen auch künstlich hervorgerufenen)

Zufallsereignis

Zufallsereignissen, die regelmäßig stattfinden. Es kommen u.a. atmosphärisches Rauschen, radioaktiver Zerfall, Widerstandsrauschen, Entladung von Kapazitäten etc. in Betracht. Die Zufallswerte können daraus gebildet werden, indem man die Zeitdifferenz zweier aufeinander folgender Ereignisse berechnet und über eine vorher festgelegte Funktion das Ergebnis-Bit bestimmt. Allerdings ist es in der Praxis schwierig, diese Verfahren in die Kryptosysteme zu integrieren; meist ist eine spezielle Hardware erforderlich.

deterministische
Berechnungen

Auch gängige Hardwarekomponenten eignen sich als Basis zur Berechnung von Pseudozufallszahlen. Pseudozufällige Bitfolgen lassen sich unter anderem durch Messung der Tastaturverzögerung, also der Zeit zwischen zwei Tastenanschlägen, durch Analyse der Mausbewegungen oder auch durch eine Kombination aus der Uhrzeit der lokalen Systemuhr, der Systemidentifikation und dem aktuellen Datum erzeugen.

Standard X9.17

Mit dem ANSI-Standard X9.17 ist ein Verfahren festgelegt, das mithilfe eines symmetrischen Algorithmus (z.B. AES) eine zufällige Bitfolge bestimmt. Das Verfahren gewinnt aus einem Zeitwert T_i und einem Startwert V_i einen Zufallsschlüssel S_i und den Startwert V_{i+1} für den nächsten Schritt. Ausgangsbasis für die Schlüsselerzeugung ist ein vorab vereinbarter, geheimer Schlüssel K sowie ein geheimer Initialwert V_0 . Um den i -ten Schlüssel S_i zu generieren, ist z.B. unter Nutzung von AES und einem 128-Bit Initialwert V_0 wie folgt vorzugehen:⁷

$$S_i := \text{AES}(\text{AES}(T_i, K) \oplus V_i, K)$$

Den Startwert für den nächsten Generierungsschritt erhält man durch

$$V_{i+1} := \text{AES}(\text{AES}(T_i, K) \oplus S_i, K).$$

Damit lassen sich unter Nutzung vorhandener, kryptografischer Verfahren auf effiziente Weise geeignete Zufallsschlüssel generieren, solange der geheime Schlüssel K nicht offen gelegt ist. An dessen Sicherheit, einschließlich seiner sicheren Speicherung in zugriffskontrollierten, geschützten Speicherbereichen, werden somit hohe Anforderungen gestellt.

9.2.2 Schlüsselspeicherung und -vernichtung

Benutzergedächtnis

Die sichere Aufbewahrung kryptografischer Schlüssel ist eine weitere wesentliche Aufgabe eines Schlüsselmanagements. In vielen Systemen wird diese Aufgabe dem Benutzer selber überlassen, der den Schlüssel im einfachsten Fall in seinem Gedächtnis speichert und ihn bei Bedarf eingibt. Klassisches Beispiel hierfür sind die Benutzerpasswörter zur Authentifikation (vgl. Kapitel 10.2.1). Da aber in heutigen Systemen eine Vielzahl

⁷ \oplus bezeichnet wie immer die XOR-Verknüpfung.

unterschiedlicher Schlüssel verwendet werden, ist es einem Benutzer nicht zuzumuten, sich alle diese Schlüssel auswendig zu merken und sie nicht, was leider oft genug der Fall ist, auf einem Stück Papier zu notieren. Die Verlagerung der Managementaufgabe zur sicheren Schlüsselspeicherung auf den Benutzer eröffnet somit Sicherheitsprobleme, die durch geeignete, maschinelle Unterstützung vermieden werden können. Einen noch höheren Grad an Sicherheit als dies durch eine reine Softwarelösung möglich ist, bieten Smartcard-basierte Schlüsselspeicher.

Persönliche Sicherheitswerkzeuge

Eine sehr attraktive Lösung des Speicherungsproblems bieten so genannte persönliche Sicherheitswerkzeuge wie Chipkarten (vgl. Kapitel 11) oder USB-Token. Kryptografische Schlüssel werden im ROM des Chips bzw. des Tokens gespeichert und können nur unter Verwendung von speziellen Lesegeräten ausgelesen werden. Auch wenn es in der Vergangenheit Angriffe auf Chipkarten gegeben hat (u.a. [6]), so ist doch der Aufwand, der dafür zu leisten ist, sehr hoch und die Angriffe sind sehr schwierig.

Chipkarte

Ein persönliches Sicherheitswerkzeug kann darüber hinaus durch den Einsatz einer PIN (Personal Identification Number) zugriffsgeschützt werden, so dass nur der berechtigte Benutzer nach erfolgreicher PIN-Überprüfung auf die gespeicherten Daten zugreifen kann. Zunehmend werden auch biometrische Zugangskontrollen (vgl. Abschnitt 10.3) in die Sicherheitswerkzeuge integriert, so dass der berechtigte Benutzer anhand seines Fingerabdrucks oder seiner Handschriftendynamik authentifiziert wird. Die offensichtlichen Vorteile der einfachen Nutzbarkeit werden jedoch getrübt durch die Probleme, die nach wie vor beim Einsatz biometrischer Techniken zu bewältigen sind (vgl. dazu auch Abschnitt 10.3.5). Um die Gefahr einer unabsichtlichen Offenlegung kryptografischer Schlüssel zu verringern, kann auch der direkte Zugriff durch den autorisierten Benutzer unterbunden werden, so dass durch die PIN-Eingabe ausschließlich festgelegte, interne Berechnungsschritte auf der Chipkarte, wie das Berechnen digitaler Signaturen, angestoßen werden.

PIN-Schutz

Biometrie

Die Sicherheit kann weiter erhöht werden, wenn man den Schlüssel auf mehrere Komponenten aufspaltet und z.B. einen Teil auf der Smartcard und einen anderen auf der Workstation oder dem PC verwaltet. Nur durch das Zusammenwirken aller Komponenten kann der Gesamtschlüssel gewonnen werden. Schützt man jeden Schlüsselteil durch ein eigenes Zugriffsverfahren, so führt der Verlust bzw. die Kompromittierung einer dieser Schlüsselkomponenten noch nicht zur Kompromittierung des Gesamtschlüssels.

Schlüsselaufteilung

Da Signierschlüssel die Chipkarte nicht verlassen sollten, müssen die zu signierenden Daten, wie beispielsweise die auf einem externen Rechner berechneten Hashwerte von Dokumenten, auf die Chipkarte zum Signieren

Darstellungsproblem

übertragen werden. Hierbei tritt das so genannte Darstellungsproblem auf, da die Chipkarte bzw. der Chipkarteninhaber den übertragenen Wert nicht sehen kann und dem Rechner vertrauen muss, dass dieser die korrekten Werte übermittelt. Wird das IT-System jedoch von einem Angreifer kontrolliert, dem eine gezielte Manipulation der Systemfunktionalität zum Beispiel durch die Integration eines Trojanischen Pferdes gelungen ist, so ist diese Voraussetzung nicht mehr gegeben. Zurzeit gibt es noch keine zufrieden stellenden Lösungen für dieses Problem. Es verdeutlicht erneut das Grundproblem sicherer Systeme, dass deren Sicherheit nur so stark ist wie das schwächste Glied in der Sicherheitskette.

Persistente Speicherung

Dateisystem

Häufig werden Schlüssel in Dateien gespeichert, die vom Betriebssystem des jeweiligen Rechners verwaltet werden. Dies gilt beispielsweise für die Schlüsselringe, die das Verschlüsselungsprogramm PGP (Pretty Good Privacy) (vgl. Kapitel 14.6) verwendet, oder auch für die Verwaltung von Passworten, die für die Benutzerauthentifikation benötigt werden. Der Zugriff auf diese Dateien ist natürlich besonders zu kontrollieren und zu beschränken. Alternative Realisierungsmöglichkeiten bestehen darin, dass man nur unter Eingabe eines Passwortes Zugriff auf die Schlüsseldatei erhält oder Schlüssel nicht als Klartexte abgelegt werden. Letzteres ist besonders wichtig, da Dateien durch Backup-Verfahren archiviert oder Inhalte von Dateien in Zwischenspeichern (Caches) gehalten werden können, so dass ein Angreifer unter Umständen die Zugriffskontrolle, die das Betriebssystem bei jedem Dateizugriff automatisch durchführt, umgehen kann.

Trusted Computing

Mit den Entwicklungen rund um das Trusted Computing (TC) (vgl. Abschnitt 11.4) stehen weitere Möglichkeiten für hardware-basierte, sichere Schlüsselspeicher zur Verfügung. Das Trusted Platform Module (TPM) erweitert herkömmliche Rechnerarchitekturen um einen zusätzlichen Chip, der im Groben vergleichbar ist mit einer Smartcard, im Gegensatz zu dieser aber fest an einen Rechner und nicht an einen Benutzer gebunden ist. Der TPM unterstützt durch Hardware-Schutzmaßnahmen einen sicheren Schlüsselspeicher für Laptops, PCs etc. Dieser wird zum Beispiel von Microsoft Vista verwendet, um den Schlüssel, der von dem Festplattenverschlüsselungsprogramm BitLocker verwendet wird, sicher in dem Chip auf dem Rechner abzulegen. Dieser Schlüssel wird durch die vom Chip angebotene, so genannte Sealing-Operation an den aktuellen Zustand des Rechners gebunden. Das BitLocker Programm Bitlocker verschlüsselt das gesamte Volume, inklusive Betriebssystem, Registry, temporäre Dateien und Hibernation File. Die Entschlüsselung der Festplatte setzt voraus, dass sich der Rechner noch in dem gleichen Zustand (Konfiguration) wie zum Zeitpunkt des Versiegelns des Schlüssels befindet; d.h. wenn die Konfiguration zw-

BitLocker

schenzeitlich absichtlich oder unabsichtlich modifiziert wurde, schlägt die Entsiegelung des Festplattenschlüssels fehl.

Schlüsselvernichtung

Dem sicheren Löschen von Schlüsseln wird oftmals erheblich weniger Aufmerksamkeit gewidmet als deren sicherer Erzeugung und Speicherung. Das kann zu gravierenden Sicherheitsproblemen führen, wenn ein als gelöscht vermuteter Schlüssel in die Hände eines Angreifers gelangt. Falls er darüber hinaus Gelegenheit hatte, die mit diesem Schlüssel verschlüsselten Kryptotexte aufzuzeichnen, ist er nun am Ziel seiner Wünsche.

Löschen

Abhängig von dem Medium, auf dem ein Schlüssel gespeichert ist, sind unterschiedliche Maßnahmen erforderlich. Wird der Schlüssel im EEPROM (Electronically Erasable Programmable Read Only Memory) eines Chips einer Smart Card gehalten, so kann man ihn durch mehrfaches Überschreiben löschen. Bei Speicherung im EPROM oder PROM ist die vollständige Zerstörung des Chips erforderlich, um die Schlüsselinformation zu löschen.

Chip

Werden Schlüssel in Dateien auf Festplatten und auf Hintergrundspeichern gespeichert bzw. archiviert, so reichen die von üblichen Betriebssystemen angebotenen Systemdienste zum Löschen von Dateien nicht aus. Diese begnügen sich in der Regel damit, nur den Eintrag für die Datei in der Dateitabelle zu löschen. Die Daten sind aber dennoch nach wie vor zugreifbar. Um Dateien tatsächlich zu löschen, muss der entsprechende Speicherbereich bereinigt, also mehrfach überschrieben werden. Zusätzliche Probleme ergeben sich durch das Konzept des virtuellen Speichers, das für heutige PCs, Laptops, Server, aber auch Smartphones gängig ist. Im Zuge der virtuellen Speicherverwaltung lagert das Betriebssystem Daten transparent für den Benutzer vom Arbeitsspeicher auf den Hintergrundspeicher aus und speichert sie in Caches oder auch in Swap-Bereichen zwischen. Beim Löschen eines Schlüssels ist somit sicherzustellen, dass auch alle angelegten Kopien gelöscht und die verwendeten Speicherbereiche bereinigt werden.

Archivierung

9.3 Schlüsselaustausch

Nachdem im vorherigen Abschnitt die Schlüsselerzeugung und -speicherung erklärt wurde, wollen wir uns nun mit der Frage des sicheren Schlüsselaustausches beschäftigen. Hierbei ist der geschützte Austausch geheimer Schlüssel symmetrischer Kryptosysteme von Interesse. Die einfachste Lösung besteht darin, dass jeder Kommunikationsteilnehmer mit jedem anderen Teilnehmer auf geschützte Weise einen Schlüssel vereinbart. Dies kann ex-

Kopien

Jeder mit Jedem

tern beispielsweise per Briefpost oder auf elektronischem Weg geschehen. Bei einem Netz von n Kommunikationsteilnehmern müssten also $n(n-1)/2$ Schlüssel ausgetauscht werden. Kommt ein neuer Partner hinzu, so muss dieser n neue Schlüssel generieren und diese sicher austauschen. Man erkennt somit unmittelbar, dass diese Vorgehensweise nur in Umgebungen mit einer geringen Anzahl von Teilnehmern praktikabel ist.

Verteilung

Um den Verteilungsaufwand zu reduzieren, führt man Schlüsselverteilungsdienste ein, die die Aufgabe haben, Schlüsselinformationen vertrauenswürdig zu verwalten und auf Anfrage von Teilnehmern Schlüssel für deren Kommunikation auszutauschen. Ein entsprechender Dienst sollte jedoch nicht durch eine zentrale Komponente in einem großen Netz erbracht werden, da dadurch die Gefahr eines Engpasses mit langen Wartezeiten entsteht. Zentrale Komponenten bilden ferner Schwachpunkte in Bezug auf die Zuverlässigkeit (single point of failure). Durch einen möglichen Ausfall der zentralen Komponente könnte die sichere Kommunikation generell infrage gestellt sein.

9.3.1 Schlüsselhierarchie

unterschiedliche Schlüssel

In einem IT-System werden Schlüssel für ganz unterschiedliche Zwecke eingesetzt, so dass entsprechend viele verschiedene Schlüssel erzeugt und verwaltet werden müssen. Beispiele sind kurze Sitzungsschlüssel, die nur zum gesicherten Transfer geringer Datenmengen erzeugt und danach wieder gelöscht werden, oder langlebige Schlüssel, mit deren Hilfe andere Schlüssel ausgetauscht werden. Die Schlüsselaustausch-Schlüssel, wie beispielsweise private Schlüssel eines asymmetrischen Kryptosystems, werden relativ selten genutzt, aber da deren Offenlegung die Sicherheit aller damit ausgetauschten Sitzungsschlüssel infrage stellt, müssen für deren Verwaltung hohe Sicherheitsanforderungen gewährleistet sein.

Benötigt werden weiterhin Signierschlüssel, die unter Umständen eine sehr lange Gültigkeit haben, da die Rechtsverbindlichkeit einer Unterschrift über lange Zeiträume gewährleistet werden muss. Das bedeutet, dass insbesondere lange Schlüssel (z.B. große RSA-Module) notwendig sind, um das Knacken des Signierschlüssels zu verhindern. Datenschlüssel, die zur Verschlüsselung persistenter Daten verwendet werden, besitzen in der Regel ebenfalls eine lange Gültigkeit, da das Wechseln von Schlüsseln hier sehr aufwändig und mit dem Entschlüsseln und erneuten Verschlüsseln der mit dem veralteten Schlüssel verschlüsselten Datenbestände verbunden wäre. An solche Datenschlüssel sind demnach sehr hohe Anforderungen an Maßnahmen zu deren gesicherten Aufbewahrung zu stellen. Zur Systematisierung und einfacheren Handhabung dieser Schlüsselvielfalt legt man Schlüsselklassen fest und organisiert diese hierarchisch.

Hierarchie von Kommunikationsschlüsseln

Ein wichtiger Einsatzbereich unterschiedlicher Schlüssel ist die sichere Kommunikation zwischen Benutzerprozessen auf vernetzten Rechnern. An Schlüsselklassen unterscheidet man zwischen den eigentlichen Sitzungsschlüsseln, den Schlüsseln, die zu deren Austausch eingesetzt werden, sowie den Basisschlüsseln (Master Keys, Pre-Shared Secrets), die eine initiale sichere Kommunikation ermöglichen. Wir wollen uns im Folgenden den Aufbau einer Schlüsselhierarchie anhand eines Kommunikationszenarios innerhalb eines Rechnernetzes etwas genauer ansehen.

sichere
Kommunikation

Gegeben sei ein Netz von Rechnern. Man fasst zunächst jeweils eine Menge von Rechnern zu einer Verwaltungsdomäne zusammen und führt für jede Domäne einen zentralen sowie für jeden Rechner einen lokalen Schlüsseldienst ein. Zwischen dem zentralen Server einer Domäne und jedem lokalen Server der gleichen Domäne sei ein Wurzel-Schlüssel, ein Master Key, sicher ausgetauscht. Die Master Keys bilden die Wurzel der Schlüsselhierarchie. Unter Verwendung der Wurzelschlüssel werden Serverschlüssel zwischen lokalen Servern einer Domäne ausgetauscht, die ihrerseits dazu benutzt werden, die Sitzungsschlüssel zwischen den einzelnen Prozessen auszutauschen.

Schlüsselhierarchie

Master Key

Beispiel 9.2 (WLAN-Schlüsselhierarchie)

Ein einfaches Beispiel einer solchen Schlüsselhierarchie findet man in dem WLAN-Standard 802.11i (vgl. Abschnitt 15.4). Das Ziel hierbei ist es, eine verschlüsselte Kommunikation zwischen einem Endgerät und einem WLAN-Zugangspunkt (Access Point) zu etablieren. Ausgangspunkt dafür ist ein Master-Key, der individuell zwischen Endgerät und Zugangspunkt auf sicherem Weg ausgetauscht wird. Aus diesem Master-Key wird ein zweiter Schlüssel abgeleitet, der Verbindungsschlüssel. Dieser dient wiederum als Basis, um einzelne Kommunikationsschlüssel zu generieren, wobei für jedes Paket, das übertragen wird, mit einem eigenen Schlüssel verschlüsselt wird⁸.

WLAN



Die Schlüssel der Hierarchie können individuell verwaltet und insbesondere nach Bedarf erneuert werden. Die Serverschlüssel werden in der Regel über einen längeren Zeitraum hinweg genutzt, da sie jeweils nur zum Verschlüsseln einer kurzen Nachricht, nämlich eines neu erzeugten Schlüssels, benötigt werden. Es ist selbstverständlich notwendig, diese Schlüssel in regelmäßigen Abständen, z.B. in Abhängigkeit vom Umfang der damit bereits verschlüsselten Datenmenge oder beim Verdacht, dass der Schlüssel kompromittiert sein könnte, zu erneuern. Dazu ist dann wieder auf den Master Key zurückzugreifen. Da Sitzungsschlüssel in der Regel mit einer hohen Frequenz und für größere Datenmengen verwendet werden, sollten diese

individuelle
Verwaltung

⁸ Dies ist wichtig, da die WLAN-Verschlüsselung auf einer Stromchiffre basiert.

nach einer erheblich kürzeren Zeitspanne erneuert werden. Im Kerberos Authentifikationsdienst Version 4 (vgl. Kapitel 10.4.2) sind Sitzungsschlüssel beispielsweise bis zu 21 Stunden gültig, während in der aktuellen Version 5 die Schlüssellebenszeit sogar fast unbeschränkt ist, da diese über einen Zeitstempel festlegbar ist, der eine maximale Gültigkeitsdauer bis zum 31.12.9999 haben kann. Diese u.U. ja sehr langen Gültigkeitsdauern sind auch einer der Kritikpunkte an diesem System.

Perfect Forward Secrecy (PFS)

Perfect Forward Secrecy

Eine wichtige Anforderungen an die Verfahren und Protokolle zur Schlüsselerneuerung besteht darin dafür zu sorgen, dass die Kenntnis eines Schlüssels, also dessen Aufdeckung, nicht dazu führt, dass damit auch vorherige und nachfolgende Schlüssel direkt aufdeckbar sind. Man fordert die Perfect Forward Secrecy. Um dies zu erreichen, darf ein neuer Schlüssel nicht von dem alten Schlüssel abhängen, d.h. es müssen stets neue, von einander unabhängige Schlüssel generiert werden. Beispielsweise garantiert das IPsec-Protokoll zum Aufbau einer sicheren Kommunikation in Netzen (vgl. Abschnitt 14.3) bei entsprechender Konfigurierung die Eigenschaft der Perfect Forward Secrecy. Auch im Standardprotokoll SSL/TLS für die Web-Sicherheit ist bei der Nutzung des Diffie-Hellman-Schlüsselaustausches Perfect Forward Secrecy umsetzbar, vgl. Abschnitt 14.4.4.

9.3.2 Naives Austauschprotokoll

naives Protokoll

Asymmetrische Verfahren sind gut geeignet, um geheime Sitzungsschlüssel sicher zwischen zwei Partnern Alice (A) und Bob (B) auszutauschen. Ein naives Protokoll könnte wie folgt aussehen

1. Partner Alice erzeugt den gemeinsamen Schlüssel $K_{A,B}$ und
2. verschlüsselt $K_{A,B}$ mit dem öffentlichen Schlüssel K_E^B des gewünschten Kommunikationspartners Bob
3. Partner Alice sendet diesen Kryptotext $C = E(K_{A,B}, K_E^B)$ an Partner Bob
4. Bob entschlüsselt den Kryptotext mit seinem geheimen Schlüssel, $K_{A,B} = D(C, K_D^B)$, und ist damit auch im Besitz des Sitzungsschlüssels $K_{A,B}$.

Sicherheitsprobleme

Dieses einfache Protokoll weist leider eine Fülle von Unzulänglichkeiten auf, so dass unautorisierte Modifikationen des Kryptotextes, Maskierungsangriffe oder Wiedereinspielungen nicht erkennbar sind und somit dem Missbrauch Tür und Tor geöffnet wird. Da der Schlüsselempfänger, hier Partner Bob, keine Möglichkeit besitzt zu prüfen, ob es sich bei dem

empfangenen Schlüssel $K_{A,B}$ tatsächlich um den ursprünglich von Alice generierten Schlüssel handelt, ist es für einen Angreifer X einfach, den Kryptotext C abzufangen und durch ein C' zu ersetzen, das einen von X erzeugten Schlüssel K' enthält. Gelingt es X nun auch noch dem Partner Alice diesen gefälschten Schlüssel unterzuschieben, so kann er in aller Ruhe die vertrauliche Kommunikation zwischen Alice und Bob verfolgen. Da in dem naiven Protokoll Alice und Bob sich nicht wechselseitig authentifizieren, kann der Angreifer X seinen Maskierungsangriff erfolgreich durchführen. Es handelt sich hierbei um einen klassischen Man-in-the-middle Angriff.

Man-in-the-Middle

Um diese Angriffe abzuwehren, muss man das naive Protokoll um Maßnahmen zur Berechnung von MACs oder signierten Hashwerten des auszutauschenden Sitzungsschlüssels ergänzen. Dies reicht jedoch immer noch nicht aus, um auch den Versuch einer Wiedereinspielung (Replay-Attack) erfolgreich abzuwehren. Hat ein Angreifer X den Sitzungsschlüssel $K_{A,B}$ „geknackt“ und das von Alice übertragene Nachrichtenpaket C abgefangen, so kann er dieses Paket zu einem späteren Zeitpunkt erneut einspielen. Bob glaubt, dass Alice mit ihm kommunizieren will, da er den MAC bzw. die digitale Signatur korrekt verifizieren konnte. Er wird also den Schlüssel $K_{A,B}$ erneut für die Kommunikation mit Alice verwenden. Der Angreifer X ist somit in der Lage, alle unter $K_{A,B}$ verschlüsselten Nachrichten zu entschlüsseln. Um auch Wiedereinspielungen erfolgreich abzuwehren, ist mit diesen Ausführungen klar, dass die Nachricht, die den signierten Sitzungsschlüssel enthält, auch noch einen Frischenachweis über den Schlüssel beinhalten muss. Dies kann beispielsweise über die Nutzung von Zeitstempeln bzw. Gültigkeitsdauern erzielt werden. Eine Ausprägung eines solchen Frische-Konzepts (freshness) findet man unter anderem in den beim Kerberos-Protokoll ausgetauschten bzw. genutzten Tickets (vgl. Abschnitt 10.4.2).

Angriffsabwehr?

Needham-Schroeder-Protokolle

Man sieht, dass die Entwicklung korrekter Schlüsselaustauschprotokolle sehr viel komplexer ist, als dies auf den ersten Blick erscheint. Im Folgenden werden Basisprotokolle für den Schlüsselaustausch erklärt, die so genannten Needham-Schroeder-Protokolle, die die Grundlage der meisten der in der Praxis eingesetzten Protokolle sind. Die grundlegenden Protokolle zum authentifizierten Schlüsselaustausch wurden bereits 1978 von R. Needham und M. Schroeder in [130] entwickelt. Es gibt mittlerweile eine Vielzahl von Varianten, die versuchen, die Anzahl der auszutauschenden Nachrichten zu minimieren, oder die Kommunikation unabhängig von einer zentralen Uhr im System zu halten. Wir beschränken uns hier jedoch auf die Basisprotokolle, da man an ihnen gut studieren kann, worauf bei einem korrekten Protokolldesign zu achten ist.

Basisprotokolle

Notation

Zur Beschreibung der Protokolle wird die nachfolgende Notation verwendet.

- Senden einer Nachricht M von A nach B

$$\begin{array}{ccc} \text{Absender} & \text{Empfänger} & \text{Nachricht } M \\ A & \longrightarrow & M = A, I_A \end{array}$$

- $\{M\}^{K_A}$ bedeutet, dass die Nachricht M mit dem Schlüssel K_A verschlüsselt wird, also $\{M\}^{K_A} \equiv E(M, K_A)$. Dies ist bei der Beschreibung von Protokollen eine übliche Notation, die zur kompakteren Darstellung dient.
- Mit I_A wird eine so genannte Nonce⁹ bezeichnet. Eine Nonce ist ein eindeutiger, noch nie zuvor verwendeter Identifikator, z.B. eine Zufallszahl.

9.3.3 Protokoll mit symmetrischen Verfahren

sichere Basis

Für alle betrachteten Protokolle wird eine vertrauenswürdige, sichere Basis benötigt. Grundlage dafür ist mindestens ein vertrauenswürdiger Authentifizierungs- bzw. Schlüsselverteilungsserver AS . Für die Basis wird vorausgesetzt, dass jeder Kommunikationsteilnehmer A einen geheimen Schlüssel K_A , seinen Master Key, mit dem Server AS vereinbart hat. K_A ist nur AS und dem Partner A bekannt.

An den Authentifizierungsserver sind besondere Anforderungen zu stellen: er muss vertrauenswürdig sein, d.h. seine Ist-Funktionalität entspricht (nachweisbar) seiner Soll-Funktionalität. Das bedeutet, dass der Server genau die im Folgenden beschriebenen Protokollschrifte durchführt und die geheimen Informationen, auf die er zugreifen kann (z.B. die geheimen Schlüssel), nicht an unberechtigte Dritte weiterreicht. Der Authentifizierungsserver muss darüber hinaus besonders vor Manipulationen geschützt werden.

Protokollablauf

symmetrisches
Protokoll

Möchte die Teilnehmerin Alice mit dem Teilnehmer Bob kommunizieren, so sendet Alice einen Klartext an AS , der ihre Identität, die Identität von Bob und eine Nonce I_A enthält:

$$(1) \text{Alice} \longrightarrow AS \quad \text{Alice, Bob, } I_A.$$

Nachdem AS die Nachricht (1) erhalten hat, generiert er einen neuen Sitzungsschlüssel $K_{A,B}$. AS sucht in seiner Schlüsseldatenbank nach den Masterschlüsseln K_A und K_B und schickt an Alice folgende Nachricht:

$$(2) AS \longrightarrow \text{Alice} \quad \{I_A, \text{Bob}, K_{A,B}, \{K_{A,B}, \text{Alice}\}^{K_B}\}^{K_A}.$$

⁹ Number used once

Durch die Verschlüsselung mit dem Schlüssel K_A ist sichergestellt, dass nur Alice die Nachricht entschlüsseln kann. Nur sie ist damit in der Lage, in den Besitz des generierten Sitzungsschlüssels $K_{A,B}$ zu gelangen. Da die Nachricht mit dem geheimen Masterschlüssel K_A verschlüsselt wurde, muss sie von dem Server AS stammen, da nur dieser neben Alice diesen Schlüssel kennt. Der Absender von Nachricht (2) ist also authentisch.

Da es sich aber auch um eine Wiedereinspielung handeln könnte, überprüft Alice nach der Entschlüsselung, ob die Nachricht (2) den korrekten Namen des Partners Bob und die richtige Nonce (hier I_A) enthält. Das heißt, sie prüft, ob sie Gefahr läuft, einen alten Sitzungsschlüssel $K_{A,B}$ noch einmal zu verwenden. Durch den Abgleich des Empfängernamens wird kontrolliert, ob ein Eindringling die Nachricht (1), die von Alice an AS versandt wurde, modifiziert hat, und einen anderen Empfänger, z.B. X an Stelle von Bob, eingefügt hat. Das heißt, Alice kann den Versuch einer Maskerade aufdecken. Sie wird nicht irrtümlich mit einem falschen Empfänger X , der sich für Bob ausgibt, kommunizieren.

Wiedereinspielung?

Im nächsten Schritt erhält der Teilnehmer Bob den Sitzungsschlüssel $K_{A,B}$ von Alice:

$K_{A,B}$ an Bob

$$(3) \text{Alice} \rightarrow \text{Bob} \quad \{K_{A,B}, \text{Alice}\}^{K_B}.$$

Da nur Bob und der Server AS den Masterschlüssel K_B kennen und die Nachricht ursprünglich vom Server AS in (2) verschlüsselt worden ist, kann nur Bob die Nachricht entschlüsseln. Insbesondere konnte diese Nachricht nicht von Alice unbemerkt manipuliert werden. Bob erhält nach der Entschlüsselung den neuen Sitzungsschlüssel und die Identität seines Kommunikationspartners Alice.

Was ist bis jetzt klar? Alice weiß, dass jede Nachricht, die mit $K_{A,B}$ verschlüsselt wird, von Bob stammen muss¹⁰, und dass Nachrichten, die sie mit $K_{A,B}$ verschlüsselt, von Bob verstanden werden. Bob kann jedoch nicht wissen, ob es sich bei der Nachricht (3) um eine Wiedereinspielung handelt. Um dies zu kontrollieren, generiert er eine Nonce I_B , verschlüsselt sie mit $K_{A,B}$ und sendet die Nachricht an Alice:

$$(4) \text{Bob} \rightarrow \text{Alice} \quad \{I_B\}^{K_{A,B}}.$$

Handshake

Bob erwartet von Alice eine Antwort, in der I_B mit einer vorher vereinbarten Funktion f modifiziert wird, z.B. decrementieren von I_B um 1:

$$(5) \text{Alice} \rightarrow \text{Bob} \quad \{f(I_B)\}^{K_{A,B}}.$$

¹⁰ Wie gesagt, gehen wir ja davon aus, dass der Server AS seine Kenntnis über den Schlüssel nicht missbraucht.

Nach dem Erhalt dieser Nachricht ist Bob sicher, dass Alice mittels des Schlüssels $K_{A,B}$ mit ihm kommunizieren möchte, da nur Alice in der Lage war, die Nachricht (4) zu entschlüsseln und auf den Klartext I_B die allgemein bekannte Funktion f anzuwenden. Die Teilnehmer sind authentifiziert und können den ausgetauschten Schlüssel nutzen. Den Austausch der Nachrichten (4) und (5) nennt man Handshake. Das Zusammenspiel der Partner und der Austausch der Nachrichten ist in Abbildung 9.5 skizziert.

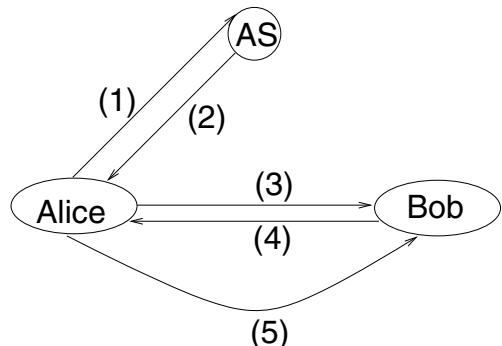


Abbildung 9.5: Schlüsselaustausch mit symmetrischen Verfahren

Probleme und Lösungsansätze

Sicherheitsproblem

Das vorgestellte Protokoll ist sicher, wenn man davon ausgeht, dass keiner der ausgetauschten Schlüssel $K_{A,B}$ einem Eindringling jemals bekannt sein kann. Da dies aber eine unrealistische Annahme ist, kann man sich das folgende Szenario vorstellen:

Ein Eindringling X hat den Sitzungsschlüssel $K_{A,B}$ „geknackt“, die Nachricht (3) aufgezeichnet und spielt diese wieder ein:

$$(3') X \longrightarrow \text{Bob} \quad \{K_{A,B}, \text{Alice}\}^{K_B}.$$

Bob glaubt, dass Alice mit ihm kommunizieren will, und beginnt den Handshake:

$$(4') \text{Bob} \longrightarrow \text{Alice} \quad \{I_B\}^{K_{A,B}}.$$

X fängt die Nachricht ab, entschlüsselt sie mit dem ihm bekannten Schlüssel $K_{A,B}$ und antwortet, indem er sich als Alice maskiert:

$$(5') X \longrightarrow \text{Bob} \quad \{f(I_B)\}^{K_{A,B}}.$$

X kann nun Nachrichten von Bob, die mit $K_{A,B}$ verschlüsselt sind, entschlüsseln und selber Nachrichten an Bob schicken, wobei Bob glaubt, dass der Absender der Nachricht Alice ist.

Das Problem kann man lösen (vgl. [52]), wenn man die Nachrichten mit einer Uhrzeit T (engl. *timestamp*) versieht. Der Empfänger einer Nachricht

muss die in der Nachricht protokolierte Zeitmarke mit seiner lokalen Uhrzeit abgleichen und entscheiden, ob es sich um eine veraltete Nachricht handelt. Das um Timestamps erweiterte Protokoll sieht dann wie folgt aus:

- (1') Alice $\rightarrow AS$ Alice, Bob
- (2') $AS \rightarrow Alice$ $\{Bob, K_{A,B}, T, \{Alice, K_{A,B}, T\}^{K_B}\}^{K_A}$
- (3') Alice $\rightarrow Bob$ $\{Alice, K_{A,B}, T\}^{K_B}$

Problem
Problematisch an dieser Lösung ist, dass die lokalen Uhren der am Protokoll beteiligten Rechner synchronisiert sein müssen, da von einer globalen Systemzeit ausgegangen wird. Falls es einem Angreifer gelingt, über einen manipulierten Zeitdienst oder durch direkte Eingriffe die lokale Uhr eines Zielrechners zurückzustellen, dann ist auf diesem Rechner eine unentdeckbare Wiedereinspielung möglich.

Das Problem, die Wiedereinspielung eines Schlüssels zu erkennen, kann man auch über die Nutzung von Nonces (vgl. [131]) lösen, die vor den angegebenen Protokollschriften zwischen den Partnern ausgetauscht werden.

- (0.1) Alice $\rightarrow Bob$ Alice
- (0.2) Bob $\rightarrow Alice$ $\{Alice, J\}^{K_B}$

Im ersten, eigentlichen Protokollschrift muss Alice dann die vereinbarteNonce J zusammen mit den anderen Informationen an AS senden:

- (1') A $\rightarrow AS$ Alice, Bob, $I_A, \{Alice, J\}^{K_B}$.

AS überprüft, ob die beiden Namen, die zum einen im Klartext und zum anderen unter dem Schlüssel K_B verschlüsselt übertragen wurden, identisch sind.

- (2') $AS \rightarrow Alice$ $\{I_A, Bob, K_{A,B}, \{K_{A,B}, Alice, J\}^{K_B}\}^{K_A}$.
- (3') Alice $\rightarrow Bob$ $\{K_{A,B}, Alice, J\}^{K_B}$.

Bob kann die in der Nachricht enthaltene Nonce J mit seiner ursprünglich versandten abgleichen.

9.3.4 Protokoll mit asymmetrischen Verfahren

Jeder Teilnehmer A sowie der Server AS besitzen ein Schlüsselpaar, bestehend aus einem geheimen Schlüssel K_D^A und einem öffentlichen Schlüssel K_E^A . Wir gehen davon aus, dass jeder Teilnehmer seinen öffentlichen Schlüssel beim Server AS authentisch, z.B. unter Verwendung von Zertifikaten, registriert hat, und dass jeder Teilnehmer den authentischen, öffentlichen Schlüssel des Servers AS kennt.

sichere Basis

Protokollablauf

Möchte nun wieder Teilnehmerin Alice (A) mit dem Teilnehmer Bob (B) kommunizieren, so sendet sie einen Klartext an AS , in der sie ihre eigene

asymmetrisches
Protokoll

Identität und die Identität von Bob angibt. Mit dieser Nachricht erfragt Alice beim AS den öffentlichen Schlüssel von Bob:

$$(1) A \rightarrow AS \quad A, B.$$

Der Server sucht in seiner Schlüsseldatenbank nach dem öffentlichen Schlüssel K_E^B von B und erzeugt eine mit seinem geheimen Schlüssel K_D^{AS} signierte Nachricht:

$$(2) AS \rightarrow A \quad \{K_E^B, B\}^{K_D^{AS}}.$$

Da Alice den öffentlichen Schlüssel des Servers kennt, kann sie die Nachricht (2) entschlüsseln und den öffentlichen Schlüssel von Bob lesen. Die Verschlüsselung in (2) dient somit nicht der Geheimhaltung, sondern dazu, sicherzustellen, dass die Nachricht vom Server AS stammt und der öffentliche Schlüssel von Bob, also K_E^B , authentisch ist. Über den in der Nachricht enthaltenen Namen B kann Alice prüfen, dass sie nicht eine wiedereingespielte Nachricht benutzt, die zwar ursprünglich auch vom AS stammte, aber den öffentlichen Schlüssel eines anderen Teilnehmers beinhaltet.

K_E^B an A

Kommunikationswunsch

Alice schickt an Bob eine Nachricht bestehend aus ihrer Identität und einer Nonce I_A verschlüsselt mit dem öffentlichen Schlüssel von Bob:

$$(3) A \rightarrow B \quad \{I_A, A\}^{K_E^B}.$$

Bob entschlüsselt die Nachricht (3) mit seinem geheimen Schlüssel K_D^B , stellt fest, dass ein Teilnehmer mit der Identität A mit ihm kommunizieren will und besorgt sich daraufhin den öffentlichen Schlüssel von Alice bei der vertrauenswürdigen Instanz AS :

$$(4) B \rightarrow AS \quad B, A.$$

$$(5) AS \rightarrow B \quad \{K_E^A, A\}^{K_D^{AS}}.$$

doppelter Handshake

Um einen möglichen Maskierungsangriff aufzudecken, wird ein doppelter Handshake durchgeführt.

$$(6) B \rightarrow A \quad \{I_A, I_{B'}\}^{K_E^A}.$$

Alice überprüft, ob die Nonce I_A mit derjenigen übereinstimmt, die sie im Schritt (3) an Bob übermittelt hat. Anschließend schickt sie die Nonce von Bob, also $I_{B'}$, die B für den Handshake erzeugt hat, mit dem öffentlichen Schlüssel von Bob verschlüsselt zurück.

$$(7) A \rightarrow B \quad \{I_{B'}\}^{K_E^B}.$$

Bob überprüft die Nonce. Nur Alice konnte die Nachricht (6) entschlüsseln und in den Besitz der Nonce $I_{B'}$ gelangen. Danach sind die Partner authentifiziert. Das Zusammenspiel der Partner und der Austausch der Nachrichten ist in Abbildung 9.6 skizziert.

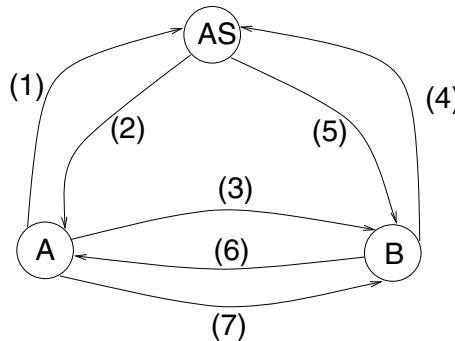


Abbildung 9.6: Wechselseitige Authentifikation mit asymmetrischen Verfahren

Schlüsselaustausch

Die Partner können jetzt einen geheimen Schlüssel eines symmetrischen Verfahrens austauschen. Dazu generiert einer von beiden, hier z.B. Alice, einen Sitzungsschlüssel $K_{A,B}$ (z.B. einen AES-Schlüssel), signiert die Nachricht, die den Schlüssel enthält, mit ihrem geheimen Schlüssel K_D^A und verschlüsselt die signierte Nachricht zusammen mit ihrer Identität A mit dem öffentlichen Schlüssel von Bob:

$$(8) \quad A \longrightarrow B \quad \{\{K_{A,B}\}^{K_D^A}, A\}^{K_E^B}.$$

Bob und nur Bob kann die Nachricht mit seinem geheimen Schlüssel K_D^B entschlüsseln. Er verifiziert die digitale Signatur mit dem ihm bekannten öffentlichen Schlüssel von Alice und ist damit ebenfalls im Besitz des geheimen Schlüssels $K_{A,B}$.

Da der öffentliche Schlüssel des Partners Bob zum Verschlüsseln von Nachrichten verwendet wurde, weist das Protokoll in diesem Schritt noch eine Lücke auf. Bob kann nämlich nicht überprüfen, ob die Nachricht wirklich frisch von Alice stammt und nicht etwa eine Wiedereinspielung ist. Um entsprechende Angriffsversuche abzuwehren, sind zum einen die Namen der Kommunikationspartner verschlüsselt mitzutragen und die Nachrichten sind mit Zeitstempeln T_A oder Sequenznummern zu versehen.

Die verbesserte Version von Schritt 8 könnte wie folgt aussehen:

$$(8') \quad A \longrightarrow B \quad \{\{A, B, T_A, K_{A,B}\}^{K_D^A}, A\}^{K_E^B}.$$

9.3.5 Leitlinien für die Protokollentwicklung

Die beiden Protokolle zum Schlüsselaustausch wurden etwas ausführlicher dargestellt, um ein grundlegendes Verständnis für die Problematik der Protokollentwicklung zu vermitteln. Dies erleichtert es, die im Folgenden diskutierten einfachen Leitlinien zur Entwicklung kryptografischer Protokolle, die auf Arbeiten von Abadi und Needham [3] basieren, zu verstehen und nachzuvollziehen.

Schlüsselaustausch

vollständige
Information

Die erste zu beachtende Regel beim Protokollentwurf betrifft die Frage nach dem Umfang der zwischen den Kommunikationspartnern in einem Protokollschnitt auszutauschenden Information. Die in den diskutierten Protokollen verwendeten Nachrichten waren vielfach nicht selbsterklärend, sondern bedurften eines ausführlichen Kommentars. Damit es nicht zu falschen Interpretationen kommen kann, die von einem Angreifer gezielt hervorgerufen werden, sollten alle relevanten Informationen in einer Protokollnachricht codiert werden. So lehrt der Protokollschnitt (8) im obigen asymmetrischen Protokoll, dass insbesondere auch die explizite Angabe des Namens der jeweiligen Kommunikationspartner Bestandteil der verschlüsselten Nachrichten sein sollte.

Verschlüsselungs-
zweck

In Protokollen werden Verschlüsselungen zu sehr unterschiedlichen Zwecken verwendet, so dass man sich bei einer Protokollentwicklung in jedem Schritt fragen muss, was genau mit der Verschlüsselung erreicht werden soll. Das heißt, es ist zu klären, ob Vertraulichkeit erzielt, die Authentizität bewiesen oder ob damit nur unterschiedliche Bestandteile einer Nachricht miteinander verknüpft werden sollen. In letzterem Fall sollte man besser eine digitale Signatur verwenden.

Doppel-
verschlüsselung

Durch den fehlerhaften Einsatz von mehrfachen Verschlüsselungen besteht die Gefahr, Redundanz zu erzeugen und damit unnötige Kosten zu verursachen. Ein Beispiel für eine redundante Doppelverschlüsselung ist in Nachricht (2) des symmetrischen Protokolls zu sehen. Das folgende Beispiel verdeutlicht, dass zu viele Verschlüsselungsschritte unter speziellen Randbedingungen sogar Sicherheitsprobleme verursachen können.

Beispiel 9.3 (Sicherheitslücke durch Doppelverschlüsselung)

Gegeben seien zwei Partner A und B , die unter Nutzung eines asymmetrischen Verfahrens eine vertrauliche Information M austauschen möchten. Die entsprechenden Protokollschnitte seien:

$$\begin{aligned} (1) \quad A \rightarrow B & \quad A, \{M, A\}^{K_E^B}, B \\ (2) \quad B \rightarrow A & \quad B, \{M, B\}^{K_E^A}, A. \end{aligned}$$

Führt man nun aufgrund eines falsch verstandenen Sicherheitsbedürfnisses eine doppelte Verschlüsselung durch, so erhält man:

$$\begin{aligned} (1') \quad A \rightarrow B & \quad A, \{\{M\}^{K_E^B}, A\}^{K_E^B}, B \\ (2') \quad B \rightarrow A & \quad B, \boxed{\{\{M\}^{K_E^A}, B\}^{K_E^A}}, A. \end{aligned}$$

Betrachten wir einen Angreifer X , der die Nachricht (2') abfängt, sie mit seinem Namen als Absender versieht, den umrandeten Teil übernimmt, die notwendigen Verschlüsselungen vornimmt und das Paket an A sendet:

$$(1') \quad X \rightarrow A \quad X, \boxed{\{\{M\}^{K_E^A}, B\}^{K_E^A}}, X \}^{K_E^A}, A.$$

A antwortet protokollgemäß mit:

$$(2') A \longrightarrow X \quad A, \{\{\{M\}^{K_E^A}, B\}^{K_E^X}, A\}^{K_E^X}, X.$$

X sendet nun:

$$(1'') X \longrightarrow A \quad X, \{\{M\}^{K_E^A}, X\}^{K_E^A}, A$$

und A antwortet wieder mit:

$$(2'') A \longrightarrow X \quad A, \{\{M\}^{K_E^X}, A\}^{K_E^X}, A$$

X ist am Ziel und kann mit seinem privaten Schlüssel K_D^X die vertrauliche Nachricht M entschlüsseln!



Die dritte Regel betrifft den Einsatz digitaler Signaturen. Falls in einem Protokollschnitt eine bereits verschlüsselte Nachricht signiert wird, so ist nicht unbedingt davon auszugehen, dass der Signierer über den wahren Inhalt der Nachricht informiert ist. Soll der entsprechende Protokollschnitt die Urheberschaft des Absenders der Nachricht zweifelsfrei nachweisen, so ist zunächst die unverschlüsselte Nachricht bzw. deren Hashwert zu signieren, bevor die Nachricht verschlüsselt übermittelt wird.

Signieren

Beispiel 9.4 (CCITT X.509)

Im CCITT X.509 Protokoll, mit dem signierte, verschlüsselte Nachrichten ausgetauscht werden sollen, ergibt sich aus einer falsch eingesetzten Signatur ein Sicherheitsproblem. Wir betrachten dazu einen vereinfachten Auszug aus dem Protokoll und beschränken uns auf die wesentliche Nachricht, mit der der Absender A sich selbst als Urheber der verschlüsselten Nachricht $\{Y_A\}^{K_E^B}$ ausweisen möchte.

$$A \longrightarrow B \quad A, \{T_A, N_A, B, X_A, \{Y_A\}^{K_E^B}\}^{K_D^A}.$$

Ein Angreifer X kann die Signatur von A entfernen und das verschlüsselte Y_A mit seinem eigenen Schlüssel signieren und diese modifizierte Nachricht an B senden.

$$X \longrightarrow B \quad X, \{T_A, N_A, B, X_A, \{Y_A\}^{K_B}\}^{K_D^X}.$$

B hält nun X für den Urheber von $\{Y_A\}^{K_E^B}$. Mögliche Konsequenzen kann sich der Leser leicht ausmalen.

Abhilfe schafft man in diesem Fall sehr einfach, indem man vor dem Verschlüsseln signiert:

$$A \longrightarrow B \quad A, \{T_A, N_A, B, X_A, \{\{Y_A\}^{K_D^A}\}^{K_E^B}\}^{K_D^A}.$$



Eine weitere Regel betrifft die Aktualität ausgetauschter Sitzungsschlüssel. Die Verwendung eines Schlüssel in einer aktuell erhaltenen Nachricht sagt

Sicherheitsproblem

frischer Schlüssel

nämlich nichts darüber aus, ob der Schlüssel selber frisch ist oder ob er schon eine sehr lange Lebenszeit hinter sich hat und evtl. sogar schon kompromittiert ist. Diesem Problem sind wir bereits beim Needham-Schroeder Protokoll mit symmetrischen Verfahren begegnet. Der dort in Nachricht (2) ausgetauschte Sitzungs-Schlüssel konnte eine Wiedereinspielung sein. Soll die Frische eines Schlüssels vom Empfänger überprüfbar sein, so muss über Zeitstempel oder Nonces ein Frischenachweis in den Protokollschrift integriert werden.

formale Analyse

Auch wenn man die genannten Leitlinien beim Protokollentwurf beherzigt, so sind damit nur einige häufig auftretende Fehlerquellen behandelt. Um ein größeres Vertrauen in die Korrektheit eines Protokollentwurfs zu erhalten, ist der Einsatz formaler Modellierungs- und Analysetechniken unabdingbar.

9.3.6 Diffie-Hellman Verfahren

Von Diffie und Hellman [55] wurde bereits 1976 das nach ihnen benannte Diffie-Hellman Verfahren (DH) zur sicheren Schlüsselvereinbarung vorgestellt. Das DH-Verfahren unterscheidet sich von den bislang erklärten Protokollen grundlegend darin, dass jeder Kommunikationspartner einen gemeinsamen, geheimen Sitzungsschlüssel berechnet. Das bedeutet, dass für die Schlüsselvereinbarung dieser geheime Schlüssel nicht über ein unsicheres Medium transportiert werden muss, sondern unter Nutzung öffentlich bekannter sowie privater, geheimer Informationen von den Kommunikationspartnern dezentral berechnet wird.

Zu beachten ist jedoch, dass im Gegensatz zu den in den vorherigen Abschnitt vorgestellten Needham-Schroeder-Protokollen die Kommunikationspartner durch das DH-Verfahren nicht authentifiziert werden. Zu beachten ist weiterhin, dass das DH-Verfahren, obwohl es auch private und öffentliche Schlüsselpaare verwendet, nicht zum Ver- und Entschlüsseln verwendbar ist, also kein asymmetrisches Kryptosystem ist. Die Funktionalität des Diffie-Hellman-Verfahrens besteht somit allein darin, einen gemeinsamen Schlüssel dezentral zu vereinbaren.

Mathematische Grundlagen

Das Verfahren beruht auf dem diskreten Logarithmusproblem und die arithmetischen Operationen des Protokolls werden auf einem endlichen Galois-Feld über einer großen Primzahl q durchgeführt. Zur Berechnung des gemeinsamen Schlüssels benötigt man ein allen bekanntes Element aus der Trägermenge des Galois-Feldes, mit dem alle anderen Elemente erzeugbar sind. Das heißt, man benötigt eine primitive Einheitswurzel.

*dezentrale
Berechnung*

Aufgaben

Grundlage

Definition 9.1 (Primitive Einheitswurzel)

Gegeben sei eine Primzahl q und die Trägermenge $M = \{0, \dots, q - 1\}$ von $GF(q)$. Ein Element $a \in M$ heißt primitive Einheitswurzel, wenn gilt:

primitive
Einheitswurzel

$$\begin{aligned} \forall m \in \{1, \dots, q - 1\} : \exists p \in \{0, \dots, q - 1\}, \text{ so dass} \\ m \equiv a^p \bmod q. \end{aligned}$$

□

Eine ganze Zahl a einer Trägermenge M eines Galois-Feldes ist somit eine primitive Einheitswurzel der Menge $M' = M \setminus \{0\}$, wenn es für jedes Element von M' einen Exponenten gibt, so dass das Element durch die primitive Einheitswurzel und den Exponenten erzeugt wird. Primitive Einheitswurzeln sind somit Generatoren für die Elemente des Galois-Feldes.

Beispiel 9.5 (Primitive Einheitswurzeln)

Gegeben sei die Primzahl $q = 11$ und damit die Menge $M = \{0, \dots, 10\}$ des von q erzeugten Galois-Feldes $GF(q)$. Die primitiven Einheitswurzeln von M sind 2, 6, 7 und 8 und nur diese. Das Element 3 ist beispielsweise kein Generator, weil es keinen Exponenten $p \in \{0, \dots, 10\}$ gibt, der die Gleichung

$GF(11)$

$$3^p \equiv 2 \bmod 11$$

erfüllt. Das heißt, dass mit der 3 das Element 2 des Feldes $GF(11)$ nicht erzeugt werden kann.

Die Generatoreigenschaft der Einheitswurzel 2 zeigen wir, indem wir die Elemente der Menge M , also des Galois-Feldes $GF(11)$ vollständig erzeugen:

$$\begin{array}{lll} 2^{10} = 1024 \equiv 1 \bmod 11 & 2^1 = 2 \equiv 2 \bmod 11 & 2^8 = 256 \equiv 3 \bmod 11 \\ 2^2 = 4 \equiv 4 \bmod 11 & 2^4 = 16 \equiv 5 \bmod 11 & 2^9 = 512 \equiv 6 \bmod 11 \\ 2^7 = 128 \equiv 7 \bmod 11 & 2^3 = 8 \equiv 8 \bmod 11 & 2^6 = 64 \equiv 9 \bmod 11 \\ 2^5 = 32 \equiv 10 \bmod 11 & & \end{array}$$

▲

Das Bestimmen von primitiven Einheitswurzeln für ein Galois-Feld $GF(q)$ ist nicht schwierig, da man zeigen kann, dass es für eine zyklische Gruppe der Ordnung n mindestens $\lceil n/(6 \ln \ln n) \rceil$ Generatoren gibt. Wenn gilt, $n = 2q$ für eine Primzahl q , dann gibt es sogar $q - 1$ Generatoren. Das heißt, dass fast die Hälfte aller Gruppenelemente die Gruppe erzeugen, so dass man bei einer zufälligen Wahl eines Elementes $a \in GF(q)$ eine gute Chance besitzt, eine primitive Einheitswurzel zu finden.

Generator-
bestimmung

Problematischer ist jedoch die tatsächliche Verifikation der Generatoreigenschaften eines Elementes a . Es gibt dafür ein effizientes Verfahren, wenn die

Generator-
verifikation

Zahl $q - 1$ faktorisierbar ist. Falls sogar $q - 1 = 2r$ gilt, wobei r eine Primzahl ist, so vereinfacht sich die Verifikation auf zwei Kongruenzüberprüfungen. Falls nämlich gilt:

$$a^2 \not\equiv 1 \pmod{q} \text{ und } a^r \not\equiv 1 \pmod{q},$$

dann ist a eine primitive Einheitswurzel.

Beispiel 9.6 (Primitive Einheitswurzel (Forts.))

Betrachten wir noch einmal das Beispiel 9.5 mit der Primzahl $q = 11$. Es gilt $q - 1 = 2 \cdot 5$, also $r = 5$. Um zu prüfen, ob 3 eine primitive Einheitswurzel ist, muss man also verifizieren, dass gilt:

$$3^2 \not\equiv 1 \pmod{11} \text{ und } 3^5 \not\equiv 1 \pmod{11}$$

Dies gilt nicht, da $3^5 \equiv 1 \pmod{11}$, also ist 3 keine primitive Einheitswurzel für $GF(11)$.



Mit den erklärten mathematischen Grundlagen sind wir nun in der Lage, das Diffie-Hellman Verfahren (DH) zur Schlüsselvereinbarung einzuführen.

DH-Verfahren

gemeinsame Basis

Informationen, die allen Kommunikationsteilnehmern bekannt sein müssen und die nicht gegenüber Dritten geheim zu halten sind, bilden die gemeinsame Basis des DH-Verfahrens.

Gemeinsame, nicht geheime Basisinformationen:

1. Es ist eine große Primzahl q zu wählen, die das Galois-Feld $GF(q)$ bestimmt.
2. Es ist eine primitive Einheitswurzel a , $2 \leq a \leq q - 2$, von $GF(q)$ zu bestimmen, so dass jedes Element des Feldes $GF(q)$ erzeugbar ist.

Die Primzahl q und der Wert a sind die öffentlich bekannten Diffie-Hellman-Parameter.

Dezentrale Berechnung eines Sitzungsschlüssels

Schlüsselpaar

1. Jeder Teilnehmer i wählt eine Zufallszahl $1 \leq X_i \leq q - 1$.
 X_i ist der geheime Schlüssel von Teilnehmer i . Der Schlüssel X_i entspricht dem privaten Schlüssel eines asymmetrischen Kryptosystems.
2. Teilnehmer i berechnet

$$Y_i = a^{X_i} \pmod{q}.$$

Y_i ist der öffentliche Schlüssel von Teilnehmer i .

Die öffentlichen Schlüssel werden in einer Schlüsseldatenbank abgelegt oder bei Bedarf potentiellen Kommunikationspartnern mitgeteilt.

3. Möchte Teilnehmer i mit dem Teilnehmer j kommunizieren, so berechnen jeder der Teilnehmer einen Sitzungsschlüssel $K_{i,j}$ bzw. $K_{j,i}$, wobei durch die Berechnungsvorschrift sichergestellt ist, dass gilt:

$$K_{i,j} = K_{j,i} = a^{X_i X_j} \bmod q.$$

Zur Berechnung von $K_{i,j}$ benötigt i den öffentlichen Schlüssel Y_j von j . Damit berechnet er:

$$K_{i,j} = Y_j^{X_i} \bmod q.$$

Der Teilnehmer j berechnet den Schlüssel $K_{j,i}$ analog mit dem öffentlichen Schlüssel Y_i von i :

$$K_{j,i} = Y_i^{X_j} \bmod q.$$

Zusammen gilt dann:

$$K_{i,j} = (a^{X_j})^{X_i} \bmod q = a^{X_j X_i} \bmod q = a^{X_i X_j} \bmod q = K_{j,i}.$$

Nach dem Protokollschrift (3) haben also beide Kommunikationspartner unabhängig voneinander einen gemeinsamen, geheimen Schlüssel berechnet, den sie für ihre vertrauliche Kommunikation verwenden können.

Beispiel 9.7 (Diffie-Hellman-Schlüsselvereinbarung)

Die beiden Kommunikationsteilnehmer Alice (A) und Bob (B) möchten einen gemeinsamen, geheimen Sitzungsschlüssel vereinbaren. Sie verständigen sich vorab über die zu verwendende Primzahl q , hier $q = 11$, sowie über die primitive Einheitswurzel a , hier $a = 2$. Die Erzeugungseigenschaft der Einheitswurzel $a = 2$ für das Galois-Feld $GF(11)$ ist in Beispiel 9.5 bereits gezeigt worden.

Alice wählt ihren geheimen Schlüssel X_A , hier $X_A = 9$, berechnet damit ihren öffentlichen Schlüssel $A = 2^9 = 512 \equiv 6 \bmod 11$ und sendet $Y_A = 6$ an Bob.

X_A, Y_A

Bob wählt seinerseits seinen privaten Schlüssel X_B , hier $X_B = 5$, berechnet seinen öffentlichen Schlüssel $Y_B = 2^5 = 32 \equiv 10 \bmod 11$ und sendet $Y_B = 10$ an Alice.

X_B, Y_B

Alice berechnet den gemeinsamen Schlüssel $K_{A,B}$:

$$Y_B^{X_A} \bmod 11 = Y_B^9 \bmod 11 = 2^{5 \cdot 9} \bmod 11 = 10 = K_{A,B}.$$

Unabhängig von Alice berechnet auch Bob den gemeinsamen Schlüssel:

$$Y_A^{X_B} \bmod 11 = Y_A^5 \bmod 11 = 2^{9 \cdot 5} \bmod 11 = 10 = K_{A,B}.$$



Sicherheit

DH-Problem

Ein Angreifer kennt die Primzahl q , die primitive Einheitswurzel a sowie die öffentlichen Schlüssel der Kommunikationspartner i und j , also Y_i und Y_j . Es ist jedoch für ihn sehr schwierig, allein mit dieser ihm zugänglichen Information den Sitzungsschlüssel $K_{i,j}$ der Partner zu berechnen, da er keine Kenntnisse über die geheimen Schlüssel X_i bzw. X_j besitzt. Ohne diese Zusatzinformationen muss ein Angreifer das diskrete Logarithmusproblem

$$X_i = \log_a Y_i \bmod q$$

lösen, um den Schlüssel $K_{i,j} = Y_i^{\log_a Y_j} \bmod q$ zu berechnen.

Berechnungs-aufwand

Die Berechnung des Sitzungsschlüssels basiert somit auf einer Einwegfunktion mit Falltür, wobei für jeden Teilnehmer der eigene geheime Schlüssel die benötigte geheime Zusatzinformation darstellt. Was bedeutet dies nun für den zu leistenden Berechnungsaufwand? Sei $q \in \mathbb{Z}$ und $b \in \mathbb{N}$. Wenn für die verwendete Primzahl q gilt, $q < b$, dann lassen sich alle Werte als b -Bit Binärzahlen darstellen. Die Exponentiation erfordert maximal $2b$ Multiplikationen modulo q . Demgegenüber benötigt die Berechnung des Logarithmus $q^{1/2} = 2^{b/2}$ Operationen. Das bedeutet, dass der Aufwand des Angreifers exponentiell zum Aufwand des berechtigten Benutzers steigt. Sei beispielsweise $b = 200$, dann benötigt man 400 Multiplikationen zur Berechnung von Y_i aus X_i . Die Berechnung des Logarithmus erfordert dagegen einen Aufwand von $2^{100} \approx 10^{30}$ Operationen.

Man beachte aber, dass die Berechnung des diskreten Logarithmus das bislang einzige bekannte allgemeine Verfahren ist, um das Diffie-Hellman Problem zu lösen, also die geheimen Sitzungsschlüssel zu bestimmen. Es ist jedoch nicht bewiesen, dass es tatsächlich die einzige Methode ist, um dieses Problem zu lösen.

Man-in-the-Middle-Angriff

Man-in-the-Middle

Wie eingangs bereits erwähnt, beinhaltet das DH-Verfahren keine Vorehrungen zur Überprüfung der Authentizität der Kommunikationspartner. Dadurch ist das Protokoll Bedrohungen durch Man-in-the-Middle-Angriffen ausgesetzt. Zur Beschreibung eines solchen Angriffs greifen wir erneut auf das Beispiel 9.7 zurück, also die Berechnung eines gemeinsamen Sitzungsschlüssels $K_{A,B} = 10$ durch die Partner Alice und Bob. Beide vertrauen darauf, dass sie miteinander kommunizieren und verwenden diesen Schlüssel, um sensible Informationen auszutauschen.

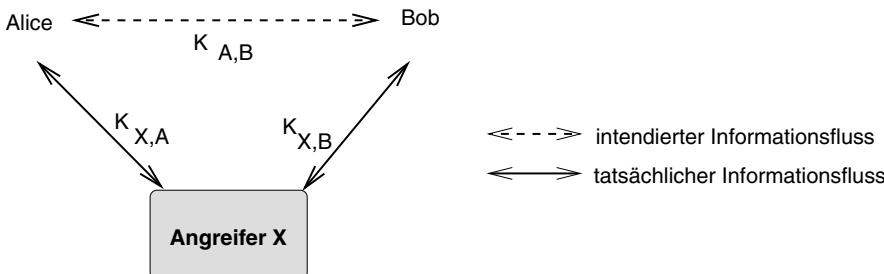


Abbildung 9.7: Man-in-the-Middle-Angriff auf das DH-Verfahren

Betrachten wir nun aber einen Angreifer X, der es geschafft hat, sich gegenüber Bob als Alice und gegenüber Alice als Bob auszugeben und mit beiden jeweils einen Schlüssel $K_{X,B}$ bzw. $K_{X,A}$ zu vereinbaren. Dann ergibt sich die in Abbildung 9.7 skizzierte Situation, in der Bob glaubt, den Schlüssel $K_{A,B} = 10$ zu verwenden, in Wahrheit aber seine Daten mit dem Schlüssel $K_{X,B}$ verschlüsselt. X fängt diese Nachrichten, die ja eigentlich für Alice bestimmt sind, ab, entschlüsselt sie, verschlüsselt sie mit dem mit Alice vereinbarten Schlüssel $K_{X,A}$ und sendet sie an Alice. Alice kann ebenfalls nicht erkennen, dass sie gar nicht mit $K_{A,B}$ sondern mit $K_{X,A}$ arbeitet. Weder Alice noch Bob sind in der Lage, das Umleiten ihrer Nachrichten über den Angreifer X zu erkennen. Der Angreifer erlangt auf diese Weise Zugriff auf den Klartext und kann den beiden darüber hinaus beliebige Informationen „unterschieben“. Zur Abwehr von Man-in-the-Middle-Angriffen muss deshalb auch die Authentizität der Kommunikationspartner überprüft werden. Hierzu werden in der Regel Zertifikate, die die Vertrauenswürdigkeit der verwendeten öffentlichen DH-Schlüssel attestieren, zusammen mit einem asymmetrischen Challenge-Response Handshake verwendet.

Angriff

Abwehr

authentifiziertes
DH

Das Station-to-Station (STS) Protokoll ([56]) ist ein Schlüsselaustausch-Protokoll basierend auf dem Original DH-Verfahren, das zusätzlich eine wechselseitige Authentifizierung der Partner auf der Basis digitaler Signaturen durchführt. Die zugrunde liegende Idee ist sehr einfach: Bevor zwei Parteien Alice (A) und Bob (B) einen DH-Schlüsselaustausch durchführen, benötigen sie jeweils ein asymmetrisches Schlüsselpaar, bestehend aus einem Signier- und einem zugehörigen Verifikationsschlüssel, sowie ein Zertifikat für ihren jeweiligen öffentlichen Schlüssel. In Erweiterung des ursprünglichen DH-Protokolls signieren beide Parteien die von ihnen jeweils übertragenen öffentlichen DH-Schlüssel $Y_A = a^{X_A} \text{mod } q$, bzw. $Y_B = a^{X_B} \text{mod } q$ mit ihren jeweiligen privaten Signaturschlüsseln. Mit den ebenfalls übertragenen Zertifikaten sind die Partner in der Lage, die Signatur des jeweils anderen zu verifizieren und damit die Authentizität des Partners zu prüfen. Aus Sicht des Partners Alice ergibt sich damit

1. Signieren des eigenen DH-Schlüssels:

$$Sig_A = Sig(Y_A, K_D^{Alice}),$$

2. Verifizieren des empfangenen, signierten DH-Schlüssels von Bob:

$$Y_B = Verif(Sig_B, K_E^{Bob}).$$

Diffie-Hellman auf elliptischen Kurven (ECDH)

ECDH

Elliptische Kurven (vgl. Abschnitt 7.7) werden häufig zur Schlüsselvereinbarung mittels des Diffie-Hellman Verfahrens verwendet. Dazu müssen sich die beteiligten Parteien zunächst über die gemeinsamen Domänen-Parameter einigen. Bei ECDH sind dies eine Primzahl p , die zu nutzende elliptische Kurve E über dem Galoisfeld $GF(p)$ und ein Basispunkt P auf der Kurve. Die Schlüsselvereinbarung erfolgt dann analog zu den Schritten der klassischen Variante. Seien wie immer Alice (A) und Bob (B) die beiden Partner, zwischen denen ein ECDH-basierter Schlüssel vereinbart werden soll.

- Beide Partner Alice (A) und Bob (B) wählen ihre jeweiligen privaten Schlüssel, also jeweils eine große ganze Zahl $a \in \{2, 3, \dots, |E|\}$ bzw. $b \in \{2, 3, \dots, |E|\}$.
- Mit diesem skalaren Wert berechnen beide Partner ihren jeweiligen öffentlichen Schlüssel: $Q_A = aP$, bzw. $Q_B = bP$.
- Beide Partner tauschen ihre öffentlichen Schlüssel Q_A, Q_B wechselseitig aus.
- Beide Partner berechnen den gemeinsamen ECDH-Punkt R :

$$\begin{aligned} A \text{ berechnet: } R &= aQ_B \\ B \text{ berechnet: } R &= bQ_A \end{aligned}$$

- Da die Punkt-Addition auf elliptischen Kurven assoziativ ist, gilt:
 $R = aQ_B = (ab)P = (ba)P = bQ_A = R$. Das heißt, beide Partner berechnen mit Kenntnis ihres jeweils privaten Schlüsselanteils den gleichen gemeinsamen Punkt R .

Die x- oder die y-Koordinate des gemeinsamen Punktes R kann nun wie üblich als Basis für eine Schlüsselgenerierungsfunktion dienen. In der Praxis wird häufig die x-Koordinate als gemeinsamer ECDH-Schlüssel k gewählt und als Eingabe in eine Hashfunktion, beispielsweise SHA-2, zur Generierung von Schlüsselbits für einen symmetrischen Schlüssel, wie zum Beispiel einen 128 Bit AES-Schlüssel, verwendet.

ECDH	klassisches DH
160	1024
224	2048
256	3072
384	7680

Tabelle 9.3: Vergleich der Schlüssellängen von ECDH und klassischem DH

Um ein mit dem klassischen DH-Verfahren vergleichbares Sicherheitsniveau zu erzielen, benötigt ECDH deutlich kürzere Schlüssel k . Tabelle 9.3 fasst den hier relevanten Ausschnitt aus Tabelle 7.6 noch einmal zusammen und stellt die unterschiedlichen Schlüssellängen mit vergleichbarem Sicherheitsniveau einander gegenüber.

Schlüssellänge

Sicherheit von ECDH

Die Sicherheit von ECDH hängt von der Güte der gewählten elliptischen Kurve E ab. Einem Angreifer, der das ECDH-Problem lösen und den gemeinsamen Schlüssel k berechnen möchte, verfügt über die öffentlich verfügbaren Informationen: e, p, P, Q_A, Q_B . Um den gemeinsamen Punkt $R = abP$ zu berechnen, muss er entweder $a = \log_P Q_A$ oder $b = \log_P Q_B$ berechnen. Das nennt man das Diffie-Hellman-Problem auf elliptischen Kurven (ECDHP). Ist die Kurve E gut gewählt, so sind alle bislang bekannt gewordenen Angriffe aus ECDH deutlich schwächer als Angriffe auf den klassischen DH-Algorithmus. Sie sind auch schwächer als bekannte Faktorisierungs-Angriffe auf den RSA-Modul n .

9.4 Schlüsselrückgewinnung

Bereits heute ist die Verwendung von Verschlüsselungstechniken in Unternehmen an der Tagesordnung. Verschlüsselt werden neben ausgetauschten E-Mails insbesondere auch Dokumente, die auf Festplatten abgelegt oder langfristig auf Hintergrundspeichermedien gespeichert werden. Damit ergibt sich das Problem, den Zugriff auf diese verschlüsselten Daten auch dann noch zu ermöglichen, wenn der verwendete Schlüssel verloren gegangen oder beispielsweise der Mitarbeiter, der für die verschlüsselte Erstellung des Dokuments verantwortlich war, nicht mehr verfügbar ist. Hierfür werden zunehmend Systeme zur Schlüsselrückgewinnung (engl. *key recovery*) bzw. zur Schlüsselhinterlegung (engl. *key escrow*) eingesetzt.

Da ein Hauptziel von Rückgewinnungs- bzw. Hinterlegungsverfahren darin besteht, einen Zugriff auf vertrauliche Daten zu ermöglichen, ohne im Besitz des Originalschlüssels zu sein, müssen wir die Sicherheitsrisiken, die mit einer solchen Technologie verbunden sind, kritisch untersuchen.

key recovery

Mit Systemen zur Schlüsselrückgewinnung sollen Managementdienste zur Verfügung gestellt werden, um in Ausnahmesituationen die Entschlüsselung verschlüsselter Dokumente zu ermöglichen, ohne dass der Originalschlüssel direkt verfügbar ist und ohne dass die Verschlüsselung mit kryptoanalytischen Methoden gebrochen wird. Als Ausnahmesituationen werden Schlüsselverluste, die nicht rechtzeitige Verfügbarkeit des Originalschlüssels und auch die mutwillige oder unabsichtliche Zerstörung von Schlüsseln betrachtet. Eine Schlüsselrückgewinnung soll daneben auch staatlichen Institutionen ermöglicht werden, die (in Deutschland nach Art. 10, Abs. 2 Grundgesetz) zum Zweck der Bekämpfung des Terrorismus oder des organisierten Verbrechens dazu berechtigt sind.

key escrow

Die Schlüssel-Recovery-Systeme haben ihren Ursprung in Systemen zur Hinterlegung von Schlüsseln (engl. *key escrow*). Die Hinterlegungstechnik geht zurück auf eine Initiative der amerikanischen Regierung, die 1994 den Escrowed Encryption Standard (EES) [129] festlegte, um einerseits den amerikanischen Bürgern die Verwendung starker Verschlüsselungen zu erlauben und andererseits staatlichen Strafverfolgungsbehörden den Zugriff auf verschlüsselte Daten zu ermöglichen.

Im Folgenden werden die technischen Aspekte einer Schlüsselrückgewinnung beleuchtet, während die Würdigung der rechtlichen Aspekte einer Schlüsselhinterlegung und der damit verbundenen staatlichen Überwachungsmöglichkeiten über den Rahmen des Buches hinausgeht. Von Interesse sind hier weniger spezifische, kommerzielle Produkte, sondern vielmehr deren konzeptionelle Grundlagen, woran sich die Grenzen und Risiken dieser Techniken aufzeigen lassen. Aus diesem Grund erklären wir zunächst ein einfaches Modell, das diese Grundkonzepte widerspiegelt, und verdeutlichen sie anhand des Beispiels des bekannten und auch sehr umstrittenen, mittlerweile aber auch veralteten Clipper-Chips, dessen Funktionalität in den früheren Auflagen des vorliegenden Buches (bis einschließlich Aufgabe 9) detailliert beschrieben wurde.

9.4.1 Systemmodell

Das hier vorgestellte Modell basiert auf einem Vorschlag von D. Denning und D. Branstad in [49]. Danach werden Recoverysysteme aus drei Komponenten zusammengesetzt: dem Kryptomodul, der Rückgewinnungskomponente sowie der Schlüsselhinterlegungskomponente.

Kryptomodul

Das Kryptomodul führt die Ver- und Entschlüsselungsoperationen durch und unterstützt automatisch eine Rückgewinnungsfunktionalität. Diese besteht darin, dem Kryptotext eine Information hinzuzufügen, nämlich die Rück-

Kryptomodul

gewinnungsinformation (RI), die für eine Wiederherstellung des Klartextes notwendig ist. Der zur Verschlüsselung der RI verwendete Schlüssel heißt Recovery-Schlüssel und wird von der dritten Komponente des Modells, der Hinterlegungskomponente, verwaltet. Beispiele für solche Recoveryschlüssel sind private Schlüssel der Sender bzw. Empfänger, Sitzungsschlüssel oder spezielle Produktschlüssel, die dediziert einem Kryptomodul zugeordnet sind. Damit im Falle einer Rückgewinnung der verwendete Recovery-Schlüssel gezielt von der Hinterlegungskomponente angefordert werden kann, muss die RI diesen sowie die zur Verschlüsselung verwendeten Algorithmen eindeutig beschreiben. Vom Kryptomodul wird gefordert, dass dessen Funktionalität nicht verändert oder umgangen werden kann.

Rückgewinnungskomponente

Die Rückgewinnungskomponente (RK) enthält die Algorithmen und Protokolle, die benötigt werden, um aus einem Kryptotext, aus dessen assoziierter RI und aus Daten, die die Hinterlegungskomponente zur Verfügung stellt, den Klartext zu ermitteln. Hat die Komponente mit den ihr zur Verfügung gestellten Informationen den verwendeten Schlüssel wiederhergestellt, so kann sie im Folgenden ohne weitere Interaktion mit der Hinterlegungskomponente Kryptotexte entschlüsseln. Um diese unbeschränkte Nutzung des wiederhergestellten Schlüssels einzuschränken, können durch die Hinterlegungskomponente Zeitintervalle festgelegt werden, nach deren Ablauf der wiederhergestellte Schlüssel zu löschen ist. Die Benutzer des Recoverysystems müssen darauf vertrauen, dass die RK diese Beschränkungen einhält. Weiterhin muss sichergestellt sein, dass die Dienste der RK, also die Aufzeichnung und insbesondere die Entschlüsselung von Kryptotexten, nur von dazu autorisierte Subjekten in Anspruch genommen werden können.

Rückgewinnung

Hinterlegungskomponente

Die Schlüsselhinterlegungskomponente (HK) speichert und verwaltet Schlüssel, die zur Rückgewinnung benötigt werden. Diese Komponente kann Bestandteil einer Zertifizierungsinfrastruktur sein. Da sie höchst sensible Informationen verwaltet, werden hohe Anforderungen an deren sicheren und zuverlässigen Betrieb gestellt. Die Administration dieser Komponente ist die Aufgabe vertrauenswürdiger Agenten, die dazu von staatlichen Stellen oder privaten Organisationen akkreditiert bzw. lizenziert werden. Um die Gefahren zu reduzieren, die sich aus einer zentralen, vertrauenswürdigen Instanz ergeben können, kann man mehrere Agenten einsetzen, die jeweils nur einen Teil der zur Rückgewinnung benötigten Informationen sicher verwalten (secret sharing). Eine gezielte Rückgewinnung erfordert die Kooperation dieser Agenten, so dass ein einzelner, nicht vertrauenswürdig agierender Agent ohne diese Kooperation erfolglos bleiben wird.

Hinterlegung

Die HK muss in der Lage sein, auf Anfrage die zur Rückgewinnung benötigten Daten zu liefern. Da diese natürlich nicht beliebig zugänglich sein dürfen, muss die HK vor der Datenübermittlung den Anfrager authentifizieren. Zu den Verwaltungsaufgaben der HK gehört auch, die gespeicherten Daten nach festgelegten oder mit den Recoveryschlüsseln verbundenen Zeitintervallen wieder zu löschen.

Die Komponenten des Modells sind in Abbildung 9.8 zusammengefasst.

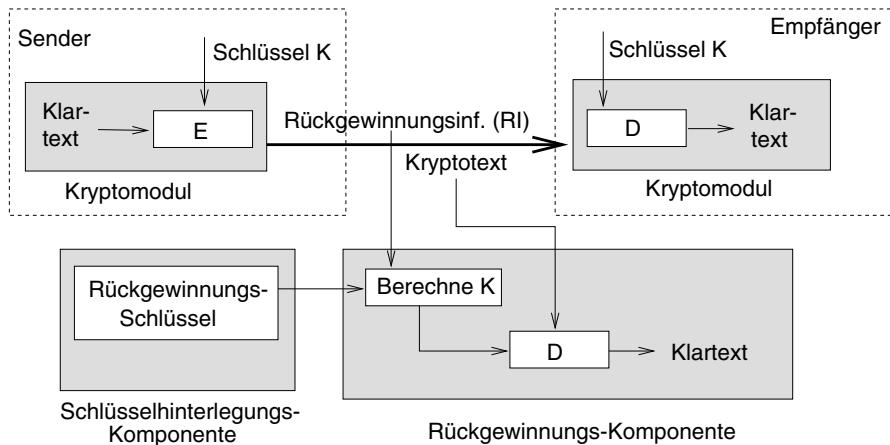


Abbildung 9.8: Modell eines Systems zur Schlüssel-Wiederherstellung

Zu den bekannten Beispielen von Systemen mit Recovery-Funktionalität, die heute auch im Einsatz sind, zählen das Softwareprodukt PGP ab Version 5.0 und das Encrypting File System (EFS) als Bestandteil der Windows-Betriebssysteme.

Beispiel 9.8 (Key-Recovery bei EFS)

Eine sehr einfache Ausprägung eines Schlüsselrückgewinnungskonzepts findet man in dem Encrypting File System (EFS), das Bestandteil der Microsoft Betriebssystem-Familie Windows ist (vgl. dazu auch Abschnitt 13.2.5).

Kryptomodul

Das Kryptomodul bei EFS hat die Aufgabe für den zur Verschlüsselung einer Datei f generierten bzw. verwendeten Schlüssel K_f die notwendige Rückgewinnungsinformation RI zu erzeugen. Dies wird einfach dadurch erreicht, dass der Schlüssel K_f mit einem öffentlichen Recovery Schlüssel K_{rec} verschlüsselt und als zusätzliches Attribut der Datei f in dem so genannten Data Recovery Field (DRF) abgespeichert wird. Als Verschlüsselungsverfahren wird RSA eingesetzt und der zur Verschlüsselung erforderliche öffentliche

Recovery-Schlüssel wird aus einem X.509 Zertifikat ausgelesen, das in einer speziellen Datei des EFS, nämlich der Encrypted Data Recovery Agent Policy, abgelegt ist. Der eigentliche Recovery Schlüssel ist der zu K_{rec} zugehörige private Schlüssel.

Die Rückgewinnungskomponente ist bei EFS trivial. Da der zu verwendende Algorithmus mit dem RSA-Verfahren festgelegt ist, wird keine zusätzliche Information über das Verfahren benötigt. Mit dem privaten Recovery-Schlüssel, der durch die Hinterlegungskomponente bereitgestellt wird, und dem Kryptotext des DRF-Feldes kann die Rückgewinnung des Dateischlüssels K_f direkt erfolgen. Die Gültigkeit des Recovery Schlüssels wird durch die im Zertifikat festgelegte Lebensdauer begrenzt.

Die Hinterlegungskomponente ist unter EFS ebenfalls sehr einfach gehalten. Bei den vertrauenswürdigen Recovery-Agenten handelt es sich standardmäßig um die Systemadministratoren. Der private Recovery-Schlüssel wird entweder manuell durch den Administrator auf ein sicheres Speichermedium exportiert oder, wovon natürlich abzuraten ist, auf der Festplatte des Rechners abgelegt. Die Hinterlegungskomponente ist somit entweder das externe Speichermedium oder ein Bereich auf der Festplatte. Unter EFS ist ein kooperatives Zusammenwirken mehrerer Recovery Agenten (Mehraugenprinzip) nicht vorgesehen.

Rückgewinnung

Hinterlegungs-komponente

9.4.2 Grenzen und Risiken

Bei der Beschreibung der Komponenten des Systemmodells sind einige der mit einer Rückgewinnungstechnik verbundenen Sicherheitsrisiken bereits angeklungen. Besonders hervorzuheben ist, dass sich durch Recoverysysteme neue Risiken ergeben, die ohne dieses gar nicht vorhanden sind. Dazu gehören die zusätzlichen Möglichkeiten, außerhalb des Einflussbereiches des autorisierten Schlüsselbesitzers Zugriff auf verschlüsselte Klartexte zu erhalten sowie die Ansammlung sicherheitskritischer Informationen an zentralen Stellen, die neue und sehr lohnende Angriffsziele bieten. Diese sensitive Information eröffnet darüber hinaus auch eine Vielzahl von Missbrauchsmöglichkeiten durch autorisierte Benutzer.

neue Risiken

Die auftretenden Risiken hängen stark vom Anwendungsbereich eines Recoverysystems ab. Man unterscheidet zwischen kommerziellen und staatlichen Anwendungen. Im kommerziellen Bereich ist man vordringlich an der Wiederherstellung gespeicherter Daten interessiert, während im staatlichen Bereich der Zugriff auf den Klartext verschlüsselt übertragener Daten wie Telefon, Fax, E-Mail, zum Zweck der Verbrechensbekämpfung im Mittelpunkt des Interesses steht. Der zweite Bereich erfordert Systeme, die die Wiederherstellung verwendeter Sitzungsschlüssel ermöglichen, während diese Funktionalität normalerweise von Nutzern der Verschlüsselungstechnologie

Einsatzbereiche

nicht benötigt wird. Probleme im nicht staatlichen Interessensumfeld, die durch einen verlorenen oder unbrauchbar gewordenen Sitzungs-Schlüssel auftreten, lassen sich direkt beheben. Dazu muss lediglich ein neuer Schlüssel generiert, ggf. ausgetauscht und die betroffenen Nachrichten müssen erneut übertragen werden. Hierfür ein Recoverysystem vorzusehen ist unnötig und eröffnet nur neue Risiken.

Systemarchitektur

dezentral

Im kommerziellen Bereich sollten lokale Systeme, zugeschnitten auf die Bedürfnisse der Benutzer, eingesetzt werden. Dies ermöglicht es, die Anzahl der zu speichernden Recoveryschlüssel so gering wie möglich zu halten und die sensiblen Recoveryinformationen dezentral durch die jeweilige Behörde bzw. das Unternehmen und durch vertrauenswürdige Mitarbeiter zu verwalten. Recoveryschlüssel können lokal erzeugt und in Abstimmung und mit Kenntnis der Betroffenen ausgegeben werden. Wichtig ist, dass so die Kontrolle über diese Schlüssel bei derjenigen Organisation verbleibt, die für die Verschlüsselung der Daten die Verantwortung trägt.

flächendeckende
Hinterlegung

Für den Einsatz von Recoverysystemen im Bereich der Verbrechensbekämpfung sind demgegenüber globale Systemarchitekturen erforderlich, durch die die Recovery-Schlüssel an zentralen Punkten extern erzeugt und verwaltet werden. Führt man sich die immense Anzahl von Schlüsseln vor Augen, die von einem flächendeckenden Hinterlegungssystem zu verwalten wären (ein Sitzungsschlüssel für jede Telefonverbindung, jede E-Mail, jede Websitzung etc.), so wird klar, dass dazu eine äußerst komplexe Infrastruktur erforderlich ist. Die Gewährleistung der sicheren und vertrauenswürdigen Funktionalität dieser komplexen Architektur erfordert im Gegensatz zu den kleinen, lokal und abgeschottet zu betreibenden Systemen einen sehr hohen Aufwand und es ist unklar, wie die Sicherheit beim Stand der heutigen Technik wirklich zu garantieren wäre. Problematisch ist weiterhin, dass die Datenrückgewinnung in der Regel transparent, also unbemerkt für die beteiligten Bürger erfolgt. Der Sinn eines solchen Recovery-Systems ist äußerst fraglich, da sich die zu überwachenden Kommunikationsteilnehmer dieser Überwachung einfach entziehen können, während die zusätzlichen Sicherheitsrisiken für die Allgemeinheit bestehen bleiben.

komplexe
Architektur

Nutzung vorhandener Infrastrukturen

Trust Center

Die Nutzung von Zertifizierungsinfrastrukturen als Schlüsselhinterlegungskomponente, wie es im Modell vorgeschlagen wurde, erscheint sehr problematisch. Ein Trust Center verwaltet nämlich in der Regel keine sensitive Information, deren Vertraulichkeit zu garantieren ist, sondern hat vorrangig die Aufgabe, Authentizitätseigenschaften zu gewährleisten. Das bedeutet, dass durch eine Recoveryfunktion gänzlich neue und sehr sicherheitskritische Aufgaben auf Trust Center übertragen werden, ohne dass diese

dafür vorgesehen sind. Darüber hinaus wird von Trust Centern nur ein Bruchteil der tatsächlich in einem System verwendeten Schlüssel verwaltet, so dass eine Anbindung eines Recoveries an die Zertifizierungsfunktion unsinnig ist und nur zu komplexeren und fehleranfälligeren Systemen führt.

Wiederherstellung von Signierschlüsseln

Kryptografische Schlüssel werden häufig eingesetzt, um mittels MACs oder Signaturen die Authentizität eines Dokumentenursprungs zu bescheinigen. Die Hinterlegung eines Signierschlüssels ist aber ebenfalls nicht nur unsinnig, sondern sogar sicherheitsgefährdend, da die dem Authentizitätsnachweis zugrunde liegende Eigenschaft nicht mehr gewährleistet ist, wenn ein Signierschlüssel in mehreren, unterschiedlichen Händen gehalten wird. Hinzu kommt, dass der Verlust eines Signierschlüssels kein Problem darstellt, das dessen Wiederherstellung erforderlich macht. Es bedeutet ja nur, dass keine neue Signatur mehr erstellt werden kann, aber bereits vorhandene Signaturen nach wie vor verifizierbar sind. Der Verlust lässt sich somit auf einfache Weise durch die Ausstellung eines neuen Signierschlüssels beheben. Problematisch wird jedoch die Situation, wenn man ein Schlüsselpaar sowohl zum Verschlüsseln als auch zum Signieren verwendet und der Verschlüsselungsschlüssel wiederherstellbar sein muss. Dies ist ein weiterer Grund, bei Softwareprodukten auf eine Trennung dieser Schlüssel zu achten bzw. eine solche Trennung, so weit dies möglich ist, durchzuführen.

Signierschlüssel

Verwaltung der Recoveryinformationen

Die Recoveryschlüssel sind sicher zu speichern und dürfen nur an authentifizierte Berechtigte ausgegeben werden. Das heißt, dass aufgrund der Recoveryfunktionalität geheime Schlüssel an mehreren verschiedenen Orten vorhanden sind und sich damit die Möglichkeiten für Angriffe darauf drastisch vergrößern. Vor allem Schlüssel, die eigentlich sicher auf einer Chipkarte verwaltet werden und diese bei deren Nutzung nicht verlassen, sind auf diese Weise Risiken ausgesetzt, die ohne ein Recovery gar nicht vorhanden sind. Führt man sich vor Augen, welche Unzulänglichkeiten heutige Standardbetriebssysteme in den Bereichen der Zugriffskontrolle und Authentifikation aufweisen oder welche Sicherheitsprobleme noch immer in heutigen Netzen bestehen, ganz zu schweigen von den durch administrative Fehler verursachten Problemen heutiger IT-Systeme, so besteht leider wenig Anlass zur Hoffnung, dass solche Systeme in einer offenen Umgebung tatsächlich sicher zu betreiben sind. Eine sichere Verwaltung von Recovery-Schlüsseln erfordert ihrerseits Schlüssel zu deren Verschlüsselung, wofür sich asymmetrische Systeme eignen, da nur geringe Datenvolumina, nämlich einzelne Schlüssel, zu chiffrieren sind. Natürlich besteht nun das Problem, diese Metaschlüssel ihrerseits sicher zu verwalten.

sichere Schlüsselspeicherung

RI-Verwaltung

Auch durch die Rückgewinnungsinformation (RI) ergeben sich Risiken mit unterschiedlichen Konsequenzen. Wird die RI zusammen mit den wiederzugewinnenden Daten gespeichert, so kann jeder Benutzer, der im Besitz eines Recoveryschlüssels ist und Zugriff auf die Daten besitzt, diese ohne weitere Kontrollen und Beschränkungen entschlüsseln. Speichert man die RI getrennt von den Daten, so muss ein Angreifer, der sich in den Besitz eines Recoveryschlüssels gebracht hat, noch die zusätzliche Hürde der Zugriffskontrolle beim Zugriff auf die RI überwinden. Andererseits ergeben sich aus der Anhäufung sensitiver Informationen wieder lohnende Angriffsziele, so dass beide Vorgehensweisen ihre Nachteile besitzen und eine sichere Konfigurierung und Administrierung der einzelnen Systeme unabdingbar ist.

Granularität der Recoveryschlüssel**Message Recovery**

Von großer Bedeutung für die Sicherheit des Gesamtsystems ist die Granularität der Recoveryschlüssel. Wird durch das System nur ein dedizierter Sitzungsschlüssel zurückgewonnen, so beschränken sich die Entschlüsselungsmöglichkeiten auf die unter diesem einen Schlüssel chiffrierten Daten, man nennt entsprechende Systeme dann auch Message Recovery Systeme. Eine gänzlich andere Tragweite besitzt ein System, das den privaten Schlüssel eines Teilnehmers als Rückgewinnungsinformation preisgibt. Dieser ermöglicht es, alle damit verschlüsselt ausgetauschten Sitzungsschlüssel wiederherzustellen und somit nicht nur eine spezifische, sondern alle Kommunikationsverbindungen des Teilnehmers abzuhören. Da die rechtzeitige Bereitstellung von Recoveryschlüsseln eine wichtige Anforderung an Systeme ist, die zur Verbrechensbekämpfung eingesetzt werden sollen, besteht eine hohe Wahrscheinlichkeit, dass hier aus Vereinfachungsgründen versucht wird, solche privaten Schlüssel wiederherzustellen.

Key Recovery**Fazit**

Die Diskussion der Grenzen und Risiken einer Rückgewinnungstechnik sollte klar gemacht haben, dass im kommerziellen Bereich ein Rückgewinnungssystem zur Wiederherstellung von Speicherschlüsseln sinnvoll sein kann, während ein Recovery von Sitzungsschlüsseln unsinnig und ein Recovery von Signierschlüsseln sogar sicherheitsgefährdend wäre. Mit Recoverysystemen sind eine Vielzahl von Sicherheitsrisiken verbunden, die beim Konfigurieren und Verwenden solcher Systeme zu beachten sind. In Beispiel 9.8 wurde die Recovery Funktion des verschlüsselnden Dateisystems EFS erläutert, dessen Nutzung aber durchaus nicht unproblematisch ist. Auf die spezifischen Probleme dieses Recovery-Systems gehen wir in Abschnitt 13.2.5 näher ein. Die Diskussion hat ebenfalls verdeutlicht, dass durch eine flächendeckende, globale Schlüsselhinterlegung nicht beherrschbare Systeme entstehen, deren Risiken in keinem Verhältnis zum erwarteten Nutzen stehen.

10 Authentifikation

Nach einer kurzen Einführung werden in diesem Kapitel die wichtigsten Konzepte und Protokolle zur Authentifizierung vorgestellt. Abschnitt 10.2 stellt Verfahren vor, die darauf basieren, dass ein Subjekt ein spezifisches Wissen vorweisen kann. Hiezu gehören die klassischen Passwort-basierten Verfahren ebenso wie das verallgemeinerte Prinzip der Challenge-Response-Verfahren, die heute in einer Vielzahl von Szenarien eingesetzt werden. Grundlegende Techniken biometrischer Verfahren sowie einige Beispiele hierzu führen wir dann in Abschnitt 10.3 ein. Abschnitt 10.4 beschäftigt sich schließlich mit Authentifikationsprotokollen in Client-Server-Architekturen. Als wichtige Vertreter werden das RADIUS- und das Kerberos-Protokoll vorgestellt.

10.1 Einführung

Die Überprüfung der Authentizität der Objekte und Subjekte ist die Voraussetzung für die Realisierung weiterer Sicherheitsanforderungen wie Integrität und Vertraulichkeit. Gemäß Definition 1.4 erfordert eine korrekte Authentifikation eines Objekts bzw. Subjekts dessen eindeutige Charakterisierung durch wohldefinierte Eigenschaften, so dass über diese zweifelsfreien Merkmale eine eindeutige Identifizierung möglich ist. Die Aufgabe der Sicherheitsgrundfunktion der Authentifikation (vgl. Abschnitt 4.7) besteht darin, mit geeigneten Maßnahmen die Korrektheit einer behaupteten Identität zu kontrollieren, d.h. diese muss von den Objekten bzw. Subjekten nachgewiesen werden. Ganz allgemein spricht man in diesem Zusammenhang häufig von Credentials, die ein Subjekt zum Nachweis seiner Identität vorzulegen hat.

Beispiele für derartige Credentials sind Benutzerkennung und zugehöriges Passwort oder Tickets, die von vertrauenswürdiger Stelle ausgestellt werden und die Identität des Besitzers bestätigen¹.

Es werden Mechanismen benötigt, die eine hohe Fälschungssicherheit der Identität garantieren bzw. es ermöglichen, Maskierungsversuche zuverlässig

¹ Der Personalausweis ist ein Beispiel für ein solches Ticket in nicht-technischen Umgebungen.

zu erkennen. Wir beschäftigen uns im vorliegenden Kapitel mit Authentifizierungsmechanismen und -maßnahmen, wobei wir uns auf die Authentifikation von Subjekten beschränken. Die gängigen Authentifikationsmechanismen für Objekte wurden ja mit den kryptografischen Hashfunktionen und Signaturen bereits in Kapitel 8 behandelt.

Subjekt

In heutigen IT-Systemen sind die Subjekte in der Regel Benutzer, Prozesse wie beispielsweise WWW-Server oder Hardwarekomponenten wie PCs. Prozesse werden entweder explizit von den Benutzern des Systems oder implizit bei der Systemnutzung gestartet oder sie sind Systemprozesse, die zur Verwaltung von Systemressourcen, wie Speicher, Prozessor oder Netzwerk, beim Booten des Systems bzw. bei Bedarf automatisch erzeugt und ausgeführt werden. Das Betriebssystem verwaltet für jeden Prozess Datenstrukturen, die den Prozess eindeutig identifizieren und Informationen darüber enthalten, ob es sich bei dem Prozess um einen Benutzer- oder Systemprozess handelt, welchem Benutzer der Prozess assoziiert ist und welche Berechtigungen dem Prozess gewährt werden.

offene Systeme

In heutigen Systemen greift eine Vielzahl parallel ausgeführter Prozesse sowohl konkurrierend als auch gezielt kooperierend auf gemeinsame Ressourcen wie zum Beispiel Dateien zu. Da solche Zugriffe auch über ein Netz, also als entfernte Zugriffe, abgewickelt werden können, ist die herkömmliche Authentifikation im Rahmen der Zugangskontrolle vieler Betriebssysteme nicht mehr ausreichend. Sie beschränkt sich nämlich nur darauf, einen Benutzer beim Systemzugang zu authentifizieren. Nach dessen erfolgreicher Authentifizierung werden automatisch alle seine Prozesse, die er im Verlauf der Login-Sitzung erzeugt, als authentisch betrachtet. Das heißt, der zugreifende Prozess wird allein durch seine eindeutige Identität, die in zentralen Systemen vertrauenswürdig durch das Betriebssystem verwaltet wird, authentifiziert. Vernetzte Systeme sind noch vorwiegend durch Client-Server-Architekturen geprägt. Die Server wie beispielsweise Dateiserver, Druckerserver oder Datenbankserver bieten Dienste an, die Clientrechner unter Verwendung von (unsicheren) Kommunikationsnetzen nutzen. Der Zugriff auf diese Dienste sollte natürlich nur authentischen Subjekten gewährt werden. Um die Authentizität von Client-Prozessen nachzuweisen reicht es aber nicht aus, lediglich eine Identität anzugeben.

In offenen Systemen können sehr einfach Identitätsangaben gefälscht und Spoofing-Angriffe durchgeführt werden. Aus diesem Grund sind Authentifikationsprotokolle zwischen anfragendem Client und anbietendem Server abzuwickeln. Wie wir bereits in Abschnitt 9.3 im Zusammenhang mit den Schlüsselverteilungsprotokollen gesehen haben, ist eine korrekte Protokollspezifikation die unabdingbare Voraussetzung für qualitativ hochwertige Implementierungen. Hilfreich sind hierbei formale Ansätze, die es erlauben,

Client-Server-Architektur

bereits in einem frühen Stadium der Protokollentwicklung Fehler, die zu Sicherheitslücken führen können, aufzudecken.

Das genutzte System und dessen Systemprozesse authentifizieren sich in der Regel aber nicht gegenüber ihren Benutzern. Das hat zur Folge, dass ein Benutzer sich nicht sicher sein kann, ob er tatsächlich mit dem entsprechenden System kommuniziert. Gelänge es beispielsweise einem Angreifer, unbemerkt ein Trojanisches Pferd als „falsches“ Login-Programm einzuschleusen und als Login-Prozess auszuführen, könnte dieser mit der üblichen Login-Aufforderung den Benutzer veranlassen, seine Kennung und sein Passwort einzugeben. In weiteren Schritten könnte diese Information in eine Datei des Angreifers kopiert, dem Benutzer eine Fehlermeldung gesendet und das Original-Login-Programm gestartet werden, so dass die Aktivitäten des maskierten Systemprozesses für den Benutzer nicht erkennbar wären. Nachdem sich ein Angreifer auf diese Weise unter der Maske eines Systemprozesses in den Besitz sensibler Informationen gebracht hat, kann er diese für weitere Maskierungsangriffe nutzen.

Maskierung

Bei den in der Praxis eingesetzten Authentifikationstechniken unterscheidet man Maßnahmen, die auf der Kenntnis eines spezifischen Wissens, auf einem persönlichen Besitz oder auf biometrischen Merkmalen basieren. Diese Klassen von Authentifizierungsmaßnahmen können sowohl in zentralen, geschlossenen, als auch in offenen und verteilten Systemen eingesetzt werden.

Techniken

Zur Realisierung von Authentifikationsverfahren werden häufig Kombinationen von ein oder mehr Authentifikationstechniken verwendet, um auf diese Weise die Sicherheit zu erhöhen und um die Vorteile der unterschiedlichen Techniken gezielt auszunutzen. Bei einer Kombination aus zwei Techniken spricht man deshalb von einer zwei Faktor-Authentifikation, die in der Regel darauf basiert, dass man etwas wissen und etwas besitzen muss. Ein allgemein geläufiges Beispiel dafür ist die Authentifikation eines Bankkunden an einem Geldautomat. Der Kunde beweist seine Identität zum einen durch den Besitz seiner ec-Karte und zum anderen durch die Kenntnis seiner PIN (spezifisches Wissen). Da sich in diesem Szenario nur der Kunde gegenüber dem Geldautomaten authentifiziert, sehen wir hier aber auch ein klassisches Beispiel für eine fehlende wechselseitige Authentifikation.

Mehr-Faktor-Authentifikation

10.2 Authentifikation durch Wissen

Techniken zur Authentifikation auf der Basis von spezifischem Wissen sind noch immer am häufigsten in der Praxis anzutreffen. In der Regel wird dazu ein Verfahren verwendet, bei dem sich der Benutzer oder ein Gerät durch die Kenntnis eines Geheimnisses authentifizieren muss. Dazu verwendet man so genannte Challenge Response-Verfahren (vgl. Abschnitt 10.2.3). Ein

in der Praxis nach wie vor am häufigsten eingesetzter Spezialfall dieser Authentifizierungsmethoden stellen Passwort-basierte Verfahren dar.

10.2.1 Passwortverfahren

Geheimnis

Mittels eines Passwortverfahrens authentifiziert sich ein Subjekt (in klassischen Betriebssystemen ist dies ein Benutzer) indem es die Kenntnis eines mit dem System vereinbarten Geheimnisses nachweist. Passwortbasierte Authentifikation wird in nahezu allen Betriebssystemen für PCs und Arbeitsplatzrechner wie die der Windows-Familie (siehe Abschnitt 13.2) oder der Unix/Linux-Familie eingesetzt. In Abschnitt 10.2.2 wird als ausführliches Fallbeispiel die Authentifikation unter Unix/Linux erklärt.

Das System hat die Passwörter sicher zu verwalten, so dass keine unautorisierten Zugriffe darauf möglich sind. In der Regel setzt man kryptografische Verfahren bzw. kryptografische Hashfunktionen ein, um die Vertraulichkeit der im System gespeicherten Passwörter zu gewährleisten. Dazu wird (z.B. in Unix- und Windows-Systemen) ein kryptografischer Hashwert von dem eigentlichen Passwort berechnet und zusammen mit Informationen, die den zugehörigen Benutzer charakterisieren, wie z.B. dessen Benutzerkennung und Namen, in einer Passworddatei gespeichert. Unter Unix-Systemen ist dies in der Regel die Datei `/etc/passwd` bzw. `.secure/etc/passwd`, während in Windows dazu der SAM (Security Accounts Manager) verwendet wird. Durch die Eigenschaft der Einwegfunktion (vgl. Definition 7.8) der verwendeten Hashfunktion ist sichergestellt, dass eine direkte Wiederherstellung des ursprünglichen Passwortes aus der Kenntnis des Hashwertes nicht ohne erheblichen Aufwand möglich ist.

Passwort-Hashfunktion

An Hashfunktionen, die im Zusammenhang mit Passworten eingesetzt werden, werden andere Anforderungen als bei herkömmlichen kryptografischen Passworten gestellt. Um systematische Brut-Force Passwort-Cracking-Angriffe zu erschweren, soll die Überprüfung eines Hashwertes gewollt langsam sein, d.h. dass das wiederholte Eingeben von falschen Passworten lange Zeit für die Verifikation auf dem Zielsystem in Anspruch nimmt. Zudem sollten die genutzten Hashverfahren so konstruiert sein, dass neben dem hohen CPU-Bedarf auch ein hoher Speicheraufwand erforderlich ist. Beispiele für kryptografische Hashfunktionen für Passwort-basierte Authentisierung sind bcrypt², Argon2³, der Gewinner der Passwort-Hash-Challenge aus dem Jahr 2015 oder auch PBKDF2 (Password-Based Key Derivation Function 2). Über Parametrisierung kann gesteuert werden, wie lange die Hashberechnung dauert.

² <https://passwordhashing.com/BCrypt>

³ <https://tools.ietf.org/id/draft-irtf-cfrg-argon2-03.html>

Die Integrität gespeicherter Passwort-Hashwerte ist über die Vergabe von Zugriffsrechten an dieser Passworddatei und über die Kontrolle der Zugriffe zu gewährleisten. Über eine beschränkte Vergabe von Schreibrechten kann das unautorisierte Überschreiben eines alten Passwortes mit einem neuen verhindert werden und die Beschränkung der Leseberechtigungen erschweren die Durchführung von Passwort-Cracking-Angriffen (s.u.).

Die Sicherheit der verwalteten Passwörter hängt somit zum einen von der Stärke der kryptografischen Hashfunktion und zum anderen von der Qualität der Rechtevergabe und Zugriffskontrolle ab. Sicherheitsprobleme ergeben sich in vielen Systemen dann aber noch daraus, dass die Passworddatei mit der sensiblen Information für lesende Zugriffe zugänglich ist. Dies eröffnet Möglichkeiten für Angriffe mit gewählten Klartexten (vgl. Abschnitt 7.8.1). Dazu rät der Angreifer das Passwort seines Opfers, lässt es vom Zugangskontrolldienst des Systems verschlüsseln und vergleicht das Ergebnis mit dem Kryptotext in der Passworddatei. Diese Art von Angriffen sind unter dem Namen **Password-Cracking** bekannt und es existieren frei verfügbare Softwareprodukte, die unter Nutzung von Wörterbüchern versuchen, die Einträge einer gegebenen Passworddatei systematisch zu brechen.

sichere Verwaltung

Password-Cracking

Für Unix-Systeme sind das beispielsweise die frei verfügbaren Programme *Crack* oder *John the Ripper*⁴. Um im Windows-Umfeld die zum Cracken erforderlichen Eingabedateien zu erhalten, stehen mit *pwdump2*, *pwdump3*⁵ ebenfalls frei verfügbare Programme zur Verfügung. Die Programme erstellen einen Dump der Passwort-Hashwerte aus der SAM-Datei eines Rechners. Zur Ausführung und damit zum Zugriff auf diese Datei sind jedoch Administrator-Privilegien erforderlich. Durch die Beschränkung der Berechtigungen zum Lesen der Passworddatei lassen sich somit die Ansatzpunkte für Passwort-Crack-Angriffe reduzieren. Ferner sollte die Sicherheit der passwortbasierten Authentifikation nicht von der Geheimhaltung der Benutzerkennungen, die zur Identifikation benötigt werden, abhängen, da diese in der Regel vom Systemadministrator unter Nutzung eines einfachen Schemas, wie die ersten acht Buchstaben des Nachnamens, vergeben werden.

Sind von der Betriebssystemseite die erforderlichen Vorkehrungen für eine sichere Passwortverwaltung getroffen, so steht und fällt die Sicherheit dieser Authentifikationstechnik mit der Wahl der Passwörter und mit dem Umgang damit außerhalb des technischen Systems. Denn was nützt eine hoch sichere Verwaltung der gespeicherten sensiblen Authentifikationsinformationen, wenn ein Angreifer auf einfachste Weise in deren Besitz gelangen kann. Klassische Beispiele hierfür sind das Beobachten der Benutzereingabe oder

Passwortwahl

⁴ <http://www.openwall.com/john/>

⁵ <http://www.openwall.com/passwords/dl/pwdump/pwdump3v2.zip>

das Lesen vermeintlich gut versteckter „Spickzettel“, auf denen das Passwort niedergeschrieben wurde,

Social Engineering

Mangelhaftes oder fehlendes Sicherheitsbewusstsein bei Nutzern, aber auch Administratoren erleichtert es Angreifern häufig erheblich, geheime Passworte an den Besitz zu bekommen. So sind Fälle durchaus keine Seltenheit, in denen sich ein Benutzer erfolgreich(!) an die telefonische Online-Hilfe seines Systems wendet mit der Bitte: „Meine Kennung ist Root und ich habe mein Passwort vergessen“. Solchen Anrufern werden noch immer häufig ohne weitere Nachfragen und insbesondere ohne weitere Überprüfung von deren Identität das angeblich vergessene Passwort mitgeteilt. Eine Sensibilisierung der Nutzer eines Systems ist deshalb unabdingbar, so dass sie sich über die Konsequenzen eines fahrlässigen und unachtsamen Umgangs mit Authentifikationsinformationen im Klaren sind. Das vorliegende Buch dient nicht zuletzt dazu, das erforderliche Sicherheitsbewusstsein zu schärfen und zu verdeutlichen, dass jeder Einzelne zur Sicherheit und insbesondere Un Sicherheit eines komplexen Systems beitragen kann.

Im Folgenden konzentrieren wir uns aber im Wesentlichen auf die technischen Aspekte einer sicheren Authentifikation. Als Basis für die weitere Diskussion ist es nützlich, sich zunächst den allgemeinen Ablauf einer passwortbasierten Zugangskontrolle vor Augen zu führen.

Passwortbasierte Zugangskontrolle:

Login-Protokoll

1. Das System verlangt die Identifikation/Kennung des Benutzers, z.B. durch ein Login-Prompt
Login :
2. Der Benutzer identifiziert sich durch seine Kennung, z.B. mit
Login : eckertc
3. Das System verlangt die Authentifikation, z.B. über den Prompt
Password : .
4. Der Benutzer gibt ein Passwort ein, das aber (hoffentlich!) nicht auf dem Bildschirm erscheint.
5. Das System vergleicht das Passwort mit dem in der Passworddatei unter der Benutzeridentifikation abgespeicherten, wofür ggf. das eingegebene Passwort zunächst verschlüsselt werden muss.
6. Im Fehlerfall wird der Benutzer in der Regel informiert, z.B. mit
Login failed.

Probleme

Verwendet nun der Benutzer ein einfache zu bestimmendes Passwort, so ist es für einen Angreifer nicht schwierig, sich unter der Benutzerkennung seines Opfers Zugang zum System zu verschaffen. Die Probleme und Risiken

des Passwortverfahrens sind bereits seit langer Zeit allgemein bekannt. So lieferte eine in [124] durchgeführte Untersuchung an 3289 von Benutzern frei gewählten Passworten das in Tabelle 10.1 zusammengefasste Ergebnis.

Anzahl	Prozent	Eigenschaften
15	0.5%	Passwort aus einem ASCII Zeichen
72	2%	Passwort aus zwei ASCII Zeichen
464	14 %	Passwort aus drei ASCII Zeichen
477	14 %	Passwort aus vier alphanumerischen Zeichen
706	21%	Passwort aus 5 Groß- oder Kleinbuchstaben
605	18%	Passwort aus 6 Kleinbuchstaben
492	15%	Passwort stand in einem Wörterbuch o.ä.

Tabelle 10.1: Schwächen benutzerwählbarer Passworte

Insgesamt fielen 86% aller Passworte in eine der in Tabelle 10.1 angegebenen Kategorien und waren somit sehr einfach zu „knacken“. Um ein Mindestmaß an Sicherheit zu gewähren, sollten die gewählten Passworte und die realisierten Zugangsverfahren die folgenden Anforderungen erfüllen.

Anforderungen:

1. Jedes Passwort sollte mindestens eine Länge von 12 Zeichen besitzen, aus Groß- und Kleinbuchstaben, sowie Sonderzeichen zusammengesetzt sein und kein Wort sein, das in einem Wörterbuch nachschlagbar ist.
2. Ferner sollte es kein Eigenname oder gar der eigene Vor- oder Nachname sein.
3. Es sollte mindestens ein Sonderzeichen enthalten und
4. nicht nur aus einer Folge von Zeichen bestehen, die auf der Tastatur unmittelbar nebeneinander liegen, wie zum Beispiel die Sequenz uiop [.
5. Am Besten sollte es möglichst viele unterschiedliche Zahlen oder Buchstaben enthalten und
6. in regelmäßigen Abständen, zum Beispiel spätestens nach einem Jahr, geändert werden.
7. Der Systemdienst der Zugriffskontrolle sollte durch frühzeitige Überprüfungen verhindern, dass ein Benutzer bei der Wahl seines Passwortes gegen die Längenforderungen aus (1.) und/oder gegen die Anforderungen (2.) und (3.) verstößt.
8. Das System sollte nur eine geringe Anzahl von Fehlversuchen beim Login tolerieren (z.B. 3) und danach die Kennung sperren.

Beispiel 10.1 (Passwortwahl)

Ein relativ einfacher Weg, um zu guten Passworten zu kommen, besteht darin, sich einen längeren Satz zu überlegen und von jedem der Worte dieses Satzes nur den Anfangsbuchstaben zu verwenden. Der Satz sollte natürlich für den Benutzer eine Bedeutung haben, so dass er sich daran auch nach einer längeren Zeit noch erinnern kann. Beispiele hierfür sind Passagen aus Büchern oder Filmen, wie `I s D i d A K !`. Des Rätsels Lösung ist: „Ich schau' Dir in die Augen Kleines!“ (aus dem Film Casablanca).



Systemgenerierte Passwörte

Da die benutzerbestimmte Wahl von Passworten bis heute immer wieder große Sicherheitsprobleme verursacht, gibt es Ansätze den Benutzer zu zwingen, systemgenerierte Passwörte zu verwenden. Die so generierten Passwörte genügen zwar meist den gestellten Anforderungen, da sie in der Regel als Zufallszeichenfolgen erzeugt werden; sie können jedoch aufgrund dieser Zufälligkeit nur schwer memoriert werden. Das führt dann dazu, dass der überforderte Benutzer diese kryptischen Folgen aufschreibt und sie vorzugsweise in der Nähe seines Arbeitsplatzes deponiert. Klassisch sind hier die Zettel unter der Schreibtischunterlage, unter der Tastatur oder sogar für alle einsehbar am Bildschirm. Systemgenerierte Passwörter werden häufig für den Zugang zu entfernten Rechnern bzw. Servern vergeben. Hierbei ist ein Benutzer bereits auf seinem lokalen System eingeloggt und muss sich für den Fernzugriff erneut authentifizieren. Um Benutzer zu entlasten, bieten heutige Betriebssysteme Dienste an, um derartige, in der Regel sehr kryptische Passwörter auf der Festplatte zu speichern und beim Login diese automatisch zu übermitteln. Dies ist natürlich sehr benutzungsfreundlich, birgt aber gleichzeitig die Gefahr, dass solche Passwörter mittels Trojanischen Pferden oder Würmern ausgespäht und für Spoofing-Angriffe missbraucht werden.

In Anbetracht dieser Probleme verwenden einige Systeme eine Kombination von systembestimmten und benutzerbestimmbaren Passwortfestlegungen. Dazu generiert das System eine Anzahl von Zeichen, die in dem Passwort, das der Benutzer generiert, enthalten sein müssen.

Beispiel 10.2 (PIN-Erzeugung)

Ein bekanntes Beispiel eines systemgenerierten Passwortes ist die PIN (Personal Identification Number) für Eurocheque-Karten. Dazu werden die 10-stellige Kontonummer, die fünf letzten Zahlen der Bankleitzahl und eine einstellige Kartensequenznummer konkateniert und mittels BCD (jede Zahl durch vier Bit codiert) in einen 64-Bit Wert codiert. Dieser dient als Eingabeblock für den 3DES, wobei als einer der DES-Schlüssel der 56 Bit

lange Institutsschlüssel derjenigen Bank verwendet wird, die die ec-Karte ausstellt. Die 64-Bit Ausgabe wird in 16 Hexadezimalzeichen transformiert. Die PIN ergibt sich aus den 3ten bis 6ten Zeichen, wobei vorkommende Buchstaben A–F des Hexcodes durch die Ziffern 0–5 ersetzt werden. Diese PIN ist zwar systemgeneriert, erfüllt aber nicht die Anforderungen, die an die Länge oder auch an die Zufälligkeit der erzeugten Werte gestellt wurden. Dies war und ist zusammen mit der Entscheidung, einen „kurzen“ DES-Schlüssel zur PIN-Berechnung zu verwenden, Gegenstand heftiger Kritik an dem PIN-Erzeugungsprozess.



Einmal-Passwörte

Eine interessante Alternative zur klassischen Vorgehensweise stellen die einmal benutzbaren Passwörte OTP (engl. *one-time password*) dar. Wie der Name bereits suggeriert, werden bei dieser Technik Passwörte nur jeweils einmalig zur Authentifikation verwendet. Dadurch ist sichergestellt, dass die Kenntnis eines solchen Passwortes einem Angreifer nicht zu einem erfolgreichen Maskierungsangriff verhilft. Einsatzbereiche für derartige Verfahren sind in erster Linie Systeme, auf die entfernt (entweder innerhalb des LANs oder von außerhalb, beispielsweise durch einen mobilen Mitarbeiter) zugegriffen wird (u.a. per ftp) und für die keine anderen Sicherheitsmechanismen wie das Kerberos-Protokoll oder Secure Shell etabliert sind. Nun stellt sich natürlich unmittelbar die Frage, wie einmal benutzbare Passwörte so erzeugt werden können, dass der Benutzer selber von dieser Generierung weitestgehend entlastet ist und gleichzeitig die Wahrscheinlichkeit gering ist, dass ein Angreifer ein noch nicht verwendetes Einmal-Passwort korrekt bestimmen kann. Die Lösungsidee hierfür basiert auf der Verwendung von Einweg-Funktionen. Ein Schema für Einmal Passwörte wurde konzeptionell bereits 1981 in dem Artikel [106] von Lesli Lamport vorgestellt. Das Konzept wurde in dem S/Key-Verfahren [76] später umgesetzt und fand insbesondere in der Unix-Welt eine relativ weite Verbreitung. Im Folgenden wird das Lamport-Konzept am Beispiel der konkreten Umsetzung durch das S/Key-Verfahren vorgestellt. Auf der Basis von S/Key wurden weitere One-Time-Passwort (OTP) Verfahren entwickelt, die auch als RFC 2289 allgemein verfügbar sind.

Einmal-Passwort

Lösungsidee

S/Key-Verfahren⁶

S/Key ist ein Authentifikationsprogramm, das entwickelt wurde, um Password-Sniffer Angriffe in Client-Server-artig kommunizierenden Unix-Systemen wirkungslos zu machen.

S/Key

⁶ S/Key ist ein Trademark der Firma Bellcore.

Der Client in dem OTP-Verfahren ist der Arbeitsplatzrechner (PC) eines Benutzers, von dem aus ein entfernter Zugang auf einen Serverrechner mittels Passworten realisiert wird. Das Prinzip ist aber natürlich auch auf andere Szenarien übertragbar wie zum Beispiel eine Authentifikation zwischen einer persönlichen Chipkarte eines Benutzers und einem Rechner. Hierbei übernimmt dann der Rechner (u.a. PC, Laptop) die Rolle des Servers.

Das S/Key-Verfahren ist aufgeteilt in eine Initialisierungsphase, die einmal durchzuführen ist, und die Authentifizierungsschritte, die bei jedem Login-Vorgang ausgeführt werden.

Initialisierungsphase

Geheimnis

Ausgangspunkt des Verfahrens ist ein geheimes Passwort s , das vorab zwischen dem Benutzer und seinem Arbeitsplatzrechner, dem Client, vereinbart wurde. Wichtig hierbei ist, dass der Server das Geheimnis s nicht benötigt, so dass es nicht zwischen Client und Server auf einen gesicherten Kanal ausgetauscht werden muss. Der Server muss dieses Geheimnis auch nicht verwalten, so dass ein möglicher Angriffspunkt konzeptionell vermieden ist.

1. Unter Verwendung einer kryptografisch sicheren Hashfunktion als Einweg-Funktion f berechnet der Clientrechner aus dem Geheimnis s und einem Seed-Wert⁷ k die Folge der einmal benutzbaren Passwörter p_1, \dots, p_n , es gilt

$$p_i = f^i(s | k) \text{ für } i = 1, \dots, n.$$

Zur Authentifikation des Clients gegenüber dem Server werden ausschließlich diese Einmalpasswörter p_i verwendet. Das bedeutet, dass das Geheimnis s niemals beim Login vom Client zum Server übertragen wird.

2. Zur Initialisierung sendet der Client das letzte Passwort der Folge, also p_n , zusammen mit dem Seed-Wert k an den Server. Das Seed ist eindeutig für den Benutzer auf dem entsprechenden Server und dient dazu, dass ein Benutzer das gleiche geheime Passwort s für verschiedene Rechner verwenden kann. Damit übernimmt das Seed eine ähnliche Funktion, wie der Salt-Wert beim herkömmlichen Unix-Login (vgl. Seite 463).

S/Key-Authentifikation

Aus Benutzersicht ergibt sich keine Änderung im Authentifikationsablauf. Ein Benutzer weist sich nach wie vor unter Kenntnis seines Geheimnisses s , bzw. einer PIN, falls der Client eine Chipkarte ist, gegenüber dem Clientrechner als authentisches Subjekt aus. Dieses wird aber, wie gesagt,

⁷ Der Wert wird häufig verwirrender Weise als Key bezeichnet.

bei einem anschließenden entfernten Login vom Client auf den Server nicht mehr verwendet, sondern es kommen hierbei die Einmal-Passworte wie folgt ins Spiel:

1. Nach i erfolgreichen Authentifikationen des Benutzers beim Server gilt, dass dieser das i -te Passwort p_i bereits überprüft hat und es zusammen mit dem Index i protokolliert.
2. Bei einer erneuten Login-Anfrage durch den Clientrechner fordert der Server den Clientrechner auf, das $(i - 1)$ -te Passwort vorzuweisen. Dazu sendet der Server die laufende Nummer $i - 1$ sowie den mit dem Clientrechner vereinbarten Seed-Wert k an den Clientrechner.
3. Dieser berechnet (oder wählt es aus der Liste aus) das entsprechende Passwort p_{i-1} und überträgt es zum Server. Gemäß der oben angegebenen Konstruktion, erhält man p_{i-1} durch $i - 1$ -maliges Anwenden der Funktion f auf das Geheimnis s konateniert mit dem Seed k .
4. Da das Benutzer-Geheimnis s dem Server nicht bekannt ist, wohl aber der Wert p_i , kann er die Korrektheit des Passwortes p_{i-1} durch die einmalige Anwendung der Funktion f auf p_{i-1} überprüfen. Die Passwörte wurden nämlich gerade so konstruiert, dass gilt:

$$p_i = f^i(s \mid k) = f(f^{(i-1)}(s \mid k)) = f(p_{i-1}).$$

Es ist somit zu prüfen, ob gilt:

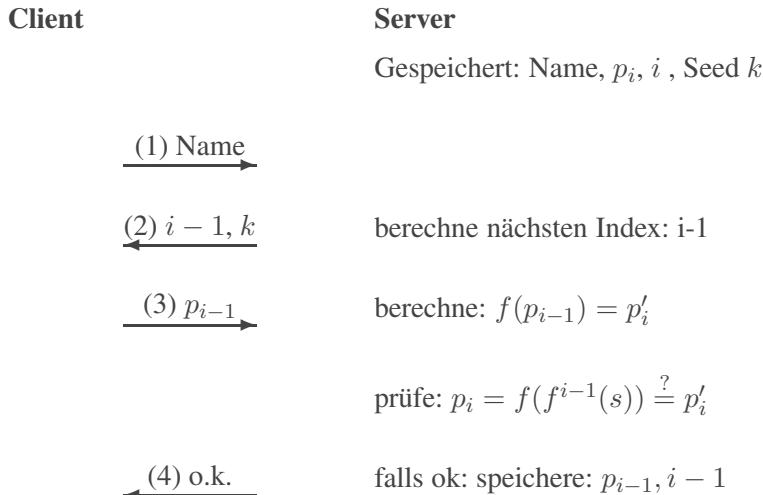
$$f(p_{i-1}) \stackrel{?}{=} p_i.$$

Da f eine Einwegfunktion ist, kann der Server den Wert $f(p_{i-1})$ effizient berechnen.

5. Falls die Überprüfung positiv verläuft, ist die Authentifikation erfolgreich und der Server ersetzt den Wert p_i durch das nunmehr verbrauchte Passwort p_{i-1} und dekrementiert seinen Sequenzzähler.

Die Schritte sind in der folgenden Übersicht noch einmal zusammengefasst.

Das System ermöglicht es auch, dass der Benutzer sich eine Folge von Einmal-Passwörtern generieren lässt, um z.B. von einem unsicheren Rechner aus ein sicheres Login zu starten. Dazu muss er die vorab generierten Passwörte auf Papier oder einem externen Medium speichern und bei einem Login dann per Hand eingeben.



In einem System würde die beschriebene Situation jetzt also beispielsweise wie im folgenden Szenario aussehen:

```
madwand.10pht.com> telnet some.host.somewhere
Trying 199.99.99.99... Connected to some.host.somewhere.
Escape character is '^]'.
```

```
login: jdoe
s/key 99 k113355
Password: WELD GUY CHIMP SWING GONE
```

Hierbei wird eine Telnet-Verbindung zu einem Server aufgebaut, der S/Key verwendet. Nach der Anmeldung über den Benutzernamen *jdoe* sendet der Server eine Challenge, hier die Zahl 99, und das Seed des Servers, hier der String *k113355*. Das S/Key-Verfahren auf dem Client-Rechner berechnet das Passwort wie oben beschrieben, also durch das 99-malige Anwenden der Hashfunktion auf das geheime Passwort konkatiniert mit dem Server-Seed und liefert *WELD GUY CHIMP SWING GONE* als Einmal-Passwort zurück.

S/Key-Sicherheit

Sicherheit

Die Sicherheit des Verfahrens basiert auf der Einwegeigenschaft der verwendeten Funktion f und auf der Qualität des zugrunde liegenden geheimen Passwortes. Da die Einmal-Passwörte p_i unverschlüsselt übertragen werden, kann ein Angreifer diese beobachten (Sniff-Angriff). Mit der Kenntnis eines Passwortes p_i ist es ihm jedoch nicht möglich, das nächste Passwort p_{i-1} zu bestimmen. Dazu müsste er nämlich das Urbild für das abgefangene Passwort p_i berechnen, $p_{i-1} = f^{-1}(p_i)$, da er das Geheimnis s nicht kennt⁸.

⁸ Wir gehen hier davon aus, dass dem Angreifer s nicht zur Verfügung steht, da sonst natürlich auch die Einmal-Passwörte keine zusätzliche Sicherheit bieten.

Eine erfolgreiche Maskierung ist somit deutlich erschwert, aber nicht gänzlich ausgeschlossen.

Gelingt es nämlich dem Angreifer, die Verbindung zwischen Benutzerrechner und Server so zu kontrollieren, dass er das korrekte Passwort p_i , das der Client in Schritt (3) sendet, abfangen kann und es nicht an den Server weiterleitet⁹, so wird dieser den Authentifikationsvorgang abbrechen, ohne seinen Passwortzähler zu erniedrigen, falls das zuvor erläuterte Protokoll so „naiv“ implementiert wird. Der Angreifer kann nun seinerseits einen Login Versuch unter der Identität des Clientrechners starten. Wird er dann vom Server aufgefordert, das i -te Passwort vorzuweisen, so kann er mit dem korrekten Wert p_i antworten. Ein Maskierungsangriff ist somit erfolgreich durchführbar. Eine einfache Lösung dieser Protokollschwachstelle besteht darin, dass der Server seinen Passwortzähler auch bei einem abgebrochenen Login-Versuch dekrementiert. In diesem Fall muss er dann bei einem erneuten Login die Hashfunktion entsprechend oft auf das neue Passwort anwenden, damit er den korrekten Vergleich durchführen kann, da er ja nach wie vor nur das letzte korrekt verwendete Passwort als Referenzwert besitzt.

Eine Verallgemeinerung des zuletzt beschriebenen Angriffs besteht darin, dass sich ein Angreifer als Server maskiert und dem Client eine Sequenznummer j als Challenge übermittelt, die deutlich kleiner ist, als die zuletzt regulär verwendete Sequenznummer i . Muss sich der Server nicht gegenüber dem Client authentifizieren, so wird der Client auf die Challenge mit dem korrekten Passwort p_j antworten. Damit ist der Angreifer in der Lage, alle Sequenznummer-Challenges $r \in \{j, \dots, i - 1\}$ des Original-Servers mit korrekten Passworten zu beantworten. Dazu muss er lediglich, ausgehend von dem erschlichenen Passwort p_j , den Wert

$$f^{r-j}(p_j) = p_r$$

berechnen.

Es ist damit klar, dass auch ein Einmal-Passwort-Verfahren nicht vor Phishing-Angriffen (vgl. Seite 1.3.2) schützt, wenn der Clientrechner auf Anfragen von beliebigen Rechnern ein angefragte Einmal-Passwort zurück sendet. Der Angreifer profitiert hierbei auch zusätzlich noch davon, dass die von ihm „abgefischten“ Passwörter nicht nach einer kurzen Zeit automatisch verfallen.

Ein weiteres Problem besteht darin, dass durch das Abfangen von Einmal-Passworten zusammen mit dem Abfangen der Sequenznummer i (Challenge) und dem Seed k , ein Angreifer in der Lage ist, Wörterbuchangriffe auf das zugrunde liegende geheime Passwort s durchzuführen. Dazu muss der An-

Clientmaskierung

Servermaskierung

Phishing

Wörterbuchangriff

⁹ Das ist eine spezielle Ausprägung einer Phishing-Attacke.

greifer systematisch mit allen möglichen Worten s' aus dem Wörterbuch eine S/Key-Berechnung durchführen, also die bekannte Hashfunktion i -mal auf s' konkatiniert mit dem ebenfalls bekannten Seed anwenden, d.h. $f^i(s' \mid k)$. Eine einfache Lösung dieses Problems besteht hierbei darin, dass das geheime Passwort s kein vom Benutzer wählbares Passwort ist, sondern dass das System eine gute Zufallszahl als Ausgangsbasis für die Passwortberechnung wählt.

OPIE-Verfahren

Eine heute noch relativ weit verbreitete Umsetzung des One-Time-Password Verfahrens basierend auf dem S/Key-Verfahren ist das OPIE¹⁰-Verfahren (One-time Passwords In Everything). Das Verfahren lässt sich zum Beispiel unter Linux unter Nutzung des Pluggable Authentication Moduls (PAM) relativ einfach als Erweiterung z.B. von SSH integrieren. Benötigt werden Erweiterungen sowohl auf der Seite des Servers als auch beim Client. Zur Initialisierung muss der Benutzer sein Geheimnis s wählen, das unter OPIE ein mindestens 10-stellige Passphrase ist.

TOTP

Das ursprüngliche OTP-Verfahren wurde in den letzten Jahren weiter entwickelt. Besonders erwähnenswert sind das TOTP¹¹ (Time-Based One-Time Password Algorithm) (vgl. RFC 6238) als Erweiterung des HOTP-Verfahrens¹² (HMAC-Based One-Time Password Algorithm) (vgl. RFC 6238). TOTP unterstützt im Gegensatz zum zählerbasierten Ansatz von HOTP einen so genannten zeitbasierten, dynamischen Faktor. Dieser Faktor ist ein Wert, der sich für jedes neu generierte Passwort ändern muss. Das TOTP wird beispielsweise von Dropbox¹³ oder auch von Google mit dem Google Authenticator¹⁴ implementiert.

Beispiel 10.3 (Google-Authenticator)

Authenticator

Seit 2011 bietet Google eine One-Time Passwortvariante (OTP) für Web-Authentisierung unter dem Namen **2 Step Verification** oder auch *Google Authenticator* für Smartphones (Android, iOS, Blackberry) an. Dazu muss der Nutzer eine Authenticator-App aus dem Google Android App Market herunterladen. Diese App generiert die OTPs. Bevor eine Authentisierung mit zusätzlichen OTPs beim Login für Web-Dienste genutzt werden kann, muss eine Registrierung stattfinden. Mit dieser Registrierung wird zwischen der App und dem Web-Dienst bzw. dem zugehörigen Web-Server ein Pre-shared Secret verabredet. Dazu generiert ein solcher Web-Server, der die

¹⁰ Vgl. http://static.usenix.org/publications/library/proceedings/security95/full_papers/mcdonald.pdf

¹¹ siehe <http://tools.ietf.org/html/rfc6238>

¹² siehe <http://tools.ietf.org/html/rfc4226>

¹³ siehe <https://www.dropbox.com/help/363/en>

¹⁴ siehe <http://code.google.com/p/google-authenticator/>

erweiterte Authentisierungsfunktion unterstützt, ein 80 Bit Secret s . Das Geheimnis s kann auf verschiedenen Wegen zum Smartphone übermittelt werden. Üblich ist eine Übermittlung per SMS oder Voice-Mail, als 16 Zeichen base32 codierter String oder via QR Code. Das Secret s wird in der Authenticator-App auf dem Smartphone gespeichert.

Beim Login in den Web-Dienst muss dann neben Nutzerkennung und Passwort auch ein aktuelles OTP eingegeben werden. Das OTP, unten als AccessCode bezeichnet, wird von der Authenticator-App unter Nutzung des Secrets s berechnet:

$$\text{AccessCode} = \text{HMAC_SHA1}(s \parallel \text{Uhrzeit}).$$

Der AccessCode wird auf eine 6-stellige Zahl reduziert. Der Parameter *Uhrzeit* ist ein Counter, der die 30 Sekunden Slots seit Beginn der Unixzeit (1.1.1970) zählt. Der Server berechnet seinerseits auch den AccessCode auf der Basis seiner eigenen Daten:

$$\text{AccessCode}' = \text{HMAC_SHA1}(s' \parallel \text{Uhrzeit}')$$

und vergleicht die beiden Codes. Aufgrund der Verwendung von Timeslots haben die AccessCodes nur eine kurze Gültigkeitsdauer. Wie bei allen OTP-Verfahren ist damit aber ein Man-in-the-Middle Angriff mittels Phishing nicht prinzipiell ausgeschlossen. Die OTPs können immer noch abgefangen und missbraucht werden. Aber das Zeitfenster zur erneuten Verwendung des OTPs ist sehr klein, so dass das Risiko für Phishing-Angriffe deutlich reduziert wird.

Problematisch an dem verfolgten Ansatz ist jedoch, dass das Geheimnis s auf unsicherem Weg übertragen wird und der Server sich nicht authentisieren muss. Zudem kann die Authenticator-App, die die Access-Codes generiert, manipuliert sein. Die App wird zwar signiert, was eine Vertrauensbildende Maßnahme ist, ob die Apps wirklich Schadcode-frei und korrekt sind, wird durch eine Signatur aber nicht per se gewährleistet.

Zur Erhöhung der Sicherheit beim Login bei Web-Anwendungen soll ab 2014 eine stärkere 2-Faktor-Authentisierung mit dem Konzept U2F, der Universal 2 Factor Authentication¹⁵ eingeführt werden. Das U2F-Konzept wird in Abschnitt 11.3 vorgestellt.



Neben Software-Lösungen, wie den S/Key-basierten Verfahren existieren eine ganze Reihe von Hardware-basierten Lösungen, die ein spezielles Hardware-Token zur Generierung der Einmal Passwörter erfordern.

¹⁵ vgl. <https://sites.google.com/site/oauthgoog/gnubby>

Beispiel 10.4 (RSA SecureID als OTP-Token)

SecureID

Ein in der Praxis häufig eingesetzte Hardware-basierte Lösung sind die RSA SecureID-Token. Die Authentifikation eines Benutzers gegenüber einem Server, dem RSA ACE Server, erfolgt als eine zwei Faktor-Authentifikation, bei der der Benutzer sich durch die Kenntnis einer PIN sowie durch den Besitz eines Tokens ausweisen muss. In einem initialen Schritt erhält der Benutzer ein solches Token von dem Systemadministrator. Jeder Token besitzt eine eindeutige Seriennummer, die auch dem Server bekannt ist. Darüber hinaus enthält das Token einen symmetrischen Schlüssel, das Seed, das ebenfalls dem Server bekannt ist. Der Benutzer besitzt auf dem Server eine Kennung, die mit der Seriennummer des Tokens sowie dem Seed verknüpft ist.

Zeit-Synchronisation

Anders als bei dem S/Key-Verfahren basiert die Authentifikation nicht auf einem Frage-Antwort-Spiel, sondern diesem System liegt eine zeitlich gesteuerte Synchronisation zwischen dem Token und dem Server zugrunde. Das bedeutet, dass der Server und das ausgegebene Token eine synchronisierte Uhr besitzen. Das Token erzeugt in fest gelegten Zeitintervallen, in der Regel ist das nach 30 oder 60 Sekunden ein neues Passwort, den so genannten Tokencode, der auf dem Display des Tokens angezeigt wird. Der Tokencode ist das Einmal-Passworte, dessen Generierung unter Rückgriff auf die Seriennummer des Tokens, das Seed und die aktuelle meist unter Verwendung des AES als Hashfunktion erfolgt. Sowohl der Server als auch das Token generieren zeitsynchronisiert das gleiche Einmal-Passwort.

Benutzer-authentifikation

Zur Authentifikation beim Server muss der Benutzer seine Kennung angeben und sich über die Kenntnis einer PIN sowie des aktuellen Einmal-Passwortes ausweisen, das er von dem Display des Tokens manuell in das Login-Fenster auf seinem Rechner übertragen muss. Falls das Token zusätzlich zum Display auch über eine Eingabemöglichkeit verfügt, dann handelt es sich um eine PIN-Pad Karte. Der Benutzer gibt seine PIN direkt auf diesem Token ein und der PIN-Wert fließt auch in die Berechnung des AES-Hashs ein. Wiederum wird das Ergebnis als Tokencode auf dem Display angezeigt und muss dann noch vom Benutzer manuell an den Server übermittelt werden. Dieses Token hat jedoch den Vorteil, dass der Benutzer nicht mehr seine PIN im Klartext an den Server übermitteln muss. In allen Fällen prüft der Server, ob das vor vom Benutzer eingegebene Tokencode mit dem auf dem Server zeitgleich generierten Code übereinstimmt. Bei Gleichheit hat sich der Benutzer erfolgreich authentifiziert.

Hacker-Angriff

Für einiges Aufsehen sorgte ein Hacker-Angriff auf die RSA-Server im Jahr 2011. Im Zuge dieses Angriffs gelang den Einbrechern der Diebstahl von sicherheitsrelevanten Informationen (Seeds und Seriennummern). Man vermutete, dass die Angreifer vordringlich Zugang zu militärischen Anla-

gen und Rüstungsbetrieben mittels dieser Information erlangen wollten, da sich mit der gestohlenen Information OTPs berechnen ließen. So wurde ein erfolgreicher Angriff auf die Rüstungsfirma Lockheed Martin auf diesen Datendiebstahl zurückgeführt¹⁶. Der RSA-Hacking-Angriff führte dazu, dass rund 40 Millionen Hardware-Token ausgetauscht werden mussten.

SecureID Token der beschriebenen Ausprägung stellen eine sehr einfache und kostengünstige Lösung dar, die eine attraktive Alternative zur flächen-deckenden Einführung von Smartcards zur Authentifikation darstellen.



Obwohl mit den Einmalpasswort-Verfahren pragmatische Lösungen zur Verbesserung der Passwortproblematik zur Verfügung stehen, ist der klassische Passwort-Ansatz im heutigen Unternehmensumfeld, in Behörden und auch beim Zugang zu Internetdiensten (u.a. Amazon, eBay, Freemailer) immer noch dominierend. In Unternehmen sind in der Regel eine Vielzahl unterschiedlicher Softwarelösungen im Einsatz (u.a. Dokumentenmanagement-Systeme, SAP-Systeme, Projektdatenbanken), die jede für sich eine eigene Benutzerverwaltung zusammen mit einer Rechtevergabe für die Benutzer implementiert. Zusammen mit der ständig zunehmenden Zahl an Internet-Diensteanbietern führt diese Situation dazu, dass ein Benutzer in heutigen Systemlandschaften nicht nur eine Kennung mit einem Passwort verwalten muss, sondern sich eine Vielzahl von Kennungen mit zugehörigen Passwörtern merken muss. Dies führt dazu, dass Benutzer häufig das gleiche, meist auch noch schlechte Passwort als Zugangsausweis für verschiedene Dienste und Anwendungen festlegen oder sich die vielen Passwörter aufschreibt, um sie sich merken zu können. Beide Wege sind sicherheitskritisch.

Passwort-Vielzahl

10.2.2 Authentifikation in Unix

Das Unix Betriebssystem wird in diesem und im nächsten Kapitel als Fallbeispiel herangezogen, um die typischen Techniken der Authentifikation und Zugriffskontrolle exemplarisch zu verdeutlichen. vergleichbare Konzepte und findet man auch bei Linux und seinen Derivaten. Auch wenn einige der geschilderten konkreten Sachverhalte heute nicht mehr überall im Einsatz sind, so dienen sie doch weiterhin dazu, abstrakte Sachverhalte an konkreten Umsetzungen zu verdeutlichen. Dies ist das vordringliche Ziel der Fallbeispielbeschreibungen.

Im Rahmen dieses Buches kann Unix natürlich nicht mit allen seinen Konzepten und Systemdiensten vorgestellt werden, sondern wir müssen uns darauf beschränken, einige seiner Sicherheitsdienste zu beleuchten.

¹⁶ siehe <http://www.rsa.com/node.aspx?id=3891>

Grundlagen

Hintergrund

Sicherheitsfragen standen bei der Entwicklung des Mehrbenutzer- und mehrprozessfähigen Betriebssystems Unix nicht im Vordergrund. Unix wurde für Umgebungen wie Forschungslabors oder Universitäten konzipiert, in denen Benutzer offen miteinander kooperieren wollen und müssen, ohne durch Sicherheitskontrollen zu stark in ihrer Zusammenarbeit beeinträchtigt zu werden. Erst nachdem in den frühen 80er Jahren Unix immer stärker in den Bereichen der öffentlichen Verwaltung und in Firmen eingesetzt wurde, erhielten Sicherheitsfragen ein größeres Gewicht. Dies verstärkte sich noch durch den Übergang auf vernetzte Systeme, da dafür aus Sicherheitssicht zunächst keinerlei spezifische Vorkehrungen getroffen wurden. Analoge Probleme sind uns bereits im Zusammenhang mit den Sicherheitsproblemen des verteilten Dateisystems NFS (vgl. Abschnitt 3.4.2) oder der ungeschützten Übertragung sensitiver Informationen bei entfernten Zugriffen mittels FTP (vgl. Abschnitt 3.4.3) begegnet. Im Folgenden konzentrieren wir uns auf die klassischen Unix-Dienste zur Authentifikation. Mit dem Kerberos-System werden wir dann in Abschnitt 10.4.2 einen sehr wirksamen Authentifikationsdienst für vernetzte Systeme kennenlernen, der sowohl im Unix- als auch im Windows-Umfeld in der Praxis sehr häufig eingesetzt wird.

Systemarchitektur

monolithischer Kern

Unix ist ein monolithisches Betriebssystem, das alle Betriebssystemdienste als Prozeduren in einem großen und komplexen Systemkern zusammenfasst. Die Grobstruktur ist in Abbildung 10.1 skizziert. Der Betriebssystemkern enthält die Dienste zur Datei-, Prozess- und Prozessorverwaltung sowie zur virtuellen Speicherverwaltung, das Ein/Ausgabe- und Unterbrechungs-System sowie die Gerätetreiber. Die Systemdienste können von Prozessen über Systemaufrufe, wie beispielsweise `open`, `read`, `write`, `close` zum Zugriff auf das Dateisystem genutzt werden. Jeder Systemaufruf (engl. *trap* oder *system call*) führt zu einem Moduswechsel, mit der Konsequenz, dass die Systemdienste mit besonderen, privilegierten Berechtigungen ausgeführt werden.

Sicherheitsdienste

Im Kontext des vorliegenden Buches interessieren wir uns für die in der Abbildung grau unterlegten Dienste, also den Bereich der Benutzer-Authentifikation, der durch das Login-Programm `/bin/login` und den Kerndienst der Zugangskontrolle realisiert wird, sowie für den Bereich des Dateisystems mit der Zugriffskontrolle.

Identifikation und Authentifikation

Identifikation

Jeder Benutzer besitzt eine Benutzerkennung, die bis zu acht Zeichen lang ist. Systemintern wird jeder Benutzer über eine eindeutige 16-Bit Zahl, die `user id (uid)`, identifiziert. Es ist die Aufgabe des Systemadministrators,

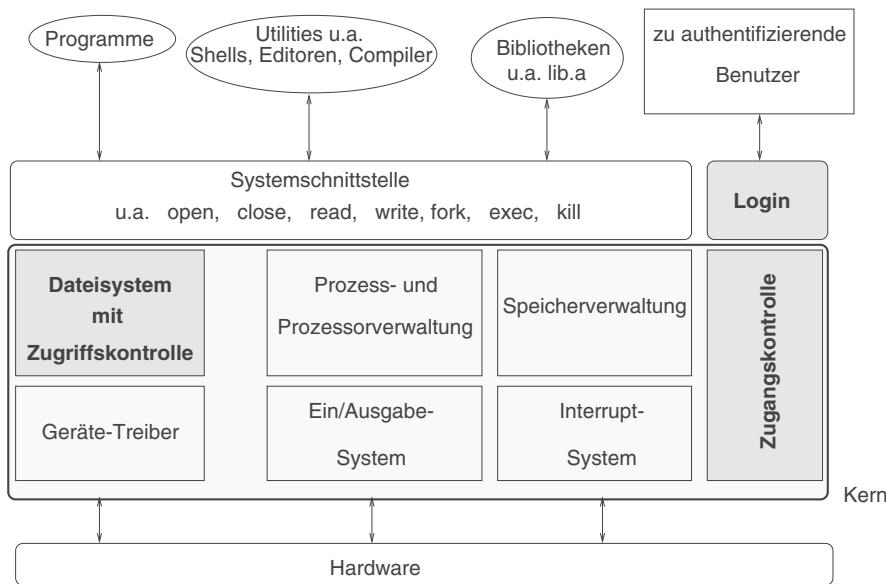


Abbildung 10.1: Grobarchitektur des Unix Betriebssystems

tors dafür zu sorgen, dass diese `uids` eindeutig sind. Standardmäßig sind die `uids` 0 bis 9 für die Systemadministration reserviert. So ist zum Beispiel die `uid` 0 der Kennung `root` mit Superuserberechtigungen zugeordnet. Jedem vom Benutzer gestarteten Prozess und jeder Datei, deren Eigentümer der Benutzer ist, wird dessen `uid` zugeordnet. Die `uid` eines Prozesses kennzeichnet damit den Benutzer, in dessen Auftrag der Prozess im System aktiv ist, während die `uid` einer Datei den Eigentümer der Datei identifiziert.

Zusätzlich besitzt jeder Benutzer eine Gruppenkennung, die intern ebenfalls durch eine eindeutige Zahl, die `group-id (guid)`, dargestellt wird. Eine Gruppe repräsentiert eine Zusammenfassung von Benutzern, an die Zugriffsrechte vergeben werden können. Beispiele sind die Gruppen der Studenten oder wissenschaftlichen Mitarbeiter in einem universitären Umfeld oder Gruppen für unterschiedliche Projektteams in einer Softwarefirma. Gruppen werden vom Systemadministrator eingerichtet und verwaltet. Die Datei `/etc/group` enthält die im System aktuell eingerichteten Gruppen.

Benutzerwählbare Passwörte

In herkömmlichen Unix Systemen erfolgt eine Authentifikation auf der Basis benutzerwählbarer Passwörte. Diese werden zusammen mit zusätzlicher Verwaltungsinformation standardmäßig in der Datei `/etc/passwd` gespeichert, die, wie alle Unix-Dateien, der Zugriffskontrolle (vgl. Kapitel 12.4) unterliegt. Jeder Eintrag in der Datei `/etc/passwd` besteht aus

`/etc/passwd`

sieben Feldern, die jeweils durch ein ':' getrennt sind. Ein /etc/passwd-Eintrag könnte beispielsweise wie folgt aussehen:

Feld 1	2	3	4	5	6	7
eckertc:	*	207:	1009:	Eckert:	/usr/wiss/eckertc:	tcsh

- Feld 1 Benutzerkennung: bis zu acht Zeichen lang.
- Feld 2 Verschlüsseltes Passwort (siehe Anmerkung unten): es ist 13 Zeichen lang. Steht, wie im Beispiel oben ein * an Stelle der 13 Zeichen, so bedeutet das, dass die Passworte der Datei in einer speziellen geschützten Datei gespeichert werden (siehe unten).
- Feld 3 uid: eindeutige Zahl, die den Benutzer intern identifiziert.
- Feld 4 guid: interne Darstellung der dem Benutzer zugeordneten Gruppenkennung.
- Feld 5 Kommentarfeld: enthält häufig, wie im obigen Beispiel, den vollständigen Namen des Benutzers, eine Aufgabenbeschreibung oder seine organisatorische Zugehörigkeit.
- Feld 6 Home Directory: das Heimatverzeichnis des Benutzers.
- Feld 7 Login Shell: der vom Benutzer gewählte Kommandointerpreter, den er für seine Interaktion mit dem System verwendet.

Anmerkung

Zum Passwort-Eintrag in Feld 2 ist anzumerken, dass dort streng genommen nicht das verschlüsselte Passwort steht, sondern eine mit dem Passwort als Schlüssel verschlüsselte Konstante (s.u.). Aus Vereinfachungsgründen behalten wir jedoch diese im Unix-Umfeld übliche Sprechweise bei.

Passwortverschlüsselung

crypt(3)

Zur Verschlüsselung der Passwörte dient die C-Funktion `crypt(3)`. Sie ist nicht mit dem Verschlüsselungsprogramm `crypt(1)` zu verwechseln, das zum Chiffrieren von Dateien standardmäßig in Unix Systemen zur Verfügung steht, aber nur eine relativ einfach zu brechende Substitutionschiffre realisiert. `crypt(3)` ist eine Variante des DES-Algorithmus (vgl. Kapitel 7.5.4), die zwei Eingabeparameter, nämlich `key` und `salt`, benötigt.

Schlüssel

Das benutzergewählte Passwort dient als Schlüssel, so dass es den Wert des `key`-Parameters bestimmt. Mit diesem Schlüssel wird ein 64-Bit Eingabeblock verschlüsselt, der eine Konstante bestehend aus Nullen ist. Die Ausgabe wird mit dem gleichen Schlüssel (Passwort) erneut verschlüsselt und dieser Vorgang wiederholt sich 25 mal. Das Ergebnis der iterierten

Verschlüsselung ist eine 64-Bit Ausgabe, die abschließend in 11 Zeichen transformiert wird. Das iterierende Verschlüsseln ist eigentlich ein Anachronismus und trägt heutzutage nicht mehr zur Sicherheitssteigerung des Verfahrens bei. Zum Zeitpunkt der Entwicklung des `crypt(3)`-Verfahrens hatte es zum Ziel, einen Angriff mit gewähltem Klartext erheblich zu verlangsamen.

Der Salt¹⁷ ist eine 12-Bit Zahl, die Werte zwischen 0 und 4095 annehmen kann. Der Saltwert einer Kennung wird von dem Programm `/bin/passwd` beim erstmaligen Einrichten der Kennung bzw. nach einem Passwortwechsel unter Einbeziehung der aktuellen Uhrzeit und der Prozessidentifikation erzeugt. Er ist in den ersten beiden Zeichen (als Klartext) des Passworteintrages (Feld 2) abgelegt. Der Salt beeinflusst die Verschlüsselung, so dass für Benutzer, die (zufällig oder absichtlich) das gleiche Passwort gewählt haben, dieses in unterschiedlichen Repräsentationen in der `/etc/passwd`-Datei erscheint. Der Salt-Wert dient ferner dazu, dass ein Benutzer das gleiche Passwort auf unterschiedlichen Rechnern verwenden kann, ohne dass dies anhand des Passwort-Eintrages direkt durch einen einfachen Vergleich von Hashwerten ersichtlich ist. Da durch den Saltwert jedes Passwort in 4096 verschiedenen Repräsentationen vorliegen könnte, wird ein Angriff mit gewählten Klartexten erschwert. Mit dem Saltwert wird eine von 4096 verschiedenen Verschlüsselungsvarianten ausgewählt. Die Varianten ergeben sich durch eine Veränderung der Expansionsabbildung E des DES (vgl. Tabelle 7.4 auf Seite 322), die in jeder der 16 DES-Runden benötigt wird, um einen 32-Bit Block auf einen 48-Bit Block zu erweitern.

Salt

Salt-Einfluss

Authentifikation

Beim Login muss ein Benutzer zunächst seine Kennung und danach sein Passwort angeben, das, wie oben beschrieben, als Schlüsselparameter der `crypt(3)`-Funktion dient. Der benötigte Wert des Saltparameters wird aus der `/etc/passwd`-Datei ausgelesen. Die 11 Bytes der Ausgabe des Verschlüsselungsschrittes werden dann mit den restlichen 11 Bytes des Passworteintrags in der `/etc/passwd`-Datei verglichen (vgl. Abbildung 10.2). Bei der Authentifikation muss also keine Entschlüsselung stattfinden, so dass die Klartextpasswörte nicht gespeichert werden.

Authentifikation

Sicherheit

Auf das Unix-Authentifikationskonzept treffen alle generellen Probleme einer passwortbasierten Zugangskontrolle zu. Dazu gehören die Gefahr einer schlechten Passwortwahl oder der unachtsame Umgang mit Passworten durch die Benutzer außerhalb des IT-Systems. Für klassische Unix Systeme

Probleme

¹⁷ Der Salt ist vergleichbar mit dem Seed beim S/Key oder RSA-SecureID-Verfahren.

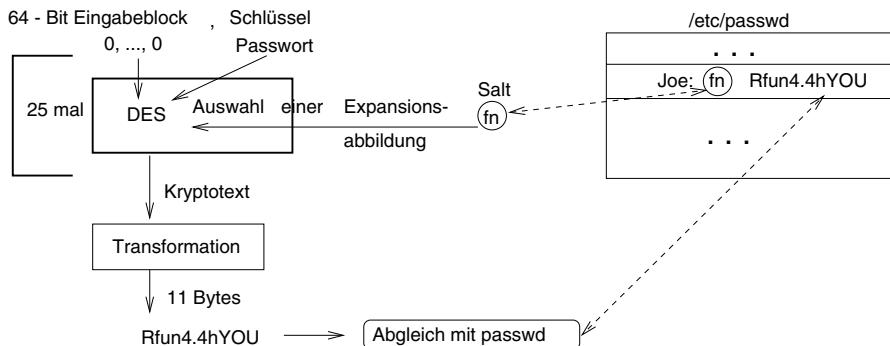


Abbildung 10.2: Authentifikation in Unix Systemen

gilt darüber hinaus, dass die Datei `/etc/passwd` von jedermann lesbar (world readable) ist. Da ferner der Algorithmus¹⁸ `crypt(3)` von jedem Benutzer aufgerufen werden kann, ist es möglich, geratene Passwörte zu verschlüsseln und mit den in der `/etc/passwd` abgespeicherten Daten zu vergleichen. Um solche Passwort-Cracking Angriffe zu verhindern, sollte also der Lesezugriff auf die `/etc/passwd`-Datei beschränkt werden. Weiterhin ist die unautorisierte Modifikation dieser Datei zu verhindern. Dazu muss der Systemadministrator darauf achten, dass nur die Kennung `root` Schreibrechte an der `/etc/passwd`-Datei und an dem Verzeichnis `/etc` erhält. Andernfalls könnte nämlich ein Angreifer als autorisierter Systembenutzer die Passwortdatei durch eine neue Passwortdatei ersetzen, in der er z.B. für die Kennung `root` ein ihm bekanntes Passwort einge tragen hat, so dass er Superuserrechte erlangen kann. Da beim regulären Ändern eines Benutzerpasswortes der Benutzer schreibenden Zugriff auf diese Datei benötigt, würde man ohne ein Zusatzkonzept eine unbrauchbare Lösung schaffen. Das unter Unix verwendete Konzept hier ist das setuid-Bit, durch das temporär Rechte an Dateien vergeben werden können (vgl. Abschnitt 12.4.2).

Passwort Shadowing:

Schattendatei

Um Passwort-Cracking Angriffe abzuwehren, verwenden heutige Unix Systeme in der Regel das Konzept einer Schattenpassworddatei, das `shadowed password file`. Diese enthält die verschlüsselten Passwörte, die dann aus der `/etc/passwd`-Datei entfernt werden. Die Berechtigung zum lesenden Zugriff auf die Schattendatei erhält nur die Kennung `root`. Unter HP-UX wird beispielsweise beim Übergang zu einem so genannten Trusted (secure) System Modus eine neue Passwortdatei `.secure/etc/passwd`¹⁹ erzeugt und die verschlüsselten Passwörte

¹⁸ Alle Unix Systeme nutzen die gleiche `crypt`-Funktion.

¹⁹ Unter Linux oder Solaris heißt diese Datei `/etc/shadow`.

werden aus `/etc/passwd` entfernt und in diese neue Datei kopiert. In `/etc/passwd` wird an Stelle des Passworts ein `*` eingetragen²⁰. Die Datei `.secure/etc/passwd` ist nur mit Superuserrechten zugreifbar.

10.2.3 Challenge-Response-Verfahren

Challenge-Response-Techniken (CR) sind Verallgemeinerungen von Authentifizierungsverfahren, die auf Wissen beruhen. Mit dem Einmal-Passwortverfahren haben wir bereits eine spezielle Ausprägung von Challenge-Response-Verfahren kennen gelernt. Die wesentliche Idee bei den CR-Verfahren ist wie bei den Einmal-Passworten, dass das geheime Wissen nicht mehrfach zum Login-Rechner übermittelt wird, sondern dass bei jedem Login ein neues Passwort berechnet wird. Dabei wird dem zu authentifizierenden Subjekt eine Frage bzw. eine Folge von Fragen vorgelegt, die es zu beantworten hat. Solche Frage/Antwort-Protokolle können im einfachsten Fall als Erweiterung des herkömmlichen Login-Vorganges realisiert sein. Dazu vereinbart ein Benutzer nicht nur ein geheimes Passwort mit dem System, sondern eine Folge von Frage/Antwort-Paaren.

Die bekannte PIN/TAN-Technik beim Homebanking stellt eine Variante dieser Vorgehensweise dar. Diese sehr einfache Vorgehensweise birgt jedoch eine Reihe von Problemen. Da der Klartextraum, aus dem das authentifizierende System seine Challenges auswählt, sehr klein ist, kann ein Angreifer durch Abhören der Leitungen die gesendeten Anfragen und zugehörigen Antworten abspeichern. Aufgrund des kleinen Raumes, wird das System mit großer Wahrscheinlichkeit, eine bereits früher gesendete Challenge wiederholen. Der Angreifer ist dann im Besitz der korrekten Antwort und kann seinen Spoofing-Angriff erfolgreich durchführen.

Als Lösung für dieses Problem wird in heutigen Systemen in der Regel eine Zufallszahl als Challenge generiert, so dass die Wahrscheinlichkeit der erneuten Verwendung einer Zufallszahl von der Güte des verwendeten Zufallszahlengenerators abhängt.

Challenge-Response-Verfahren werden sehr häufig in Authentifikationsprotokollen eingesetzt, die zwischen zwei oder mehr Geräten abgewickelt werden. Beispiele hierfür sind die Authentifikation in Mobilfunknetzen, wobei sich Mobiltelefone unter GSM oder UMTS gegenüber dem Netz authentifizieren, oder aber auch die Authentifikation von Endgeräten gegenüber einem Access Point in drahtlosen lokalen Netzen, den WLANs (vgl. Kapitel 15.4). Die Technik des Challenge-Response wird jedoch heutzutage auch in vielfältigen anderen Szenarien eingesetzt, um beliebige Geräte zu authentifizieren. So prüfen manche Mobiltelefonhersteller, ob die im Gerät

Frage/Antwort-Spiel

Probleme

Einsatz

²⁰ Unter Linux/Solaris enthält die Passworddatei einen X-Entrag anstelle des `*`.

verwendete Batterie von einem zertifizierten Hersteller stammt, manche Drucker prüfen, ob die eingesetzte Toner-Kasette authentisch ist, oder eine Sony Playstation prüft, ob die Speicherplatte von Sony stammt.

Im Folgenden werden allgemeine Challenge-Response-Abläufe zum einen basierend auf symmetrischen und zum anderen basierend auf asymmetrischen Kryptosystemen erklärt. Wir betrachten jeweils ein Szenario, in dem ein Client sich gegenüber einem Partner, worin ihn den Server, authentifizieren möchte.

Verfahren mit symmetrischen Kryptosystemen

Basis

Voraussetzung für die Durchführung einer Authentifikation ist, dass die beteiligten Geräte den gleichen Verschlüsselungsalgorithmus E sowie den gleichen Schlüssel, der z.B. aus dem Passwort des Benutzers abgeleitet sein kann, verwenden. Sei CID die Identifikation des Clients und K_{CID} der verwendete Schlüssel, der mit dem Client vereinbart ist. Sei K_r der Schlüssel, den der Server lokal zur Kommunikation mit dem Client CID in einer Schlüsseldatenbank gespeichert hat. Ist der Client authentisch, so gilt natürlich $K_{CID} = K_r$.

Authentifikation

Beim Login sendet der Client eine Login-Anfrage an den Server, in der er seine Kennung CID bekannt gibt. Der Server sucht den zu CID in seiner Schlüsseldatenbank abgelegten Schlüssel K_r . Falls ein Eintrag vorliegt, stellt der Server dem zu authentifizierenden Client eine Frage (Challenge), indem er ihm eine neu erzeugte Zufallszahl $RAND$ zusendet. Der Client muss $RAND$ mit dem vereinbarten Verfahren E und ihrem Schlüssel K_{CID} verschlüsseln,

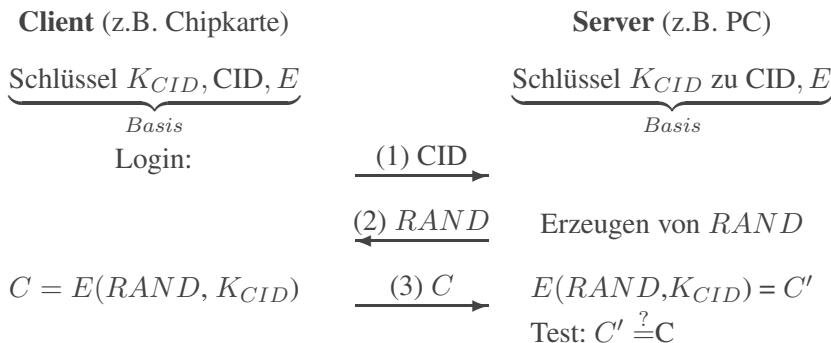
$$C = E(RAND, K_{CID}).$$

Response

Das Ergebnis C dieser Verschlüsselung wird an den Server gesendet (Response). Dieser verschlüsselt seinerseits die Zufallszahl $RAND$ mit dem ihm bekannten Schlüssel K_r ,

$$C' = E(RAND, K_r).$$

Der Client hat die Challenge korrekt beantwortet, wenn Gilt, $C' = C$. Die Schritte sind nachfolgend noch einmal zusammengefasst. Da der Server in der Regel die Kryptonachricht C nicht entschlüsselt, um die Authentizität zu prüfen, werden an Stelle von symmetrischen Kryptoverfahren auch häufig MAC-Verfahren verwendet.



Der skizzierte Ablauf kann noch erweitert und zur wechselseitigen Authentifikation eingesetzt werden, so dass sich der Server dann auch gegenüber dem Client authentifiziert. In diesem Fall stellt der Client die Challenge, die vom Server korrekt zu beantworten ist.

Sicherheit

Beachtenswert an dem vorgestellten Verfahren ist, dass das Passwort des Benutzers zur Authentifikation nicht übermittelt werden muss. Da bei jeder Authentifizierung eine andere Zufallszahl $RAND$ verwendet wird, kann ein Angreifer analog zum Einmal-Passwort-Verfahren einen abgefangenen und bereits zur Authentifikation verwendeten Kryptotext $E(RAND, K_{CID})$ nicht erfolgreich wieder einspielen, um sich als CID zu maskieren. Wie beim Einmal-Passwort-Verfahren besteht aber auch hier die Gefahr, dass ein Angreifer die Verbindung zwischen dem zu authentifizierenden und dem authentifizierenden System kontrolliert. Da symmetrische Challenge-Response Verfahren heutzutage vielfach auch in offenen Umgebungen eingesetzt werden, erfolgt der Austausch von Challenges und zugehöriger Responses häufig über unsichere und sehr einfach abzuhörende Funkschnittstellen.

Dies eröffnet vielfältige Möglichkeiten für Known-Plaintext-Angriffe. Hierbei zeichnet der Angreifer die Klartext-Nachricht $RAND$ sowie den zugehörigen Kryptotext, die Response, auf und versucht mittels kryptoanalytischer Techniken den verwendeten geheimen Schlüssel abzuleiten. Derartige Angriffe werden beispielsweise in den GSM-Systemen erfolgreich durchgeführt.

Mit den abgehörten Klartext-Kryptotext-Paaren lassen sich aber auch Wörterbuchangriffe durchführen, wie man sie in herkömmlichen Passwort-Systemen kennt. Dazu rät der Angreifer den verwendeten geheimen Schlüssel, verschlüsselt den $RAND$ -Wert mit diesem geratenen Schlüssel und vergleicht sein Ergebnis mit der Response des berechtigten Benutzers. Zur Abwehr derartiger Angriffe kann man zum einen die Protokolle erweitern, so

RAND

Maskierung?

Known-Plaintext-Angriff

Wörterbuch-Angriff

dass die Challenge verschlüsselt übermittelt wird (u.a. [17]) oder man setzt Zero-Knowledge-Verfahren ein (vgl. Abschnitt 10.2.4).

geheime Schlüssel

Problematisch ist, dass der authentifizierende Server für jeden autorisierten Client sensitive Informationen, nämlich den gemeinsamen geheimen Schlüssel, sicher zu verwalten hat. Der Server wird als vertrauenswürdige Instanz angesehen; dennoch besteht ein Risiko, dass ein unberechtigter Dritter, z.B. durch das ausnutzen einer Schwachstelle Zugriff auf die sensitive Information erlangen kann. Um Server von der Verwaltung geheimer Schlüssel zu entlasten und die damit verbundenen Sicherheitsprobleme zu verringern, kann man asymmetrische Kryptosysteme zur Challenge-Response-Authentifikation verwenden.

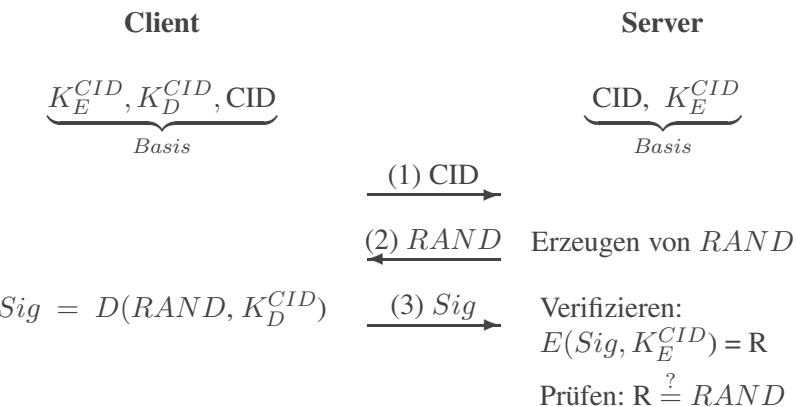
Verfahren mit asymmetrischen Kryptosystemen

Basis

Beim Einsatz asymmetrischer Verfahren benötigt der Server den öffentlichen Schlüssel K_E^{CID} des Clients CID .

Authentifikation

Nach einer Login-Anfrage erzeugt der Server auch hier eine Zufallszahl $RAND$, die er als Challenge an den Client sendet. Dieser signiert die Zahl $RAND$ mit seinem geheimen Schlüssel K_D^{CID} und sendet die digitale Signatur $Sig = D(RAND, K_D^{CID})$ als Response zurück. Der Server verifiziert die Signatur durch Anwendung des öffentlichen Schlüssels K_E^{CID} des Clients und prüft, ob nach dieser Entschlüsselung die Zufallszahl $RAND$ vorliegt und somit die Signatur eine aktuelle Antwort auf die Anfrage ist. Die Schritte sind nachfolgend noch einmal zusammengefasst.



Sicherheit

Der Vorteil des asymmetrischen Verfahrens gegenüber dem symmetrischen besteht darin, dass Server und Client keine Geheimnisse vorab absprechen müssen, die der Server zu speichern hat. Für die Sicherheit des Verfahrens ist aber wesentlich, dass die öffentlichen Schlüssel authentisch sind, wozu

öffentlicher Schlüssel

eine Public-Key-Infrastruktur (PKI) benötigt wird. Ferner muss der Server die Integrität der gespeicherten öffentlichen Schlüssel sicherstellen, so dass ein Angreifer nicht erfolgreich einen authentischen Schlüssel gegen einen gefälschten austauschen kann. Alternativ kann der Server bei jedem Login neben der Response auch ein gültiges Zertifikat des Clients einfordern und anhand dessen die Gültigkeit des öffentlichen und damit auch privaten Schlüssels stets aufs Neue überprüfen. Eine Variante dieser Vorgehensweise wird beispielsweise im SSL/TLS-Protokoll angewandt.

Für beide vorgestellten Authentifizierungstechniken benötigt man gute Zufallszahlengeneratoren. Werden durch einen schlechten Generator beispielsweise Zufallszahlen $RAND$ in kurzen Abständen periodisch wiederholt, so muss ein Angreifer nur Frage/Antwort-Paare sammeln und auf die periodische Wiederholung einer gespeicherten Frage warten, um die abgefangene Antwort auf die so abgepasste Challenge einzuschleusen.

Probleme können auch durch eine fehlende wechselseitige Authentifikation auftreten. Dadurch ist es möglich, dass sich ein Angreifer gegenüber dem Client als Server maskiert und eine beliebige, geratene Challenge $RAND$ stellt. Mit den aufgezeichneten, korrekt verschlüsselten Antworten des gutgläubigen Clients kann er anschließend eine eingehende Kryptoanalyse durchführen, um den Schlüssel zu brechen. Eine fehlende Server-seitige Authentifizierung führt u.a. in Mobilfunknetzen oder auch in WLANs zu Problemen.

Zufallszahlen-generator

Man-in-the Middle

10.2.4 Zero-Knowledge-Verfahren

Zero-Knowledge-Verfahren kann man als spezielle Challenge-Response-Techniken betrachten. Das zugrunde liegende Problem lautet informell wie folgt: „Wie kann eine Teilnehmerin Alice einen anderen Teilnehmer Bob davon überzeugen, dass sie ein Geheimnis s kennt, ohne dass Alice auch nur einen Teil dieses Geheimnisses Bob oder einem Dritten preisgibt?“

Problem

Von Protokollen, die dieses Problem lösen, fordert man, dass der Teilnehmer Bob keine geheimen Informationen über Alice speichern muss und dass ein Angreifer beliebig viele Kommunikationsschritte beobachten darf, ohne dass er dadurch das Geheimnis s erfährt oder es ermitteln kann. Fiat und Shamir haben in [64] eines der ersten Zero-Knowledge-Verfahren vorgestellt, dessen Grundzüge im Folgenden besprochen werden.

Fiat-Shamir Verfahren

Obwohl das Fiat-Shamir-Verfahren Schlüssel-Paare, bestehend aus öffentlichen und geheimen Schlüsseln, verwendet, erfüllt es nicht die Anforderun-

gen, die Definition 7.7 an ein asymmetrisches Kryptosystem stellt, und kann deshalb nicht zum Ver- und Entschlüsseln verwendet werden.

Szenario

Zur Beschreibung des Verfahrens gehen wir von einem Szenario aus, in dem eine vertrauenswürdige Instanz Z existiert, die die Aufgabe hat, Smartcards an Teilnehmer auszugeben. Die Teilnehmer müssen sich dazu der Instanz gegenüber zweifelsfrei identifizieren. Diese Schritte sind jedoch nicht Bestandteil des Protokolls. Die Instanz schreibt die für die spätere Authentifikation benötigten Informationen auf die Chipkarte. Das Zielsystem, zu dem ein Benutzer Zugang wünscht, verfügt über ein Smartcardlesegerät. Hier gilt ebenfalls, dass das Verfahren auch in anderen Szenarien anwendbar ist, in denen z.B. die Chipkarte durch einen PC ersetzt wird.

Basis

Die Instanz Z wählt zwei große Primzahlen p und q und bildet das Produkt $n = p \cdot q$. p und q werden geheim gehalten, während n und eine Funktion

$$f : \{0, \dots, n - 1\} \times \mathbb{N} \longrightarrow \{0, \dots, n - 1\}$$

allen Teilnehmern, also hier allen Smartcards und allen authentifizierenden Rechnern bekannt sind.

Schlüssel

Sei $I \in \{0, \dots, n - 1\}$ eine Zahl, die eine Chipkarte eindeutig repräsentiert (z.B. als Funktion über deren Verfallsdatum und Seriennummer). Aus I berechnet Z eine Folge von Zahlen

$$v_1 = f(I, 1), v_2 = f(I, 2) \dots$$

und wählt davon $k \in \mathbb{N}$ aus. Für jedes gewählte v_{i_j} muss gelten, dass es ein s_j gibt, mit

$$s_j^2 \cdot v_{i_j} \equiv 1 \pmod{n}.$$

Da die Instanz Z die Primzahlen p und q kennt, ist es für sie einfach, das Inverse $v_{i_j}^{-1}$ zu $v_{i_j} \pmod{n}$ und damit $s_j = \sqrt{v_{i_j}^{-1}}$ zu berechnen.

Die $\{s_1, \dots, s_k\}$ sind die geheimen und die $\{v_{i_1}, \dots, v_{i_k}\}$ sind die öffentlichen Schlüssel des Kartenbesitzers. Die Kennzahl I und die Indizes i_1, \dots, i_k sind frei lesbar auf der Karte gespeichert, während die geheimen Schlüssel vor unauthorisierten Lesezugriffen geschützt abgelegt werden.

Notation

Zur Vereinfachung der Schreibweise wird o.B.d.A. angenommen, dass gilt: $\{v_{i_1}, \dots, v_{i_k}\} = \{v_1, \dots, v_k\}$.

Authentifikation

Bei einer Authentifikation beweist die Chipkarte mit der Kennung I , dass sie die Geheimnisse s_1, \dots, s_k kennt, ohne jedoch deren Werte preiszugeben. Der Ablauf des Protokolls ist nun wie folgt:

Protokoll

1. Die Kennung I der Chipkarte sowie die Indizes i_1, \dots, i_k werden gelesen und zum authentifizierenden Rechner R übertragen.

2. Für $j = 1, \dots, k$ berechnet R den Wert

$$v_j = f(I, j).$$

Damit kennt R die öffentlichen Schlüssel der Karte I ohne eine dritte Instanz befragen zu müssen.

Danach werden die Schritte 3 bis 6 t -mal durchlaufen, wobei t eine frei wählbare, natürliche Zahl ist.

3. Sei t festgelegt und $m \in \{1, \dots, t\}$. I wählt eine Zufallszahl $r_m \in \{0, \dots, n - 1\}$ und berechnet:

$$x_m \equiv r_m^2 \bmod n.$$

Der Wert x_m wird an R übertragen.

4. R sendet einen k -stelligen binären Zufallsvektor $(e_1, \dots, e_k) \in \mathbb{IB}^k$ an I , d.h. $e_j \in \{0, 1\}, j \in \{1, \dots, k\}$. Challenge

Dieser Vektor entspricht einer Challenge, da R damit auswählt, welche der k geheimen Schlüssel die Chipkarte I anwenden muss, um ihre Authentizität zu beweisen. D.h. für jedes e_j , für das gilt, $e_j = 1$, muss I in einer korrekten Response den geheimen Schlüssel s_j anwenden.

5. I antwortet mit:

$$y_m \equiv r_m \prod_{e_j=1} s_j \bmod n.$$

Response

6. R überprüft für den Wert y_m und für den in Schritt (3) von I erhaltenen Wert x_m die folgende Äquivalenz:

$$x_m \stackrel{?}{=} y_m^2 \prod_{e_j=1} v_j \bmod n.$$

Nur die Chipkarte I , die die geheimen Schlüssel s_j kennt, konnte den Wert y_m korrekt so berechnen, dass gilt:

$$\begin{aligned} y_m^2 \prod_{e_j=1} v_j \bmod n &\equiv (r_m \cdot s_1^{e_1} \dots s_k^{e_k})^2 \cdot v_1^{e_1} \dots v_k^{e_k} \bmod n \\ &\equiv r_m^2 \cdot (s_1^2 \cdot v_1)^{e_1} \dots (s_k^2 \cdot v_k)^{e_k} \bmod n \\ &\equiv r_m^2 \cdot 1 \dots 1 \bmod n \\ &\equiv r_m^2 \bmod n \equiv x_m. \end{aligned}$$

Sicherheit

Ein Angreifer, der sich als Subjekt mit der Kennung I maskieren möchte und dessen öffentliche Schlüssel $\{v_1, \dots, v_k\}$ kennt, kann versuchen, eine Challenge (e_1, \dots, e_k) im Voraus zu erraten. Dann wählt er sich ein y_m ,

Maskierung?

berechnet den zugehörigen x -Wert:

$$(*) \quad x_m \equiv y^2 \cdot v_1^{e_1} \cdots v_k^{e_k} \bmod n$$

und lässt sich diesen Wert von R verifizieren. Das gelingt immer dann, wenn R tatsächlich den Vektor (e_1, \dots, e_k) auswählt, den der Angreifer zur Berechnung in $(*)$ geraten hat. Es stellt sich somit die Frage, wie groß die Wahrscheinlichkeit dafür ist, dass dieser Angriff erfolgreich sein kann. Die Wahrscheinlichkeit, dass R eine falsche Karte akzeptiert, ist kleiner als 2^{-kt} , weil der Angreifer alle Werte der von R versendeten Bitvektoren (e_1, \dots, e_k) korrekt voraussagen muss. Da für jedes e_i zwei Möglichkeiten infrage kommen, ist die Anzahl der möglichen k -Vektoren gleich 2^k . Damit ist die Wahrscheinlichkeit, in einer Runde den richtigen Wert zu raten, gleich 2^{-k} und bei t Runden also 2^{-kt} .

Das Fiat-Shamir-Verfahren minimiert die Anzahl der Berechnungen, die ein zu authentifizierender Teilnehmer pro Iterationsschritt durchführen muss. Dafür ist jedoch eine große Anzahl von Iterationen notwendig, wobei jede Iteration den aufwändigen Austausch von Informationen zwischen den Authentifikationspartnern erfordert. Das ist allerdings gerade für Smartcards, für die ja eine solche Authentifikationstechnik von großem Interesse ist, keine ideale Situation. Verbesserungen liefern hier u.a. die Verfahren von Guillou und Quisquater [146]. Diese beschränken sich auf nur einen Iterationsschritt, der jedoch eine aufwändigere Berechnung als beim Fiat-Shamir-Verfahren erfordert.

10.3 Biometrie

In diesem Abschnitt behandeln wir Authentifikationstechniken, die auf biometrischen Merkmalen (griechisch *bios* = Leben und *metron* = Mass) basieren. Unter einem solchen Merkmal versteht man physiologische oder verhaltenstypische Eigenschaften einer Person, die diese eindeutig charakterisieren, wie zum Beispiel Fingerabdrücke.

10.3.1 Einführung

Der Einsatzbereich biometrischer Technologie (u.a. [137]) ist weit gefächert und reicht vom Bereich der Strafverfolgung, der Grenzkontrollen und der Terrorismusfahndung, über Eintrittskontrollen für Hochsicherheitsbereiche bis hin zur Kontrolle der Zugriffe auf Datenbestände, zur Kontrolle der Initiierung von Transaktionen beispielsweise an einem Geldautomat, oder zur Aktivierung von Smartcards und Handys anstelle einer herkömmlichen PIN-Authentifikation.

Techniken zur eindeutigen Identifikation von Personen anhand biometrischer Merkmale haben bereits eine längere Entwicklungsgeschichte hinter sich. Ihre breite Akzeptanz im Massenmarkt wurde jedoch über lange Zeit durch die hohen Kosten, die mit der benötigten Geräteausstattung verknüpft waren, gebremst. Die Fortschritte bei der Entwicklung der Biometriegeräte sorgen aber mittlerweile dafür, dass diese Technologie auch im Massenmarkt Fuß fasst, zum Beispiel als Fingerabdrucksensoren als PIN-Ersatz bei Mobiltelefonen, in Tastaturen oder aber auch in Schlüsseln für hochpreisige Automobile.

biometrisches
Merkmal

Biometrische Techniken zur Authentifikation und Zugangskontrolle kann man grob in zwei Klassen einteilen. Wir unterscheiden Verfahren, die auf physiologischen, statischen Eigenschaften einer Person beruhen und verhaltenstypische, dynamische Verfahren. Beispiele für physiologische Eigenschaften sind die Iris oder die Retina, das Gesicht oder Fingerabdrücke. In der Forschung werden auch bereits weitere physiologische Eigenschaften wie der Geruch oder der Hautwiderstand untersucht. Verhaltenstypisch ist beispielsweise ein spezifisches Tippverhalten (z.B. Rhythmus), die Stimme oder die Dynamik einer handschriftlichen Unterschrift. Noch im Forschungsstadium befinden sich Untersuchungen auch anderer Verhaltensmuster wie beispielsweise das Sitzverhalten einer Person. In der Praxis sind heute noch überwiegend Geräte im Einsatz, die sich auf physiologische Merkmale beziehen.

statisch

dynamisch

An ein biometrisches Merkmal, das zur Authentifikation von natürlichen Personen eingesetzt werden kann, werden folgende allgemeine Anforderungen gestellt.

Anforderungen

Universalität: Jede Person besitzt das biometrische Merkmal.

Eindeutigkeit: Das biometrische Merkmal ist für jede Person verschieden.

Beständigkeit: Das Merkmal ist unveränderlich.

Quantitative Erfassbarkeit: Das Merkmal lässt sich mittels Sensoren quantitativ erfassen.

Performanz: Die Erfassung des Merkmals kann mit einer erforderlichen Genauigkeit erfolgen und der Vorgang ist performant durchführbar.

Akzeptanz Die Verwendung des Merkmals wird von Benutzern akzeptiert.

Fälschungssicherheit Das Merkmal ist fälschungssicher.

Die körpereigenen Merkmale erfüllen die aufgelisteten Anforderungen in unterschiedlichem Maß, so dass die Qualität eines biometrischen Verfahrens auch von der Qualität abhängt, mit der die Anforderungen erfüllt werden.

Beispiel 10.5 (Fingerabdruck)

Fingerabdruck

Betrachtet man als Beispiel das biometrische Merkmal *Fingerabdruck*. Techniken zum Scannen von Fingerabdrücken sowie zum automatischen Abgleich von Fingerabdrücken und Referenzdaten wurden bereits Ende 1960 vom amerikanischen FBI eingesetzt. Fingerabdruckverfahren haben somit ihren Ursprung zwar im Bereich der Verbrechensbekämpfung, sind aber durch ihr zunehmend besser werdendes Preis/Leistungsverhältnis geeignet, auch im Massenmarkt als Zugangskontroll-Technologie für PCs, Smartcards oder Handys eingesetzt zu werden. So wurde in das iPhone 5S von Apple ein Fingerabdrucksensor integriert. Auf die biometrische Authentifikation mittels Fingerabdrücke geht Abschnitt 10.3.4 noch genauer ein.

Das biometrische Merkmal Fingerabdruck erfüllt die Anforderungen der Universalität und Eindeutigkeit (auch eineiige Zwillinge besitzen unterschiedliche Fingerabdrücke) und es ist beständig, da es sich ohne Fremdeinwirkungen während der Lebenszeit des Menschen nicht verändert. Durchaus problematisch ist dagegen die Erfüllung der Anforderungen der quantitativen Erfassbarkeit und Performanz. So sind die Fingerabdrücke nicht bei jedem Menschen mittels Sensoren tatsächlich quantitativ erfassbar, weil sie beispielsweise zu wenige der erfassbaren Charakteristika aufweisen. Zum anderen ist die Anforderung, dass die Erfassung mit der erforderlichen Genauigkeit zu erfolgen hat, sehr unpräzise. Gibt sich das Erkennungssystem mit der Erfassung sehr weniger Fingerabdruck-Charakteristika zufrieden (z.B. aus Effizienzgründen), so kann es bei der Überprüfung der Fingerabdrücke zu Fehlern kommen und ggf. auch Unberechtigten der Zugang gewährt werden. Wodurch natürlich auch gleichzeitig die Anforderung der Fälschungssicherheit in Frage gestellt wird. Abhängig von der Qualität einer Fingerabdruck-Technik kann diese bereits durch die Nutzung von Gummifingern etc. überlistet werden. Da Fingerabdrücke öffentliche Daten sind, die der Mensch vielfältig hinterlässt, hängt die Fälschungssicherheit des Systems auch davon ab, inwieweit ein Angreifer solche öffentliche Daten einer autorisierten Person unberechtigterweise als seine eigenen Daten ausgeben kann. Die Anforderungen der Benutzer-Akzeptanz scheint dagegen durch Fingerabdruck-Verfahren erfüllt zu sein, da im Gegensatz zum Scannen der Retina kaum Hemmschwellen bestehen, den Finger auf einen Sensor zur Überprüfung zu legen.



10.3.2 Biometrische Techniken

Handgeometrie

Die ersten kommerziellen Erfolge erzielte 1970 ein biometrisches System, das zur Zugangskontrolle zu Hochsicherheitsbereichen eingesetzt wurde und die Handgeometrie, insbesondere die Fingerlänge, der zu authentifizierenden

Personen maß. Techniken zur Analyse der Handgeometrie werden heute in vielfältiger Weise verwendet, wie zum Beispiel auf dem New Yorker John F. Kennedy Flughafen. Dort können sich Geschäftsreisende, die häufiger in die USA fliegen, lange Schlangen an der Passkontrolle ersparen, indem sie eine Smartcard verwenden, die den Referenzwert ihrer Handgeometrie enthält. Bei der Einreise muss nur noch die Hand auf ein Lesegerät gelegt werden, das den aktuellen mit dem gespeicherten Wert vergleicht.

Zu den Ansätzen, die den höchsten Grad an Sicherheit bieten, zählen biometrische Verfahren, die das menschliche Auge analysieren. Hier unterscheidet man irisorientierte und retinaorientierte Systeme. Charakteristisch für beide Merkmale ist, dass sie nach dem Tod sehr schnell zerstört werden. Ab Mitte 1980 wurden Geräte entwickelt, die die eindeutigen Muster der Retina analysieren. Die Analyse der Retina stellt, zusammen mit den heute bereits im Einsatz befindlichen Techniken zur Irisanalyse, die qualitativ hochwertigste Möglichkeit zur biometrischen Authentifikation dar.

Iris und Retina

Die Iris des Auges umfasst 266 charakteristische Attribute, die von Person zu Person variieren können und auch bei einigen Zwillingen unterschiedlich sind. Mit einer Videokamera wird die Regenbogenhaut aus einem sehr kurzen Abstand von ca. 20 cm aufgenommen und aus dem Film werden die angesprochenen Charakteristika abgeleitet. Kontaktlinsen oder ungünstige Lichtverhältnisse haben keinen Einfluss auf die Güte des Erkennungssystems (u.a. [132]). Die Retina beschreibt die Blutgefäßanordnung des Auges und gilt genau so wie die Iris als unverfälschbares Merkmal einer Person. Zur Erfassung der Gefäßstruktur muss die zu authentifizierende Person einige Sekunden auf einen festen Punkt im Analysegerät blicken, so dass ein Abbild der Gefäßanordnung erstellt werden kann.

Ein Beispiel für ein solches System, IrisIdent vom amerikanischen Hersteller IriScan²¹, wurde beispielsweise bei den Olympischen Winterspielen im japanischen Nagano eingesetzt, um den Zugang zur Waffenkammer der Sportschützen zu kontrollieren.

Eine klassische Technik, mit der Menschen sich gegenseitig wiedererkennen und authentifizieren, basiert auf der Erkennung von Gesichtern. Bei entsprechenden biometrischen Geräten handelt es sich um Gesichtserkennungssysteme, die in der Lage sind, eine Identifikation auch auf größere Distanz durchzuführen. Die Identifikation einer Person anhand ihrer Gesichtszüge [140] ist schwierig, da sich das Aussehen von Personen häufig durch unterschiedliche Mimik, Brillen, veränderte Frisuren etc. ändert. Zur Gesichtserkennung werden meist handelsübliche Videokameras eingesetzt, um zunächst für jede später zu authentifizierende Person mehrere Referenz-

Gesichtserkennung

²¹ <http://iriscan.com>

werte aufzunehmen. Dies ist notwendig, da unterschiedliche Lichteinflüsse oder der Aufnahmewinkel die Güte des Systems beeinflussen können. Produkte, die zu sehr niedrigen Preisen zurzeit bereits auf dem Markt angeboten werden, zeigen bei der zuverlässigen Authentifikation noch erhebliche Schwächen, so dass sie noch nicht reif für den Einsatz in sicherheitsrelevanten Bereichen sind. So sind einige Produkte bereits durch die Vorlage eines Fotos zu überlisten. Verbesserungen lassen sich durch den Einsatz thermografischer Verfahren erzielen. Mit einer Infrarotkamera wird dabei ein Profil aufgenommen, das durch die Wärmeabstrahlung der Blutgefäße des Gesichts entsteht. Durch die Verwendung von Infrarotkameras ist diese Erkennungstechnik zwar unabhängig von den herrschenden Lichtverhältnissen, dafür aber erheblich teurer als der Einsatz von Standardkameras. Weitere Verbesserungen bei der Qualität der Erkennungsleistung lassen sich durch 3D-Verfahren erzielen. Trotz der nach wie vor erheblichen Schwächen, werden Gesichtserkennungssysteme bereits heute in verschiedenen Bereichen, die jedoch keine hohen Sicherheitsanforderungen besitzen, eingesetzt. Ein Beispiel ist das Sozialamt in Toronto. Hier werden derartige biometrische Techniken eingesetzt, um eine mehrfache Auszahlung von Unterstützungs-geldern an Sozialhilfeempfänger zu verhindern.

Elastic Bunch Graph

Eine mögliche Technik zur Gesichtserkennung ist das Elastic Bunch Graph Matching-Verfahren. Hierbei wird ein Gitternetz in Form eines markierten Graphen über das Gesicht gelegt und markante Stellen werden mit Knoten des Graphen, den Landmarken, markiert. Als Referenzwerte werden die Länge und der Winkel jeder Kante des Graphen gespeichert. Eine Authentifikation läuft dann im Groben wie folgt ab: zunächst erfolgt die Lokalisierung des Gesichts und der Landmarken, anschließend wird der Graph des aktuell erfassten Gesichts mit dem als Referenzwert abgelegten verglichen, d.h. es erfolgt ein Vergleich der Längen und der Winkel der Kanten.

Stimme

Stimmerkennungssysteme, die im biometrischen Umfeld eingesetzt werden, dürfen nicht mit Spracherkennungsverfahren verwechselt werden. Die Stimmerkennung beruht auf dem Geräusch, das durch Resonanzen abhängig von der Mundform oder auch der Nasenhöhle, hervorgerufen wird. Meist werden textabhängige Erkennungen durchgeführt, bei denen der Sprecher ein festgelegtes Passwort oder eine bestimmte Phrase aussprechen muss.

Hybridsystem

Durch die Kombination verschiedener biometrischer Merkmale zur Authentifikation lässt sich die Qualität des Authentifikationsdienstes steigern. Eine Möglichkeit besteht beispielsweise in einer Kombination aus Mimik (Lippenbewegung), Stimme und Gesichtserkennung. Die zu authentifizierende Person muss sich in einem ersten Schritt mittels ihrer Stimme authentifizieren, indem sie ihren Namen nennt. Gleichzeitig wird sowohl die Lippenbewegung aufgezeichnet als auch ein Standbild des Gesichts angefer-

tigt. Da in einem derartigen Hybridsystem jede Person über drei Merkmale authentifiziert wird, können temporäre Störeinflüsse wie krankheitsbedingte Stimmveränderungen oder ein leicht verändertes Erscheinungsbild toleriert werden, solange zwei der drei Merkmale noch innerhalb der Toleranzgrenzen liegen.

Biometrische Geräte, die auf neueren Techniken wie der DNA-Analyse, der Anordnung der Venen auf der Handrückseite oder der Dynamik des Tastenanschlages beim Eingeben von Texten basieren, sind zum Teil erst als Prototypen in Forschungslabors verfügbar und werden noch weiterentwickelt und erprobt. Mit den kommerziellen Produkten steht aber bereits heute ein großes Spektrum an biometrischen Techniken zur Verfügung. Da biometrische Verfahren wichtige Bestandteile zukünftiger Sicherheitsinfrastrukturen sein werden, beschäftigen wir uns im Folgenden mit deren allgemeinen Charakteristika und weisen schließlich auf Risiken bei deren Einsatz hin.

neue Techniken

10.3.3 Biometrische Authentifikation

Biometrische Techniken arbeiten alle nach dem gleichen Schema. Zunächst muss das Analysesystem Referenzwerte der zu analysierenden biometrischen Eigenschaften mittels Sensoren erfassen und digitalisieren. Diese Referenzwerterfassung wird auch als „Lernen“ bezeichnet. Aus den Referenzwerten werden dann charakteristische Eigenschaften extrahiert und als personenbezogene, komprimierte Referenzdatensätze auf dem biometrischen Gerät selber, auf einer personenbezogenen Smartcard oder in anderen Speichermedien abgelegt. Zur Authentifikation einer Person benötigt man spezielle Geräte (u.a. Fingerabdrucksensoren oder Videokameras), die das betreffende biometrische Merkmal bei jeder durchzuführenden Authentifikation erfassen. Die aktuell erhobenen biometrischen Merkmale sind ihrerseits in Merkmalsätze digital zu transformieren und im abschließenden Schritt mit den gespeicherten Referenzdaten zu vergleichen.

Authentifikations-schema

Abgleich

Die Hauptschwierigkeit biometrischer Verfahren besteht darin, korrekt zu entscheiden, ob aktuelle Werte mit den Referenzwerten übereinstimmen. Auch bei den bisher behandelten Verfahren zur Authentifikation musste ein solcher Abgleich durchgeführt werden. Es handelte sich dabei um die Prüfung auf Gleichheit von PINs und gehaschten Passworten oder um die Überprüfung der Korrektheit von Antworten bei Challenge-Response-Protokollen. Hier waren einfache Tests auf Gleichheit ausreichend, um mit einer 100-prozentigen Genauigkeit eine Übereinstimmung der zu vergleichenden Werte oder eine Abweichung festzustellen. Bei biometrischen

Problem

Merkmalen sind wir demgegenüber mit Eigenschaften einer Person konfrontiert, die sich über die Zeit ein wenig ändern können. Man denke dabei an Stimmungsschwankungen, die sich auf die Stimme oder die Gesichtsmotorik niederschlagen, oder an Stresssituationen, in denen sich der Tipp-Rhythmus oder die Unterschrift ändert, oder an leichte Verletzungen an der Hand, so dass die Fingerabdrücke nicht mehr übereinstimmen oder gar nicht mehr erfassbar sind. Abweichungen können sich auch durch veränderte Umgebungseinflüsse ergeben, die zum Zeitpunkt der Erstellung der aktuellen Authentifikationsprofile herrschen. Dazu gehören unterschiedliche Lichtbedingungen, Verschmutzungen oder verschiedene Aufnahmewinkel (z.B. im Profil, frontal). Da die aktuellen Werte häufig von den Referenzwerten abweichen, ist es notwendig, Toleranzschwellen festzulegen. Mittels Korrelationstests sind die Abweichungen von den Referenzwerten zu bestimmen und es ist dann mit geeigneten Auswertungsalgorithmen zu prüfen, ob sich diese noch unterhalb der Schwellen bewegen. Die Analyse- und Auswertungsalgorithmen sind die Herzstücke biometrischer Geräte.

Fehlerraten

Fehlerraten

Man unterscheidet zwei Typen von Fehlern: (1) ein berechtigter Benutzer wird abgewiesen und (2) ein unberechtigter Benutzer wird authentifiziert und akzeptiert. Tritt ein Fehler der ersten Klasse auf, so sind die Kontrollen unter Umständen zu streng und ein häufiges Auftreten derartiger Fehler kann zu Akzeptanzproblemen seitens der Benutzer führen. Dies ist jedoch tragbar, falls gleichzeitig auch Fehler der zweiten Art mit hoher Wahrscheinlichkeit vermieden werden und durch das Zugangsverfahren ein höchst sensibler Bereich zu schützen ist. Sind die Toleranzschwellen zu locker, so erhalten im Zuge der Fehler der zweiten Klasse unberechtigte Benutzer Zugang zum System, was für sicherheitskritische Anwendungsbereiche sicherlich nicht tolerabel ist.

Gleichfehlerrate

Als Leistungsmaß zur Bewertung der Güte eines Erkennungssystems dienen Angaben über Fehlerraten zur Abweisung autorisierter Benutzer (engl. *false rejection rate* FRR) bzw. zur Akzeptanz unautorisierter Benutzer (engl. *false acceptance rate* FAR) sowie über die Gleichfehlerrate (engl. *equal error rate* EER). Die Gleichfehlerrate ist der Schnittpunkt der Kurven der beiden Fehlerraten, d.h. die Zahl, bei der die Anzahl der fälschlich abgewiesenen Benutzer und die Anzahl der fälschlich akzeptierten Benutzer gleich groß ist. Wünschenswert ist also eine niedrige Gleichfehlerrate. Abbildung 10.3 veranschaulicht den Zusammenhang zwischen den beiden Raten. Je mehr Merkmale bzw. Parameter man dem Erkennungsalgorithmus hinzufügt (x-Achse), desto aufwändiger werden die Verfahren und die Rate der fälschlich als autorisierte akzeptierte Benutzer sinkt stark ab. Die Qualität, d.h. die Sicherheit nimmt zu. Gleichzeitig besteht aber die Tendenz, dass aufgrund

der niedrigen Toleranzschwellen die Rate der fälschlich abgewiesenen steigt, wodurch der Nutzungskomfort stark beeinträchtigt werden könnte.

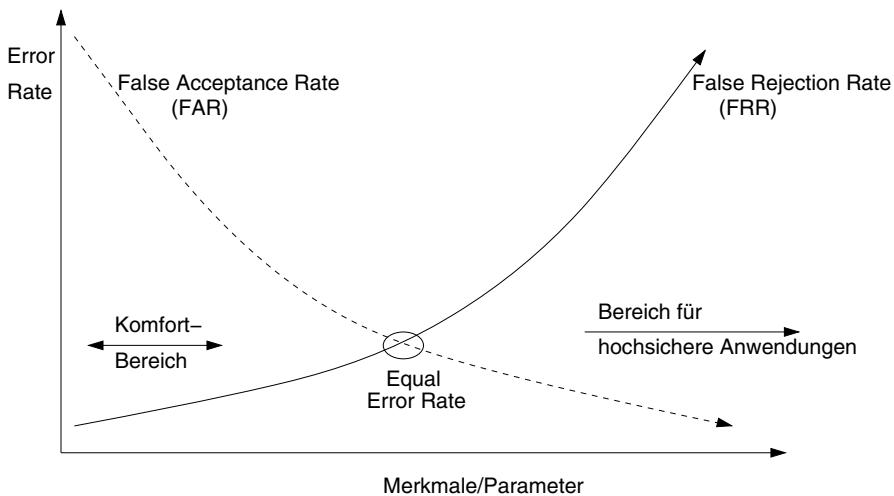


Abbildung 10.3: Zusammenhang zwischen FAR und FRR

Zu beachten ist, dass in der Regel die Angaben über Fehlerraten von den jeweiligen Herstellern biometrischer Geräte stammen. Solche Angaben sind aber mit großer Vorsicht zu bewerten, da meist keine Unterlagen über das Zustandekommen der Werte verfügbar sind. Werden die Daten beispielsweise in einheitlichen, gleichbleibenden Testumgebungen berechnet, so sind variierende Umgebungseinflüsse, wie unterschiedliche Lichtverhältnisse, Verschmutzungen oder krankheitsbedingte Veränderungen der Merkmale, meist nicht erfasst. Derartige Einflüsse gehören jedoch zum Alltag biometrischer Erkennungsgeräte und können deren Leistungsfähigkeit erheblich beeinflussen.

Vorsicht!

Internationale Standards bzw. standardisierte Testverfahren zur Überprüfung der Leistungsfähigkeit der Geräte werden zurzeit entwickelt und sind bereits teilweise verabschiedet, wie beispielsweise der ISO/IEC 19795²². Der Standard ISO/IEC 29794 gibt Rahmenbedingungen für die Beurteilung der Qualität biometrischer Verfahren vor. Er legt dafür eine Skala mit Werten von 0 bis 100 fest, wobei die Werte 0-25 eine inakzeptable Qualität, die Werte 26-50 eine mangelhafte, die Werte 51-75 eine ausreichende und die Werte 76-100 eine exzellente Qualität charakterisieren. Jedoch enthält der Standard keine exakten Definitionen, so dass die Bedeutung dieser Qualifikation unscharf bleibt. Weitere Standards sind zurzeit noch in der Entwicklung, wie

²² Biometric Performance Testing and Reporting Standard

beispielsweise die ISO/IEC 29794-Familie, wobei bislang jedoch meist nur die Aufnahmegerätqualität beurteilt wird.

10.3.4 Fallbeispiel: Fingerabdruckerkennung

Im Folgenden wird exemplarisch am Beispiel der Fingerabdruckerkennung das Vorgehen bei der biometrischen Authentifikation verdeutlicht.

Wie bereits einführend erwähnt, sind besonders Fingerabdruckverfahren geeignet, um auch auf dem Massenmarkt eingesetzt zu werden. Fingerabdrücke entstehen durch die Papillarleisten in Form von spezifisch hervortretenden Erhebungen der Haut. Papillarleisten zeigen Charakteristika, die in ihrer Form und Lage zueinander einmalig sind. Die Papillarleistenstruktur bildet sich in der Regel bis zum vierten Embryonalmonat aus und verändert sich danach ohne Fremdeinwirkung nicht mehr.

Minutien

Die kommerziell verfügbaren Systeme unterscheiden sich darin, welche der charakteristischen Eigenschaften von Fingerkuppen, die so genannten Minutien²³, sie zur Analysegrundlage wählen. Bei Minutien handelt es sich um Unterbrechungen von Fingerlinien, wie beispielsweise Eckpunkte, Gabelungen, Wirbel oder Inseln. Merkmale zur Authentifikation sind Art und Ausrichtung der Minutien.

Eine mögliche Technik bei der Fingerabdrucksensoren besteht darin, charakteristische Rillen und Vertiefungen in Fingerkuppen zu erfassen, um daraus die Minutien zu extrahieren, die eine Person eindeutig identifizieren. Andere Techniken analysieren die Abstände zwischen den Rillen oder die Porenstruktur des Fingers.

Verarbeitungsschritte

Bei der Verarbeitung von Fingerabdrücken sind mehrere Arbeitsschritte zu durchlaufen.

Scannen

In einem ersten Schritt wird ein Bild der Fingerkuppe aufgenommen, wozu der Finger meist auf einen Scanner aufgelegt werden muss. Für den Scan-Vorgang können verschiedene Sensoren verwendet werden. So arbeiten optische Verfahren mit Lichtquellen und sensorbasierte Techniken erfordern das Auflegen des Fingers auf den Sensor, der beispielsweise die Temperatur misst, die charakteristischen Rillen erfasst und/oder eine dreidimensionale Aufnahme des Abdrucks erstellt, um z.B. das Auflegen von Fotos anstelle eines realen Fingers direkt als Maskierungsangriff zu erkennen.

Neben den bereits angesprochenen optischen Verfahren und thermischen Sensoren unterscheidet man grob folgende weitere Klassen von Sensoren.

²³ Lateinisch für Kleinigkeit oder kleines Detail.

- **Kapazitive Sensoren:** diese nutzen zur Erstellung des Fingerabdrucks das Ladungsfeld des Menschen, das mit einer sensitiven, metallischen Beschichtung erfasst wird.
- **Drucksensoren** messen pixelweise den Druck des aufliegenden Fingers und
- **Ultraschallsensoren** messen die durch die Kontaktstreuung verursachten Signale.

Das Ergebnis des Scan-Vorgangs ist ein Bild im Graustufen-Format.

Der nächste Schritt besteht darin, störenden Hintergrund soweit herauszufiltern, dass nur noch die Rillen sichtbar sind. Auf diese Weise versucht man, Verzerrungen durch beispielsweise Schmutz und Verletzungen aus dem Fingerabdruck zu entfernen. Anhand der charakteristischen Rillenendpunkte, Verzweigungen etc. erfolgt dann die so genannte Feature-Extraktion, das heißt, die Berechnung der Minutien. Je nach Erkennungssystem werden in diesem Schritt 10–100 Minutien extrahiert.

Filterung

Abschließend wird ein Abgleich (engl. *match*) des gefundenen Minutien-Musters mit den gespeicherten Referenzwerten durchgeführt. Dazu werden schrittweise benachbarte Minutien des Verifikationsdatensatzes mit entsprechenden Nachbarschaftsbeziehungen in den Referenzdaten verglichen. Falls der Grad der Abweichung den festgelegten Toleranzwert nicht übersteigt, wird ein Match festgestellt. Eine derartige Prüfung erfolgt für alle möglichen Nachbarschaftsbeziehungen so lange, bis die vorab festgelegte Anzahl von Minutien-Matches (z.B. 5) erzielt wurde, bzw. bis alle Kombinationen getestet wurden und ggf. nicht genügend Übereinstimmungen ermittelt werden konnten.

Match

Häufig²⁴ betrachtete Merkmale für Fingerabdruck-Minutien sind die Ortskoordinaten x und y sowie der Linienwinkel (Tangentenwinkel) τ . Eine Minutie wird hierbei als ein Tripel (x, y, τ) dargestellt. Ein Fingerabdruck wird als Folge von Tripeln der Art $((x_1, y_1, \tau_1), (x_2, y_2, \tau_2), \dots, (x_n, y_n, \tau_n))$ charakterisiert.

Für den Abgleich (Match) von Fingerabdrücken wird der Abstand und die Winkeldifferenz zweier Minutien $(x_i, y_i, \tau_i), (x_j, y_j, \tau_j)$ wie folgt berechnet. Der Abstand d ergibt sich durch:

$$d = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$$

²⁴ Matching-Verfahren

Die Winkeldifferenz ergibt sich durch:

$$\Delta\tau = \begin{cases} |\tau_i - \tau_j|, & \text{falls } |\tau_i - \tau_j| \leq 180^0 \\ 360^0 - |\tau_i - \tau_j|, & \text{falls } |\tau_i - \tau_j| > 180^0 \end{cases}$$

Als Vergleichskriterium für zwei Fingerabdrücke werden Toleranzschwellen sowohl für den Abstand $dTol$ als auch die Winkeldifferenz τTol festgelegt und geprüft. Weiterhin wird ein so genannter Match-Score k bestimmt, dessen Größe abhängig von dem angestrebten Sicherheitsniveau ist. Zwei Minuten gelten als übereinstimmend, falls $d \leq dTol$ und $\Delta\tau \leq \tauTol$. Zwei Fingerabdrücke gelten als übereinstimmend, wenn mindestens k übereinstimmende Minuten im Rahmen der Toleranzen $dTol$ und τTol gefunden wurden.

Prüfungs-Varianten

Man unterscheidet bei der Prüfung zwischen der Verifikation (One To One Matching) und der Identifikation (One To Many Matching). Bei der Verifikation wird der Verifikationsdatensatz mit einem gespeicherten Referenzmuster (Template) verglichen. Die Prüfung findet direkt auf dem Speichermedium statt, auf dem der Referenzdatensatz abgelegt ist. Bei der Identifikation wird in einer Datenbank nach einem passenden Referenzmuster für den Abdruck gesucht, so dass hierzu ein höherer Aufwand als bei der Verifikation erforderlich ist, da alle gespeicherten Referenzmuster für den Vergleich überprüft werden müssen.

Die im Einsatz befindlichen Systeme zur Auswertung des Fingerabdrucks sind klein, kostengünstig und einfach zu bedienen. Fingerprint-Sensoren benötigen nur einen geringen Flächenbedarf (z.B. $12,8 \times 12,8 \text{ mm}^2$), so dass sie in sehr unterschiedlichen Anwendungen eingesetzt werden können. Beispiele sind die Zugangskontrolle zum PC/Laptop mittels Fingerprint-Sensor auf der Tastatur, die Zugangskontrolle zu Räumen oder aber auch die Identifizierung Einreiseberechtigter an Flughäfen.

10.3.5 Sicherheit biometrischer Techniken

Die rasante Entwicklung auf dem Gebiet der biometrischen Geräte und deren Vordringen sowohl auf dem Massenmarkt als auch in viele gesellschafts- und sozialpolitisch relevante Bereiche ist bei allen Vorteilen, die man unmittelbar sehen kann, aber nicht unproblematisch (vgl. [160]).

Vorteile

enge Kopplung

Biometrische Merkmale bieten den Vorteil, dass sie nicht absichtlich oder unabsichtlich weitergegeben werden oder gar verloren gehen können und dass das Fälschen in der Regel sehr schwierig ist. Diese Eigenschaften machen biometrische Merkmale zu geeigneten Kandidaten für die Ablösung

klassischer, passwortbasierter Zugangskontrollen heutiger IT-Systeme. Die enge Kopplung zwischen den Authentifikationsdaten und der zu authentifizierenden Person hat zur Folge, dass der Besitz bzw. die Kenntnis der Referenzwerte, anders als bei den Passwortverfahren, nicht ausreicht, um sich korrekt zu authentifizieren. Doch stimmt das wirklich? Die Aussage ist nämlich nur für solche Szenarien zutreffend, in denen die erhobenen Daten direkt auch vom Analysegerät überprüft oder über eine sichere Verbindung vom Lesegerät zum Verifizierer übertragen werden. Ganz anders stellt sich jedoch die Situation dar, wenn die aktuellen Authentifikationsdaten zur Verifikation über ein unsicheres Medium an ein entferntes Ziel gesendet werden müssen. In diesem Fall stehen einem Angreifer alle Möglichkeiten für aktive Angriffe offen. Das heißt, dass ein Angreifer die Authentifikationsdaten einer Person Alice abfangen und unter der Identität von Alice wiedereinspielen kann. Eine erfolgreiche Maskierung ist somit in einem solchen Szenario sehr wohl möglich, auch wenn der Angreifer nicht die realen biometrischen Merkmale von Alice besitzt oder sie gefälscht hat.

Angriff

Aus der engen Kopplung zwischen biometrischen Merkmalen und der zu authentifizierenden Person ergeben sich aber auch eine ganze Reihe von Problemen, die man grob in drei Klassen aufteilen kann. Es handelt sich (1) um Gefahren für das informationelle Selbstbestimmungsrecht, (2) um die Gefahr gewaltssamer Angriffe gegen Personen und (3) um Gefahren, die aus der Unveränderbarkeit der biometrischen Eigenschaften resultieren.

Probleme

Informationelle Selbstbestimmung

Biometrische Merkmale sind eindeutige Identifikatoren für Personen, die an einer Vielzahl von Orten und zu vielen Gelegenheiten auch unbemerkt erfasst und analysiert werden können. Man denke hier beispielsweise an Techniken zur Gesichtserkennung mittels in Räumen installierter Kameras. Biometrische Merkmale lassen sich nicht einfach ablegen, um sich dadurch bewusst einer Überwachung zu entziehen. Durch flächendeckende, ggf. auch unbemerkte, Erhebungen von Aufenthaltsdaten einzelner Personen oder auch von Personengruppen besteht somit die Gefahr, dass Bewegungsprofile erstellt oder illegale Rasterfahndungen anhand der erhobenen Daten durchgeführt werden. Der Arbeitsplatz ist ein weiterer sensibler Bereich, an dem das informationelle Selbstbestimmungsrecht des Einzelnen gefährdet sein kann. Biometrische Geräte wie zum Beispiel eine Gesichtskontrolle zur Zugangskontrolle an Arbeitsplätzen könnten auf einfache Weise zu einer unbemerkten, allgemeinen Mitarbeiterüberwachung missbraucht werden.

Datenerhebung

Erschwerend kommt hinzu, dass mit den biometrischen Daten häufig sehr viele Informationen über eine Person erfasst werden. Anhand der Iris und der Netzhaut lassen sich zum Beispiel Rückschlüsse auf gewisse Krankheitsbilder wie Diabetes und Bluthochdruck ziehen. Jeder Blick in ein

zusätzliche Informationen

entsprechendes Analysegerät liefert somit eine Auskunft über den aktuellen Gesundheitszustand oder bei anderen Geräten über die aktuelle Gemütslage wie z.B. Stress. Es erscheint nicht unrealistisch, dass eine derartige Information für manche Arbeitgeber von Interesse sein könnte. Man benötigt somit Verfahren zur Anonymisierung, so dass hinterlegte Referenzwerte nicht mit einer natürlichen Person identifiziert werden können. Denkbar sind hier zum Beispiel Verfahren mit blinden Signaturen [41] oder Pseudomisierungsverfahren, die analog zu den asymmetrischen Verschlüsselungssystemen nicht die sensitiven, persönlichen Merkmale einer Person, sondern nur ein entsprechendes, unkritisches Pendant veröffentlichen. In diesem Bereich sind jedoch noch weitere Forschungsanstrengungen erforderlich.

Gewaltkriminalität

menschliche
Schlüssel

Skeptiker biometrischer Verfahren befürchten ein Anwachsen krimineller Delikte, die sich dann nicht mehr „nur“ darauf beschränken, Informationen als immaterielles Gut zu stehlen oder zu modifizieren, sondern sich verstärkt direkt gegen Personen richten könnten. In dem Maß, in dem der ganze Mensch oder Teile von ihm als Schlüssel zum Zugriff auf sensible Bereiche dient, steigt die Gefahr gewaltsamer Aktionen, um sich in den „Besitz“ dieser menschlichen Schlüssel zu bringen. Neben Erpressungsversuchen, durch die ein Opfer unter Androhung von Gewaltanwendung dazu gezwungen wird, seinen Körper zur Authentifikation einzusetzen, kursieren vermehrt die zwar etwas makabren, aber durchaus nicht unrealistischen Vorstellungen von abgetrennten Körperteilen wie Fingern, Fingerkuppen oder gar Augen, mit deren Hilfe eine unautorisierte Authentifikation erfolgreich durchführbar ist.

spezielle
Kontrollen

Durch spezielle Vorkehrungen, den so genannten Lebend-Checks wie thermografischen Überprüfungen, können sich aber bereits heute viele Systeme gegen das Vorlegen verstümmelter oder abgestorbener Körperteile wappnen. Zur Abwehr von erpresserisch herbeigeführten unautorisierten Zugriffen könnten Systeme eine Notfallversorgung vorsehen, so dass beispielsweise eine Art „Notfinger“ festgelegt wird, den das Opfer in einem solchen Fall zur Signalisierung seiner Situation anstelle des normalen Fingerabdrucks vorlegt. Bei einer Stimmerkennung könnte vorab ein spezielles Notfallwort vereinbart werden.

Unveränderliche Merkmale

Kompromittierung

Ein weiteres Problem biometrischer Merkmale liegt darin begründet, dass sie unveränderbar sind. Eine Authentifikation erfolgt auf der Basis von statisch festgelegten Authentifizierungsdaten. Ist der Datensatz über die Referenzwerte einer Person in irgendeiner Weise kompromittiert, so kann die betroffene Person nicht einfach ihre Merkmale erneuern, wie sie dies bei einem aufgedeckten Passwort oder einem offengelegten Signatur- oder kryptografischen Schlüssel leicht machen könnte. Diese Inflexibilität hat

zur Folge, dass das Zugangskontrollsysteem in einem derartigen Fall unter Umständen für die betroffene Person überhaupt nicht mehr benutzbar ist. Konsequenzen, die sich daraus für den Einzelnen ergeben könnten, zum Beispiel am Arbeitsplatz oder auch allein schon beim Geldabheben am Automat, kann man sich leicht ausmalen.

Bedrohungen für die biometriebasierte Authentifikation können sich nicht nur durch das Abfangen der aktuellen Authentifikationsdaten, sondern insbesondere aus der Verwaltung und Bearbeitung sowohl der Referenzdaten, als auch der jeweils aktuell erhobenen Daten ergeben. Der Benutzer muss sich darauf verlassen können, dass die eingesetzten Analyse- und Verifikationssysteme vertrauenswürdig und korrekt arbeiten. Es sind ferner sehr hohe Anforderungen an die Integrität und Authentizität der ggf. über lange Zeiträume hinweg in Datenbanken verwalteten Referenzdaten zu stellen. Gelingt es nämlich einem Angreifer, diese durch seine eigenen Daten zu ersetzen, so sind nicht allein erfolgreiche Maskierungen möglich, sondern es resultieren daraus natürlich auch Denial-of-Service-Angriffe, da die betroffene Person, obwohl sie dazu autorisiert ist, sich nicht oder im schlimmsten Fall sogar nie mehr korrekt authentifizieren kann.

sicherheitskritische
Referenzdaten

Eine Schwachstelle, die in der Vergangenheit auch bereits für Angriffe ausgenutzt wurde, ist der Datenübertragungsweg zwischen Biometricsensor und dem Gerät, das die Vergleiche durchführt. Die mittels des Biometricsensors aufgenommenen Verifikationsdaten werden häufig über ein USB-Kabel (Universal Serial Bus) zum PC transferiert. Aufgrund der in der Regel fehlenden Authentifikation der beteiligten Geräte (man könnte eine Authentifikation zum Beispiel durch ein Challenge-Response-Protokoll erzielen) ist es einem Angreifer möglich, sich als Biometricscanner zu maskieren und Daten, die vorab über die USB-Schnittstelle übertragen wurden, wieder einzuspielen. Zum Abhören der Daten, die vom USB-Adapter zum Gerätetreiber des PCs übertragen werden, existieren bereits frei verfügbare Sniffer-Programme. So schaltet sich beispielsweise das Programm **USB-Snoop²⁵** für Windows-Systeme zwischen den USB-Adapter und den Gerätetreiber und schreibt die abgefangenen Daten in eine Logdatei.

USB-
Schwachstelle

Fazit

Die datengesetzkonforme Verwaltung der Referenzdaten durch zuverlässige und vertrauenswürdige Systeme ist eine wesentliche Voraussetzung dafür, dass biometrische Techniken von Benutzern akzeptiert und eingesetzt werden. Biometrische Verfahren können dann erheblich zu einer Qualitätssteigerung auch in offenen IT-Systemen beitragen, wenn die Integrität und Authentizität der Referenzdaten mit hoher Zuverlässigkeit garantiert wird

Akzeptanz

²⁵ <http://www.sourceforge.net/projects/usbsnoop>

und Biometrie nicht als „Allzweckwaffe“ und Kryptografieersatz zweckentfremdet wird. Genauso, wie man auch nicht das gleiche Passwort für zwei unterschiedliche Rechner oder den gleichen Datenschlüssel für unterschiedliche Anwendungen verwenden sollte, so sollte auch ein und dasselbe biometrische Merkmal nicht zur Zugangskontrolle für ganz unterschiedliche Anwendungsbereiche wie private Daten, E-Mails und vertrauliche Patientenakten oder gleichzeitig zum Deaktivieren der Autowegfahrsperrre und zum Aktivieren des Handys sowie der ec-Karte genutzt werden.

sinnvoller Einsatz

Wie auch schon an anderen Stellen ist auch hier natürlich wieder darauf hinzuweisen, dass die Eignung von Verfahren und Mechanismen unmittelbar davon abhängt, welche Sicherheitsanforderungen gestellt werden. Nicht für jeden Anwendungsbereich ist eine hohe Mechanismenstärke der biometrischen Verfahren notwendig, so dass diese trotz der zurzeit noch offensichtlichen Defizite durchaus für einige Bereiche sinnvoll einsetzbar sind. Dies gilt unter anderem überall dort, wo eine relativ einfache Plausibilitätskontrolle ausreicht und eine Vielzahl von Überprüfungen effektiv und mit möglichst wenigen Ressourcen durchgeführt werden sollen. Falls ohne biometrische Kontrolle aufgrund von Ressourcenengpässen gar keine Kontrollen stattfinden, könnten in solchen Fällen auch schwache flächendeckende Überprüfungen Fortschritte bedeuten.

10.4 Authentifikation in verteilten Systemen

Die im vorliegenden Kapitel eingeführten Authentifizierungstechniken werden auch in unterschiedlichen Ausprägungen und Kombinationen eingesetzt, um Subjekte in offenen Systemumgebungen zu authentifizieren. Die jeweiligen Maßnahmen sind Bestandteil von Authentifikationsprotokollen, die zwischen räumlich verteilten Endsystemen abgewickelt werden.

Heutige vernetzte IT-Systeme sind noch vorwiegend durch Client-Server Architekturen geprägt. Die Server bieten Dienste (Services) an, die von den Clients in Anspruch genommen werden können. Beispiele für solche Dienste sind Dateisystem- oder Datenbankdienste. Im Folgenden wird zunächst mit dem RADIUS-Protokoll ein weit verbreitetes, nachrichtenorientiertes Authentifizierungsprotokoll erklärt. Danach wird das Kerberos-Protokoll als Beispiel eines Single-Sign-On Protokolls für verteilte Systeme erläutert.

10.4.1 RADIUS

entfernte Einwahl

RADIUS (Remote Authentication Dial In User Service) ist ein UDP-basiertes Client-Server-Protokoll²⁶, das ursprünglich eingeführt wurde, um

²⁶ Der offizielle Port von RADIUS ist 1645.

Benutzer, die über einen Service Provider einen Einwahldienst (dial in) zum Internet nutzen (vgl. Abbildung 10.4) und über deren Einwahlknoten einen Systemzugang erlangen [150], zu authentifizieren. Da ein Benutzer sich an unterschiedlichen Einwahlknoten anmelden kann, fungiert der Einwahlknoten, der Network Access Server (NAS), als RADIUS-Client, der die Authentifizierungsdaten (in der Regel Name und Passwort) von Benutzern entgegen nimmt und mit dem RADIUS-Protokoll an den RADIUS-Server zur Überprüfung weiterleitet. Mittlerweile wird das RADIUS-Protokoll auch in anderen Umgebungen eingesetzt, wie beispielsweise zur Absicherung eines WLAN-Zugangs oder zur Authentifizierung von Roaming-Partnern. Abbildung 10.4 zeigt eine typische Architektur eines Zugangssystems mittels RADIUS. Bei einem Verbindungsaufbau werden in einem solchen System grob folgende Schritte durchlaufen:

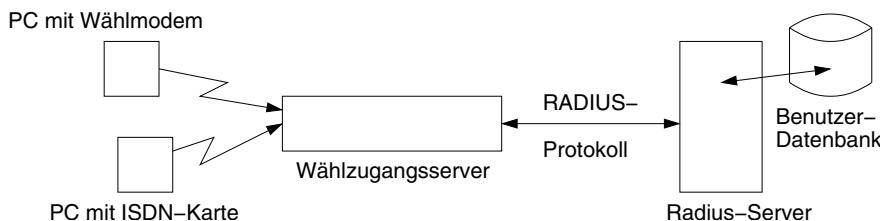


Abbildung 10.4: Zugangssystem mittels RADIUS

- Der Benutzer baut mit Hilfe eines Modems oder einer ISDN-Karte eine Verbindung zum Wählzugangsserver auf.
- Nach dem erfolgreichen Verbindungsaufbau übergibt er seinen Benutzernamen und das dazugehörige Passwort.
- Der Wählzugangsserver, der als RADIUS-Client fungiert, sendet diese Daten mit Hilfe des RADIUS-Protokolls (s.u.) an den RADIUS-Server. Dieser überprüft die Gültigkeit des Benutzernamens und des Passwortes anhand seiner Benutzerdatenbank, und sendet das Ergebnis zurück an den Wählzugangsserver.

Die Authentifikation basiert auf einem vorab vereinbarten Geheimnis (pre-shared secret) zwischen den RADIUS-Clients und -Server. Ein RADIUS-Server kann verschiedene Konzepte und Protokolle zur Authentifizierung von Benutzern einsetzen, wie beispielsweise PAP, CHAP oder aber auch Unix-Login-Varianten.

Um auch eine dezentrale Verwaltung von Benutzerkennungen und Passworten für verschiedene administrative Bereiche (RADIUS Zonen) durch dafür zuständige RADIUS-Server zu ermöglichen, ist es üblich, ein Proxy-

Konzept bestehend aus einem oder mehr Proxy-Servern zu verwenden (vgl. Abbildung 10.5). Der Proxy-Server leitet Anfragen an bzw. Antworten vom RADIUS-Server weiter und dient damit als Gateway.

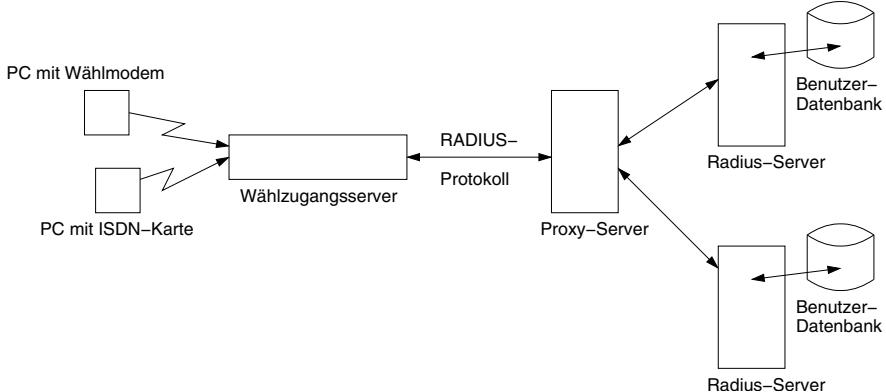


Abbildung 10.5: Dezentrale Zugangskontrolle über Proxy-Server

Das RADIUS-Protokoll

Ausgangspunkt für das Protokoll ist ein Pre-Shared Secret S , das zwischen RADIUS-Client, also dem Einwahlknoten oder einem Proxy, und dem RADIUS-Server vorab vereinbart wurde. Das Protokoll trifft keine Aussage darüber, auf welche Weise dieser gemeinsame Schlüssel auszutauschen ist.

Access-Request

Bei einer Verbindungsaunahme sendet der RADIUS-Client eine *Access-Request* Nachricht zum Server. Die Nachricht enthält die IP-Adresse des RADIUS-Clients, die Benutzererkennung sowie entweder das gehashte Benutzer-Passwort (s.u.) oder ein CHAP-Passwort. Über einen 8-Bit Identifier (ID) wird die Anfrage identifiziert und der Anfrager fügt ihr noch eine 128 Bit Zufallszahl, den so genannten Request Authenticator (RA) hinzu. Optional können noch weitere Attribute ergänzt werden, die aber vom Server nicht unbedingt bearbeitet werden müssen. Der Client wiederholt seine Anfrage, falls nach Ablauf eines Timer-Intervalls keine Antwort auf die Anfrage vorliegt.

gehashtes Passwort

Das Benutzer-Passwort wird in der Anfrage nicht im Klartext übertragen, sondern wie folgt mittels der Hashfunktion H bearbeitet. Gegeben sei das Pre-Shared-Secret²⁷ S und die 128-Bit Zufallszahl RA. Es werden schrittweise folgende Werte berechnet:

- $c_1 = p_1 \text{ xor } H(S \mid RA)$

²⁷ Es sollte laut dem RFC 128 Bit lang sein.

- $c_2 = p_2 \text{ xor } H(S \mid c_1)$
- ...
- $c_n = p_n \text{ xor } H(S \mid c_{n-1})$

wobei p_i die i-ten 128-Bit des Original-Passwortes sind. Der letzte 128-Bit Block p_n des Passwortes wird, falls notwendig, mit Paddingzeichen aufgefüllt.

Die Request-Nachricht besitzt grob folgendes Format:

Request: IP-Adresse_Client, ID, RA, Benutzer-Name, c_1, \dots, c_n, \dots

Der Server extrahiert aus der empfangenen Passwort-Information c_1, \dots, c_n unter Einbeziehung des Pre-Shared Secrets S das ursprüngliche Passwort²⁸. Das extrahierte Passwort wird geprüft (bzw. zur Prüfung weitergeleitet, falls der Server als Proxy fungiert) und der Server antwortet mit der *Access-Accept*-Nachricht, die den Identifikator ID der Anfrage sowie einen 128-Bit Response Authenticator enthält. Der Response Authenticator ist ein MAC-Wert über die Daten aus der Anfrage-Nachricht:

$\text{ResponseAuth} = \text{Hash}(\dots \mid ID \mid length \mid RA \mid S).$

Mit dem Nachweis der Kenntnis von S bei der Berechnung des Response Authenticators beweist der RADIUS-Server seine Authentizität. Die *Accept*-Nachricht besitzt grob folgendes Format, wobei die Attribute, die der Server noch zusätzlich zurücksenden kann, nicht einzeln aufgeführt sind:

Access Accept: ID, ResponseAuth, ...

Sollte eines der in der Anfrage-Nachricht übermittelten Attribute (natürlich insbesondere das Passwort-Attribut) ungültig sein, so sendet der Server eine *Access-Reject*-Nachricht zurück, die die Request-ID sowie den Response Authenticator (s.o.) enthält.

Reject

Beim Empfang der Server-Nachricht prüft der Client, ob es sich um eine Antwort (passende ID) auf seine Anfrage handelt. Andernfalls wird die Nachricht nicht weiter bearbeitet. Im nächsten Schritt wird die Korrektheit des Response Authenticators überprüft, indem der Client die gleiche MAC-Berechnung wie der Server durchführt und die Ergebnisse vergleicht.

Alternativ zur *Accept*- bzw. *Reject*-Nachricht kann der RADIUS-Server auch dem Client eine Challenge zurücksenden, die ebenfalls die Request-ID und den Response-Authenticator enthält. Eine Challenge kann einen Text-String enthalten, der dann vom RADIUS-Client dem Benutzer anzuzeigen ist. Die Challenge-Nachricht fordert den Client auf, eine erneute Access-Anfrage zu stellen. Falls der Benutzer ein neues Passwort eingeben muss, wird dieses in der erneuten Anfrage übergeben (gehashed wie oben beschrieben).

Challenge

²⁸ $p_i = c_i \text{ xor } \text{Hash}(S \mid c_{i-1}S)$

Sicherheit des Protokolls

Da das RADIUS-Protokoll Passwort-basiert ist, stellen die verwendeten Benutzerpassworte natürlich eine mögliche Schwachstelle dar. Eine weitere Verwundbarkeit bilden die Daten aus der Request-Nachricht und dem Authenticator aus der Server-Antwort. Ein Angreifer kann zusammengehörende Paare abfangen und diese Paare offline für eine Known-Plaintext Analyse des verwendeten geheimen Schlüssels S verwenden. Mit der Offenlegung des Schlüssels S sind alle Passworte, die zwischen dem RADIUS-Client und dem Server übertragen werden, vom Angreifer direkt „entschlüsselbar“.

Wesentlich für das Protokoll ist der Einsatz eines guten Zufallszahlengenerators, um bei der Anfrage-Nachricht den Wert RA zu generieren. Falls der Generator zu kurze Zykelzeiten aufweist und Werte wiederholt, so kann ein Angreifer die ersten 128-Bit des Passwortes aus dem gehaschten Strom herausfiltern. Ferner kann ein Angreifer auch RA-Werte und die zugehörigen Antworten des Servers aufzeichnen und bei einem erneuten Auftreten eines aufgezeichneten RA-Wertes den Server durch die Vorlage des Authenticators maskieren.

Trotz dieser möglichen Angriffspunkte hat sich das RADIUS-Protokoll in der Praxis bewährt und wird, wie oben bereits erwähnt, zunehmend auch im Bereich der drahtlosen Vernetzung bei WLAN-Zugängen eingesetzt.

AAA-Architektur

RADIUS ist eine vereinfachte Implementierung der so genannten AAA-Architektur. Diese ist eine generische Architektur zur Authentisierung, Autorisierung und Abrechnung (engl. *authentication, autorisation, accounting*), die von der AAA Working Group der Internet Engineering Task Force entwickelt wurde [176]. Das AAA-Framework beschreibt Verfahren, wie Benutzer für verschiedene Dienste autorisiert werden können, ohne jedoch festzulegen, welche Authentifikationsverfahren die beteiligten Organisationen dazu zu verwenden haben. Darüber hinaus spezifiziert das Framework auch ein Verfahren zum Roaming zwischen unterschiedlichen Organisationen. In einer AAA-Architektur unterscheidet man

- den Benutzer (user), der Zugriff auf einen Dienst oder eine Ressource erlangen möchte,
- die User Home Organization (UHO), die dem Benutzer bestimmte Dienste anbietet, welche vertraglich mit ihm festgelegt werden,
- den Service Provider, der Benutzern oder Organisationen Dienste anbietet,
- den AAA-Server eines Service Providers, der einem Benutzer die Nutzung eines Dienstes gestattet oder verweigert, wobei die Autorisierung

- sich aus seiner Beziehung zur UHO des zugreifenden Benutzers ergibt. Der Server benötigt also keine Kenntnisse über den Benutzer,
- das Service Equipment des Providers, das den Dienst bereitstellt, wie beispielsweise einen Einwähldienst.

In diesem Framework hat also ein Benutzer eine Beziehung zu seiner UHO, die ihrerseits eine Beziehung zu Service Providern haben kann. Der Benutzer hat jedoch keine Kenntnis, welche Dienste seine UHO von welchem Provider bezieht.

Die Architektur schreibt vor, dass jedes AAA-Protokoll eine Basismenge von Verschlüsselungsverfahren verwenden muss, wobei explizit nur Public-Key-Verfahren und Verfahren zur digitalen Signatur gefordert werden. Ferner wird empfohlen, zur Verschlüsselung vollständiger AAA-Nachrichten Protokolle niedrigerer Protokollsichten einzusetzen. Empfohlen wird hierfür explizit IPSec (vgl. Kapitel 14.3).

Neben der Authentifikation bietet das RADIUS-Protokoll auch die Möglichkeit, eine Protokollierung für ein Accounting durchzuführen. Ein Client kann eine derartige Protokollierung initiieren, indem er eine Account-Request-Nachricht vom Typ Start an den Server sendet. Protokolliert werden können u.a. die Anzahl der Pakete, die ein Client während der Dauer der Verbindung empfangen und gesendet hat, oder auch wieviele Verbindungen der Benutzer parallel verwendet. Für einen Einsatz im Zusammenhang mit drahtlosen Netzen über einen Access Point als RADIUS-Client (vgl. Seite 914) wäre es wünschenswert, auch die IP-Adresse des Endpunktes protokollieren zu können. Dafür sieht das RADIUS-Protokoll jedoch kein dediziertes Attribut vor, so dass diese Information in anwendungsspezifisch festlegbaren Attributen übermittelt werden muss. Dies erfordert allerdings, dass Client und Server sich vorab über die korrekte Interpretation von solchen Attributwerten verständigen müssen.

Accounting

Da das RADIUS-Protokoll keine Nachrichtenverschlüsselung verwendet, sondern nur Mechanismen zur Datenintegrität einsetzt, erfüllt es nicht die Sicherheitsanforderungen, die das AAA-Framework stellt. Diesen Forderungen trägt der RADIUS-Nachfolger, das DIAMETER-Protokoll, Rechnung. Dieses Protokoll fordert, dass die Kommunikation zwischen Client und Server mittels IPSec sowie zwischen unterschiedlichen RADIUS-Servern über TLS (Transport Layer Security) (vgl. Abschnitt 14.4) abgesichert wird.

DIAMETER

10.4.2 Kerberos-Authentifikationssystem

Das Kerberos-Authentifikationssystem [165] wurde ursprünglich (Versionen 1–4) im Rahmen des Athena Projektes (Start 1983) am MIT in Kooperation

mit IBM und DEC entwickelt. Das Protokoll wurde nach dem Dreiköpfigen Hund Zerberus (lat. *Cerberus*) benannt, der gemäß der griechischen Mythologie den Eingang zur Unterwelt bewacht und jeden freudig wedelnd in die Unterwelt hineinlässt, aber niemanden zurück ans Tageslicht lässt²⁹.

Es basiert auf dem Needham-Schroeder Protokoll für symmetrische Kryptosysteme (siehe Abschnitt 9.3), das um die von Denning und Sacco vorgeschlagenen Zeitstempel (siehe Seite 422) erweitert wurde. Auf die Verwendung von asymmetrischen Verfahren wurde verzichtet, da in den USA zur Zeit der Entwicklung von Kerberos die Public-Key Verschlüsselung noch unter patentrechtlichen Beschränkungen lag. In dem RFC 4556³⁰ wird mit dem Protokoll PKINIT eine Kerberos-Erweiterung vorgeschlagen, um die Verwendung von asymmetrischen Verfahren mit Kerberos zu kombinieren. In einer erweiterten initialen Anfrage sendet der Client sein X.509-Zertifikat und signiert seine Anfragedaten. Diese Kerberos-Erweiterung wurde von Microsoft für ihre proprietäre Kerberos-Version implementiert.

Version 5

Eine Reihe von Mängeln, die noch bis einschließlich der Version 4 auftraten, sind in der aktuellen Version 5 (siehe auch RFC 4120) beseitigt. Dazu gehört, dass nicht mehr der DES als Verschlüsselungsverfahren fest vorgeschrieben ist, sondern geeignete Verfahren von Clients und Servern wählbar sind. Der aktuelle RFC schreibt vor, dass zur Verschlüsselung der AES256 und als Hashverfahren HMAC-SHA1-96 unterstützt werden müssen und dass die Verfahren AES128 und DES3-CBC-SHA1 unterstützt werden sollten, während u.a. DES-CBC-MD5 nicht mehr unterstützt wird. Die Angabe des verwendeten Verschlüsselungsverfahrens wird als Information der versandten Nachricht hinzugefügt. Außerdem können in Version 5 Authentifikationsinformationen an andere Subjekte weitergereicht werden, so dass ein Server mit den Authentifikationsmerkmalen eines Benutzers in dessen Auftrag (Delegation-Prinzip) tätig sein kann. Die Authentifikationsinformationen werden dezentral durch eine Hierarchie von Authentifikationsservern verwaltet, die jeweils in ihrem Verantwortungsbereich (engl. *realm*) autonom agieren können und miteinander kooperieren, um bereichsübergreifende Zugriffe von Benutzern transparent für diese zu authentifizieren. Auf diese Weise wird ein Single-Sign-On-Konzept realisiert, das verschiedene Verwaltungsbereiche überspannt. Das heißt, dass sich auch in einem verteilten System ein Benutzer nur einmal beim Zugang zu einem der vernetzten Rechner wie z.B. ein Datei-Server authentifizieren muss. Der authentifizierte Zugang zu anderen Rechnern des verteilten Systems wird automatisch, ohne

²⁹ Der Original-Cerberus bewachte also den Ausgang der Unterwelt, während das Kerberos-Protokoll natürlich den Eingang zu dem geschützten System bewacht.

³⁰ Public Key Cryptography for Initial Authentication in Kerberos

Single-Sign-On

Interaktion mit dem Benutzer, durch die „kerborisierten“ Clients und Server abgewickelt.

Kerberos erfüllt zwei Aufgaben: Es stellt Hilfsmittel zur Verfügung, um (1) Subjekte (u.a. Arbeitsplatzrechner, Benutzer oder Server), sie heißen im Kerberos-Kontext Principals, zu authentifizieren und (2) Sitzungsschlüssel auszutauschen. Der Authentifikationsdienst wird von einem vertrauenswürdigen Server (engl. *trusted third party*) erbracht. Dieser ist jedoch nicht in der Lage, Denial-of-Service-Attacken zu erkennen und abzuwehren. Das heißt, dass ein Angreifer die Durchführung von Authentifizierungsschritten für Teilnehmer (Clients) stören und diese dadurch an der Nutzung von Diensten hindern kann. Der Authentifikationsdienst basiert auf symmetrischen Verfahren und geht davon aus, dass die beteiligten Principals ihre geheimen Schlüssel sicher verwalten. Da das Protokoll Zeitstempel mit einer globalen Zeit einsetzt, wird darüber hinaus von den beteiligten Clientrechnern erwartet, dass deren Systemuhren lose mit den Uhren des Authentifizierungsdienstes sowie der Server, die von den Clients genutzt werden sollen, synchronisiert sind. Auf diese Weise wird sichergestellt, dass ihre Uhren sich nur um einen festgelegten Toleranzwert unterscheiden können.

Zu beachten ist, dass Kerberos ein reiner Authentifikations- und Schlüsselaustauschdienst ist. Die Vergabe von Rechten zur Nutzung von netzwerkweiten Diensten sowie die Kontrolle der Zugriffe gehört nicht zu seinem Funktionsumfang.

Aufgaben

Anforderungen

keine Zugriffs-kontrolle

Kerberos im Überblick

Kerberos basiert auf einem Client-Server-Modell (vgl. Abbildung 10.6), in dem ein Principal einen Dienst eines Servers S nur dann in Anspruch nehmen darf, wenn er dafür dem Server eine Authentizitätsbescheinigung, ein Ticket, das ein vertrauenswürdiger Kerberos-Dienst ausgestellt hat, vorweisen kann. Kerberos ist zusammen mit herkömmlichen Netzwerkdiensten, z.B. mit NFS, Remote Login oder Electronic Mail, verwendbar. Dazu müssen diese Dienste, d.h. die Server, die diese Dienste anbieten, um die nachfolgend beschriebenen Fähigkeiten zur Überprüfung von Authentizitätsnachweisen erweitert werden.

Client-Server

Die Benutzer eines um Kerberosdienste erweiterten Systems melden sich bei ihrem Arbeitsplatzrechner (in Abbildung 10.6 ist das der Client C) über die Vorlage eines Passwortes wie üblich an. Die für die entfernten Server-Zugriffe erforderlichen Authentizitätsbescheinigungen werden danach vom Clientrechner, transparent für den Benutzer, beim vertrauenswürdigen Authentifikationsserver AS angefordert. Dieser arbeitet mit einem weiteren vertrauenswürdigen Server, dem Ticket-Granting Server (TGS), zur Ausstellung der Tickets zusammen. Die Dienste des AS und TGS werden häufig

Login

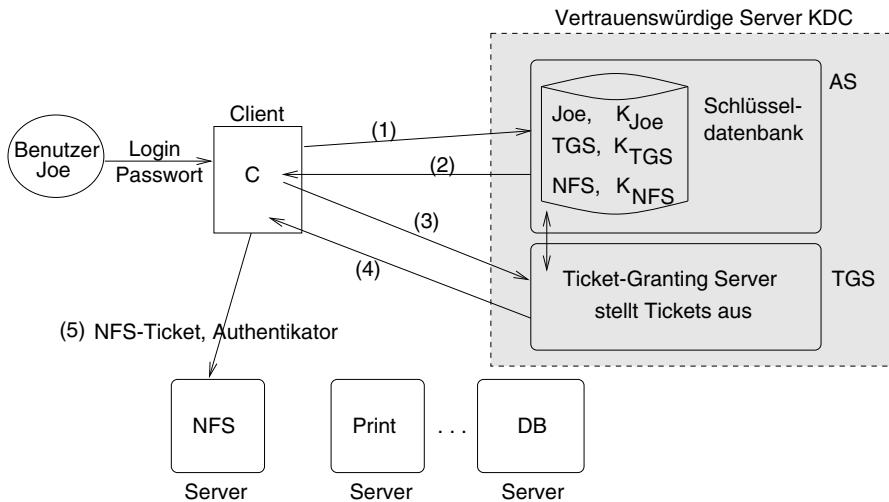


Abbildung 10.6: Kerberos-Grobarchitektur

in einem Server zusammengefasst, der als Schlüsselverteilungsserver KDC (Key Distribution Center) bezeichnet wird. In einem großen System können mehrere derartige Server vorhanden sein.

Basis

Jeder Principal A hat vorab mit dem Kerberos-Schlüsselverteilungs-Server KDC einen geheimen Master Key K_A vereinbart, der vom KDC in einer Schlüsseldatenbank verwaltet wird. Für einen Benutzer ist dieser Schlüssel aus dessen Passwort i.d.R. unter Verwendung eines HMAC-Hash-Verfahrens abgeleitet, so dass der KDC nur die Hashwerte der Benutzerpasswörte sicher verwalten muss und der jeweilige Client-Rechner den Master-Schlüssel bei Bedarf, d.h. beim Login des Benutzers, berechnen kann. Der Benutzer ist von einer Verwaltung dieser Master-Schlüssel entlastet. Der Benutzer präsentiert beim Login-Vorgang wie gewöhnlich sein Passwort, das der Client-Rechner verwendet, um den Master-Key für die Kommunikation mit dem KDC zu bestimmen. Ein KDC verwaltet die Master-Keys sicher, indem er sie mit seinem eigenen Schlüssel verschlüsselt. Zur Verschlüsselung wird in der Regel der AES verwendet, wobei wie bereits angemerkt, die Version 5 auch den Einsatz anderer Verschlüsselungsverfahren ermöglicht. Dazu ist ein Datenfeld in den Datenpaketen vorgesehen, in das der Identifikator des zu verwendenden Krypto-Verfahrens eingetragen werden kann. Alle Master-Keys von Principals werden mit dem gleichen Schlüssel des KDC, dem KDC-Master-Key, verschlüsselt.

Identifikation

Im KDC wird jeder Principal durch ein Tupel (*Name*, *Instanz*, *Bereich*) beschrieben. Falls der Principal ein Benutzer ist, so ist der *Name* die Benutzerkennung und die *Instanz* ist entweder Null oder sie repräsentiert spezifische Attribute, wie beispielsweise Root. Handelt es sich bei dem Prin-

cipal um einen Service, so bezeichnet *Name* dessen Namen und die *Instanz* ist die IP-Adresse des Rechners, auf dem der Dienst implementiert ist. Der *Bereich* (engl. *realm*) wird verwendet, um zwischen unterschiedlichen Authentifikationsbereichen differenzieren zu können.

Das Login unterscheidet sich bei den Versionen 4 und 5. Bei der Version 4 sendet der Clientrechner den Benutzernamen zum KDC, dieser prüft nach, ob es sich um einen angemeldeten Benutzer handelt und sendet ein mit dem Passwort des Benutzers (bzw. mit dem Schlüssel, der daraus abgeleitet wird) verschlüsseltes Initialticket zurück. Der Clientrechner promptet nun seinen Benutzer nach dem Passwort, verwendet dieses, um den notwendigen Schlüssel daraus zu berechnen und um das Ticket zu entschlüsseln. Danach kann das Passwort gelöscht werden. Auf diese Weise sollte die Zeit, in der das Benutzerpasswort auf dem Clientrechner gespeichert werden muss, minimiert werden. Leider eröffnet sich damit jedoch ein möglicher Angriffspunkt, da ein Angreifer sich als ein Benutzer ausgeben und das Initialticket anfordern kann. Mit dem Besitz des Tickets ist er nun in der Lage, offline eine Wörterbuch-Attacke auf das Passwort des Benutzers, für den das Ticket ausgestellt wurde, durchzuführen.

Login

Mit der Version 5 wurde diese Angriffsmöglichkeit ausgeschaltet, indem der KDC bereits für die Ausstellung des Initialtickets von dem Clientrechner fordert, dass der Benutzer die Kenntnis des korrekten Passwortes nachweist. Das heißt, dass beim Login unter der Version 5 der Clientrechner den Benutzer nach Kennung und Passwort fragt, aus dem Benutzer-Passwort, wie bereits erwähnt, den Master-Key des Benutzers berechnet und diesen verwendet, um dem KDC eine verschlüsselte Nachricht, die den Benutzernamen und die aktuelle Uhrzeit enthält, zu schicken. Der KDC versucht, diese Nachricht zu entschlüsseln und überprüft die Aktualität der Uhrzeit (nicht älter als 5 Minuten). Erst wenn alles korrekt ist, wird er ein Initialticket ausstellen.

Ticket

Ein vom TGS ausgestelltes Ticket ist nur für einen Server gültig. Tickets dürfen nicht mit Capabilities (vgl. Kapitel 12) verwechselt werden, da sie nur zur authentifizierten Nutzung eines Dienstes dienen³¹. Jedes Ticket $T_{c,s}$ enthält im Wesentlichen folgende Informationen:

Ticket

$$T_{c,s} = S, C, \text{addr}, \text{timestamp}, \text{lifetime}, K_{c,s}.$$

S ist der Name des Servers, der in Anspruch genommen werden soll, C ist der Name des anfordernden Clients, präziser, es ist der Name des Principals, der authentifiziert wurde, und addr ist dessen IP-Adresse. Das

³¹ Dennoch ist natürlich das zugrunde liegende Prinzip der Ausstellung von Ausweisen das Gleiche wie bei Capabilities.

Ticket enthält ferner den Sitzungsschlüssel $K_{c,s}$, der vom KDC für die Kommunikation zwischen S und C erzeugt wird und der auch C bekannt gegeben wird (siehe Nachricht (2) bzw. (4) des Protokolls in Tabelle 10.2). Solange die Lebenszeit des Tickets, angegeben durch einen Anfangs- und einen Endwert, nicht abgelaufen ist, kann es der Client für Zugriffe auf den Server S verwenden. Das Ticket wird mit dem Master Key K_s des Servers S verschlüsselt, $\{T_{c,s}\}^{K_s}$, und so vom TGS dem Client zugestellt, der es seinerseits bei Bedarf dem Server S präsentiert.

Initialticket

Da der TGS ebenfalls ein Server ist, dessen Service, nämlich die Ticket-Ausstellung, nur durch Vorlage eines gültigen Tickets in Anspruch genommen werden darf, wird ein Initialticket, das Ticket-Granting-Ticket (TGT), benötigt, das der Clientrechner automatisch beim KDC nach dem erfolgreichen Login eines Benutzers anfordert. Die Verwendung solcher TGTs hat den Vorteil, dass der KDC im Wesentlichen zustandslos arbeiten kann, da er nach einer Authentifizierungsanfrage alle Antwortdaten in das Ticket verpackt und keine Zustandsinformationen über den Client anlegen und verwalten muss.

Authentikator

Authentikator

Damit S die Authentizität desjenigen Principals, der ihm ein Ticket $T_{c,s}$ vorweist, verifizieren kann, muss dieser einen so genannten Authentikator erzeugen und diesen zusammen mit dem Ticket versenden. Die Authentikator-Erstellung erfolgt, ebenfalls transparent für den Benutzer, durch dessen Clientrechner. Mit dem Authentikator beweist der Principal, also der Client, dass auch er im Besitz des Sitzungsschlüssels $K_{c,s}$ ist. Um Wiedereinspielungen abzuwehren, muss der Authentikator einen aktuellen Zeitstempel enthalten. Jeder Authentikator enthält im Wesentlichen folgende Informationen:

$$A_c = c, \text{addr}, \text{timestamp} .$$

Der Client C verschlüsselt den Authentikator mit dem gemeinsamen Schlüssel $K_{c,s}$ bevor er ihn dem Server S vorlegt. Diesen gemeinsamen Schlüssel erhält der Client (siehe Protokollschrifte in Tabelle 10.2) vom TGS, indem dieser ihn mit einem Schlüssel verschlüsselt, der zwischen dem TGS und dem Client vereinbart ist. Beim initialen Schritt ist dies natürlich der Masterkey des Principals.

Protokollschrifte

Der Ablauf des Kerberos-Protokolls wird am Beispiel eines Benutzers namens Joe, der ein Authentifizierungs-Ticket für den NFS-Server NFS benötigt, erläutert. Die Protokollschrifte zur Erlangung des Initialtickets für den Principal Joe (Schritte (1) und (2)) sowie zur Ausstellung eines Tickets (Schritte (3) bis (5)) zur Nutzung des NFS-Servers sind in Tabelle 10.2 zusammengefasst. Die Nummerierung entspricht den bereits in Abbildung 10.6

Protokollablauf

angegebenen Protokollschriften. Der in dem Protokoll angegebene Client C ist ein Prozess, der im Auftrag eines Benutzers, in dem Beispiel ist es der Benutzer namens Joe, tätig ist.

Von	An	Nachricht
1. Client	KDC	$\text{Joe}, \text{TGS}, \text{Nonce1}, \{\text{Joe}, \text{Time}\}^{K_{\text{Joe}}}$
2. KDC	Client	$\{K_{\text{Joe}, \text{TGS}}, \text{Nonce1}\}^{K_{\text{Joe}}}, \{T_{\text{Joe}, \text{TGS}}\}^{K_{\text{TGS}}}$
3. Client	TGS	$\{A_{\text{Joe}}\}^{K_{\text{Joe}, \text{TGS}}}, \{T_{\text{Joe}, \text{TGS}}\}^{K_{\text{TGS}}}, \text{NFS}, \text{Nonce2}$
4. TGS	Client	$\{K_{\text{Joe}, \text{NFS}}, \text{Nonce2}\}^{K_{\text{Joe}, \text{TGS}}}, \{T_{\text{Joe}, \text{NFS}}\}^{K_{\text{NFS}}}$
5. Client	NFS	$\{A_{\text{Joe}}\}^{K_{\text{Joe}, \text{NFS}}}, \{T_{\text{Joe}, \text{NFS}}\}^{K_{\text{NFS}}}$

Tabelle 10.2: Kerberos-Protokollablauf (Version 5)

Zur Erlangung des Initialtickets sendet der Client in Nachricht (1) den Namen des zu authentifizierenden Principals, hier Joe, zusammen mit einer Nonce zum KDC. Mit $\{\text{Joe}, \text{Time}\}^{K_{\text{Joe}}}$ weist der Benutzer nach, dass er Kenntnis über das korrekte Passwort besitzt. Falls der Benutzer in der Datenbank des KDC registriert ist, stellt dieser ein Ticket-Granting Ticket $T_{\text{Joe}, \text{TGS}}$ für Joe aus und verschlüsselt es mit dem geheimen Schlüssel K_{TGS} des Servers TGS,

$$\{T_{\text{Joe}, \text{TGS}}\}^{K_{\text{TGS}}}.$$

*Ticket-Granting
Ticket*

Das Ticket enthält den vom KDC erzeugten Sitzungsschlüssel $K_{\text{Joe}, \text{TGS}}$. Dieser wird in Schritt (2) zusammen mit dem Ticket an den Client zurückgesandt. Damit diese Information vertraulich bleibt, ist sie mit dem Masterschlüssel von Joe verschlüsselt, also

$$\{K_{\text{Joe}, \text{TGS}}, \text{Nonce1}\}^{K_{\text{Joe}}}.$$

Die verwendeten Nonces dienen zur Erkennung von Replays. Um das Ticket-Granting-Ticket entschlüsseln zu können, benötigt der Clientrechner den Masterschlüssel K_{Joe} des Benutzers Joe. Da dieser ja aus dem Benutzerpasswort abgeleitet ist, berechnet der Client ihn aus den ihm vorliegenden bzw. interaktiv vorzulegenden Passwortinformationen. Tickets und Sitzungsschlüssel sind von den Clientrechnern sicher zu verwalten. Der Client cashed alle verwendeten Tickets in dem `/tmp`-Verzeichnis und löscht sie, wenn sich der Benutzer ausloggt.

Um mit dem TGS bzw. KDC Kontakt aufnehmen zu können, muss der Clientrechner diesen lokalisieren. Die entsprechende Information kann zum

Beispiel manuell in einer Konfigurationsdatei eingetragen sein. In der Regel wird hierfür die Datei `krb5.conf` verwendet. Alternativ kann ein KDC seine Dienste aber auch über DNS bekannt machen.

Ticket-Anfrage

Der TGS stellt Tickets auf Anfragen gemäß Nachricht (3) aus. In der Anfrage ist neben dem Namen des Principals und einer Nonce auch der Name des Servers angegeben, für den ein Ticket benötigt wird. Im Beispiel ist dies der *NFS*-Server. Über die Vorlage des Ticket-Granting-Tickets, hier $T = \{T_{Joe,TGS}\}^{K_{TGS}}$, und eines Authentikators dafür, hier $A = \{A_{Joe}\}^{K_{Joe,TGS}}$, weist der Client seine Authentizität gegenüber dem TGS nach. Der TGS entschlüsselt das Ticket T unter Anwendung seines Schlüssels K_{TGS} und besitzt danach ebenfalls den Sitzungsschlüssel $K_{Joe,TGS}$. Mit diesem kann er im nächsten Schritt den Authentikator A entschlüsseln und die Aktualität der darin enthaltenen Zeitmarke überprüfen. Veraltete Authentikatoren werden als ungültig zurückgewiesen.

Nach einer erfolgreichen Authentifikation erzeugt das KDC einen Sitzungsschlüssel, hier $K_{Joe,NFS}$, für die Kommunikation zwischen dem Benutzer Joe und dem NFS-Server. Der TGS sendet dann in Schritt (4), analog zu Schritt (2), ein NFS-Ticket sowie diesen Schlüssel zurück.

Server-Authentifikation

Wird eine wechselseitige Authentifikation gefordert, so ist das angegebene Protokoll um einen Schritt (5') zu erweitern. Der Server authentifiziert sich gegenüber dem Client, indem er den um eins erhöhten Zeitstempel aus dem Authentikator $\{A_{Joe}\}^{K_{Joe,NFS}}$ verschlüsselt mit dem gemeinsamen Schlüssel $K_{Joe,NFS}$ zurückschickt. Damit hat der Server bewiesen, dass er den Schlüssel $K_{Joe,NFS}$ sowie den geheimen Schlüssel K_{NFS} kennt.

Beispiel 10.6 (Kerberisiertes NFS)

Der Einsatz von Kerberos erfordert, dass alle beteiligten Server und Clients um Kerberos-Funktionalität erweitert werden, um das Authentifikationsprotokoll wie beschrieben durchzuführen. Normalerweise ist dazu der Zugriff auf den Quellcode des Dienstes notwendig. Betrachten wir als Beispiel das Network File System NFS. Für das Standard NFS (siehe Kapitel 3.4.2) gilt, dass ein Server, der einen Teil seines Dateisystems an einen Clientrechner exportiert hat, diesem implizit vertraut, dass er seine Benutzer korrekt authentifiziert. Falls sich nun ein beliebiger Benutzer, nennen wir ihn Joe, auf dem Client einloggt, darf er auf die exportierten Dateien zugreifen, wenn er die entsprechenden Rechte besitzt. Unter NFS ist dann folgender Maskierungsangriff nicht erkennbar. Nehmen wir an, dass der Angriff durch einen Benutzer Joe erfolgt, der Superuser-Rechte auf dem Clientrechner C besitzt. Nun kann er seine interne eindeutige Benutzerkennung `uid` ändern und stattdessen diejenige eines Opfers Bill annehmen. Zugriffe unter der Mas-

NFS

Maskierungsangriff

ke von Bill auf dessen Daten in dem exportierten Dateisystem sind danach unmittelbar möglich.

Sind dagegen sowohl der Clientrechner als auch der NFS-Dateiserver mit Kerberos-Funktionalität ausgestattet, so geschieht Folgendes: Der Benutzer Joe loggt sich ein und die Software des Clientrechners sorgt dafür, dass die benötigten Tickets für den Zugriff auf den Dateiserver ausgestellt werden. Wichtig hierbei ist, dass jedes Ticket den Benutzernamen enthält. Betrachten wir nun erneut das oben skizzierte Angriffszenario. Der Benutzer Joe versucht sich als Bill auszugeben, ohne sich jedoch unter der Benutzerkennung Bill auf dem Clientrechner eingeloggt zu haben. Das bedeutet, dass Joe kein Ticket für den Principal Bill vom KDC besitzt und seine Versuche, mit seinem eigenen Ticket unter der Maske von Bill auf den Dateiserver zuzugreifen, von diesem direkt erkannt und abgewehrt werden können.



Kerberosiertes NFS

Sicherheit von Kerberos

Kerberos stellt ohne Zweifel einen erheblichen Fortschritt in Bezug auf die Sicherheit in offenen Netzen dar, da auf netzwerkweite Dienste authentifiziert zugegriffen werden kann und Benutzerpassworte dazu nicht über das Netz übertragen werden müssen. Dennoch weist das Protokoll einige Defizite auf. Die meisten Mängel, deren Ursache in der ursprünglichen Ausführungsumgebung, dem Athena-Projekt, zu finden waren, sind mit der Version 5 bereits behoben worden. Nach wie vor problematisch ist jedoch die Verwaltung von Sitzungsschlüsseln, die in der Verantwortung eines jeden Clientrechners liegt, sowie die Verwendung von IP-Adressen. Im Athena-Projekt wurden nur Einbenutzerrechner verwendet, so dass die sichere Speicherung von Sitzungsschlüsseln und Tickets keine spezifischen Maßnahmen erforderte. Dies trifft bei offenen Mehrbenutzerumgebungen natürlich nicht mehr zu, so dass die sichere Verwaltung dieser Informationen nun eine wesentliche Voraussetzung für einen sicheren Betrieb von Kerberos ist. Da Tickets automatisch im `/tmp`-Verzeichnis abgelegt werden, ist es in Mehrbenutzerumgebungen sehr einfach, Tickets anderer Benutzer zu stehlen und zu missbrauchen.

Athena-Projekt

Um zu verhindern, dass ein Angreifer eine Offline-Attacke auf einen Master-Key eines Benutzers durchführen kann, indem der Angreifer sich einfach ein Ticket für diesen Benutzer ausstellen lässt, werden durch den KDC/TGS keine Tickets für Benutzer ausgestellt.

Die Authentizität eines Clients wird anhand des Authentikators und der darin enthaltenen IP-Adresse überprüft. Da IP-Adressen fälschbar sind, könnte ein Angreifer einen Authentikator abfangen und in eine maskierte TCP/IP-Verbindung einschleusen. Während die Version 4 hierfür keine

Abwehrmaßnahmen vorsieht, bietet die Version 5 ein Challenge-Response-Protokoll an. Dies ist aber lediglich eine wählbare Option, die für einen sicheren Betrieb aber unbedingt genutzt werden sollte.

Passworte

Wesentlicher Schwachpunkt von Kerberos ist sicherlich das Passwortkonzept, das die Basis für die initiale Vergabe von Tickets an Benutzer und Server bildet. Mit erfolgreichen Passwort-Cracking-Angriffen auf Clientrechner lässt sich der gesamte Authentifikationsprozess unterlaufen. Wer es geschafft hat, sich auf einem Clientrechner unter einer falschen Identität einzuloggen, kann unter dieser Maske auch netzwerkweit aktiv werden. Aber auch die Masterkey-Verwaltung auf den KDCs ist problematisch. Der KDC verschlüsselt alle diese Schlüssel mit dem gleichen Server-Master-Key, der ebenfalls auf der Festplatte gespeichert ist. Falls der KDC oder ein Backup kompromittiert ist, müssen alle Passworte von Benutzern erneuert werden.

vertrauenswürdige Zeit

Die Verwendung der Authentikatoren zur Erkennung von Replayangriffen ist ebenfalls nicht ganz unproblematisch, da die Erkennung auf der Basis der Aktualität von Zeitstempeln erfolgt. Kerberos basiert darauf, dass sich die beteiligten Rechner bezüglich einer globalen Zeit synchronisieren. Falls es aber einem Angreifer gelingt, einem Server eine falsche Zeit (eine bereits abgelaufene Zeit) als aktuelle Zeit „unterzuschieben“, so ist es ihm möglich, bereits verwendete Authentikatoren für Maskierungssangriffe zu missbrauchen. Der Server kann dann nämlich beim Testen der Aktualität des Authentikators nicht mehr erkennen, dass es sich um eine Wiedereinspielung handelt. Da viele Rechner auf nicht vertrauenswürdige Synchronisationsprotokolle für ihre Uhren zurückgreifen, sind solche Angriffe keineswegs unrealistisch. Abhilfen könnten hier Challenge-Response-Schritte unter Verwendung von Nonces schaffen. Diese sind aber auch in der Version 5 nicht vorgesehen.

Sitzungsschlüssel

Zu beachten ist auch noch, dass die vom KDC erzeugten Sitzungsschlüssel keine Sitzungsschlüssel im eigentlichen Sinn sind. Vielmehr kann ein solcher Schlüssel von einem Client für jede Kommunikationsverbindung mit dem Server, für den das Ticket gilt, verwendet werden. Da Tickets eine unter Umständen lange Gültigkeit haben, unter Version 4 beträgt die maximale Gültigkeitsdauer 21 Stunden, während unter Version 5 sogar eine maximale Dauer bis zum 31.12.9999 möglich ist, wird in diesem Fall auch der Sitzungsschlüssel über einen sehr langen Zeitraum verwendet und ist währenddessen möglichen Angriffen ausgesetzt. Die Version 5 sieht jedoch eine Option vor, einen Sitzungsschlüssel zu erneuern bzw. Tickets mit langer Lebensdauer zurückzurufen. Von dieser Möglichkeit sollte bei Tickets mit langer Gültigkeit unbedingt Gebrauch gemacht werden.

Single-Sign-On

In der Version 4 gibt es keine Möglichkeit, Tickets in einem größeren Netzwerk weiterzureichen (engl. *forwarding*). Tickets sind nur innerhalb des

Bereiches desjenigen TGS gültig, der sie ausgestellt hat. Möchte ein Client einen authentischen Netzdienst eines Rechners, der außerhalb des Gültigkeitsbereichs seines für ihn zuständigen KDCs liegt, in Anspruch nehmen, so muss er dafür ein gültiges Ticket vorweisen, wofür das Passwort über das Netz zu übertragen ist. Dies läuft jedoch der Grundphilosophie von Kerberos zuwider. In der Version 5 wird das Problem dadurch gelöst, dass es eine Hierarchie von KDCs gibt, die das Forwarding transparent und insbesondere ohne eine Passwortübertragung durchführen. Die KDCs realisieren somit ein Single-Sign-On. Dies ist einerseits für eine einfache Handhabung des Systems sicher sehr wünschenswert und fördert die Akzeptanz von Sicherheitsmaßnahmen beim Endbenutzer; andererseits entfallen durch das automatisierte Vorgehen Interaktionen mit dem Benutzer, die für eine erneute Überprüfung der Authentizität nützlich sein könnten. Ein erfolgreicher Maskierungssangriff in einem Single-Sign-On-System kann somit einen großen Wirkungsbereich haben, so dass der Zugriff auf sensible Daten durch zusätzliche, explizite Kontrollen differenziert geschützt werden sollte.

Fazit

Kerberos sieht keine Maßnahmen vor zur Gewährleistung der Nachrichtenintegrität und der nicht abstreitbaren Zuordenbarkeit von Nachrichten zu Absendern. Dies wäre mit dem Einsatz von Message Authentication Codes (MAC) oder mit digitalen Signaturen erreichbar. Durch den sich rasant verbreitenden Einsatz von Smartcards steht zu erwarten, dass in nachfolgenden Kerberos-Versionen das sicherheitskritische Passwortverfahren durch ein smartcardbasiertes Challenge-Response-Protokoll mit öffentlichen Schlüsseln ersetzt wird. Trotz der aufgezeigten Mängel stellt die Version 5 des Kerberos-Protokolls eine deutliche Verbesserung der Sicherheit in vernetzten IT-Systemen dar.

Microsoft verwendet Kerberos Version 5 in den aktuellen Versionen seiner Betriebssystem-Familien, um Benutzern, die sich mit ihrem Domänenamen anmelden, also nicht bei ihrer lokalen Kennung, zu authentifizieren. Das originäre Kerberos ist ein Authentifizierungs- aber kein Autorisierungsdienst. Dies führt in Betriebssystemen wie Windows oder Unix zu Problemen, da sie für ihre jeweiligen Zugriffskontroll-Dienste (vgl. Kapitel 12) nicht den Namen des Principals, wie er im Ticket enthalten ist, benötigen, sondern eindeutige Benutzer-Identifikatoren (uid) bzw. auch Gruppen-Identifikatoren (guid). Fehlt dies, so müsste jeder Server eine Tabelle verwalten, die den Principal-Namen auf diese Identifikatoren abbildet. In der Kerberos-Version von Windows wurde deshalb das Ticket um das *authorization data*-Feld erweitert, wodurch das Ticket nicht mehr kompatibel mit der ursprünglichen Spezifikation ist.

Microsoft
Windows

11 Digitale Identität

Digitale Identitäten für Personen, aber auch für Geräte und smarte Objekte gewinnen mit der Digitalisierung erheblich an Bedeutung. Das Kapitel beleuchtet deshalb das Thema der digitalen Identität aus verschiedenen Perspektiven. Smartcards sind eine wichtige Technologie, um eine digitale Identität vertrauenswürdig nachzuweisen, so dass wir zunächst in Abschnitt 11.1 auf Smartcard-Architekturen und deren Sicherheitseigenschaften eingehen. Hoheitliche, digitale Ausweis-Dokumente, wie der elektronische Reisepass und der elektronische Personalausweis werden im Abschnitt 11.2 detailliert als wichtige Fallbeispiele für die digitale Identität diskutiert.

Zur einfachen, sicheren Identifizierung von Nutzern im Internet hat die FIDO Allianz (Fast IDentity Online) Standards entwickelt, wie das Universal Second Factor-Protokoll (U2F). Abschnitt 11.3 stellt das U2F Protokoll detailliert vor und diskutiert dessen Sicherheitseigenschaften.

Spezielle Smartcards, die an Rechnerplattformen gebunden sind, sind die Trusted Plattform Module (TPM), die im Rahmen der Trusted Computing Initiative entwickelt werden. Abschnitt 11.4 erläutert die wesentlichen Konzepte des Trusted Computings.

Abschließend werden in Abschnitt 11.5 mit den Physically Unclonable Functions (PUF) neue Hardware-basierte Sicherheitsansätze erläutert, die es ermöglichen, Objekte eindeutig anhand ihrer unverfälschbaren, physikalischen Merkmalen zu identifizieren.

11.1 Smartcards

Besitzbasierte Authentifikationstechniken stützen sich in der Praxis meist auf den Besitz von Smartcards¹ ab, wobei aber zunehmend auch andere Formfaktoren wie USB-Token eingesetzt werden. In diesem Abschnitt beschränken wir uns aber auf Smartcards, da die meisten Smartcard-Konzepte, die wir besprechen, auch auf andere Formfaktoren übertragbar sind. Als Geburtsstunde der Chipkarte gilt das Jahr 1974, in dem dem französischen Wirtschaftsjournalisten R. Moreno ein Patent für ein „System zur Spei-

¹ Im Folgenden wird der Begriff Smartcard und Chipkarte synonym verwendet.

Speicherkarte

cherung von Daten in einem unabhängigen, tragbaren Gegenstand“ erteilt wurde. Die heutige Chipkarte ist eine Plastikkarte, die in unterschiedlichen Ausprägungen vorkommt. Die einfachsten Typen sind reine Speicherkarten, die nur über einen nicht flüchtigen EEPROM-Speicher von einigen Hundert Bytes bis 8 KByte verfügen, keine Rechenfähigkeit, also keine CPU besitzen und monofunktional sind. Derartige Karten können sehr preiswert produziert werden. Beispiele für Speicherkarten sind Telefon- oder Krankenversicherungskarten. Die nächste Stufe bilden die intelligenten Speicherkarten, die eine zusätzliche Sicherheitslogik besitzen, die meist für eine PIN-Speicherung und -Überprüfung verwendet wird. Weiterhin verfügen sie in der Regel über einen Zähler für Fehlversuche, um nach einer Anzahl solcher Fehlversuche die Karte zu sperren. Sind die Karten auch noch mit einem eigenen Mikroprozessor und einem programmierbaren Speicher ausgerüstet, so spricht man von Smartcards. Das Ur-Patent für eine Smartcard (Patent-Nr. DE 19 45 777 C3) datiert vom 10. September 1968 und der Patent-Inhaber ist der Deutsche Jürgen Dethloff.

Smartcard

Durch das schnell sich verbessernde Preis/Leistungsverhältnis von Smartcards kommt diesen bereits heute eine große Bedeutung als Sicherheitswerkzeug und insbesondere als Mittel zur Authentifikation zu. Smartcards sind in vielfältigen Bereichen einsetzbar, als GeldKarte oder Visa-Cash Karte im Bankensektor, als Campus- und Firmen-Karte oder als SIM-Karte in Mobiltelefonen. Mit der Gesundheitskarte (eGK) oder auch dem digitalen Personalausweise werden bereits seit einigen Jahren Smartcards für alle Bürger millionenfach ausgerollt. Einen sehr guten Überblick über die Architektur, Betriebsssoftware und Anwendungsbereiche für Smartcards gibt das Buch von Rankl und Effing [178].

Einsatzbereich

11.1.1 Smartcard-Architektur

ISO 7816

Der ISO 7816 Standard enthält Vorgaben für die Karten-Dimensionen, die Anordnung der Kartenkontakte, die zulässigen Signale und Spannungswerte sowie die Übertragungsprotokolle. Kontaktbehaftete Smartcards interagieren mit ihrer Umwelt mittels fünf Kontakten (Vcc, GND, RST, GLK, I/O) während kontaktlose über Funk kommunizieren. Ein nur optional vorhandener Magnetstreifen enthält meist keine sicherheitskritischen Informationen, sondern Daten wie die Personalnummer des Besitzers oder das Verfallsdatum der Karte.

Kartentypen

Kartentypen

Man unterscheidet drei Kartengrößen, nämlich die Typen ID-1, ID-000 und ID-00. Die Standardkarte ist vom Typ ID-1 und besitzt die Größe einer Kreditkarte (siehe Abbildung 11.1. Bitte beachten Sie, dass Abbildung die Größenverhältnisse nicht 1:1 widerspiegelt.).

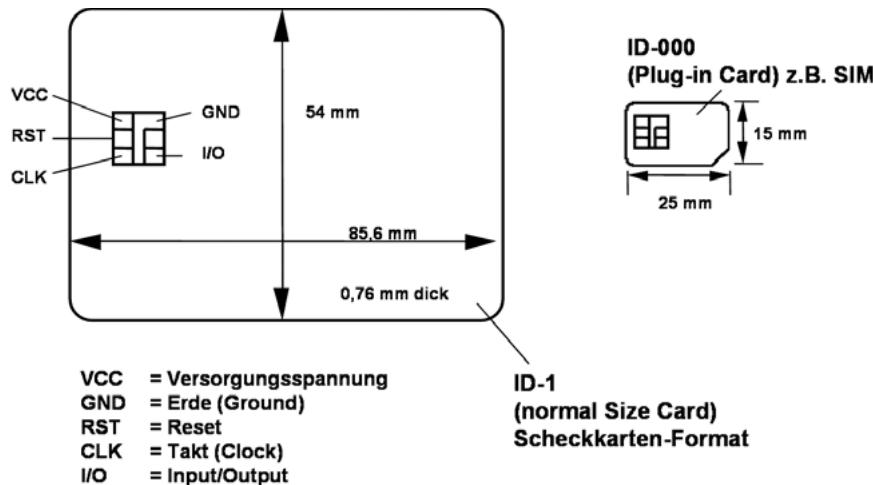


Abbildung 11.1: Standardisierte Kartenformate ID-1 und ID000

Die ID-000 Karte mit der Größe von $15\text{mm} \times 25\text{mm}$ ist eine Plug-in Karte, die ursprünglich für Mobiltelefone im Rahmen des GSM (Global System for Mobile Communication) (siehe Abschnitt 15.1) entwickelt wurde. Die ID-00 Karten, auch bekannt unter dem Namen Minikarten, haben eine Größe von $66\text{mm} \times 33\text{mm}$. Da deren Kontakte auf die gleiche Art angeordnet sind wie diejenigen bei Karten des Typs ID-1, können Kartenlesegeräte beide Kartentypen akzeptieren. Die kleineren Karten verzichten im Vergleich zum Typ ID-1 schlicht auf überflüssiges Plastik. Die durch die Normung vorgegebene Beschränkung der Grundfläche eines Chips auf 25mm^2 trägt der physischen Zerbrechlichkeit der Siliziumträger Rechnung.

Smartcard-Mikrocontroller

Der zentrale Baustein einer Chipkarte ist der Mikrocontroller, der unter dem Kontaktfeld der Karte eingebettet ist. Die Architektur eines Smartcard-Mikrocontrollers ist in Abbildung 11.2 skizziert. Wie man deutlich sieht, besitzt sie eine große Ähnlichkeit mit der Architektur eines Standard-PCs. Sie besteht aus einer CPU, einem ROM (Read only memory), einem RAM (Random access memory), Ein/Ausgabe-Kanälen sowie einem EEPROM (Electrically Erasable Programmable ROM) Speicher, dessen Aufgabe mit der einer Festplatte vergleichbar ist. High-End Microcontroller verwenden heute bereits einen 32-Bit Prozessor, während einfache Chips nach wie vor 8-Bit oder 16-Bit Prozessoren besitzen mit einem 16-Bit Adressbus.

Mikrocontroller

Der ROM-Speicher mit einer Größe, die von wenigen KByte (17–32 KByte) bei einfachen Controllern bis hin zu 320 KByte bei High-End Controllern reichen kann, ist maskenprogrammiert, so dass er keine Programme und Daten enthalten kann, die kartenspezifisch sind oder die sich über die

ROM

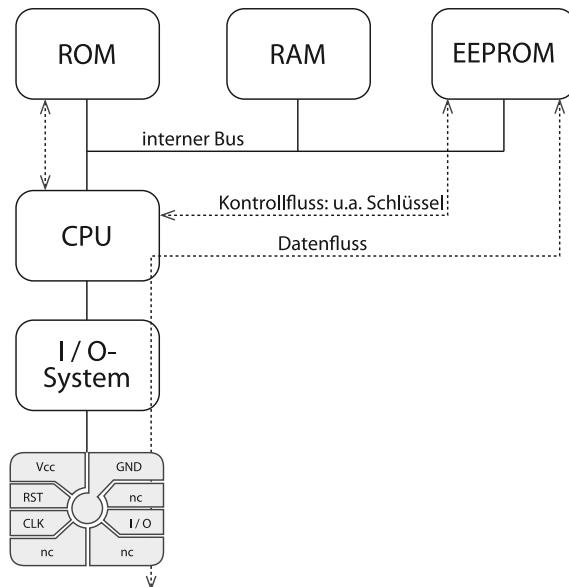


Abbildung 11.2: Architektur einer Smartcard

Lebenszeit der Karte hinweg ändern. Eine Änderung des ROM-Codes erfordert eine neue Maske zum Belichten und Ätzen der Siliziumscheiben. Typischerweise enthält der ROM das Karten-Betriebssystem, das bei der Chip-Produktion geladen wird, Verfahren zur PIN-Überprüfung und kryptografische Verfahren zum Verschlüsseln, Signieren und zur Berechnung von Hashwerten.

RAM

Der RAM-Speicher mit einer Größe, die ebenfalls stark variiert und von einigen hundert Byte (256–512 Byte) bis zu 16 KByte reichen kann, ist ein flüchtiger Speicher, dessen Inhalt verloren geht, sobald keine Stromversorgung mehr existiert. Er dient der CPU als Arbeitsspeicher zur Ausführung der Algorithmen. Lese- und Schreibzugriffe auf den RAM sind sehr schnell und erfordern nur einige Mikrosekunden. Sie sind damit um einen Faktor Tausend schneller als ein Schreibzugriff auf einen (E)EPROM. Da aber der RAM einen großen Flächenbedarf hat, ist er im Vergleich zu den beiden anderen Speicherkomponenten klein, damit die zugelassene Gesamtgröße der Karte nicht überschritten wird.

EEPROM

Der EEPROM mit einer Größe von typischerweise 8KByte bis 32 KByte im Low-End Bereich und von 132 KByte bis 400 KByte im High-End Bereich ist ein nicht-flüchtiger Speicher zur langfristigen Speicherung von benutzer-spezifischen, veränderbaren Daten wie z.B. die PIN, die Sitzungsschlüssel oder die Kontonummer. Die EEPROM-Technologie ermöglicht mindestens 100.000 Umprogrammierungen mittels elektrischer Signale, so dass Karten

mit diesem Speicher sehr lange und relativ flexibel nutzbar sind. Die einfacheren EPROMs lassen sich dem gegenüber nicht durch elektrische Signale ändern, sondern erfordern das Einwirken von UV-Licht, so dass sie nur für solche Anwendungen geeignet sind, deren Daten sich nur selten oder gar nicht ändern.

Das I/O-System dient zum bitseriellen Datentransfer mit einer Datenrate von mindestens 9600 Bit/s. Zur Datenübertragung wurden zwei asynchrone Halbduplex-Protokolle standardisiert, die als T=0 und T=1 bezeichnet werden. T=0 ist ein byteorientiertes Protokoll, das für jedes empfangene Byte dessen Paritätsbit prüft und im Falle eines Fehlers eine erneute Übertragung veranlasst. Das T=1 Protokoll arbeitet demgegenüber blockorientiert.

Protokolle

Fortgeschrittene Karten können über zusätzliche Komponenten, die Additional Units (AU), verfügen. Ein Beispiel ist ein Co-Prozessor zur beschleunigten Ausführung modularer Arithmetik oder zur Beschleunigung der Exponentiationen, die ja u.a. beim RSA-Verfahren benötigt werden. Weitere mögliche Zusatzkomponenten einer Smartcard sind eine Memory Management Unit (MMU), ein Zeitgeber oder ein Universal Asynchronous Receiver/Transmitter (UART). Ferner kann ein Chip auf seiner Oberfläche ein Foto des Kartenbesitzers, ein Hologramm oder eingestanzte Daten besitzen.

weitere Funktionen

Die Komponenten einer Chipkarte sind noch einmal in der Tabelle 11.1 zusammengefasst.

	ROM	Betriebssystem, Kryptoverfahren, PIN-Prüfung, Übertragungsprotokolle
CPU	RAM	Arbeitsspeicher
AU	EEPROM	u.a. Schlüssel, PIN, Kontonr.

Tabelle 11.1: Komponenten einer Smartcard

11.1.2 Betriebssystem und Sicherheitsmechanismen

Jeder Zugriff auf den Speicher und insbesondere auf die Daten des im EEPROM verwalteten Dateisystems wird durch das Kartenbetriebssystem überprüft. Der Zugriff auf die Dateien kann durch eine PIN oder durch

Kontrollen

kryptografische Schlüssel geschützt werden. Der Kartenaussteller legt fest, nach wie vielen Fehlversuchen bei der PIN-Eingabe die Karte zu sperren ist. Multi-Anwendungskarten erlauben es, für verschiedene Verzeichnisbäume des Dateisystems unterschiedliche PINs festzulegen. Zusammen mit einer logischen Dateisystemstruktur, die Anwendungen gegeneinander abschottet, können diese Anwendungen unterschiedlich zugriffsgeschützt werden. Problematisch an dieser Vorgehensweise ist natürlich, dass der Benutzer damit belastet wird, sich für jede Applikation eine eigene PIN merken zu müssen. Das führt schnell dazu, dass die eigentlich beabsichtigte Erhöhung des Sicherheitsniveaus durch den häufig fahrlässigen und unachtsamen Umgang mit PINs seitens der Benutzer stark gefährdet wird.

Kryptoverfahren

Auf Smartcards werden noch immer vordringlich symmetrische Verfahren zur Verschlüsselung eingesetzt, da diese zum einen in der Regel für ihre Ausführung weniger Speicherplatz als asymmetrische Verfahren benötigen und zum anderen die Ver- und Entschlüsselungsoperationen schnell durchgeführt werden können. Zur effizienten RSA-Realisierung wird eine Langzahlarithmetik benötigt, wofür die vorhandenen RAM-Größen jedoch noch nicht ausreichen. Auch die Generierung von RSA-Schlüsseln ist sehr aufwändig und wird noch häufig außerhalb der Chipkarte erledigt. Da der Berechnungsaufwand bei der RSA-Verschlüsselung stark von der Länge der verwendeten Schlüssel abhängt, werden meist nur RSA-Schlüssel der Länge 512, 768 oder 1024 Bit verwendet. Besonders die ersten beiden Längen genügen aber bereits heute nicht mehr hohen Sicherheitsanforderungen (vgl. auch Abschnitt 7.3.2). Vorteile bieten hier Verfahren, die auf elliptischen Kurven basieren und mit deutlich kleineren Schlüsseln starke Sicherheit bieten.

Problembereiche

Einen einfachen Angriffspunkt für feindliche Kartenlesegeräte bilden die Kontakte der Karte. Passive Angriffe können jedoch durch das Verschlüsseln aller übertragenen Daten wirkungslos gemacht werden. Weitere Sicherungsmaßnahmen sind aus dem Umfeld der Geldautomaten wohlbekannt. Die Kartenterminals befinden sich dort in physisch geschützten Umgebungen und ein motorisierter Kartenleser zieht die eingelegte Karte automatisch ein und versiegelt den Zugang auf die Karte und das Lesegerät für die Dauer der Datenübertragung. Kartenterminals sind bereits heute und werden in der Zukunft in noch verstärkterem Maß in offene, komplexe Systemarchitekturen und Netze eingebunden. Es ist aber klar, dass die Absicherung unsicherer Netze und Architekturen jenseits der Möglichkeiten einer Smartcard liegen. Auch hier zeigt sich wieder, dass die Erhöhung des Sicherheitsstandards, der durch eine Chipkarte erreichbar ist, direkt wieder zunichte gemacht wird, falls das schwächste Glied in der Kette ein unsicheres Netz oder ein schlecht konfiguriertes System ist.

Smartcardbasierte Authentifikation

Eine Smartcard kann sehr gut als persönliches Sicherheitswerkzeug zur Authentifikation von Benutzern eingesetzt werden. Wir unterscheiden drei Stufen der Authentifizierung. Im ersten Schritt authentifiziert sich der Benutzer gegenüber der Smartcard und erst im zweiten Schritt erfolgt die Authentifikation mit dem Zielsystem wie dem Kartenlesegerät oder einem PC. Der dritte Schritt umfasst die Authentifikation des Zielsystems gegenüber der Karte.

Schritt 1: Benutzeroauthentifikation

Der Benutzer authentifiziert sich gegenüber der Karte in der Regel unter Nutzung einer vier- oder fünfstelligen PIN (Personal Identification Number). Im Umfeld der Smartcards wird eine PIN auch häufig als CHV, Card Holder Verification, bezeichnet. Die PIN gibt er am Kartenlesegerät oder direkt auf der Karte ein, falls diese über eine eigene Tastatur verfügt. Wird die PIN über ein Kartenlesegerät an die Karte transferiert, so sendet das Lesegerät gleichzeitig das Kommando Verify CHV, um die Karte zur Authentifikation zur Veranlassen. Die eingegebene PIN wird vom Mikroprozessor der Karte unter Nutzung seines RAMs mit derjenigen PIN verglichen, die im EEPROM steht (vgl. Abbildung 11.3). Sollte die PIN in verschlüsselter Form im EEPROM abgelegt sein, so wird zunächst von der CPU der kartenspezifische Schlüssel aus dem EEPROM gelesen, die eingegebene PIN mit dem im ROM festgelegten Verfahren unter Nutzung des RAMs verschlüsselt und erst dann der Abgleich durchgeführt.

PIN-Prüfung

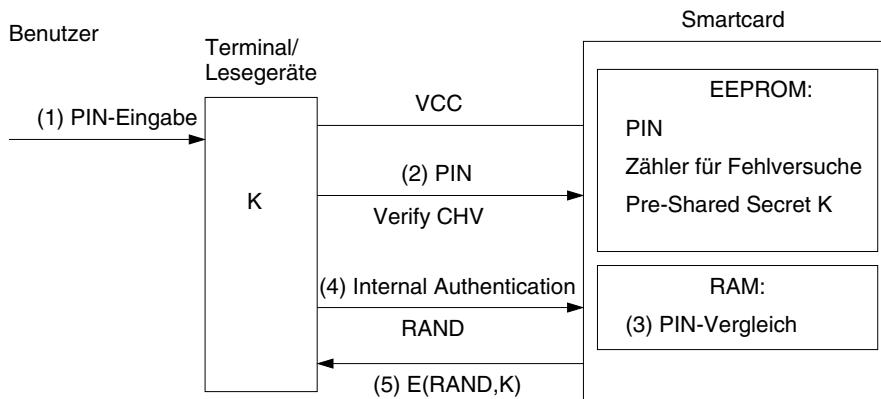


Abbildung 11.3: Authentifikation zwischen Nutzer, Kartenterminal und Karte

Der Chip führt daneben auch noch einen Zähler über die Anzahl der Fehlversuche. Falls eine kartenspezifisch festgelegte Maximalzahl erreicht ist, üblicherweise ist dies nach drei Fehlversuchen der Fall, wird die Karte

blockiert. Manche Karten ermöglichen das Entsperren durch die Eingabe einer speziellen PIN, des PUK (PIN Unblocking Key). Für dessen Eingabe wird ebenfalls ein Zähler geführt. Mehrmalige Fehlversuche führen zu einer endgültigen Sperrung der Karte. Im nächsten Schritt authentifiziert sich nun die Karte gegenüber dem Lesegerät. Dazu besitzen beide ein Pre-Shared Secret K . Das Terminal sendet der Karte eine Zufallszahl RAND und das Kommando, eine entsprechende Response zu berechnen. Die Karte antwortet mit der verschlüsselten Response, wobei zur Verschlüsselung die Kenntnis des Pre-shared Secrets nachgewiesen wird.

Biometrie

Alternativ zur PIN-Authentifikation werden in zunehmendem Maß auch biometrische Informationen zum Identitätsnachweis verwendet. Da biometrische Techniken (siehe Abschnitt 10.3) spezielle Hardwarekomponenten wie beispielsweise Fingerabdruck-Scanner zur Erfassung der biometrischen Merkmale erfordern, enthalten die Smartcards häufig nur die Referenzwerte der zu identifizierenden Person. Der Abgleich zwischen Referenzdaten und aktuellen Daten erfolgt im Gegensatz zur PIN-Überprüfung häufig nicht in der Smartcard, sondern durch das Biometriegerät. Dies kann für die Sicherheit gravierende Konsequenzen haben. Musste man nämlich bei der PIN-Technik „nur“ darauf vertrauen, dass die Chipkarte zuverlässig und vor Angriffen geschützt (engl. *tamper resistant*) ihre Funktionen erfüllt, so fordert man diese Vertrauenswürdigkeit nun zusätzlich auch von den verwendeten biometrischen Geräten. Auf Risiken im Zusammenhang mit biometrischen Authentifikationstechniken gehen wir in Abschnitt 10.3 noch näher ein.

Schritt 2: Kartauthentifikation

Challenge-Response

Im zweiten Schritt erfolgt die Authentifikation zwischen der Smartcard und dem Zielsystem. Dabei werden Challenge-Response-Protokolle verwendet. Challenge-Response-Protokolle wurden bereits in Abschnitt 10.2.3 ausführlich erklärt. Soll sich auch das Zielsystem gegenüber der Karte authentifizieren, so wird dazu wiederum ein Challenge-Response-Protokoll abgewickelt, wobei diesmal die Frage (Challenge) durch die Chipkarte gestellt wird.

11.1.3 Smartcard-Sicherheit

Zu den sicherheitsrelevanten Funktionsbereichen einer Chipkarte gehören neben den Bereichen, die direkt vom menschlichen Benutzer lesbare Daten enthalten, die Sicherheitseigenschaften des Chips und seines Betriebssystems sowie Sicherheitseigenschaften des Übertragungsmediums. Direkt vom Menschen lesbare Informationen dienen nicht zum Schutz der gespeicherten Daten, sondern zur Verhinderung einer Kartenfälschung oder ihrer

unautorisierten Nutzung. Beispiele für derartige Daten sind das Foto des Karteninhabers, seine handschriftliche Unterschrift, Hologramme oder Prägungen bzw. Hervorhebungen auf der Karte, wie das Verfallsdatum.

Secure by Design

Die Komponenten auf einem Chip werden durch unterschiedliche Maßnahmen geschützt. Je weniger Informationen über den zu analysierenden Chip verfügbar sind, um so schwieriger ist es, gezielte Angriffe durchzuführen. Beim Chip-Design werden deshalb häufig Verschleierungstechniken angewandt und/oder durch ein manuelles Layout wird versucht, regelmäßige Strukturen zu vermeiden. Diese Maßnahmen zählen zu der eigentlich ablehnenden Technik des Schutzes „security through obscurity“. Da sie hier aber nur begleitenden Schutzcharakter haben und die Sicherheit nicht allein auf der Verschleierung beruht, sind sie als sinnvolle Ergänzungen zu sehen. Die internen Busse des Microcontrollers sind nicht nach außen geführt und damit nicht kontaktierbar, so dass ein Angreifer die Adress-, Daten- und Steuerbusse weder abhören noch beeinflussen kann. Zusätzlich sind die Busse noch individuell gecrambelt, so dass von außen die Funktion der Busse nicht erkennbar ist.

Chip-Design

Eine Modifikation des ROMs, zum Beispiel mittels Laser Cutter, ist nur dann erfolgreich durchführbar, wenn der ROM in den obersten Ebenen des Chips realisiert ist. Deshalb wird er meist in tiefere Siliziumschichten verlagert. Gleichzeitig wird damit auch ein Reverse Engineering des Betriebssystems erschwert. Ionenimplantierte ROM-Codes verhindern das Auslesen des ROMs mittels Laser-Mikroskop, da die Dateninhalte weder im sichtbaren noch im Infrarot oder UV-Licht sichtbar sind, und machen eine unautorisierte Modifikation praktisch unmöglich.

ROM

Die Wahrscheinlichkeit, mit vertretbarem Aufwand einen Ein-Bitfehler zu erzeugen, wird als sehr gering angesehen, da der statische RAM heutiger Mikrocontroller besonders unempfindlich gegen Störungen ist. Fehler, die durch ionisierende Strahlung verursacht werden, betreffen eher die Software der Chipkarte. Sie führen zu einem Systemabsturz oder zu fehlerhaften Antworten, aus denen der Angreifer zwar keine sensitive Information ableiten, aber immerhin noch einen Denial-of-Service-Effekt erzielen kann.

RAM

Um zu verhindern, dass elektrische Abstrahlung von außen aufgezeichnet werden kann, wird der EEPROM-Bereich des Chips speziell abgeschirmt. Eine Entfernung des Schutzschildes führt zu einer Zerstörung des gesamten Chips. Eine solche Schutzschicht kann beispielsweise aus einer spannungsführenden Metallierungsschicht bestehen, deren Entfernung auf chemischem Weg dazu führt, dass die Spannungszuführung über diese Schicht fehlt und der Chip funktionsunfähig wird. Durch eine weitere Schutzschicht wird

EEPROM

verhindert, dass die Speicherinhalte durch UV-Strahlung gelöscht werden können. Ferner verfügen Smartcards über spezielle Sensoren, über die ein Eingriffsversuch von außen erkennbar ist, indem zum Beispiel eine zu hohe oder zu niedrige Spannung oder eine unzulässige Temperatur gemessen wird. Das heißt, im Siliziumkristall befinden sich Sensoren, die auf Licht reagieren, Wärme-Sensoren gegen eine Übertaktung des Chips, Sensoren zur Messung von Widerständen und Kapazitäten, um zu erkennen, ob die Schutzschicht über der Schaltung noch vorhanden ist. Da die Taktversorgung (CLK) der Karte von außen erfolgt, ist sie theoretisch Angriffen ausgesetzt, die versuchen, die Rechengeschwindigkeit des Controllers von außen zu steuern und einen Einzelschrittbetrieb herbeizuführen. Manche Smartcards verschlüsseln die Kommunikation zwischen den On-Chip-Komponenten, so dass auch passive Angriffe auf den lokalen Datenbus abgewehrt werden. Häufig setzt man noch zusätzlich Prüfsummen oder Signaturen ein, um die geheimen Schlüssel, die meist im EEPROM gespeichert sind, zu schützen.

Angriffe auf Smartcards

Angriffe

Die meisten direkten Angriffe auf Smartcards zielen auf das Ausspähen der geheimen Schlüssel auf der Karte ab. So geben Boneh, DeMillo und Lipton in [30] ein Kryptoanalyseverfahren an, das von der Annahme ausgeht, dass durch physikalische Einwirkungen kurzzeitige Hardware-Effekte wie die Invertierung eines Register-Bits hervorgerufen werden, so dass dadurch gespeicherte geheime Schlüssel bestimmbar sind.

gezielte Beeinflussung

Weitere Angriffsszenarien auf Smartcards und auf Sicherheitsprozessoren werden von Anderson und Kuhn beschrieben (vgl. [178]). Durch das Anlegen einer unüblichen Spannung oder durch zu hohe bzw. zu niedrige Temperaturen wird dort versucht, den Mikrocontroller in seiner Funktionsweise zu beeinflussen. Mit gezielt eingesetzten, kurzzeitigen Störungen der Spannungs- und Taktversorgung wird ferner versucht, Änderungen im Code vorzunehmen und die im Mikrocontroller gespeicherten geheimen Schlüssel zu ermitteln.

Abhören

Weitere Angriffsbeschreibungen betreffen die Herauslösung des Chips aus seiner Plastikumhüllung, so dass es mit Hilfe von Mikroprobennadeln im laufenden Betrieb möglich wird, Signale auf dem chiplokalen Datenbus abzugreifen. Um Leiterbahnen und anliegende Spannungswerte auszuspähen, könnten Elektronenmikroskope eingesetzt werden. Für weitreichende Angriffe ist jedoch eine spezifische Ausrüstung erforderlich, wie sie zwar in professionellen Halbleiterlabors verwendet wird, aber dem „Gelegenheitsangreifer“ sicherlich nicht und selbst einem „ernsthaften“ Angreifer nicht unbedingt zur Verfügung steht.

Angriffe, die den Übertragungsweg zwischen Kartenlesegerät und Zielsystem betreffen, werden hier nicht weiter aufgeführt, da sie sich nicht von den aktiven und passiven Angriffen wie Sniffing und Spoofing unterscheiden, die wir schon in herkömmlichen vernetzten Umgebungen kennen gelernt haben. Zur Abwehr derartiger Angriffe sind auch hier die bereits bekannten Mechanismen wie Verschlüsselung, MAC-Berechnung, Signieren und Verifizieren sowie eine wechselseitige Authentifikation einzusetzen.

Übertragungsweg

Seitenkanalanalysen

In [103] wurden mit der einfachen Leistungsanalyse (simple power analysis SPA) und der differenziellen Leistungsanalyse (differential power analysis DPA) Angriffe auf Smartcards publiziert, die darauf abzielen, den Stromverbrauch bei der Ausführung von Kommandos auf der Karte zu messen. Bei der differenziellen Leistungsanalyse (DPA) können gegenüber der einfachen Leistungsanalyse (SPA) zusätzlich noch geringere Unterschiede beim Stromverbrauch des Mikrocontrollers bei der Verarbeitung der Daten erkannt werden. Eine entsprechende Analyse kann man so durchführen, dass man den Stromverbrauch zunächst bei der Verarbeitung bekannter und dann bei der Verarbeitung unbekannter Daten misst und aus den Messergebnissen Mittelwerte berechnet. Diese Art von Angriffen kann auf Smartcards erfolgreich angewandt werden, da häufig Abhängigkeiten des Stromverbrauchs vom jeweiligen Maschinenbefehl und den verarbeiteten Daten bestehen. Zur wirkungsvollen Abwehr derartiger Angriffe kann beispielsweise ein Spannungsregler auf dem Chip eingebaut werden, der über einen Widerstand einen von Maschinenbefehl und Daten unabhängigen Stromverbrauch sicherstellt. Auf der Software-Seite kann man derartige Angriffe ebenfalls wirksam abwehren, indem sichergestellt wird, dass ausschließlich Maschinenbefehle benutzt werden, die einen sehr ähnlichen Stromverbrauch besitzen.

Differenzial Power Analysis

Konkrete Angriffe auf der Basis der Stromanalyse ergeben sich zum Beispiel bei PIN-Vergleichen. Eine Strommessung kann dazu beispielsweise über den Spannungsabfall an einem in die Vcc-Leitung (vgl. Abbildung 11.1) eingebrachten Widerstand erfolgen. Werden nun das Kommando und die Vergleichsdaten zur Karte gesendet, so lässt sich über die Strommessung bereits vor Erhalt des Returncodes, der von der Karte berechnet wird, feststellen, ob der Fehlbedienungszähler erhöht werden wird oder nicht. Würde bei einem positiven Vergleich, d.h. bei korrekter PIN-Eingabe, der Returncode früher zurückgesandt, als der Fehlbedienungszähler beschrieben wird, so könnte ein Angreifer dafür sorgen, dass bei einem negativen Vergleichsergebnis der Controller abgeschaltet wird bevor der Fehlbedienungszähler erhöht wird. Damit wäre ein Brute-Force Angriff durchführbar, indem der Vergleichswert in allen Varianten zur Chipkarte gesendet wird

PIN-Vergleich

und man diese immer dann, falls keine Übereinstimmung erkannt wird, vor dem Erhöhen des Fehlbedienungszählers abschaltet. Der positive Fall kann durch den entsprechenden Returncode, der vor der Erhöhung des Fehlbedienungszählers gesendet wird, eindeutig erkannt werden. Derartige Angriffe lassen sich wirksam abwehren, indem man bei der Implementierung der Vergleichsroutinen dafür sorgt, dass der Fehlbedienungszähler stets vor jedem Vergleich erhöht und bei Bedarf wieder erniedrigt wird. Seitenkanalanalysen sind Angriffe, die nicht nur für Smartcard-Architekturen relevant sind, sondern auch gegen beispielsweise Caches in herkömmlichen Architekturen gerichtet sein können, wie die Meltdown und Spectre-Probleme (siehe auch Abschnitt 2.8.1) Anfang 2018 eindrucksvoll gezeigt haben.

Fazit

Der Einsatz von Smartcards als Sicherheitswerkzeuge stellt einen wesentlichen Beitrag zur Erhöhung der Systemsicherheit dar. Das immer günstiger werdende Preis/Leistungsverhältnis, sowohl der Karten selber, als auch der zu ihrer Betreibung erforderlichen Lesegeräte und Terminals, macht diese Technologie nun auch dem Massenmarkt zugänglich. Mit der Entwicklung der JavaCard wurde ein weiterer wichtiger Schritt in Richtung auf einen breiten Einsatz von Smartcards gemacht. Mit der JavaCard ist es möglich, Java Programme (Applets) auf eine solche Karte zu laden und auf dieser auszuführen. Das Betriebssystem der JavaCard ist eine eingeschränkte Version der Java Virtual Machine, das es ermöglicht, JavaCard Klassen dynamisch in den EEPROM zu laden. Solche smartcardspezifischen Java-Klassen sind beispielsweise Funktionen zur EEPROM Verwaltung, die T=0 und T=1 Übertragungsprotokolle oder Schutzfunktionen für Daten.

JavaCard

Ausblick

Mit der zu erwartenden Verbesserung der Prozessorleistung und einer Vergrößerung der EEPROM-, ROM- und RAM-Speicher bzw. der Ausstattung von Smartcards mit zusätzlichen Hardwarekomponenten, zusammen mit einer Ablösung der PIN-Überprüfung durch biometrische Techniken, eröffnet die Smartcard äußerst interessante Zukunftsperspektiven zur Steigerung der Systemsicherheit von IT-Systemen. Smartcards sind wichtige Bausteine einer Sicherheitskette, deren Sicherheit jedoch, wie schon mehrfach betont, von der Sicherheit ihres schwächsten Glieds abhängt. Nur durch einen systematischen Einsatz der verschiedenen sicherheitssteigernden Maßnahmen, die wir in diesem Buch behandeln, ist eine wirkliche Qualitätssteigerung erreichbar.

11.2 Elektronische Identifikationsausweise

Im März 2005 beschloss das Bundeskabinett die Eckpunkte für eine eCard-Strategie der Bundesregierung. Ziel der Strategie ist es, Chipkarten flächendeckend für die verschiedenen Bereiche der Bundesverwaltung einzuführen, um eine elektronische Authentisierung sowie die Verwendung qualifizierter elektronischer Signaturen zu unterstützen. Wichtige Projekte der Bundesregierung zur Durchsetzung der Strategie sind der elektronische Personalausweis (nPA) sowie der elektronische Reisepass (ePass). Mit den neuen Ausweis- und Reisedokumenten, dem elektronischen Reisepass und dem elektronischen Personalausweis wurde in Deutschland flächendeckend eine Infrastruktur eingeführt, die jeden Bundesbürger mit einer elektronisch prüfbaren Identität versorgt. Während der Reisepass der hoheitlichen Personenkontrolle (Grenzkontrolle) vorbehalten ist, bietet der elektronische Personalausweis neben dieser hoheitlichen Aufgabe auch eine Authentisierungsfunktion an, die eID, die es erlaubt, den Personalausweis in digitalen (Geschäfts)-Prozessen zur Identifizierung zu verwenden.

Im Folgenden werden die wichtigsten Sicherheitsmechanismen dieser elektronischen Reise- und Ausweisdokumente vorgestellt. Abschnitt 11.2.1 diskutiert die Eigenschaften des elektronischen Reisepasses, dem ePass, der bereits seit 2005 in Deutschland im Einsatz ist. In Abschnitt 11.2.2 stellen wir die Eigenschaften des elektronischen Personalausweises vor, der seit dem 1. November 2010 ausgegeben wird und den alten Personalausweis ersetzt hat.

11.2.1 Elektronischer Reisepass (ePass)

Ausgangspunkt der Entwicklung elektronischer Reisedokumente war im Jahr 2004 der Beschluss des Rats der Europäischen Union, die Pässe der Bürgerinnen und Bürger der Mitgliedsstaaten mit maschinenlesbaren Personenkennzeichen, zu denen auch biometrischen Daten des Inhabers gehören, auszustatten. Diese elektronischen Reisedokumente sollten gemäß dem Standard, der von der ICAO² spezifiziert wurde, entwickelt werden. In Deutschland wurde diese EU-Richtlinie mit dem Beschluss des Bundeskabinetts vom 22.6.2005 umgesetzt, in dem die Einführung des elektronischen Reisepasses zum 1.11.2005 beschlossen wurde.

Mit dem Pass werden zwei wesentliche Ziele verfolgt, nämlich

Schutzziele

1. die Fälschungssicherheit der Daten auf dem Pass zu erhöhen und
2. den Missbrauch von Pässen zu erschweren, d.h. zu erschweren, dass echte Pässe von ähnlich aussehenden Personen (engl. *look-alike fraud*) missbraucht werden.

² International Civil Aviation Organisation

Zur Erreichung dieser Ziele enthält der ePass verschiedene Sicherheitsmerkmale. Wesentliches Merkmal des neuen Reisepasses ist ein RFID³-Chip, der in den Pass integriert ist (vgl. Abbildung 11.4).

RFID-Chip

RFID

Der RFID-Chip im elektronischen Reisepasses ist ein kontaktloser, passiver Chip. Es handelt sich hierbei um einen ISO/IEC 14443-konformen Chip⁴ mit kryptographischem Koprozessor. In deutschen Reisepässen werden Chips eingesetzt, die gemäß der Common Criteria (CC) in der Stufe EAL5 zertifiziert sind.

Ein RFID-Chip verfügt über einen Microcontroller zur Speicherung und Verarbeitung von Daten und eine Antenne zur drahtlosen Kommunikation über geringe Distanzen. Der RFID-Chip im ePass ist passiv, das heißt, er verfügt nicht über eine eigene Stromversorgung sondern wird durch ein Lesegerät mit Strom versorgt, in dem es ein hochfrequentes elektromagnetisches Wechselfeld erzeugt. Befindet sich die Antenne des Transponders in diesem Feld, so entsteht in der Antennenspule ein Induktionsstrom. Der RFID-Chip ermöglicht es, die personenbezogenen Daten des Passbesitzers zu speichern. Analog zu einer Smartcard kann ein RFID-Chip mit dem Lesegerät über Protokolle kommunizieren und insbesondere Daten auch verschlüsselt vom Chip zum Lesegerät übertragen. Darauf wird weiter unten noch genauer eingegangen.

Reichweite

Die Signalisierungsreichweite der ISO-14443-konformen, kontaktlosen Chips ist auf ca. 10 cm begrenzt. Damit und durch die integrierten Sicherheitsfunktionen unterscheidet sich der im ePass zum Einsatz kommende RFID-Chip erheblich von RFID-Chips gemäß ISO 15693 bzw. ISO 1800-6, die insbesondere in der Logistik verwendet werden. Diese RFID-Chips besitzen eine deutliche größere Signalisierungsreichweite, die zwischen einem und sieben Metern liegt.

Zugriffskontrolle

Der Zugriff auf die Daten im Chip ist zumindest über eine einfache Zugriffskontrolle, der Basic Access Control (BAC) und optional auch über eine erweiterte Zugriffskontrolle, der Extended Access Control (EAC), abgesichert. PACE (vgl. auch Seite 540 ff) ist eine Entwicklung des BSI, das als Alternative zum BAC Protokoll einsetzbar ist (vgl. [33]). Auf die Zugriffskontrollmechanismen wird weiter unten noch genauer eingegangen.

Biometrische Daten

Biometrische Daten

Das Sicherheitsziel der engeren Bindung des Passes an seinen rechtmäßigen

³ Radio Frequency Identification

⁴ ISO/IEC14443: Identification cards: Contactless Integrated Circuit Cards, Proximity cards

Besitzer soll durch biometrische Daten erreicht werden, die im Chip abgelegt werden, während zur Erhöhung der Fälschungssicherheit der auf dem Chip gespeicherten Daten klassische Mechanismen, wie Digitale Signatur, Hashfunktionen und verschiedene, spezifisch auf die Einsatzszenarien elektronischer Reisedokumente zugeschnittene Authentisierungsprotokolle zum Einsatz kommen.

Abbildung 11.4 zeigt den deutschen elektronischen Reisepass mit integriertem RFID-Chip. Deutsche Reisepässe werden von der Bundesdruckerei herausgegeben.

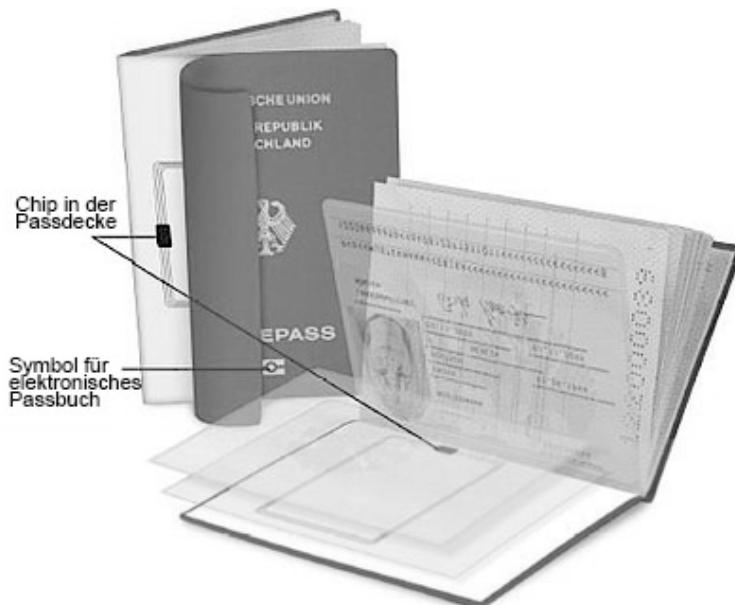


Abbildung 11.4: Der ePass mit integriertem RFID-Chip (Quelle: Bundesministerium des Inneren)

Daten auf dem ePass

Der elektronische Reisepass enthält personen- und dokumentenbezogene Daten, die teilweise auf dem Reisepass aufgedruckt, in der maschinenlesbaren Zone (MRZ⁵) codiert bzw. im RFID-Chip des Reisepasses abgespeichert sind. Die Daten zur Person umfassen Vorname, (Familien-)Name, Wohnort, Geburtsort, Geschlecht, Geburtsdatum, Körperhöhe, Augenfarbe, Passfoto, Unterschriftenprobe und zwei Fingerabdrücke. Die Daten, die zum Dokument gehören, sind die 9-stellige Seriennummer (Passnummer), der

Daten

⁵ Machine Readable Zone

ausstellende Staat, der Dokumententyp und das Gültigkeitsdatum des Passes. Deutsche Reisepässe haben eine Gültigkeit von 10 Jahren. Die 9-stellige Passnummer besteht aus zwei Teilnummern: die vierstellige Behördennummer und die 5-stellige PassID.

Die Daten im Chip werden entsprechend der ICAO-Spezifikation für maschinenlesbare Reisedokumente (vgl. [84, 85]) in 16 Datengruppen und einem sogenannten Document Security Object aufgeteilt (vgl. Tabelle 11.2).

Datengruppe	obligatorisch / optional	Inhalt	Zugriffskontrolle
DG1	obligatorisch	Daten aus der MRZ	BAC/PACE
DG2	obligatorisch	Gesichtsbild	BAC/PACE
DG3	optional	Fingerabdrücke	BAC/PACE + EAC
DG4	optional	Iris	BAC/PACE + EAC
...	optional		BAC/PACE
DG14	optional	Public Key für Chip Authentication	BAC/PACE
DG15	optional	Public Key für Active Authentication	BAC/PACE
DG16	optional	Im Notfall zu benachrichtigende Person	BAC/PACE
Document Security Object	obligatorisch	Signierte Hashwerte aller Datengruppen	BAC/PACE

Tabelle 11.2: Datenstruktur des ePass

MRZ

Datengruppe 1 enthält die Daten, die auch in der maschinenlesbaren Zone codiert abgelegt sind. Es handelt sich hierbei um den Namen, das Geschlecht und das Geburtsdatum des Passinhabers. Da diese Daten auch in der maschinenlesbaren Zone des Passes abgelegt sind, können sie optisch ausgelesen werden. Zum Auslesen der Daten aus der MRZ ist es erforderlich, dass der Pass auf ein optisches Lesegerät gelegt wird (vgl. Abbildung 11.5), während zum Auslesen der Daten aus der Datengruppe 1 des Chips eine Funkverbindung zum Chip aufgebaut werden muss.

Biometrische Daten

Die Datengruppen 2, 3 und 4 sind für die Speicherung biometrischer Merkmale vorgesehen. Datengruppe 2 enthält dabei das digitale Gesichtsbild (Foto) des Passinhabers, Datengruppe 3 die digitalen Fingerabdrücke, wäh-



Abbildung 11.5: ePass und Lesegerät für die optisch auslesbare MRZ (Quelle: BSI)

rend die Datengruppe 4 laut Spezifikation für das biometrische Merkmal Iris vorgesehen ist, was jedoch in den deutschen Pässen nicht verwendet wird. Die Datengruppen 3 und 4 sind optional. In Deutschland wird seit November 2007 auch der digitale Fingerabdruck (beide Zeigefinger) auf dem Chip gespeichert.

Weitere wichtige Datengruppen sind die Datengruppen 14 und 15, die öffentliche Schlüssel enthalten, die für die aktive Authentifizierung des RFID-Chips gegenüber einem Lesegerät verwendet werden. Zur Prüfung der Unverfälschtheit der auf dem Chip abgelegten Daten dient das Document Security Object (DSO). Es enthält die Hashwerte aller auf dem Chip gespeicherten Datengruppen. Die Hashwerte sind von der passausstellenden Instanz, das ist in Deutschland die Bundesdruckerei, digital signiert, so dass bei einer hoheitlichen Personenkontrolle die Authentizität der Passdaten überprüft werden kann⁶. Die Daten auf dem Chip sind alle nicht wiederholt beschreibbar, so dass sich die Hashwerte im Document Security Object über die Lebenszeit des Reisepasses nicht ändern.

DSO

Sicherheitsmechanismen im ePass

Der ePass sieht im Zusammenspiel mit einem Lesegerät unterschiedliche Sicherheitskontrollen vor.

Sicherheits-mechanismen

- Mittels der *passiven Authentifizierung* kann ein Lesegerät die Integrität und Authentizität der Passdaten prüfen. Dazu berechnet er den Hashwert über die aktuell ausgelesenen Daten und vergleicht diesen mit dem Wert im Document Security Object. Damit soll die Fälschungssicherheit der

⁶ Anmerkung: Diese Absicherung ist bei den elektronischen Personalausweisen nicht vorgesehen.

Passdaten im Vergleich zum früheren, nicht-elektronischen Pass erhöht werden.

- Die *aktive Authentifizierung*, bei der sich der Chip im Pass mittels eines asymmetrischen Challenge-Response-Protokolls gegenüber dem Lesegerät authentifiziert, dient darüber hinaus zur Erkennung geklonter Pässe. Der hierbei vom RFID-Chip zu verwendende Chip-private Schlüssel darf nicht vom Lesegerät ausgelesen werden, sondern er wird nur chipintern verwendet und fließt bei der Berechnung der Response ein.
- Der Zugriff auf die Daten im Chip ist mittels *Zugriffskontrollmechanismen* geschützt. Sie umfassen verpflichtend eine einfache Zugriffskontrolle, die Basic Access Control (BAC). Diese wurde in deutschen elektronischen Personalausweisen durch eine verbesserte, passwortbasierte Kontrolle, das PACE-Verfahren, ersetzt.
- Für den Zugriff auf sensitive biometrische Daten, wie den Fingerabdruck, wird eine erweiterte Kontrolle, die Extended Access Control (EAC) gefordert. Sie fordert die Authentifikation des zugreifenden Lesegeräts mittels eines gültigen Zertifikats. Dieses Protokoll wird als *Terminal Authentifizierung* bezeichnet.
Die Mitgliedsländer der EU haben sich verpflichtet, die erweiterte Zugriffskontrolle in ihren Lesegeräten zu verwenden, so dass der Zugriff auf alle Passdaten dieser verbesserten Kontrolle unterliegen wird.

Tabelle 11.3 fasst die Sicherheitsmechanismen und Protokolle, die auf dem ePass umgesetzt sind, sowie die damit verfolgten Schutzziele zusammen. Die Sicherheitsmechanismen werden nachfolgend erläutert.

Passive Authentifizierung

Bei der passiven Authentifizierung prüft ein Lesegerät die Passdaten auf Integrität und Authentizität des Datenursprungs. Die Prüfung der Passdaten durch ein Lesegerät erfolgt 2-stufig und ist verbunden mit Maßnahmen zur Zugriffskontrolle, der sogenannten Basic Access Control (BAC), auf die Abschnitt 11.2.1 noch genauer eingehht.

Lesegeräte

Bei der passiven Authentifizierung muss sich das Lesegerät nicht gegenüber dem Pass (genauer dem RFID-Chip im Pass) authentisieren. Es hat lediglich nachzuweisen, dass es die Daten aus der MRZ des Passes kennt. In der Regel erfordert dies einen optischen Zugriff auf die MRZ, aber natürlich können diese Daten auch durch einen Angreifer anderweitig beschafft werden. Die passive Authentifizierung verhindert somit nicht, dass unautorisierte Lesegeräte versuchen können, die Daten aus dem Chip des Reisepasses auszulesen. Auch wenn die geringe Signalisierungsreichweite des Chips das

Schutzziel(e)	Mechanismen	Protokolle
Integrität	Hashfunktion, Signatur	Passive Authentifikation
Authentizität der Passdaten	digitale Signatur, Zertifikat, länderübergreifende PKI	Passive und Aktive Authentifikation
Vertraulichkeit der Daten	physischer Zugriff (MRZ-Daten), Verschlüsselung	Zugriffskontrolle (BAC/PACE, EAC), Secure Messaging
Missbrauch reduzieren	Biometrische Merkmale (Foto, Fingerabdruck), Chip-Schlüssel (Signatur)	Passive Authentifikation, Terminal Authentifikation

Tabelle 11.3: Sicherheitsmechanismen und -Protokolle des ePass

einfache Auslesen erschwert. Zur Abwehr derartiger Angriffe müsste das zugreifende Lesegerät nachwesen, dass es authentisch ist. Zudem müsste es einen signierten Nachweis vorlegen, auf welche Daten es zugreifen darf. Dies wird durch das Protokoll der *Terminal-Authentifizierung* erreicht, auf das wir auf Seite 527 eingehen.

Passive Authentifikation zusammen mit der einfachen Zugriffskontrolle für den Datenzugriff (BAC) orientiert sich am herkömmlichen Schutzniveau für Daten in Reisepässen. Das heißt, zum Auslesen der Daten muss der herkömmliche Reisepass (willentlich) dem Passbeamten übergeben werden. Analoges gilt auch für den ePass, der in ein optisches Lesegerät eingelegt werden muss, damit dieses die Daten aus der MRZ des Passes auslesen kann. Aus den optisch gelesenen Daten der MRZ berechnet das Lesegerät einen Zugriffsschlüssel, der als Basis zum Aufbau eines sicheren Kommunikationskanals zwischen Lesegerät und RFID-Chip des Passes genutzt wird. Damit das Lesegerät die Daten der MRZ auf Integrität prüfen kann, benötigt es das Data Security Objekt (DSO), das auf dem RFID-Chip abgelegt ist. Diese Daten werden mittels Secure Messaging unter Verwendung eines gemeinsamen 112-Bit Sitzungsschlüssel vertraulich vom Pass zum Lesegerät übermittelt. Der gemeinsame Sitzungsschlüssel wird bei der Ausführung des BAC-Protokolls sowohl im Chip des Passes als auch im Lesegerät berechnet, indem Teilschlüssel zwischen dem Chip und dem Lesegerät mittels eines Challenge-Response-Protokolls vertraulich ausgetauscht werden. Bei deutschen Reisepässen wird als Verschlüsselungsverfahren für das Secure Messaging 3DES im CBC Modus (vgl. Kapitel 7.5.4) verwendet.

Zur Integritätsprüfung berechnet das Lesegerät die Hashwerte der ausgelese-

Integritätsprüfung

Secure Messaging

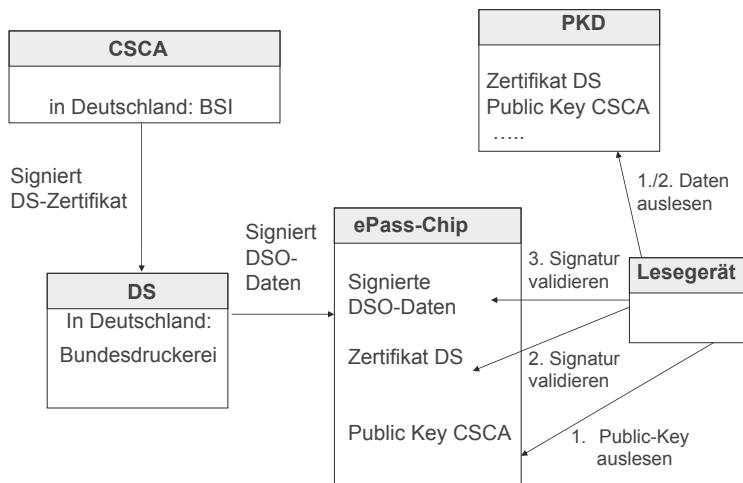


Abbildung 11.6: Prüfen der Authentizität der Passdaten durch ein Lesegerät

nen Daten aus den Datengruppen auf den RFID-Chip und vergleicht sie mit den in dem Data Security Objekt abgelegten Hashwerten.

Bei der passiven Authentifikation kann das Lesegerät auch die Integrität der Gesichtsbilddaten (Datengruppe 2) prüfen, während der Zugriff auf die Fingerabdrücke einer höheren Sicherheitsanforderung unterliegt. Hierfür ist obligatorisch die Extended Access Control (EAC) durchzuführen, die vom Lesegerät einen Nachweis erfordert, dass es zum Zugriff tatsächlich berechtigt ist.

Authentizitätsprüfung

Um die Authentizität der im Pass gespeicherten Daten überprüfen zu können, muss das Lesegerät die Signatur des vom Pass erhaltenen Data Security Objekts prüfen. Abbildung 11.6 veranschaulicht die dafür benötigten Komponenten.

Die Daten im Data Security Object werden bei der Passerstellung vom Passaussteller signiert, so dass das Lesegerät das Zertifikat dieses Passausstellers benötigt, um die Signatur zu prüfen. In Deutschland ist diese Instanz die Bundesdruckerei. In der ICAO-Spezifikation für maschinenlesbare Reisedokumente ([84, 85]) wird eine solche Instanz ein Document Signer (DS) genannt. Das Zertifikat eines Document Signers wird entweder auf dem Pass selbst abgelegt oder über das Public-Key Directory (PKD) der Standardisierungsorganisation ICAO veröffentlicht, so dass sich ein Lesegerät das erforderliche Zertifikat von dieser Stelle besorgen kann.

Das Zertifikat des Document Signers ist seinerseits digital signiert. Zur vollständigen Zertifikatsvalidierung benötigt das Lesegerät somit noch den öffentlichen Schlüssel derjenigen Instanz, die das Zertifikat des Document

Signers signiert hat. Laut ICAO-Spezifikation muss es pro Land genau eine solche Instanz, die sogenannte Country Signing Certification Authority (CSCA) geben. In Deutschland ist dies das BSI. Der öffentliche Schlüssel der CSCA wird auf dem RFID-Chip des Reisepasses abgelegt und entweder im Public-Key Directory der ICAO veröffentlicht oder über diplomatische Wege den CSCA-Instanzen anderen Ländern bekannt gemacht. Der öffentliche Schlüssel der CSCA bildet den Vertrauensanker für die Zertifikatsvalidierung.

Beim Prüfen der Signatur des Document Security Objects ist somit eine 2-stufige Vertrauenskette durch ein Lesegerät zu validieren:

1. Validieren der Signatur der CSCA (also in Deutschland des BSI). Dazu sind die Schritte (1) und (2) aus Abbildung 11.6 auszuführen.
2. Validieren der Signatur des DS (also in Deutschland der Bundesdruckerei). Dazu ist der Schritt (3) aus der Abbildung auszuführen.

Da ein deutscher Reisepass eine Geltungsdauer von 10 Jahren besitzt, müssen die Signaturen eine entsprechend lange Gültigkeit besitzen. Dies erfordert lange Signaturschlüssel und eine lange Gültigkeit von Signatur-Verifikationsschlüsseln. In Deutschland wird für das Signieren der Passdaten ein 224 Bit ECDSA-Schlüssel (Elliptic Curve DSA) des Passausstellers (Bundesdruckerei) verwendet. Während also der öffentliche Verifikations-schlüssel mindestens 10 Jahre gültig sein muss, um Signaturen prüfen zu können, besitzt ein privater Signierschlüssel nur eine kurze Gültigkeit von maximal drei Monaten. Dadurch wird sicher gestellt, dass derselbe private Schlüssel nur während je drei Monaten zur Zertifikatserstellung verwendet und anschließend durch einen neuen ersetzt wird. Damit will man erreichen, dass im Fall einer Schlüsselkompromittierung möglichst wenige Pässe betroffen sind. Wie bereits weiter oben erwähnt, werden die deutschen Reisepässe von der Bundesdruckerei ausgestellt und damit auch signiert. Der zugehörige öffentliche Verifikationsschlüssel der Bundesdruckerei wird über ein Zertifikat verfügbar gemacht. Die Zertifikate der Bundesdruckerei sind vom BSI als der deutschen CSCA-Instanz signiert. Hierfür wird ein 256-Bit ECDSA-Schlüssel verwendet. Der private Schlüssel der CSCA muss zwischen 3 und 5 Jahren gültig sein, während für den öffentlichen Schlüssel eine Gültigkeitsdauer von mindestens 13-15 Jahren gefordert wird. Um kompromittierte Schlüssel bekannt zu geben, muss jede CSCA regelmäßig CRLs (Certificate Revocation Lists)⁷ ausstellen und diese über diplomatische Kanäle oder über das Public-Key Directory (ICAO) publizieren.

In deutschen Pässen wird wie oben erwähnt zur Erstellung der digitalen

Zertifikat-Validierung

Signaturschlüssel

⁷ Rückruflisten für ungültige öffentliche Schlüssel

Signaturen ECDSA verwendet. Die Spezifikation erlaubt aber auch die Verwendung von RSA oder alternativ von DSA und empfiehlt hierfür Schlüssellängen von 3072 Bit für die CSCA-Instanzen bzw. von 2048 Bit für Document Signer Instanzen.

Aktive Authentifizierung des Chips

Klonen

Allein durch passive Authentifizierung der Passdaten kann ein Lesegerät nicht erkennen, ob der Pass vollständig gekloned wurde. Das heißt, es ist nicht erkennbar, ob es sich tatsächlich um einen authentischen Pass handelt. Die Daten in der MRZ auf der Datenseite des Passes sind zwar logisch mit den Daten in der Datengruppe 1 auf dem RFID-Chip verknüpft, so dass das Klonen eines Passes auch erfordert, dass die Datenseite des Passes kopiert wird. Dies ist schwierig, da die Datenseite durch zusätzliche physische Sicherheitsmerkmale, wie Hologramme, vor dem Kopieren geschützt wird. Dennoch ist ein Klone-Angriff nicht vollständig auszuschließen. Beim Klonen eines RFID-Chips werden die Chip-Daten ausgelesen und auf einen leeren Chip kopiert.

Abwehr

Zur Abwehr derartiger Angriffe wurde der Sicherheitsmechanismus der *Aktiven Authentifizierung* entwickelt. Bei der aktiven Authentifizierung muss der RFID-Chip des Passes dem Lesegerät beweisen, dass er den Chip-individuellen, geheimen Schlüssel kennt. Dieser Schlüssel muss deshalb im geschützten, nicht auslesbaren Speicher des RFID-Chips abgelegt sein. Ein solcher Schlüssel kann dann nur intern genutzt werden, so dass der Schlüssel nicht in einen geklonten Chip kopiert werden kann.

Zur aktiven Authentisierung gibt es zwei alternative Ansätze: die *Active Authentication* und die *Chip-Authentication*. Nachfolgend werden beide Protokolle kurz beschrieben. Da bei beiden Protokollen auf Daten des RFID-Chips lesend durch das Lesegerät zugegriffen wird, muss einer aktiven Authentisierung stets eine passive Authentisierung vorausgehen, in deren Verlauf zwischen Chip und Lesegerät ein gemeinsamer Sitzungsschlüssel etabliert wird.

Active Authentication Protokoll

AA

Die Basis einer Active Authentication (AA) bildet ein statisches Schlüsselpaar, dessen öffentlicher Bestandteil auf dem Chip in der Datengruppe 15 abgelegt ist. Der zugehörige private Schlüssel ist im geschützten Speicherbereich des Chips abgelegt. Zum Nachweis der Kenntnis des privaten Schlüssels wird ein asymmetrisches Challenge-Response-Protokoll zwischen Chip und Lesegerät abgewickelt. Dazu erzeugt das Lesegerät eine Random-Zahl, die vom Chip mit seinem geheimen Schlüssel signiert werden muss. Zur Signatur-Validierung liest das Lesegerät den zugehörigen

Schlüssel aus der Datengruppe 15 aus, der mittels Secure Messaging zum Lesegerät übertragen wird. Die ICAO-Spezifikation erlaubt für die Active Authentication den Einsatz von RSA, DSA oder ECDSA. Die empfohlenen Schlüssellänge bei RSA/DSA-Schlüsseln beträgt 1024 Bit und bei ECDSA 160 Bit. Beim heutigen Stand der Technik sind das relativ kurze Schlüssel.

Ein mögliches Sicherheitsproblem bei der Verwendung von aktiver Authentifikation wird darin gesehen, dass der Chip keine Kenntnis darüber hat, ob die Random-Nachricht, die das Lesegerät sendet, eine Semantik trägt. Ein durchaus realistisches Szenario ist das folgende: Ein Lesegerät, das bei Grenz-Kontrollen offiziell eingesetzt wird, sendet spezielle Challenges an die elektronischen Pässe. Diese Challenges sind so formatiert, dass sie in geeigneter Form das Datum, die Uhrzeit und ggf. auch Ortsinformationen enthalten. Mit dem Signieren einer solchen Challenge durch einen ePass kann an Kontrollpunkten beispielsweise nachgehalten werden, wann und wo eine bestimmte Person eine Grenze überschritten hat. Da jedoch eine solche Challenge auch von einem gefälschten Lesegerät gesendet werden kann, könnte ein Pass dazu verleitet werden, einen beliebigen Wert zu signieren, z.B. falsche Orts- und Zeitangaben. Diese signierten Daten könnten zu einem späteren Zeitpunkt dazu verwendet werden, falsche Orts- und Aufenthaltsdaten über den Passinhaber zu verbreiten.

Challenge-Semantik

Als Alternative zu der Active Authentication der ICAO hat die EU die *Chip-Authentifikation* spezifiziert, die das Problem der Challenge-Semantik behebt und zusätzlich stärkere Sitzungsschlüssel für das Secure Messaging etabliert. In den deutschen Reisepässen kommt deshalb das Verfahren der Chip-Authentifikation zum Einsatz.

Chip-Authentifikation Protokoll

Bei der Chip-Authentifikation handelt es sich um ein Diffie-Hellman-Schlüsselaustauschprotokoll (vgl. Kapitel 9.3.6) unter Nutzung klassischer Diffie-Hellman (DH) oder Elliptic Curve DH (ECDH)-Verfahren. Dadurch kann auf ein Challenge-Response Protokoll verzichtet werden und der Chip läuft nicht Gefahr, einen beliebigen Wert (die Challenge) zu signieren. Der benötigte statische, öffentliche DH-Schlüssel ist auf dem Chip in der Datengruppe 14 abgelegt. Der private Schlüssel ist wiederum im geschützten, nicht auslesbaren Speicher des Chips gespeichert. Auf der Seite des Lesegeräts benötigt das Protokoll einen flüchtigen (ephemeral) DH-Schlüssel, der vom Lesegerät speziell für die Abwicklung des Protokolls erzeugt wird. In deutschen Reisepässen wird zur Chip-Authentifikation das ECDH-Verfahren mit einem 224 Bit-Schlüssel verwendet.

Chip-Schlüssel

Zusätzlich zur Authentifizierung eines RFID-Chips ermöglicht die Chip-Authentifizierung auch den Aufbau eines stark verschlüsselten und inter-

Verschlüsselung

grittsgesicherten Kommunikationskanals fr den Datenaustausch zwischen Chip und Lesegert. In den Mitgliedslndern der EU ist der Einsatz der Chip-Authentisierung fr die verwendeten Lesegerte verpflichtend, so dass wann immer mglich (wenn der Chip dies untersttzt), eine starke Verschlsselung beim Datenaustausch mittels Secure Messaging verwendet wird und nicht nur beim Zugriff auf Fingerabdruckdaten im Kontext der Extended Access Control.

Das nachfolgend skizzierte Ablaufprotokoll nutzt, dass zwischen dem RFID-Chip und dem Lesegert bereits mittels passiver Authentifikation ein sicherer Kommunikationskanal mit einem gemeinsamen Sitzungsschlssel etabliert wurde, so dass alle unten angegebenen Nachrichten vertraulich kommuniziert werden. Fr die Abwicklung des Protokolls genigt das damit verbundene relativ schwache Sicherheitsniveau, da die Daten, die wrend der Protokollschritte ausgetauscht werden, nicht sensitiv sind und die Sicherheit der nachfolgend etablierten strkeren Schlssel nicht darauf beruht, dass diese ausgetauschten Daten sicher sind. Ist das Chip-Authentifikations-Protokoll nicht erfolgreich durchgefhrt worden, so bleibt dieser Basisschutz fr den Datenaustausch weiterhin bestehen. Das heit, dass der mittels Basis Access Control vereinbarte 112 Bit Sitzungsschlssel weiterhin fr das Secure Messaging verwendet wird. Dagegen wird bei erfolgreicher Abwicklung des Protokolls der alte Sitzungsschlssel durch den auf einer sicheren Basis ausgehandelten und damit strkeren Schlssel ersetzt.

Varianten

Das Chip-Authentisierungsprotokoll kann in zwei Versionen implementiert werden. Variante 1 sieht eine implizite Echtheitsprfung des Chips vor, wrend die Variante 2 aktive Aktionen seitens des Chips zum expliziten Echtheitsnachweis, also zum Nachweis, dass der Chip nicht geklont ist, fordert.

Protokollablauf:

1. Der Chip sendet seinen statischen DH-Public-Key und die zu dessen Erzeugung verwendeten DH-Parameter zum Lesegert.
2. Das Lesegert erzeugt ein anonymes, flchtiges DH-Schlsselpaar und sendet den ffentlichen Teil $PK_{Terminal}$ an den Chip zurck.
3. Beide Parteien berechnen nach dem DH-Verfahren den gemeinsamen Schlssel K .
4. Beide leiten aus K einen gemeinsamen MAC-Schlssel K_{MAC} und Verschlsselungsschlssel K_{Enc} ab.
5. In Version 2 des Protokolls (explizite Echtheitsprfung) berechnet der Chip ein Authentifizierungstoken $T = \text{Hash}(PK_{Terminal}, K_{MAC})$.

Mit dem Token, der lediglich ein MAC-Wert ist, weist der Chip explizit die Kenntnis des gemeinsamen Hash-Schlüssels K_{MAC} nach.

6. Wiederum nur in Version 2 prüft das Lesegerät das Token T.

Nach einer erfolgreichen Durchführung des Chip-Authentifizierungsprotokolls erfolgt die weitere Kommunikation zwischen Chip und Lesegerät mittels Secure Messaging und der Verwendung der **neuen** gemeinsamen Schlüssel K_{MAC} bzw. K_{Enc} .

Secure Messaging

Um tatsächlich von der Echtheit des Chips überzeugt zu sein, sollte laut Spezifikation das Lesegerät unmittelbar nach einer erfolgreichen Chip-Authentifizierung die Integrität der Chip-Daten anhand des Data Security Objekts prüfen. Wenn der Chip in der Lage ist, das Datenobjekt mit dem gemeinsamen Schlüssel K_{Enc} zu verschlüsseln und wenn die Daten korrekt sind, so wird der Chip als unverfälscht betrachtet.

Terminal Authentifizierung

Sensitive Daten auf dem Chip, für die eine erweiterte Zugriffskontrolle (EAC) vorgeschrieben ist, erfordern auch eine Authentisierung des Lesegeräts (Terminal) gegenüber dem Chip. Die Authentizitätsprüfung von Lesegeräten erfolgt auf der Basis von Zertifikaten und einem asymmetrischen Challenge-Response Protokoll. Hierzu ist der Aufbau einer erweiterten, ländergrenzenübergreifend funktionierenden PKI erforderlich (vgl. Abschnitt 11.2.1).

Terminal
Authentifikation

Protokollablauf:

1. Das Lesegerät sendet eine Folge von Zertifikaten, die zur Validierung seiner Signatur erforderlich sind, zum Chip. Die Folge beginnt mit einem Zertifikat, dessen Signatur durch einen öffentlichen Schlüssel prüfbar ist⁸, der auf dem Chip gespeichert ist. Das letzte Element der Zertifikatkette ist das Zertifikat des Lesegeräts.
2. Der Chip prüft die Zertifikate und extrahiert den öffentlichen Schlüssel des Lesegeräts.
3. Der Chip sendet eine Zufallszahl an das Lesegerät.
4. Das Lesegerät signiert die Zufallszahl zusammen mit der Chip-ID (auslesbar aus der MRZ des Chips).
5. Der Chip validiert die Signatur.

⁸ Da auf dem Chip das Zertifikat der Country Signing Certification Authority (CSCA) abgelegt ist, gibt es stets einen Vertrauensanker.

Nach einer erfolgreichen Terminal-Authentifikation kann das Lesegerät Daten auf dem Chip auslesen. Über das für das Lesegerät ausgestellte Zertifikat wird geregelt, welche Zugriffsberechtigungen das Lesegerät für die Daten des Chips besitzt. In Abschnitt 11.2.1 gehen wir auf die Ausstellung der Zertifikate und die länderübergreifende PKI noch genauer ein.

Zugriffskontrolle

Zur Vermeidung unautorisierter Lese-Zugriffe auf die Daten des RFID-Chips, sieht die ICAO-Spezifikation verpflichtend eine Zugriffskontrolle mit mindestens der Mechanismenstärke einer Basic Access Control (BAC) vor. Mit dieser Basis-Zugriffskontrolle soll verhindert werden, dass über drahtlose Verbindungen beliebige Lesegeräte die Passdaten unbemerkt, z.B. wenn der Passbesitzer den Pass in seiner Tasche bei sich trägt, auslesen können. Da BAC einige Mängel aufweist, wurde vom BSI eine Alternative, das PACE (Password Authenticated Connection Establishment) Protokoll entwickelt (siehe Abschnitt 11.2.2).

Für sensitive Daten im Chip, wie die biometrischen Merkmale der Fingerabdrücke des Passinhabers, wird eine stärkere Zugriffskontrolle, die erweiterte Kontrolle, Extended Access Control (EAC), benötigt. Durch sie wird geprüft, ob das zugreifende Lesegerät berechtigt ist, auf die sensitiven Daten zuzugreifen. Die erweiterte Zugriffskontrolle wird in den deutschen Reisepässen umgesetzt.

Basic Access Control (BAC)

BAC

Der Zugriff auf die Daten des RFID-Chips ist zumindest über Basic Access Control geschützt. Das Ziel ist, dass mit diesem Protokoll die Zugriffsmechanismen, die heute für nicht-elektronische Reisedokumente verwendet werden, nachgebildet werden, indem das Lesegerät physischen Zugriff auf den ePass haben muss. Diese Anforderung wird dadurch technisch umgesetzt, dass das Lesegerät die Daten aus der MRZ optisch auslesen muss. Die ausgelesenen Daten sind die 9-stellige Passnummer, das Geburtsdatum und das Ablaufdatum. Aus diesen Daten berechnet das Lesegerät den 56-Bit Zugriffsschlüssel (Access Key) K , indem ein Hashwert der Daten berechnet wird. In nachfolgenden Schritten weisen der ePass und das Lesegerät sich wechselseitig die Kenntnis dieses Access-Keys K nach und tauschen dabei Informationen zur Berechnung eines gemeinsamen Session-Keys $K_{session}$ aus. Für die nachfolgende verschlüsselte Kommunikation zwischen ePass und Lesegerät wird dann nur noch der dynamisch berechnete Sitzungsschlüssel $K_{session}$ verwendet. Der statische Schlüssel K dient nur als Initialisierungsschlüssel. Er ist aber ein bevorzugtes Angriffziel, da er der Sicherheitsanker für die weiteren dynamisch erzeugten Schlüssel ist. Die Abwicklung eines BAC-Ablaufs erfordert ca. 1 Sekunde Berechnungszeit.

Access Key

Protokollablauf:

1. Der RFID-Chip wählt eine Zufallszahl r_{chip} und eine 56-Bit Schlüsselhälfte K_{chip} . Die Zufallszahl r_{chip} wird an das Lesegerät übermittelt.
2. Das Lesegerät wählt eine Zufallszahl r_{reader} und eine 56-Bit Schlüsselhälfte K_{reader} .
3. Das Lesegerät berechnet einen Kryptotext $C1$:

$$C1 = E((r_{reader} \mid r_{chip} \mid K_{reader}), K)$$

Der Kryptotext $C1$ wird an den Chip übermittelt.

4. Der Chip entschlüsselt den Kryptotext $C1$ unter Einsatz des auf dem Chip abgelegten Schlüssels K :

$$D(C1, K) = r'_{reader} \mid r'_{chip} \mid K'_{reader}$$

Der Chip prüft die Korrektheit der Daten: $r'_{chip} = r_{chip}$.

5. Der Chip übermittelt dem Lesegerät nun seinerseits seinen Teil des Schlüssels, sowie den Nachweis, dass er über den gemeinsamen Schlüssel K verfügt, indem er einen Kryptotext $C2$ berechnet und diesen an das Lesegerät übermittelt:

$$C2 = E((r_{chip} \mid r'_{reader} \mid K_{chip}), K).$$

6. Das Lesegerät entschlüsselt $C2$ mit

$$D(C2, K) = r_{chip} \mid r'_{reader} \mid K_{chip}$$

und überprüft die Korrektheit von $r'_{reader} = r_{reader}$.

Nach der Authentifikation berechnen beide Partner einen gemeinsamen 112 Bit 3DES-CBC Sitzungs-Schlüssel und einen gemeinsamen Integritäts-Schlüssel. Dazu werden die in dem obigen Protokoll ausgetauschten Schlüsselbestandteile konkateniert:

$$K_{session} = K_{reader} \parallel K_{chip}.$$

Wie aus dem Protokollablauf ersichtlich, ist es für die Schritte 3-6 notwendig, dass das initiale Geheimnis K nicht gebrochen ist. Dies spielt bei der Bewertung der BAC in Abschnitt 11.2.1 eine entscheidende Rolle.

Extended Access Control

Die erweiterte Zugriffskontrolle, Extended Access Control (EAC) bietet einen starken Zugriffsschutz als der BAC-basierte Schutz. Laut ICAO Spezifikation muss zumindest der Zugriff auf die Fingerabdruckdaten im Pass durch die Extended Access Control abgesichert werden. Hierbei erhalten

Sitzungsschlüssel

EAC

nur berechtigte und authentische Lesegeräte Zugriff auf diese Daten. EAC erfordert, dass das zugreifende Lesegerät eine Terminal-Authentifizierung (siehe Abschnitt 11.2.1) durchführt. Dabei muss das Lesegerät ein Zugriffs-zertifikat⁹, das die Zugriffsrechte für den Pass definiert, sowie eine Kette von Zertifikaten vorweisen, mit deren Hilfe der Pass in der Lage ist, das Zugriffs-zertifikat des Lesegerätes zu validieren. Hierfür ist eine länderübergreifende PKI erforderlich.

Länderübergreifende PKI

PKI

Die elektronischen Reisedokumente werden vom Aussteller unterschrieben und zur Prüfung werden Signatur-Zertifikate der Unterzeichner benötigt. Elektronische Reisepässe bzw. Personalausweise erfordern eine Public Key Infrastruktur (vgl. Abschnitt 9.1.3). Die zur Prüfung elektronischer Reise-pässe aufzubauende PKI muss international interoperabel sein. Jedes teilnehmende Land baut hierfür eine zweistufige PKI auf. Die Wurzel-Instanz dieser länderspezifischen PKI ist die CSCA (Country Signing Certification Authority). In Deutschland wird, wie bereits weiter vorne erwähnt, die CSCA vom BSI betrieben. Die CSCA signiert die Zertifikate für die Document Signer (DS), das sind die Passaussteller. In Deutschland ist dies die Bundesdruckerei, die zum Signieren der Passdaten berechtigt ist. Abbildung 11.7 veranschaulicht die 2-stufige PKI. Der linke Teil der Abbildung umfasst die nationalen Einheiten, die für das Signieren von Daten der nationalen Reise-pässe und Zertifikaten zuständig sind.

Verifier-Instanzen

Für das Verfahren der Terminal-Authentisierung durch ein Lesegerät muss die PKI um Verifier-Instanzen und Zertifikate erweitert werden (vgl. die rechte Seite von Abbildung 11.7). Es muss dabei sicher gestellt werden, dass jedes Lesegerät im In- und Ausland in der Lage ist, seine Authenti-zität gegenüber einem beliebigen Reisepass nachzuweisen und damit den Zugriff auf die im Pass gespeicherten Daten zu erhalten. Die Terminal-Authentisierung basiert auf einem asymmetrischen Challenge-Response Verfahren. Das bedeutet, dass ein solches Lesegerät ein Zertifikat besitzen muss, das vom Chip des Reisepasses, auf den das Lesegerät zugreifen möch-te, validierbar ist.

Anhand eines einfachen Szenarios werden im Folgenden die erforderlichen Instanzen und Schritte zur Validierung von Terminal-Zertifikaten erläutert. Gegeben Sei ein deutscher ePass, der an der französischen Grenze durch ein französisches Lesegerät kontrolliert werden soll. Der ePass soll die Authentizität und Zugriffsberechtigung des Lesegeräts prüfen können.

⁹ Bem.: Im Zusammenhang mit dem elektronischen Personalausweis werden derartige Zertifikate idR als Berechtigungszertifikat bezeichnet.

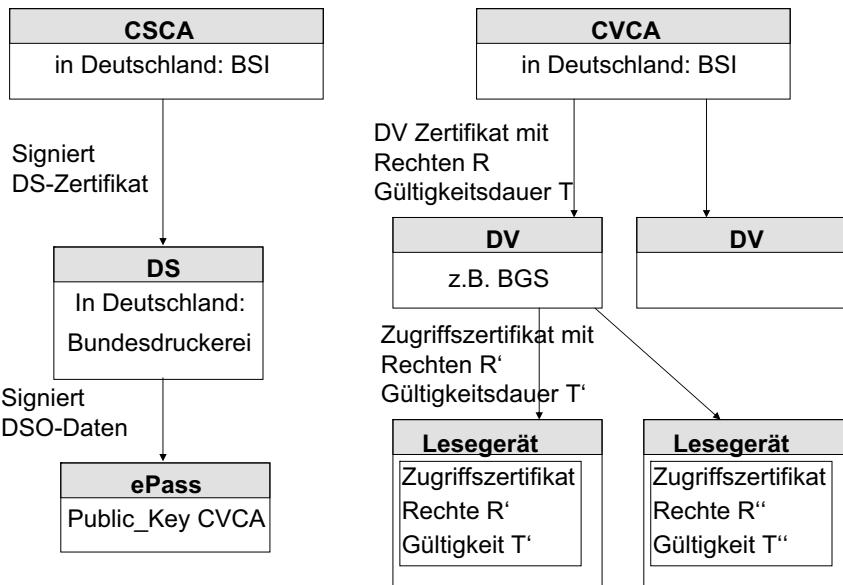


Abbildung 11.7: Nationale PKI mit Zertifikaten für Lesegeräte

Die erforderlichen Terminal-Zertifikate bzw. Zugriffszertifikate für Lesegeräte werden von den so genannten Document Verifier-Instanzen (DV) ausgehändigt. Jede DV-Instanz verwaltet eine Menge von Lesegeräten. In Deutschland ist eine solche DV-Instanz zum Beispiel der Bundesgrenzschutz (BGS), der Lesegeräte verwaltet, die bei der Grenzkontrolle zum Einsatz kommen. In dem Beispielszenario muss es einen französischen DV geben, der dem französischen Lesegerät ein solches Zertifikat aushändigt. Die nationalen Document Verifier Instanzen müssen ihrerseits dazu autorisiert werden, derartige Zertifikate für Lesegeräte auszustellen. Jedes Land ist deshalb verpflichtet, eine *Country Verifying CA* (CVCA) als vertrauenswürdige Instanz einzurichten, um eine solche Autorisierung vorzunehmen. Die Autorisierung erfolgt in Form von Zugriffszertifikaten, die von CVCAs für nationale und internationale DV-Instanzen ausgestellt werden. Auf den Reisepässen werden die öffentlichen Schlüssel der CVCA-Instanzen für die spätere Zertifikatvalidierung gespeichert.

Die von der CVCA ausgestellten Zugriffszertifikate beinhalten Zugriffsrechte auf Daten auf dem Chip. Da Reisepässe von nationalen Behörden ausgestellt werden, obliegt es der Entscheidungshoheit des jeweiligen Landes, welchem anderen Land der Zugriff auf die Pässe der eigenen Bürger gewährt wird. Für deutsche Reisepässe entscheidet dies die Deutsche Re-

Zugriffszertifikat

gierung. Für das Beispielszenario bedeutet das, dass die französischen Document Verifier Instanz von den deutschen Behörden eine solche Berechtigung erhält. Derartige Zugriffszertifikate werden u.a. über diplomatische Kanäle verteilt.

Um Missbrauch zu beschränken, sollten die Zertifikate nur eine kurze Gültigkeitsdauer besitzen. Die Festlegung der Gültigkeitsdauer der Zertifikate und des Umfangs der vergebenen Zugriffsrechte obliegt der nationalen CVCA. Ausgehend von dem Zugriffszertifikat stellt nun die DV-Instanz ein Zugriffszertifikat, das so genannte Terminal Zertifikat, für ein nationales Lesegerät aus und legt dabei die Zugriffsberechtigungen für das Lesegerät fest. Hierbei kann es sich jedoch höchstens um eine weitere Einschränkung der Rechtemenge handeln, die in dem entsprechenden Zugriffszertifikat bereits an den DV vergeben worden ist. Auch die Gültigkeitsdauer des Terminal-Zertifikats kann höchstens so lang sein, wie die des DV-Zertifikats. Für das Beispielszenario gilt somit, dass das französische Lesegerät von seiner zugeordneten französischen DV-Instanz ein Zugriffszertifikat erhält.

Validierung

Bei der Terminal-Authentisierung muss das Lesegerät ein gültiges Zugriffszertifikat vorweisen sowie eine Zertifikatskette an den Chip übermitteln, die so enden muss, dass die Validierung mit dem auf dem Chip gespeicherten öffentlichen Schlüssel der nationalen CVCA-Instanz starten kann. In unserem Beispielszenario übermittelt das französische Lesegerät sein Zugriffszertifikat sowie das Zugriffszertifikat seiner DV-Instanz, das von der deutschen CVCA ausgestellt wurde. Der deutsche Reisepass validiert das DV-Zugriffszertifikat mit dem öffentlichen CVCA-Schlüssel, der auf dem Pass abgelegt ist, und ist dann in der Lage, das Zugriffszertifikat des Leseräts zu validieren.

Sicherheit des ePasses

Im Verlauf der Einführung des ePasses wurden eine Vielzahl von Sicherheitsbedenken beim Umgang mit dem Pass geäußert. Nachfolgend werden einige Problembereiche angesprochen und diskutiert. Hauptkritikpunkte betreffen das mangelhafte Sicherheitslevel der Basic Access Control, die Gefahr des unerlaubten Mitlesen von Kommunikationsbeziehungen, die Möglichkeit, Aufenthalts- und Bewegungsprofile von Passbesitzern zu erstellen (z.B. durch nicht autorisiert betriebene Lesegeräte, die an Türen angebracht sind), sowie die Beeinträchtigung der Privatsphäre durch das unberechtigte Auslesen biometrischer Daten aus dem Reisepass. Wir gehen auf diese Punkte im Folgenden kurz ein.

Sicherheit von BAC

Der Sitzungsschlüssel weist mit 112-Bit eine bei heutigem Stand der Technologie eine noch ausreichende Schlüsselstärke auf. Die NIST hat Schlüs-

Zugriffsschlüssel

sellängen von 112 Bit bis 2030 als ausreichend eingestuft. Eine Kompro-mittierung des Sitzungsschlüssels ist aber bereits bei heutiger Technologie möglich, da die beiden 56-Bit Teilschlüssel zur Übertragung nur mit dem Zugriffsschlüssel K verschlüsselt werden. Da dieser Schlüssel lediglich aus wenigen Daten (Passnummer, Geburtsdatum, Ablaufdatum) aus der MRZ berechnet wird, besitzt er nur eine geringe Entropie. Hinzu kommt, dass dies ein statischer Schlüssel ist, der bei jedem Protokolllauf für jedes Lesegerät der gleiche ist.

Aufgrund der geringen Entropie des Schlüssels K besteht die Möglichkeit von Brute-Force Angriffen. Die Stärke des Schlüssels K entspricht etwa einem 56 Bit Schlüssel und ist damit einem DES-Schlüssel vergleichbar. Dies ergibt sich aus den Daten, die zur Schlüsselberechnung eingehten: die 9-stellige Passnummer (also 10^9 Möglichkeiten), das Geburtsdatum mit ca. $365 \cdot 10^2$ Möglichkeiten und das Ablaufdatum mit $365 \cdot 10$ Möglichkeiten, da der Pass einen Gültigkeit von 10 Jahren besitzt. Damit ergibt sich ein Suchraum von $365^2 \cdot 10^{12}$, was ungefähr dem Suchraum von 2^{56} entspricht.

Dieser Suchraum von 2^{56} kann jedoch deutlich reduziert werden, wenn In-formationen, wie zum Beispiel das exakte Geburtsdatum des Passinhabers, bekannt sind. Auch ergeben sich Schwächen daraus, dass die zur Schlüssela-bleitung verwendeten Daten des Passes nicht zufällig und gleichverteilt sind. So besteht die 9-stellige Passnummer in Deutschland aus einer 4-stelligen Behördennummer und einer 5-stelligen PassID. Ist die Behördenkennung bekannt, reduziert sich der unbekannte Raum der Passnummern auf 10^5 . Auch das Geburtsdatum kann z.B. mit Kenntnis des Passfotos geschätzt und damit auch dieser Suchraum eingeengt werden. Das Ablaufdatum des Passes ist abhängig von dem Ausstellungsdatum. Da an Feiertagen und Wochenen-den normalerweise keine Pässe ausgestellt werden, reduziert sich die Zahl der Ausgabetermine auf ca. 253 pro Jahr und damit auch dieser Suchraum. Auf diese Weise ist eine Reduktion des Raums auf 2^{30} möglich. Ein Angriff ist mit herkömmlicher Hardware innerhalb weniger Stunden möglich.

Sind die Daten der MRZ überwiegend bekannt, so kann der Suchraum sogar auf 2^{20} begrenzt werden. Anzumerken ist, dass bei der Durchführung der Brute-Force Angriffe der Angreifer Zugriff auf den Pass haben muss, um die Daten abzugleichen. Auch wenn Brute-Force Angriffe auf die Daten praktisch durchführbar sind, muss man sich klar darüber sein, dass es sich bei den Daten um solche personenbezogene Daten handelt, die i.d.R. auch auf anderem Wege leicht zu beschaffen sind. Dennoch bedeuten diese Angriffs-möglichkeiten eine Gefährdung der Privatsphäre. Aufgrund der Schwächen von BAC wird in Europa beim Zugriff auf **alle** Daten die Verwendung der Extended Access Control gefordert.

brute force

Schlüsselraum

Auslesen der Passdaten

aktiv

Mit der Kenntnis des Zugriffsschlüssels K kann ein Lesegerät versuchen, einen aktiven Leseangriff auf den Pass durchzuführen, ohne im physischen Besitz des Passes zu sein. Zum aktiven Auslesen muss sich der Pass aber in der Nähe des Gerätes befinden, da es sich bei dem in den Pässen verwendeten RFID-Chips um ISO 14443-konforme Chips handelt, die eine deutlich geringe Signalisierungsreichweite als die in der Logistik üblicherweise eingesetzten Chips besitzen. Analysen und Studien, wie die MARS-Studie des BSI (vgl. [34]) oder die Untersuchungen der Gruppe von D. Wagner (vgl. [95]) haben gezeigt, dass unter der Voraussetzung, dass die Daten der MRZ bekannt sind, ein aktives Auslesen eines ISO 14443-konformen Chips nur in einer Reichweite bis zu 25cm möglich ist.

passiv

Die oben zitierte BSI-Studie hat zudem gezeigt, dass ein passives Mitlesen (engl. *skimming*) der verschlüsselt übertragenen Daten in einer Entfernung bis zu 2m möglich ist. Da die Daten mit einem 112 Bit Schlüssel mittels 3DES-CBC verschlüsselt sind, haben einzelne Bitfehler in einem übertragenen 64-bit Block eine direkte Auswirkung auch auf den unmittelbar folgenden Block. Der Angreifer müsste, auch wenn ihm der verwendete Sitzungsschlüssel bekannt wäre, die möglichen Bitfehler manuell korrigieren. Derartige Angriffe sind möglich, aber auch relativ aufwändig.

Profilbildung

dynamische UID

Bei jeder Interaktion zwischen ePass und einem Lesegerät fordert die ICAO-Spezifikation die Übermittlung einer UID (Universal Identifier) des Chips. Dies wird zur Durchführung eines Collision Avoidance Protokolls bei der drahtlosen Kommunikation benötigt. Falls ein ePass hierbei stets die gleiche UID überträgt, könnte die Gefahr einer Profilbildung bestehen. Deshalb empfiehlt die ICAO-Spezifikation die Erzeugung dynamischer UIDs, so dass bei jeder Kontaktaufnahme eine Random-ID erzeugt und verwendet wird. Dynamische UIDs kommen in deutschen Reisepässen zum Einsatz. Zusammen mit der geringen Reichweite von 10 bis max. 25cm eines ISO 14443-konformen Chips und den Zugriffskontrollen basierend auf BAC oder EAC scheint die Bedrohung der Profilbildung relativ gering.

Zertifizierung

Zertifizierung

Zur Sicherstellung der internationalen Interoperabilität der elektronischen Reisepässe und der dazugehörigen Lesegeräte hat das BSI eine technische Richtlinie im Sinne von Prüfkriterien entwickelt. Neutrale Prüfstellen, die vom BSI hierzu zertifiziert sein müssen, können eine Konformitätsprüfung auf Grundlage der in der technischen Richtlinie definierten Prüfspezifikationen durchführen. Die Prüfung wird von der zuständigen Bestätigungsstelle

des BSI überwacht und nach erfolgreichem Abschluss mit einem Konformitätsbescheid und einem Zertifikat¹⁰ bestätigt. Im Zusammenhang mit dem Reisepass sind verschiedene technische Richtlinien von Bedeutung. Dazu gehören die Richtlinie zur Produktionsdatenerfassung, zur Produkt-Qualitätsprüfung oder aber auch die Prüfkriterien für elektronische Reisedokumente.

11.2.2 Personalausweis

Das Bundeskabinett hat am 23.7. 2008 dem Gesetz über Personalausweise und den elektronischen Identitätsnachweis zugestimmt. Als Konsequenz wurde die Einführung eines elektronischen Personalausweises¹¹, beschlossen. Der elektronische Personalausweis (nPA) wird seit dem 1. November 2010 von der Bundesdruckerei in Berlin produziert und an die deutschen Bürger ausgegeben. Der nPA wurde auf Scheckkartenformat ID-1 verkleinert. Abbildung 11.8 zeigt eine Aussenaufsicht auf den nPA. Wie der Reisepass, so ist auch der nPA mit einem kontaktlosen, ISO 14443-konformen RFID-Chip ausgestattet [32]. Für Personen ab der Vollendung des 24ten Lebensjahres ist der Ausweis 10 Jahre gültig sein; bei Personen unter 24 Jahren besitzt er eine Gültigkeit von 6 Jahren.

Seit dem 1.9.2011 produziert die Bundesdruckerei zusätzlich den elektronischen Aufenthaltstitel eAT, der an Ausländer ausgegeben wird. Grundlage dafür ist eine EU-Verordnung, die die Mitgliedstaaten zur Einführung eines solchen Dokumentes verpflichtet, um die Aufenthaltstitel der Europäischen Union für Drittstaatsangehörige zu vereinheitlichen und über die genutzten biometrischen Merkmale einen höheren Schutz gegen Missbrauch zu erzielen. Im Unterschied zum nPA können auf dem Chip, der in den eAT integriert ist, noch zusätzliche Informationen aufgenommen sein, wie die Gültigkeitsdauer der Aufenthaltsgenehmigung und mögliche Auflagen bezüglich der Arbeitsberechtigung. Diese zusätzlichen Daten sind für einen online-Zugriff nicht verfügbar und können nur von hoheitlichen Stellen, dazu zählen die Polizei oder die Ausländer- und Meldebehörden, ausgelesen werden. Wie der nPA, so verfügt auch der eAT über eine Online-Authentisierungsfunktion, die vom Inhaber aktiviert oder deaktiviert werden kann. Der eAT ist zudem auch vorbereitet, qualifizierte elektronische Signaturen durchzuführen, wofür, wie auch beim nPA, ein kostenpflichtiges Signaturzertifikat erforderlich ist, das auf dem Chip abgelegt und durch eine spezielle PIN geschützt wird.

¹⁰ Es handelt sich hierbei nicht um X509-Zertifikate, sondern Zertifikate im Sinne von Prüfsiegeln.

¹¹ Bis Anfang 2010 hieß der Ausweis noch ePA, bevor er Anfang 2010 in nPA, neuer Personalausweis, umgetauft wurde. Seit 2014 soll das Dokument nur noch unter dem Namen Personalausweis geführt werden.



Abbildung 11.8: Muster eines nPA, Quelle BMI

Funktionen des nPA

Biometrie

Der elektronische Personalausweis kann auch als Passersatz für hoheitliche Personenkontrollen, wie z.B. durch die Polizei, verwendet werden. Dazu werden körperliche Merkmale, wie Lichtbild, Größe, Augenfarbe, Alter optisch sichtbar auf dem Ausweis aufgebracht, so dass eine Identitätsprüfung durch Inaugenscheinnahme und Vergleich möglich ist. Für einen elektronischen Abgleich wird der Ausweis die ePass-Anwendung unterstützen. Analog zur hoheitlichen Personenkontrolle mittels des elektronischen Reisepasses (ePass) werden bei dieser Anwendung die Daten der MRZ sowie die auf dem Chip des elektronischen Personalausweises gespeicherten biometrischen Daten benötigt. Verpflichtend ist hierfür das Gesichtsbild (wie beim Reisepass) bei der Ausweisausstellung im Ausweis digitalisiert abzuspeichern. Anders als jedoch beim Reisepass kann der Benutzer selber entscheiden, ob er auch seinen Fingerabdruck auf den Chip des Personalausweises ablegen möchte.

Zwei wichtige Funktionen charakterisieren den elektronischen Personalausweis und unterscheiden ihn damit auch vom elektronischen Reisepass. Es handelt sich um die eID Funktion zum Nachweis der digitalen Identität und die Option der Verwendung qualifizierter Signaturen nach dem deutschen Signaturgesetz bzw. der eIDAS Verordnung. Mit der elektronischen Signatur wird, anders als mit der eID-Funktion, eine dauerhafte Zurechenbarkeit zwischen der abgegebenen Willenserklärung und der unterzeichnenden Person erreicht. Eine derartige dauerhafte Bindung wird in etlichen Vorschriften im Geschäftsverkehr bzw. bei Verwaltungsverfahren gesetzlich gefordert.

MRZ

Die maschinenlesbare Zone (MRZ) des nPA besteht aus drei Zeilen:

- Zeile 1 ist in zwei Segmente unterteilt und beschreibt die Dokumentenart (Identitätskarte Deutschland). Sie enthält zudem die Ausweisnummer

(auf der Vorderseite rechts oben) bestehend aus der Kennzahl der ausstellenden Behörde (erste bis vierte Ziffer), einer zufällig vergebenen Nummer (5. bis 9. Ziffer), einer Prüfziffer (10. Ziffer), sowie ungenutzten Felder, die durch das Zeichen < aufgefüllt werden.

- Die Zeile 2 ist in vier Segmente unterteilt und enthält das Geburtsdatum im Format JJMMTT mit abschließender Prüfziffer und nachfolgenden Füllzeichen, den letzten Tag der Gültigkeit des nPA im Format JJMMTT mit abschließender Prüfziffer, die Staatsangehörigkeit D und die Prüfziffer für die gesamte Zeile am Zeilenende.
- Zeile 3 enthält den Familiennamen, Vornamen (ggf. weitere Vornamen in derselben Reihenfolge wie in der Geburtsurkunde), ungenutzte Felder am Ende werden wieder mit dem Füllzeichen aufgefüllt.

eID-Funktion

Der elektronische Personalausweis bietet die so genannte eID-Funktion bzw. Online-Authentisierungsfunktion. Mit ihr können Daten, die auf dem RFID-Chip des Ausweises abgelegt sind, zur Online-Authentifizierung verwendet werden. Der Personalausweis liefert damit digitale Identitätsnachweise, so dass der Ausweisinhaber im Internet-basierten Geschäftsverkehr mit Verwaltung und Privatwirtschaft, beispielsweise beim Online-Shopping, Online-Banking oder Online-Auktionen, einfach und sicher identifiziert werden kann. Um die eID-Funktion des Ausweises nutzen zu können, muss sie aktiviert sein. Die eID Funktion ist jederzeit auf Antrag aktivierbar (dieser Vorgang kostet 6 Euro) und kann auch wieder deaktiviert werden. Ein aktivieren und deaktivieren der eID Funktion ist jedoch nur durch die Personalausweisbehörde zulässig. Diese muss dafür ein spezifisches Berechtigungszertifikat (s.u.) besitzen. Für Jugendliche unter 16 Jahren ist die eID-Funktion deaktiviert und kann erst ab dem 16. Lebensjahr durch die Behörde freigeschaltet werden.

eID

Für die eID Funktion sind folgende Datenfelder im nPA vorgesehen¹²:

Datenfelder

- Vornamen, Familienname,
- Doktorgrad,
- Tag und Ort der Geburt, sowie eine Funktion, die Auskunft gibt, ob ein bestimmtes Alter über- bzw. unterschritten ist (wie beispielsweise 18 Jahre),
- gegenwärtige Anschrift, sowie eine Funktion, die Auskunft gibt, ob der Wohnort mit einem bestimmten Wohnort übereinstimmt
- Dokumentenart (Personalausweis),

¹² Der nPA enthält noch weitere Daten, die hier nicht mit aufgeführt sind.

- ausstellendes Land, Abkürzung D für Bundesrepublik Deutschland,
- eine Funktion zur Abfrage der Gültigkeit des nPA

Bei der Beantragung des neuen Ausweises kann der Benutzer explizit untersagen, dass bestimmte Datenfelder zum Auslesen verfügbar sind. Die biometrischen Daten stehen ausschließlich für hoheitliche Personenkontrollen zur Verfügung. Das heißt, dass eBusiness oder eGovernment-Anwendungen keinen Zugriff auf biometrische Daten im Ausweis mittels der eID Funktion erhalten. Der Zugriff auf die Daten im Ausweis erfordert vom zugreifenden Service einen Nachweis, dass er authentisch und zum Zugriff berechtigt ist. Diensteanbieter benötigen deshalb Berechtigungszertifikate, die von dazu autorisierten Stellen auszustellen sind. Der Vorgang entspricht dem Vorgang der Ausstellung von Zugriffszertifikaten für Lesegeräte für den elektronischen Reisepass. Wir gehen weiter unten auf das Konzept des Berechtigungszertifikats noch etwas genauer ein.

PIN

Beim nPA soll der Besitzer des Personalausweises die Kontrolle darüber behalten, welcher Online-Dienstanbieter Zugriff auf welche Daten im Personalausweis erhält. Dazu wird, bei einer Kommunikation mit einem Diensteanbieter, dem Benutzer angezeigt, welcher Diensteanbieter auf welche Datenfelder im Ausweis zugreifen möchte. Der Benutzer kann explizit die Zugriffe auf einzelne Datenfelder für den Zugriff verbieten oder erlauben (ankreuzen von entsprechenden Datenfeldern). Bevor ein Zugriff dann tatsächlich erfolgen kann, muss der Benutzer noch explizit die Erlaubnis dazu erteilen. Dazu muss er seine 6-stellige Benutzer-PIN verwenden, die er nach der Beantragung des Personalausweises per PIN/PUK-Brief zugesandt bekommt. Mit der Eingabe seiner 6-stelligen Benutzer-PIN bestätigt der Benutzer, dass der Zugriff auf seinen nPA erfolgen darf.

Durch die Verknüpfung von Besitz (der Ausweis) und Wissen (PIN) beim Zugriff auf die Daten im RFID-Chip des nPA soll zudem die Sicherheit erhöht werden, dass Jemand, der im Besitz des Ausweises ist, unter der digitalen Identität des Ausweisinhabers im Online-Business aktiv wird. Eine Änderungsmöglichkeit für die PIN ist vorgesehen. Der Passbesitzer kann seine PIN selber ändern; dazu ist wie üblich die Kenntnis der alten PIN notwendig. Wurde die PIN vergessen, so kann nur die Personalausweisbehörde nach eindeutiger Identifizierung des Ausweisinhabers die PIN ändern. Die PUK dient wie üblich zum Zurücksetzen des Fehlbedienungszählers, der für die Benutzer-PIN im Chip geführt wird und nach dreimaliger Falscheingabe zum Blockieren der PIN führt.

Sperrung

Der versiegelte PIN-Brief, der per Post zugestellt wird, enthält auch das Nutzer-spezifische Sperrkennwort, mit dem der Nutzer bei Verlust den Ausweis sperren lassen kann. Über eine rund um die Uhr erreichbare

Telefon-Hotline kann der Bürger bei einem Verlust oder Diebstahl seinen Ausweis sperren lassen. Der gesperrte Ausweis wird auf eine Sperrliste gesetzt, die von jedem Diensteanbieter heruntergeladen werden kann. Mit Hilfe der Sperrliste kann jeder Dienstanbieter prüfen, ob eine Authentisierung mit einem als verloren oder gestohlen gemeldeten elektronischen Personalausweis versucht wird.

Um die eID Funktion für Online-Transaktionen nutzen zu können, benötigt ein Ausweisinhaber sowohl Hardware- als auch Software-Komponenten. Laut BMI sind folgende Komponenten gefordert:

- Funktionsfähiger Arbeitsplatzrechner,
- (zertifiziertes) Kartenlesegerät für kontaktlose Chipkarten. Auf die Kartenleser und deren Funktionsumfang gehen wir auf der Seite 550 noch ein.
- die Ausweis App. Die App ist eine Anwendung, die eine sichere Verbindung zwischen dem verwendeten Kartenlesegerät, dem Personalausweis und dem Diensteanbieter herstellt. Sie ermöglicht den verschlüsselten Datenaustausch zwischen Personalausweis und Diensteanbieter. Die Ausweis App wird vom Bund kostenlos zum Download zur Verfügung gestellt: <https://www.ausweisapp.bund.de>.
- ein Internetzugang über eine Web-Browser.

Erforderliche Komponenten

Qualifizierte Signatur

Ist bei Geschäftstransaktionen oder Verwaltungsabläufen eine eigenhändige Unterschrift erforderlich, so wird im eBusiness oder eGovernment eine elektronische Signatur als Äquivalent verlangt. Deshalb ist der elektronische Personalausweis vorbereitet, ein Signaturzertifikat nachzuladen, das der Bürger separat bei einem dafür akreditierten Anbieter erwerben muss. Das bedeutet, dass der nPA für die Aufnahme eines qualifizierten Signaturzertifikates im Prinzip vorbereitet ist.

Signatur

Schutzziele

Analog zum ePass gehören die Integrität der Ausweisdaten, die Authentizität, die Zugriffskontrolle und die vertrauliche Kommunikation zu den Schutzz Zielen des nPA. Anders jedoch als beim ePass werden bei der Online-Authentisierung Ausweisdaten nicht nur über kurze Distanzen drahtlos zwischen Chip und Lesegerät übertragen, sondern über das ungesicherte Internet, so dass die Kommunikationsverbindung stark abgesichert werden muss. Zudem sind im Sinne der mehrseitigen Sicherheit bei einer Online-Authentisierung die Sicherheitsanforderungen beider beteiligter Parteien zu berücksichtigen:

Schutzziele

1. Der Online-Dienstanbieter muss in der Lage sein zu überprüfen, ob der Nutzer des Passes auch der berechtigte Inhaber des Ausweises ist. Bei der ePass-Anwendung zur hoheitlichen Personenkontrolle erfolgt dies durch den kontrollierenden Grenzbeamten, der den Ausweis kontrolliert und das Gesichtsbild auf dem Ausweis mit der Person direkt vergleicht bzw. einen Rechner bedient, der den Vergleich durchführt.
2. Der Inhaber des Personalausweises sollte die Kontrolle darüber haben, welche Dienstanbieter auf welche Daten auf seinem Ausweis zugreifen dürfen. Zudem ist zu verhindern, dass Bewegungs- und Aufenthaltsprofile des Ausweisinhabers erstellt werden können, um die Privatsphäre nicht zu gefährden.

Zur Erfüllung der Sicherheitsanforderungen kommen die Verfahren PACE (Password Authenticated Connection Establishment) (s.u.) und EAC zur Zugriffskontrolle sowie Terminal- und Chip-Authentifizierung zur Authentisierung der Teilnehmer zum Einsatz. Anwendungen dürfen nicht auf den nPA nachgeladen werden.

PACE-Protokoll

PACE

PACE (Password Authenticated Connection Establishment) ist ein kryptografisches Protokoll, das zum Ziel hat, die Verwendung klassischer PIN-basierter Authentisierungstechniken auch für kontaktlose Karten zu ermöglichen. PACE wurde vom BSI entwickelt und in der Technische Richtlinie TR-03110 beschrieben [33]. Das PACE-Verfahren besteht im Kern aus einem Diffie-Hellman-Schlüsselaustauschprotokoll, wobei die ausgetauschten Schlüssel mittels (einfachen) Passwörtern (PINs) abgesichert werden. Auch bei BAC wird ein Passwort verwendet. Dies ist der immer gleiche Zwischenschlüssel, der aus den Daten der MRZ berechnet wird und damit jedem Lesegerät mit optischem Zugriff direkt zugänglich ist. PACE unterstützt demgegenüber mehrere, falls die Geräte es zulassen sogar dynamische Passworte bzw. PINs zur Authentisierung. PACE unterscheidet 2 Klassen von PINs:

Karten-PIN

- Karten-PIN (CardPIN): Die Karten-PIN ist keine geheime PIN. Es kann eine statische oder dynamische PIN sein. Eine dynamische PIN wird von der Karte auf dem Karten-Display angezeigt (falls ein solches vorhanden ist). Eine statische PIN ist eine Nummer, die auf das Dokument (den Pass) aufgedruckt ist. Diese Nummer kann maschinell auslesbar sein. Sie dient zur Simulation des Steckvorgangs kontaktbehafteter Karten. Analog zur BAC-Prüfung erfordert das Auslesen dieser PIN den optischen Zugriff auf den Pass.

- Nutzer-PIN (eID): Die Nutzer-PIN ist eine geheime PIN, die nur der Ausweis-Besitzer und sein Chip kennen sollten. Analog zur PIN-basierten Authentisierung bei der Verwendung von Smartcards, kann der Kartenbesitzer unter Eingabe der PIN am Lesegerät nachweisen, dass er der berechtigte Nutzer ist. Die PIN wird dem Nutzer wie weiter oben bereits beschrieben mittels PIN-Brief auf dem Postweg übermittelt.

Die Karten-PIN ist eine nicht-blockierende PIN, während mit Benutzer-PINs, die beim nPA 6 Ziffern lang sind, ein Fehlbedienungszähler verknüpft ist. Entsprechend sind mittels einer PIN-basierten Authentisierung geschützte Karten anfällig gegen Denial of Service-Angriffe, die bei den kontaktlosen Karten eine physische Nähe (ca. 10 cm) zur Karte erfordern. Zur Abschaltung derartiger Angriffe verwendet PACE das Konzept des verzögerten Blockierens. Nach Ablauf der maximalen Zahl an Fehlbedienungen (beim nPA sind das 2 Versuche) wird die Karte zunächst suspendiert und der Benutzer muss als nächstes die korrekte Karten-PIN eingeben, so dass ein Angreifer physischen Zugriff auf den Ausweis haben müsste. Erst wenn nach korrekter Eingabe der Karten-PIN erneut eine falsche Benutzer-PIN eingegeben wird, erfolgt die Sperrung der PIN. Die PUK besitzt einen Nutzungszähler (UsageCounter), der bei jeder PUK-Eingabe dekrementiert wird, unabhängig davon, ob die PUK richtig oder falsch war. Er wird nie inkrementiert. Erreicht er den Wert 0, dann wird die Karte unbrauchbar. Ein typischer Initialwert einer PUK ist 10.

eID

Verzögertes
Blockieren

Protokollablauf (vergröbert)

Im Gegensatz zu der oben beschriebenen Chip-Authentisierung, bei der ein statisches DH-Schlüsselpaar, das auf dem ePass-Chip abgelegt ist, verwendet wird, generieren bei der PACE-Authentisierung sowohl der Chip als auch das Lesegerät dynamisch flüchtige (ephemeral) DH-Schlüsselpaare, basierend auf den auf dem Chip abgelegten DH-Parametern, den sogenannten Domänen-Parametern. Die Authentizität der öffentlichen DH-Schlüssel wird über den Nachweis der Kenntnis des gemeinsamen Geheimnisses, der 6-stelligen PIN, sichergestellt. Beim nPA kommen zur Abwicklung des PACE-Protokolls die Verfahren ECDH und ECDSA mit 224 Bit-Schlüsseln zum Einsatz.

Basis

1. Der Chip wählt eine Zufallszahl s und verschlüsselt diese mit einem aus der PIN π abgeleiteten Schlüssel $K_\pi : C = E(s, K_\pi)$.
2. Der Chip überträgt den Kryptotext C und seine DH-Parameter zum Lesegerät.
3. Das Lesegerät muss die PIN π besitzen, entweder durch direktes Extrahieren aus der lesbaren Zone des Ausweises, falls es sich um eine statische Karten-PIN handelt, oder der Benutzer muss seine geheime PIN am Lesegerät eingeben.

Protokoll

4. Das Lesegerät leitet seinerseits den Schlüssel K_π aus der PIN ab und entschlüsselt den erhaltenen Kryptotext: $s = D(C, K_\pi)$.
5. Chip und Lesegerät erzeugen jeweils flüchtige DH-Schlüsselpaare basierend auf den neu berechneten DH-Parametern. Dazu wenden sie jeweils eine Abbildungsfunktion an, die als Eingabe die ursprünglichen Domänen-Parameter des Chips sowie die verschlüsselt ausgetauschte Zufallszahl s verwendet.
6. Beide Partner tauschen ihre jeweiligen öffentlichen DH-Schlüssel aus.
7. Beide Partner berechnen den gemeinsamen, geheimen DH-Schlüssel K und leiten davon einen gemeinsamen Integritäts- K_{MAC} und Sitzungsschlüssel K_{Enc} ab.
8. Beide Partner erzeugen jeweils ein Authentisierungstoken. Dies ist ein MAC über den Integritätsschlüssel K_{MAC} und den öffentlichen DH-Schlüssel des Partners.
9. Beide prüfen den MAC und verwenden danach die neuen Schlüssel für das nachfolgende Secure Messaging zwischen dem Chip und dem Lesegerät.

Sicherheit

In [19] wurde ein formaler Sicherheitsbeweis für PACE geführt und die Stärke von PACE wurde damit gezeigt. Dennoch ist das PACE-Protokoll qua Design anfällig gegen Brute-Force Angriffe auf das verwendete Passwort. Beim Einsatz von PACE im nPA ist aber durch die Blockade der PIN nach der 2-maligen Falsch-Eingabe des Passwortes eine wirksame Gegenmaßnahmen gegen solche Brute-Force Angriffe implementiert. Da die Stärke der von PACE berechneten Integritäts- und Sitzungsschlüssel nicht vom verwendeten Passwort abhängt, können auch sehr kurze Passworte, wie die 6-stellige PIN beim nPA, verwendet werden.

PIN-Klassen

Die bei PACE verwendete PIN ist im Fall einer Online-Authentisierung bei E-Business oder E-Government eine 6-stellige Benutzer-PIN, um die Personenbindung an den Ausweis durchzuführen. Wird jedoch nur ein sicherer Kanal zwischen einem bestimmten Ausweis und einem Lesegerät benötigt, wie dies bei hoheitlichen Kontrollen erforderlich ist, so wird lediglich eine Karten-PIN benötigt. Diese kann als 6-stellige PIN auf der Karte aufgedruckt oder in der MRZ abgelegt sein. Dies ist dann natürlich keine geheime PIN mehr, was aber für die hoheitliche Personenkontrolle auch nicht notwendig ist, da hierbei der Besitz des Ausweises nachgewiesen wird und die Nutzeridentität unmittelbar prüfbar ist.

nPA bei der Online-Authentisierung

In diesem Abschnitt gehen wir etwas genauer auf die Nutzung der eID-Funktion des Ausweises bei der Online-Authentisierung ein.

Berechtigungszertifikat Will ein Online-Dienstanbieter auf Daten im Personalausweis zugreifen, muss er ein Berechtigungszertifikat¹³ besitzen. Ein solches Zertifikat wird von einer staatlichen Stelle auf Antrag des Diensteanbieters ausgestellt. Am 26. Februar 2010 hat das Bundesministerium des Innern gemäß Paragraph 4 des Personalausweisgesetzes vom 18. Juni 2009 das Bundesverwaltungsamt in Köln als Vergabestelle für Berechtigungszertifikate (VfB) und als Sperrlistenbetreiber bestimmt.

Berechtigungszertifikat

Der Antrag auf Ausstellung eines Berechtigungszertifikats muss Angaben über Name und Anschrift des Dienstanbieters, Nachweis des berechtigten Interesses an den personenbezogenen Daten, Zweckbindung der Daten, Angabe über die eingesetzte Krypto-Hard- und -Software, eine Datenschutzerklärung der zuständigen Datenschutzaufsichtsbehörde, sowie eine Darlegung enthalten, dass geeignete, (zertifizierte) Software beim Diensteanbieter zum Einsatz kommt. Das Bundesverwaltungsamt hat bei der Anfrage die Zulässigkeit des Dienstanbieters, sowie die Plausibilität der angefragten Datenzugriffe in Bezug auf den intendierten Verwendungszweck zu prüfen. Ist die Prüfung erfolgreich verlaufen, wird der Diensteanbieter aufgefordert, ein Schlüsselpaar mit einer zertifizierten Krypto-Soft- und Hardware zu erstellen. Als nächstes muss der Diensteanbieter seinen öffentlichen Schlüssel, sowie ein SSL/TLS Zertifikat auf sicherem Weg an das Bundesverwaltungsamt übermitteln. Die staatliche Stelle generiert aus den Daten das Berechtigungszertifikat für den Diensteanbieter.

VfB

Das Berechtigungszertifikat ist ein CV-Zertifikat. CV Zertifikate besitzen eine besonders kompakte Form und benötigen erheblich weniger Speicherplatz als klassische X.509-Zertifikate. Sie können ausschließlich zur Authentisierung verwendet werden und enthalten deshalb weniger Informationen als die flexibleren X.509-Zertifikate. Ein Berechtigungszertifikat enthält u.a. folgende Angaben:

CV-Zertifikat

- Name, Anschrift und E-Mail Adresse des Diensteanbieters,
- Berechtigung zum Zugriff auf Datenfelder im nPA, z.B. Vor- und Nachname, Alter,
- Anbieterspezifische Kennung, die von der ausgegebenen Stelle vergeben wird; dies wird vom nPA zur Erzeugung von Pseudonymen (s.u.) benötigt,
- Zweck der Datenübermittlung,
- Angabe der zuständigen Datenschutzaufsichtsbehörde,
- Ablaufdatum des Zertifikats.

¹³ Das Berechtigungszertifikat entspricht dem Konzept des Zugriffszertifikats beim elektronischen Reisepass.

Die Berechtigungszertifikate haben eine Gültigkeit von 1-3 Tagen, da der nPA nicht in der Lage ist, Sperrlisten zu prüfen. Demgegenüber wird die Berechtigung zur Erhebung von Authentisierungsdaten einem Diensteanbieter für einen Zeitraum von bis zu drei Jahren erteilt. Folgezertifikate sollen im Rahmen dieser Gültigkeitsdauer beim Bundesverwaltungsamt online abrufbar sein.

Anhand eines generischen Szenarios werden zunächst der allgemeine Ablauf und dann die dabei verwendeten Konzepte und Protokolle beschrieben.

Allgemeiner Ablauf

Abbildung 11.9 verdeutlicht den allgemeinen Ablauf bei einer Online-Authentisierung mittels des nPA und zeigt die dabei involvierten Parteien, wobei wir hier noch auf die Einbindung eines eID-Servers verzichten. Darauf gehen wir weiter unten noch ein.

Diensteanfrage

Der Ausweisinhaber stellt über den Web-Browser seines Rechners eine Anfrage an einen Online-Diensteanbieter z.B. an *Online_Games* zur Nutzung eines Online Dienstes, z.B. zum Online-Erwerb eines Spiels *Game_X*, das erst ab 18 zu kaufen ist. Falls der Diensteanbieter im Vorfeld bereits ein Berechtigungszertifikat für seinen Dienst beim Bundesverwaltungsamt, wie oben beschrieben, beantragt und erhalten hat, wird er dieses Zertifikat an den Browser des Anfragers übermitteln. In dem Berechtigungszertifikat sind die Berechtigungen, die das Bundesverwaltungsamt dem Diensteanbieter *Online_Games* zur Durchführung des Dienstes *Game_X* prinzipiell erteilt hat, aufgelistet. In diesem speziellen Beispiel ist die Altersangabe von Bedeutung, so dass der Diensteanbieter beispielsweise die Berechtigung erlangt, eine Altersabfrage an den nPA zu stellen. Der nPA sieht dafür eine spezielle Funktion vor, so dass bei einer solchen Anfrage keine genauen Geburtsdaten übermittelt werden, sondern lediglich, wie in dem Beispiel erforderlich, die Anfrage auf Volljährigkeit mit ja oder nein beantwortet wird.

Ein Berechtigungszertifikat kann auch die Berechtigung zum Auslesen von Daten aus dem nPA enthalten. So ist in dem in Abbildung 11.9 skizzierten Szenario dargestellt, dass das Berechtigungszertifikat des Diensteanbieters den Zugriff auf die Namensfelder sowie das Alter prinzipiell ermöglicht.

kontrollierbare Datenfreigabe

Der Diensteanbieter übermittelt sein Berechtigungszertifikat an den Browser bzw. das Lesegerät des anfragenden Nutzers. Die nPA-Spezifikation fordert, dass dem Nutzer die wesentlichen Daten, die im Berechtigungszertifikat enthalten sind, optisch angezeigt werden. Der Nutzer ist damit in der Lage zu prüfen, für welchen Diensteanbieter und welchen Dienst das Zertifikat ausgestellt wurde, sowie welche Berechtigungen dem Diensteanbieter durch den Zertifikatsaussteller eingeräumt wurden. Die in dem Zertifikat festgelegten Zugriffsrechte können nun vom Benutzer explizit weiter eingeschränkt werden. Dieser Vorgang ist bei jeder Online-Authentisierung individuell

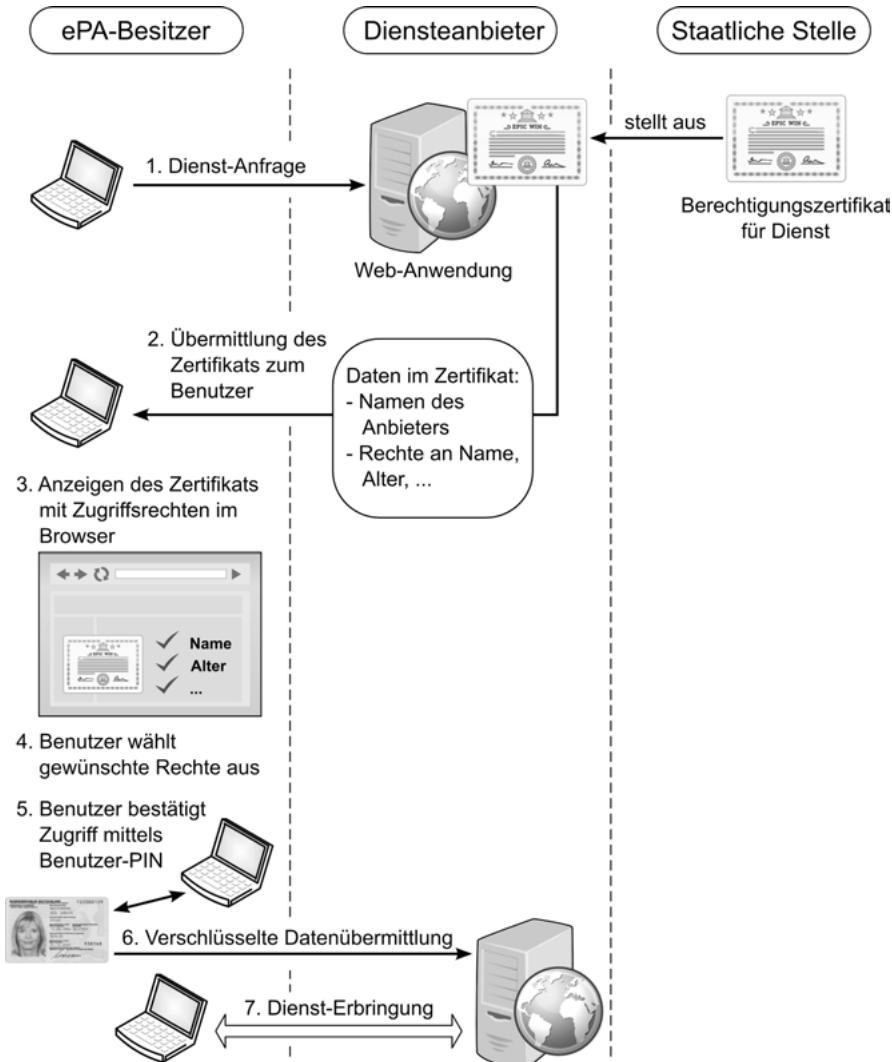


Abbildung 11.9: Nachweis der digitalen Identität mit dem nPA

durchzuführen. Im Beispiel des Online-Game-Kaufs könnte es beispielsweise sein, dass der Ausweisenutzer lediglich die Anfrage in Bezug auf Volljährigkeit zulassen, sonst aber keine persönlichen Daten übermitteln möchte. So verweigert der Benutzer in dem in Abbildung 11.9 skizzierten Szenario explizit den Zugriff auf die konkrete Angabe seines Alters, in dem er die Berechtigung entfernt (Haken wird entfernt).

Der Ausweisenhaber autorisiert den Zugriff auf die Daten in seinem Ausweis durch die Eingabe seiner 6-stelligen Benutzer-PIN. Durch die Eingabe der 6-stelligen PIN gibt der Benutzer somit explizit seine Einwilligung zu einem

Auslesevorgang und das Lesegerät kann prüfen, ob der Benutzer die PIN kennt und damit der rechtmäßige Inhaber des Personalausweises ist (falls die PIN nicht weitergegeben oder anderweitig offen gelegt wurde).

Die vom Diensteanbieter angefragten und vom Benutzer freigegebenen Daten werden aus dem nPA ausgelesen und an den Diensteanbieter zur Durchführung des Dienstes übermittelt.

Pseudonyme

Pseudonyme

Das Telemediengesetz schreibt vor, dass ein Diensteanbieter die Nutzung von Telemedien und ihre Bezahlung anonym oder unter Verwendung eines Pseudonyms zu ermöglichen hat, soweit dies technisch möglich und zumutbar ist. Laut Bundesdatenschutzgesetz bedeutet eine Pseudonymisierung, dass der Name und andere Identifikationsmerkmale ersetzt werden durch ein Kennzeichen, das den Zweck hat, die Bestimmung des Betroffenen ausschließen oder wesentlich zu erschweren.

Der nPA sieht zwei miteinander zu kombinierende Konzepte vor, um die Forderung nach Pseudomisierung zu erfüllen. Dies ist zum einen die anbieterspezifische, zufällige Kennung, die in dem Berechtigungszertifikat für den Diensteanbieter abgelegt wird. Zum anderen ist es die Anwendung *Restricted Identification*, die auf dem nPA implementiert ist. Bei der Authentisierung eines nPA-Inhabers mittels Pseudonym werden beide Konzepte kombiniert. Mittels der Restricted Identification-Anwendung wird aus einem personalausweisindividuellen, geheimen Schlüssel und der Kennung des Diensteanbieters ein bereichsspezifisches Kennzeichen berechnet. Dieses wird als Pseudonym dem Diensteanbieter übermittelt. Da unterschiedliche Diensteanbieter verschiedene Kennnummern besitzen, werden stets unterschiedliche Pseudonyme berechnet. Auf diese Weise soll verhindert werden, dass diensteanbieterübergreifend Nutzerprofile erstellt werden.

Anhand der bereichsspezifischen Kennung können Diensteanbieter zudem einen registrierten Nutzer wieder erkennen. D.h. die Restricted ID-Funktion des nPA erlaubt es, einen Personalausweis eindeutig wiederzuerkennen, ohne dass damit Ausweisdaten oder Daten über den Ausweisbesitzer gespeichert werden müssen¹⁴.

Verwendete Protokolle bei der Online Authentisierung

Abbildung 11.10 verdeutlicht die Protokolle, die bei einer Online-Authentisierung mittels des nPA verwendet werden.

Entsprechend dem oben beschriebenen Ablauf, übersendet der Diensteanbieter sein Berechtigungszertifikat und der Ausweisinhaber gibt durch die

¹⁴ Bem.: Die Verwendung der Seriennummer des Personalausweises in kommerziellen Anwendungen ist gesetzlich unzulässig.

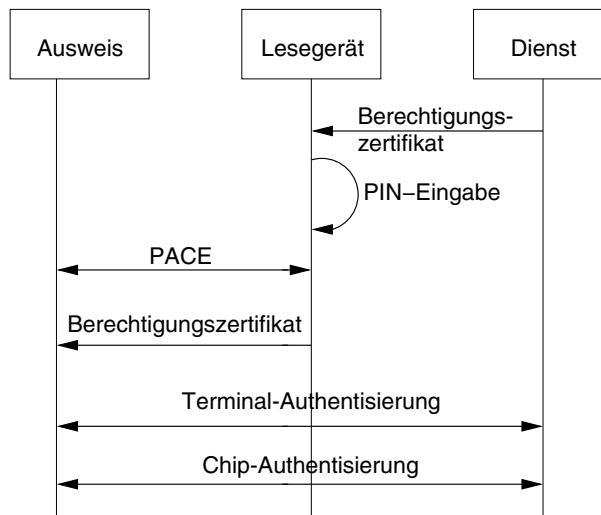


Abbildung 11.10: Online-Authentisierung mit dem nPA (Abbildung aus [20])

Eingabe seiner Benutzer-PIN die explizite Einwilligung zum Zugriff auf die Daten. Es ist nun durch den nPA zu prüfen, ob der Benutzer eine korrekte PIN eingegeben hat. Ist dies der Fall, so wird ein sicherer Kanal zwischen dem Ausweis und dem Lesegerät aufgebaut.

Zur Prüfung der Korrektheit der PIN und zum Aufbau einer verschlüsselten Verbindung zwischen Personalausweis und Lesegerät wird PACE verwendet. Nach der Eingabe der geheimen Benutzer-PIN am Lesegerät durch den Benutzer wird mittels PACE aus den öffentlichen DH-Schlüsseln der Partner (Ausweis-Chip und Lesegerät) ein gemeinsames Geheimnis berechnet. Aus diesem Geheimnis werden anschließend der Integritäts- und Sitzungsschlüssel für das Secure Messaging zwischen den Partnern abgeleitet. Zum verschlüsselten Datentransfer zwischen Chip und Lesegerät verwendet der nPA das AES-Verfahren. Nachdem über PACE gemeinsame Sitzungsschlüssel ausgehandelt wurden, überträgt das Lesegerät das Berechtigungszertifikat des Diensteanbieters zum Ausweis.

Um Tracking-Angriffe zu erschweren, verwendet auch der nPA dynamische UIDs beim Aufbau einer Kommunikationsverbindung zu einem Lesegerät.

Als nächsten Schritt führen der Chip des Ausweises und der Dienstanbieter ein Terminal-Authentisierungs-Protokoll (siehe Seite 527) durch, wobei hier der Dienstanbieter den Part des Terminals übernimmt und das Lesegerät lediglich als Gateway zwischen Chip und Dienstanbieter fungiert. Durch das Challenge-Response-Protokoll, das bei der Terminal-Authentisierung abgewickelt wird, weist der Dienstanbieter durch die Erstellung einer Signatur die Kenntnis seines privaten Schlüssels nach, der zu dem öffentlichen

PACE

dynamische UIDs

Authentisierung
des Dienste-
anbieters

Schlüssel passen muss, der im Berechtigungszertifikat enthalten ist. Der Chip prüft die Signatur, jedoch ist er nicht in der Lage, Sperrlisten abzufragen, um zu prüfen, ob das Berechtigungszertifikat noch gültig ist. Deshalb haben derartige Zertifikate nur eine sehr kurze Gültigkeit von max. 3 Tagen.

Chip-Authentisierung

Wird der Diensteanbieter vom Chip als authentisch erkannt, führen Chip und Diensteanbieter eine Chip-Authentisierung durch (siehe Seite 525). Basierend auf den dabei ausgetauschten DH-Werten berechnen der Chip und der Diensteanbieter ein gemeinsames MAC-Geheimnis sowie einen gemeinsamen, geheimen Sitzungsschlüssel K . Der Diensteanbieter kann anhand des vom Personalausweisaussteller signierten, im Chip abgelegten, öffentlichen Schlüssels des Chips dessen Authentizität prüfen.

Der Chip verschlüsselt die zu übertragenen Daten mit diesem Sitzungsschlüssel K und überträgt sie zum Diensteanbieter. Die Leserechte an den Chip-Daten sind durch das Konzept implizit an diesen Schlüssel gebunden, da die Daten mit diesem Schlüssel verschlüsselt und zum Service übermittelt werden. Der angeschlossene Rechner des Ausweisbesitzers fungiert hierbei nur als Gateway. Die verwendeten Schlüssel sollten dem Browser des Nutzers entsprechend nicht bekannt sein.

Anders als beim ePass erfolgt die Integritätsprüfung der Daten des Chips bei der eID Anwendung nicht unter Rückgriff auf die signierten Haschwerte im Document Security Object, sondern implizit über die Chip-Authentisierung¹⁵.

Schlüssel- und Zertifikatsmanagement

Die für die Nutzung der eID-Funktion notwendige PKI ist analog zu der PKI des elektronischen Reisepasses aufgebaut. Das BSI ist wiederum die Wurzelinstanz, die ein Zertifikat für den Personalausweishersteller ausgibt und dieses mit dem Root-Schlüssel signiert. Der Ausweishersteller, also die Bundesdruckerei, seinerseits signiert für jeden Personalausweis das Ausweis-individuelle, statische Diffie-Hellman Schlüsselpaar, das bei der Herstellung des nPA im Chip abgelegt wird.

Das BSI ist zudem die Wurzelinstanz für auszustellende Berechtigungszertifikate. Das Root-Zertifikat wird im Chip des nPA abgelegt. Die Wurzelinstanz autorisiert weitere staatliche Stellen zur Ausstellung von Berechtigungszertifikaten für Dienstanbieter. Ein Berechtigungszertifikat signiert ein variables DH-Schlüsselpaar des Diensteanbieters.

Bei der Online-Authentisierung wird während der Terminal-Authentisierung das Berechtigungszertifikat dem Chip vorgelegt. Dieser prüft die Echtheit

¹⁵ Bem.: Dies entspricht nicht den Empfehlungen der Spezifikationsdokumente zum ePass, die empfehlen, nach einer Chip-Authentisierung noch zusätzlich eine explizite Echtheitsprüfung der Chip-Daten durchzuführen.

des Zertifikats unter Nutzung des auf dem Chip abgelegten öffentlichen Schlüssels der Root-Instanz als Vertrauensanker für den Zertifikatsvalidierungspfad.

Ausweis App

Die Ausweis-App ermöglicht den Web-Service-orientierten Zugriff auf den nPA. Um die eID Funktion nutzen zu können, muss auf dem PC/Laptop des Nutzers ein frei verfügbares Software-Modul, die Ausweis App, installiert werden. Die Ausweis App implementiert die erforderlichen Schnittstellen und Protokolle, um zum einen mit dem Kartenleser und darüber mit dem Chip im nPA kommunizieren zu können. Zum anderen wickelt diese Client-Software die Kommunikationsprotokolle mit dem entsprechenden Modul auf der Server-Seite ab, die für die wechselseitige Authentisierung von Server und nPA (Terminal- und Chip-Authentisierung) sorgen. Zur Nutzung eines eID-basierten Anwendungsdienstes initiiert der Nutzer, wie üblich, über seinen Browser eine SSL/TLS-Verbindung zum Server des Dienstanbieters. Auf der Server-Seite wird analog ein Server-Modul benötigt.

Ausweis App

Die eCardAPI ist eine Middleware, die schichtenartig aufgebaut ist und die unterschiedliche Smartcard-Anwendungen, wie z.B. den nPA, die elektronische Gesundheitskarte oder auch die Jobkarte oder das Elster-Verfahren unterstützt. Auf der obersten Schicht, dem *Application Layer* befinden sich die verschiedenen Anwendungen, die das eCard-API-Framework für den Zugriff auf die eCards und damit verbundene Funktionen nutzen wollen. In dieser Schicht können auch weitere anwendungsspezifische Convenience-Schnittstellen existieren, in denen wiederkehrende Aufruffolgen in applikationsnahe Aufrufe gekapselt werden. Diese Schnittstellen sind jedoch derzeit nicht Gegenstand des eCard-API-Frameworks.

Schichten

Der *Identity-Layer* umfasst im eCard-Interface und im Management-Interface Funktionen für die Verwaltung und Nutzung elektronischer Identitäten sowie für das Management des eCard-API-Frameworks. Das eCard-Interface ermöglicht den Import von Zertifikaten sowie u.a. die Verschlüsselung und das Signieren von Dokumenten. Im Management-Interface existieren Funktionen für die Verwaltung von vertrauenswürdigen Identitäten u.a. von Chipkarten und Kartenlesegeräten.

Der *Service-Access-Layer* bietet insbesondere Funktionen für kryptografische Primitive und biometrische Mechanismen in Verbindung mit kryptografischen Token und umfasst das ISO24727-3-Interface. Dies ist eine Webservice-basierte Umsetzung des gleichnamigen Standards. Diese Schnittstelle enthält Funktionen, um (kryptografisch geschützte) Verbindungen zu Chipkarten herzustellen, um Chipkartenapplikationen zu verwalten, Daten zu lesen oder zu schreiben, kryptografische Operationen auszuführen sowie das entsprechende Schlüsselmaterial zu verwalten.

Chipkartenleser

Lesegerät

Um die Daten aus dem RFID-Chip des nPA auslesen zu können, werden Chipkartenlesegeräte benötigt, die je nach ihrem Einsatzumfeld unterschiedliche Ausprägungen besitzen können. Ein Chipkartenlesegerät wird auch Kartenterminal, oder kurz Terminal genannt. Das BSI hat in seiner Technischen Richtlinie TR-03119 Anforderungen an Kartenleser dreier unterschiedlicher Sicherheitskategorien beschrieben. Hersteller von Kartenterminals können ihre Produkte in Bezug auf Konformität mit der Richtlinie prüfen und zertifizieren lassen. Zertifizierte Kartenleser erhalten ein Prüfsiegel, das auf dem Produkt aufgebracht werden kann.

Klassen

Es werden drei Kategorien von Kartenlesern, der Basis-, der Standard- und der Komfort-Kartenleser, unterschieden.

Cat-B

- Der Basis-Chipkartenleser (Cat-B) bietet lediglich eine einfache Funktionalität. Er soll laut BSI Richtlinie vordringlich im Heimbereich eingesetzt werden können, um e-Government Dienste des nPA wie beispielsweise Dienste im Zusammenhang mit der Rentenversicherung zu unterstützen. Im e-Commerce-Umfeld kann der einfache Leser eingesetzt werden, um beispielsweise die Altersverifikation durchzuführen, oder aber auch für den Wohnsitz- und Identitätsnachweis bei Shopping-Anwendungen oder aber auch zum eTicketing.

Ein Basis-Leser muss verpflichtend eine Schnittstelle zum Host-Rechner anbieten und eine kontaktlose Schnittstelle nach ISO/IEC 14443. Alle anderen Funktionen sind für den Basis-Leser optional. Insbesondere muss dieser Leser keine sichere PIN-Eingabe mit dem PACE-Protokoll unterstützen. Dies eröffnet natürlich Verwundbarkeiten durch beispielsweise einen Key-Logger und andere Schadsoftware auf dem angeschlossenen Rechner des nutzenden Bürgers, so dass die die PIN-Eingabe mitgelesen werden könnte. Aber auch wenn diese Daten an einen Angreifer übermittelt würden, benötigt dieser für einen erfolgreichen Maskierungsangriff neben der PIN auch noch den nPA als physische Karte. Durch die Zweifaktor-Authentisierung wird ein einfaches Maskieren also deutlich erschwert.

Cat-S

- Der Standard-Leser (Cat-S) muss höheren Sicherheitsanforderungen genügen. Ein solcher Kartenleser muss mindestens über ein PIN-Pad zur sicheren PIN-Eingabe verfügen. Die Unterstützung eines Displays oder der elektronischen Signatur ist für den Standard-Leser jedoch optional. Anwendungen, die einen Standard-Leser erfordern, sind beispielsweise eID-Anwendungen mit erhöhten Sicherheitsanforderungen, wie man sie im e-Governmentumfeld oder aber auch im Payment-Umfeld antrifft.

Cat-K

- Die dritte Kategorie ist der Komfort-Leser (Cat-K), der eine Vielzahl von Anwendungen bedienen kann. Er besitzt mindestens ein PIN-Pad

zur sicheren PIN-Eingabe sowie ein Display zur Darstellung von 2x16 alphanumerischen Zeichen. Der Komfort-Leser wird für Anwendungen mit erhöhtem Sicherheitsbedarf benötigt; er unterstützt auch die qualifizierte elektronische Signatur, sowie neben der Schnittstelle für kontaktlose Smartcards auch die Schnittstelle für kontaktbehaftete Karten gemäß ISO/EIC 7816.

eID-Server

Damit ein Diensteanbieter seine Kunden über die eID des nPA authentisieren kann, muss eine Software installiert sein, die die Zusammenarbeit mit der Ausweis-App auf dem Rechner des Kunden abwickelt und insbesondere auch die Terminal-Authentisierung, also die Authentisierung des Diensteanbieters durchführen kann. Hierfür benötigt er wie oben gesehen auch sein Berechtigungszertifikat. Damit nicht jeder Diensteanbieter diese Protokolle selber umsetzen muss, wird diese Aufgabe durch einen eID-Server erbracht. Dieser muss die technische Richtlinie TR03130 des BSI für eID-Server erfüllen. Er kann sowohl vom Betreiber selbst als auch von einer Trusted Third Party betrieben werden. Im Folgenden wird die Einbindung des eID-Servers in den Ablauf grob skizziert. Die Technische Richtlinie TR-03130 des BSI beschreibt detailliert die Schnittstellen eines eID Servers und dessen Funktionen.

eID-Server

Greift ein Bürger auf eine Web-Seite eines Diensteanbieters zu, um einen Dienst zu nutzen, der eine nPA-basierte Authentisierung erfordert, so leitet der Diensteanbieter diese Anfrage zusammen mit seinem Berechtigungszertifikat mit TLS geschützt an den eID-Server weiter. Der eID-Server etabliert eine Verbindung zum Rechner des Bürgers, der authentisiert werden soll. Über das Versenden eines entsprechenden Objekt-Tag wird vom eID-Server ein Browser-Plug-in auf diesem Rechner des Bürgers angesteuert, wodurch das Starten der Ausweis App initiiert wird. Die Ausweis App etabliert eine eigene TLS-gesicherte Verbindung zum eID-Server. Über diese gesicherte Verbindung sendet der eID-Server das Berechtigungszertifikat des Diensteanbieters und wickelt mit der Ausweis App die Authentisierung ab. Nach erfolgreicher Authentisierung des Benutzers mittels PACE und PIN auf der Nutzer-Seite und einer erfolgreichen Chip- und Terminalauthentisierung werden über die Ausweis App die gewünschten Daten aus dem nPA ausgelesen. Diese Daten werden über den sicheren TLS-Kanal zum eID Server geleitet. Dieser reicht die Daten ebenfalls verschlüsselt und signiert über die etablierte TLS-Verbindung an die Web-Anwendung bzw. den Browser des Diensteanbieters weiter. Dieser kann die Daten prüfen und den Dienst dem Bürger zur Verfügung stellen, wenn die Prüfung erfolgreich verläuft. Auf dem eID-Server werden die Daten nach Abschluss der Transaktion wieder vernichtet.

Anwendungen

Der nPA kann überall dort zum Einsatz kommen, wo die Vorlage eines gültigen Personalausweises, also das persönliche Erscheinen, benötigt wird, wie zum Beispiel bei Antragsprozessen, oder der Einsichtnahme in laufende Verfahren, in Daten beim Finanzamt, Rentenstatus etc. Weitere Einsatzmöglichkeiten für die Authentisierungsfunktion des nPA sind im Bereich des Jugendschutzes (z.B. Altersabfrage bei der Ausleihe von Videofilmen) oder dem eGovernment. Neben authentisierten Downloads (z. B. die elektronische Ausleihe von Büchern) ist auch ein authentischer Upload möglich, z.B. in der Erwachsenenbildung.

Sicherheit des nPA

Nachfolgend werden einige technische Punkte im Hinblick auf Sicherheit diskutiert.

Sicherheitsanforderungen

Das BMI hat ein Sicherheitsrahmenkonzept definiert, das die Anforderungen an ein Sicherheitskonzept für den nPA über dessen gesamten Lebenszyklus abdeckt, also von der Beantragung bis zur Vernichtung. Das Rahmenkonzept enthält Vorgaben für die Ausweishersteller sowie Empfehlungen für die Personalausweisbehörden. Wesentliche Anforderungen entsprechend des nPA Konzeptes sind¹⁶:

1. Es muss ausgeschlossen werden, dass der nPA physisch so manipuliert wird, dass eine andere Person die Identität des Ausweisinhabers annehmen kann. Darüber hinaus muss auch eine Manipulation des Chips ausgeschlossen werden, damit eine andere Person nicht auf elektronischem Weg die Identität des Inhabers annehmen kann.
2. Jede Form von gefälschten oder manipulierten elektronischen Personalausweisen bzw. Teilen davon müssen erkannt werden können.
3. Das Abhören der Kommunikation zwischen Diensteanbieter und nPA muss ausgeschlossen werden können. Die Verschlüsselung muss auf geeignete kryptographische Verfahren zurückgreifen, die ausreichende Schlüssellängen aufweisen.
4. Lesegeräte für den nPA müssen so sicher sein, dass ein Eingriff durch Dritte in den Kommunikationsprozess mit dem nPA ausgeschlossen werden kann.
5. Der Zugriff auf die personenbezogenen Daten im elektronischen Personalausweis darf nur mit Einwilligung des Benutzers möglich sein. Für den Zugriff auf die biometrischen Daten bei hoheitlichen Kontrollen ist

¹⁶ Die aufgeführten Anforderungen sind in Teilen der Spezifikation entnommen.

ein hoheitliches Berechtigungszertifikat erforderlich, dass diese Zugriffe autorisiert.

Erfüllung der Anforderungen

Nachfolgend wird diskutiert, welche Maßnahmen in dem bestehenden Sicherheitskonzept beitragen, die oben gestellten Anforderungen zu erfüllen.

1. Kopierschutzmaßnahmen, Anti-Kloning und Abwehrmaßnahmen gegen Identitätsdiebstahl:

Durch den Einsatz der Chip-Authentisierung kann erkannt werden, ob ein Chip geklont wurde. Der öffentliche DH-Schlüssel des Chips wird signiert im Chip abgelegt, so dass dessen Modifikation erkannt werden kann. Da bei der Online-Authentisierung das Security-Object zur Integritätsprüfung nicht zur Verfügung steht, ist eine explizite Integritätsprüfung der Daten im Chip jedoch nicht möglich.

Zur Abwehr von Identitätsdiebstählen insbesondere beim Online-Authentisieren dient die 2-Faktor-Authentisierung, die die Kenntnis der Benutzer-PIN erfordert. Über den Fehlbedienzähler wird verhindert, dass eine PIN durch Brute-Force Angriffe geraten wird. Nicht verhindert werden können Denial-of-Service Angriffe, bei denen ein Angreifer die kontaktlose Schnittstelle des Chips nutzt, um als Lesegerät einen Verbindungsaufbau zu provozieren. Durch die mehrmalige Eingabe falscher PINs kann er dafür sorgen, dass die Karte suspendiert wird. Das Konzept des verzögerten Blockierens verhindert das direkte Blockieren, da für weitere Aktivitäten zunächst die Karten-PIN eingegeben werden muss.

Eine weitere Abwehrmaßnahme gegen Identitätsdiebstahl stellen die Sperrlisten dar, die für Dienstanbieter Informationen über verlorene oder abhanden gekommene Ausweise zur Verfügung stellen.

2. Abwehrmaßnahmen gegen Chip-Manipulationen:

Laut Spezifikation erfordert der Schreibzugriff die Vorlage eines speziellen Berechtigungszertifikats, das nur Personalausweisbehörden ausgestellt wird.

3. Vertrauliche Kommunikation der nPA Daten:

Durch die Abwicklung des Terminal-Authentisierungs- und Chip-Authentisierungsprotokolls werden geheime Sitzungsschlüssel zwischen dem Dienstanbieter und dem Chip etabliert. Zur Verschlüsselung wird der AES mit 128 Bit Schlüsseln verwendet, was beim heutigen Stand der Technik ein ausreichend gutes Sicherheitsniveau bietet.

4. Abwehr unberechtigter Datenzugriff

Alle Stellen, die Zugriff auf die Daten im Ausweis erhalten möchten, benötigen ein Berechtigungszertifikat. Hoheitliche Stellen, die zur

Durchführung einer Personenkontrolle auch auf biometrische Daten zugreifen müssen, benötigen ein spezielles Berechtigungszertifikat, das diesen Zugriff gestattet. Zudem müssen sie beim Zugriff die Kenntnis der auf dem Ausweis aufgedruckten Karten-PIN nachweisen. Ein Zugriff auf Daten des nPA bei der eID-Authentisierung erfordert die Autorisierung durch den Benutzer mit der Bestätigung durch dessen Benutzer-PIN. Hierfür ist wesentlich, dass zum einen die Anzeige-Funktion zum Setzen der Zugriffsrechte korrekt arbeitet und die Daten korrekt an den Chip übermittelt werden. Weiterhin ist wesentlich, dass der Chip selber die Zugriffe korrekt gemäß der festgelegten Berechtigungen durchführt.

5. Vertrauenswürdige Protokollabwicklung

Zur Abwicklung der Protokolle ist das Installieren einer Ausweis App auf dem PC/Laptop des Nutzers notwendig. Ein unzulässiges Durchsickern von vertraulichen Informationen zwischen diesem Client-Modul und der sonstigen Anwendungen auf dem Benutzer-Rechner muss selbstverständlich verhindert werden. Andernfalls könnte ein mit einem Trojaner infizierter Nutzer-Rechner die vertraulichen Identifizierungsinformationen aus dem nPA unbemerkt an unautorisierte Dritte weiterleiten. Die verfügbaren Ausweis Apps für die verschiedenen Betriebssysteme werden vom BSI in Bezug auf Konformität erstellter Richtlinien für die App geprüft.

6. PIN-Eingabe-Problematik

Das vorliegende Konzept erlaubt auch die Nutzung von Basis-Kartenlesern, so dass die Benutzer-PIN direkt an der Tastatur des Rechners des Nutzers eingegeben werden muss. Dies ist angesichts der Vielzahl der virenverseuchten PCs/Laptops mit versteckt installierten KeyLoggern natürlich ein Sicherheitsproblem. PINs können abgefangen und missbraucht werden. Abhilfe schafft der Einsatz von zertifizierten Standard- oder Komfort-Kartenlesern, die über eine integrierte Tastatur zur sicheren Eingabe der PIN verfügen.

11.3 Universal Second Factor Authentication

Ziele

Das Ziel des Universal Second Factor Protokolls (U2F) ist es, das übliche Login via Nutzerkennung und Passwort um einen weiteren Faktor zu erweitern, um einen stärkeren Schutz gegen Man-in-the-Middle Angriffe zu bieten. Gleichzeitig soll U2F eine einfach zu nutzende Lösung bieten. Es soll ein Einsatz möglich sein, ohne dass ein spezieller Treiber konfiguriert oder Software geändert werden muss. Ziel ist weiterhin, dass der Nutzer über das Konzept nicht an einen speziellen Rechner gebunden ist (universell).

Das U2F-Protokoll wurde in der FIDO Alliance standardisiert. Die FIDO (Fast IDentity Online) Alliance¹⁷ ist eine 2012 von den Firmen PayPal, Infineon, Lenovo, Nok Nok Labs, Validity Sensors und Agnitio gegründete non-Profit Organisation. Diese hat sich das Ziel gesetzt, technische Spezifikationen für die offene, skalierende und interoperable Nutzer-Authentisierung im Web zu entwickeln, um die Abhängigkeit von reinen Nutzerkennung/Passwort Verfahren zur Nutzer-Authentisierung im Web zu reduzieren. Mittlerweile haben sich sehr viele Firmen der Allianz angeschlossen, wie Google, Microsoft, NXP, Oberthur oder auch MasterCard.

FIDO

Für den im U2F-Protokoll geforderten zweiten Authentisierungs-Faktor sieht der FIDO-Standard verschiedene Formfaktoren vor, die von USB-Dongles, Dongles mit Sicherheitschips, bis hin zu einer reinen Software-Lösung reichen können. Da eine reine Softwarelösung jedoch nur ein schwaches Sicherheitsniveau bietet, wird eine Hardware-basierte Lösung empfohlen, die ein Zusatz-Gerät beispielsweise in Form eines USB- oder NFC-Dongles sein kann. Das Dongle kann als Vertrauensanker ein Secure Element, also einen Sicherheitschip, integrieren, der zur Erzeugung und Speicherung von Schlüsseln und zum Verschlüsseln und Signieren von Daten genutzt wird. Das Dongle ist jedoch ansonsten ein sehr einfaches Device, das nicht programmierbar und damit auch nicht änderbar ist.

2nd Factor

Langfristig ist eine Integration in Smartphones angestrebt, um auf ein Zusatz-Gerät verzichten zu können.

Das U2F-Einsatzszenario umfasst stets einen U2F-fähigen Web-Service, bei dem sich der Nutzer einloggen möchte, einen U2F-fähigen Browser auf einem Gerät des Nutzers und das U2F-Device. Für eine Nutzung sind zwei Schritte erforderlich:

U2F-set-up

1. Eine *einmalige Registrierung* an dem Web-Dienst, der die U2F-Authentisierung unterstützt und
2. ein *erweitertes Login*, bei dem zusätzlich zu den bereits vorhandenen Credentials, wie Benutzerkennung und Passwort, noch die U2F-Authentisierung durchgeführt wird. Es ist jedoch auch ein anonymes Login-in prinzipiell möglich, wenn der Nutzer auf dem Web-Service keinen personalisierten Account benötigt.

¹⁷ vgl. <http://www.fidoalliance.org/>

11.3.1 Registrierung eines U2F-Devices

Registrierung

Ausgangspunkt ist, dass der Nutzer entweder auf dem Web-Server bereits eine Kennung besitzt und ein Passwort (oder ein anderes Credential) für die Authentisierung beim Server festgelegt sind, oder dass der Nutzer anonym kommunizieren möchte und der Server dies unterstützt. Die Registrierung startet im Fall einer nicht-anonymen Kommunikation mit einem normalen User-Login über Kennung und Passwort (bzw. vereinbartem Credential). Der allgemeine Protokollablauf bei der Registrierung eines U2F-Devices für den Account eines Benutzers Alice ist in Abbildung 11.11 zusammengefasst.

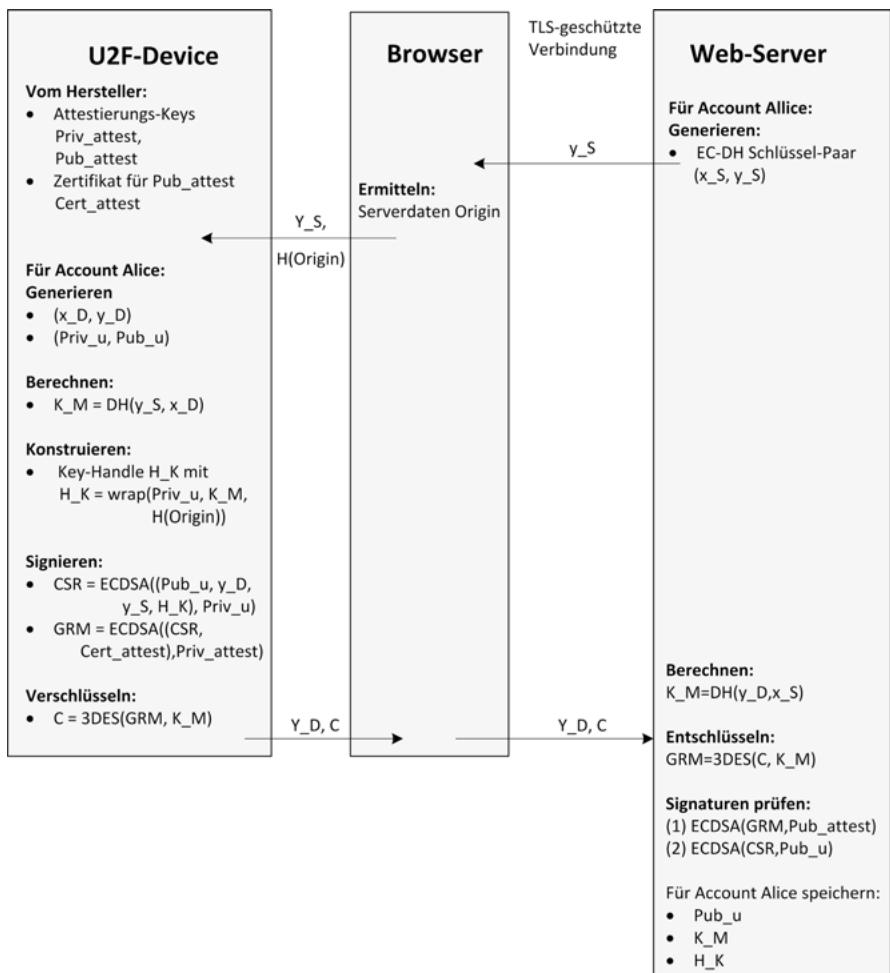


Abbildung 11.11: Ablauf der Registrierung eines U2F-Devices für den Nutzer-Account Alice

1. Aktionen des Servers:

- Nachdem im Fall eines nicht-anonymen Logins das Passwort (bzw. die entsprechenden Credentials) erfolgreich überprüft worden ist, generiert der Server ein EC-DH Schlüsselpaar (x_S, y_S) für den Web-Account. Hierbei bezeichnet y_S den öffentlichen Schlüssel, der durch einen Punkt auf der zugrundeliegenden elliptischen Kurve E definiert ist, und x_S ist der zugehörige private Schlüssel, der gemäß ECC ein skalarer Wert ist. Die zugrundeliegende elliptische Kurve E ist die NIST P256 Kurve.
- Der Server sendet einen Javascript-Call an den Client-Browser, in dem die Erzeugung eines öffentlichen Schlüssels durch das U2F-Device für das Nutzerkonto initiiert wird. Mit der Nachricht schickt er auch seinen öffentlichen EC-DH-Schlüssel y_S mit.
- Die Verbindung zwischen Server und Browser ist per SSL/TLS abgesichert.

EC-DH

2. Aktionen des Browsers:

- Der Browser ermittelt Informationen über die Ursprungs-Domäne $Origin$ des Servers und berechnet davon einen Hashwert $H(Origin)$.
- Der Browser sendet die Anfrage des Servers mit dessen öffentlichen Schlüssel y_S sowie der Information über dessen Ursprung $H(Origin)$ an das U2F-Device, also die Nachricht

$$y_S, H(Origin).$$

3. Aktionen des U2F-Devices:

- Das U2F-Device generiert ebenfalls ein EC-DH Schlüsselpaar (x_D, y_D) für den Web-Account, sowie zusätzlich noch ein EC-DSA-Schlüsselpaar $(Priv_U, Pub_U)$.
- Das Device berechnet unter Nutzung des EC-DH-Verfahrens einen gemeinsamen Schlüssel K_M mit dem Web-Server wie folgt:

$$K_M = P2AES(DH(y_S, x_D)).$$

Mit EC-DH wird ein Punkt auf der elliptischen Kurve berechnet. P2AES ist eine Key-Derivation Funktion, die aus der X-Koordinate des Punktes einen AES-Schlüssel¹⁸ ableitet. Dazu werden die 128 Bit höchstwertigen

¹⁸ Bemerkung: In den ersten Versionen des U2F Protokolls wurde noch das 3DES Verfahren verwendet, dies ist mittlerweile durch AES ersetzt worden.

Bit der X-Koordinate extrahiert und in einem Big-Endian Byte-Array gespeichert.

- Pro Account speichert das U2F-Device die Schlüssel $Priv_U$, K_M und den Hashwert $H(Origin)$ in Registern ab. Auf diese Weise werden die Schlüssel an den identifizierten Web-Server gebunden. Mittels eines *Key-Handles* H_K werden die entsprechenden Datensätze identifiziert. Dabei zeigt der Handle entweder auf ein on-device Register, das die Daten speichert, oder es enthält die Daten in einem Device-spezifischen Format:

$$H_K = \text{wrap}(Priv_U, K_M, H(Origin)).$$

Die wrap-Funktion kann auch eine Verschlüsselung sein, so dass die Informationen als Key-Blob (siehe weiter unten, Anmerkungen zum internen Device Key) auch außerhalb des Devices gespeichert werden können.

- Der öffentliche Schlüssel Pub_U des U2F-Device wird in einer spezifischen Datenstruktur, dem *Certificate Signing Request (CSR)* abgelegt. Die Datenstruktur enthält zusätzlich auch den öffentlichen EC-DH-Schlüssel y_D des U2F-Device, den öffentliche EC-DH-Schlüssel y_S des Web-Servers und das Key-Handle H_K . Die CSR Struktur ist vergleichbar mit einem X.509 Zertifikat und wird durch das Device selbst-signiert, also mit dem privaten EC-DSA-Schlüssel $Priv_U$.
- Das U2F-Device erzeugt eine *Registration Message (GRM)*, wie folgt:

$$GRM = ECDSA_{Priv_{attest}}(CSR, Cert_{attest}).$$

Die Datenstruktur GRM ist eine mit einem so genannten *Attestierungsschlüssel* $Priv_{attest}$ signierte Nachricht, mit der die Informationen im CSR attestiert werden und die auch den zur Prüfung notwendigen öffentlichen Schlüssel in einem Zertifikatformat mitliefert.

Dazu wird auf jedem Chip eines U2F-Devices vom Chip-Hersteller ein Attestierungsschlüssel $Priv_{Attest}$ zusammen mit einem Zertifikat für den zugehörigen öffentlichen Schlüssel Pub_{Attest} gespeichert. Eine mittels des Attestierungsschlüssels $Priv_{Attest}$ erstellte Signatur einer Nachricht eines U2F-Devices D bestätigt, dass das Device D genuin, also entsprechend der Spezifikation hergestellt ist. Der Attestierungsschlüssel $Priv_{Attest}$ wird fest in den Chip, also des Secure Elements eingebrannt (on-chip).

- Da die Verbindung zwischen dem Browser und dem U2F-Device ggf. unsicher ist, z.B. bei einer NFC-Verbindung, wird die Datenstruktur GRM mit dem gemeinsamen geheimen Schlüssel K_M verschlüsselt.

- Das U2F-Device sendet seinen öffentlichen DH-Schlüssel y_D sowie die verschlüsselten GRM-Daten, $C = AES_{K_M}(GRM)$, zurück an den Browser, also

$$y_D, C = AES_{K_M}(GRM).$$

4. Aktionen des Browsers:

- Der Browser leitet die Daten y_D und C direkt an den Server weiter. Er dient hier also nur als Gateway.

5. Aktionen des Servers:

- Der Server berechnet mit y_D und seinem geheimen EC-DH-Schlüssel x_S ebenfalls den gemeinsamen Schlüssel $K_M = P2AES(DH(y_D, x_S))$.
- Er entschlüsselt GRM, also

$$GRM = AES_{K_M}(C)$$

und prüft die Attestierungssignatur des Herstellers unter Nutzung des öffentlichen Hersteller-Schlüssels im Zertifikat, das in GRM enthalten ist.

- Er verifiziert die Signatur des CSR-Zertifikats mit dem öffentlichen Device Schlüssel Pub_U , der als Bestandteil der CSR-Datenstruktur im GRM mitgeliefert wurde.
- Wenn die Signaturen korrekt validiert werden konnten, speichert der Server den öffentlichen Schlüssel Pub_U aus dem CSR-Zertifikat, sowie den K_M und den Key-Handle H_K , der ebenfalls Bestandteil des CSR ist, zusammen mit dem Nutzer-Account ab.

Um auch besonders preiswerte U2F-Devices mit einem sehr kleinen oder gar keinem sicheren Speicherbereich nutzen zu können, sieht die FIDO-Spezifikationen auch vor, verschlüsselte private EC-DSA Schlüssel $Priv_U$ des U2F-Device als Key-Blob ausserhalb des Secure Elements des Device im internen oder externen Speicher abzulegen oder aber auch als Blob auf dem Server zu speichern.

11.3.2 Login beim Web-Dienst

Zunächst erfolgt, falls es kein anonymes Login ist, eine Prüfung von Kennung und Passwort wie üblich. Der allgemeine Protokollablauf ist in Abbildung 11.12 zusammengefasst.

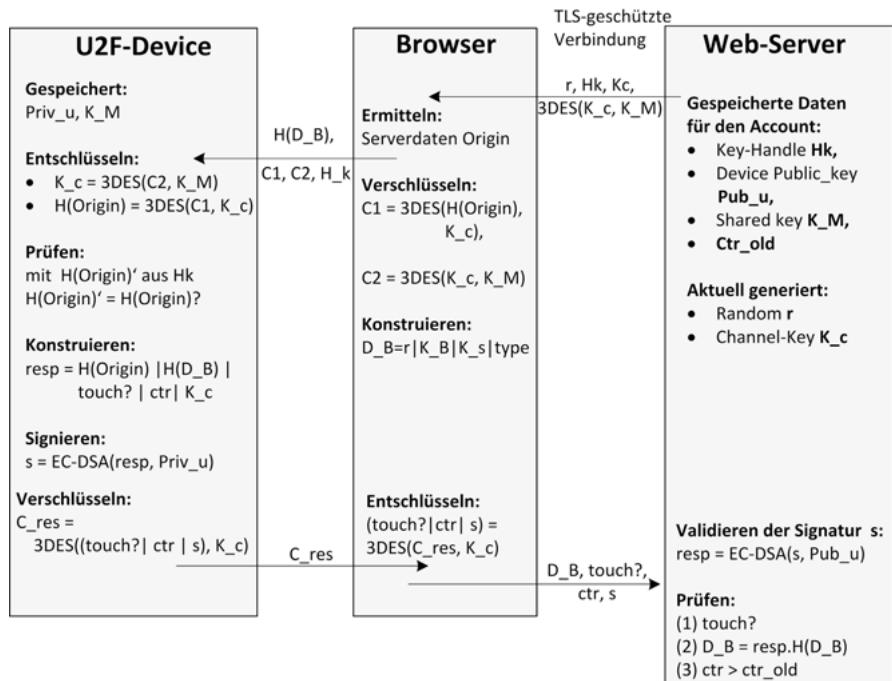


Abbildung 11.12: Ablauf eines U2F-basierten Logins

1. Aktionen des Servers: Der Web-Server sendet folgende Daten per JavaScript Call an den Browser:

- eine Challenge r ,
- den Key-Handle H_K ,
- einen neu generierten Channel-Key K_C , der in einem späteren Schritt für die sichere Kommunikation zwischen Browser und U2F-Device genutzt wird, und
- noch einmal separat den mit dem Shared Key K_M verschlüsselten K_C , also $\text{AES}_{K_M}(K_C) = \text{AES}(K_C, K_M)$.
- Ggf. wird der Key-Blob mit dem privaten Schlüssel des Devices ebenfalls zum Browser übertragen.

2. Aktionen des Browsers:

- Der Call veranlasst den Browser automatisch das angebundene U2F-Device zu lokalisieren. Ein USB-Dongle muss eingesteckt sein.

- Der Browser ermittelt die Ursprungsinformation $Origin$ des Servers, hasht diese zu $H(Origin)$ und verschlüsselt das Ergebnis mit dem Channel-Key K_C .
- Der Browser konateniert das TLS/SSL-Zertifikat K_S des Servers, mit der TLS-Session ID K_B der Verbindung zwischen dem Server und dem Browser (falls vorhanden) und mit der Challenge r zu einem Datensatz $D_B = r | K_B | K_S$, den *Browser-Daten*.
- Diese Browser-Daten D_B werden gehasht und zusammen mit der verschlüsselten Domäneninformation, dem mittels K_M verschlüsselten K_C und dem Key-Handle H_K an das U2F-Device gesandt, also

$$H(D_B), C1 = AES_{K_C}(H(Origin)), C2 = AES_{K_M}(K_C), H_K.$$

3. Aktionen des U2F-Devices:

- Das U2F-Device entschlüsselt mit seinem K_M den Channel-Key K_C und damit die gehaschten Domänendaten, also

$$K_C = AES_{K_M}(C2) \text{ und } H(Origin) = AES_{K_C}(C1).$$
- Das Device extrahiert aus dem Key-Handle H_K die gehaschten Server-Ursprungsdaten $H(Origin)'$ und vergleicht diese mit den zuvor entschlüsselten Daten, um einen potentiellen Man-in-the-Middle Angriff zu erkennen, also

$$H(Origin)' \stackrel{?}{=} H(Origin).$$

- Falls der Server einen Präsenznachweis des Nutzers erfordert (das ist optional), muss dieser einen Knopf auf dem Dongle drücken, bzw. einen Touchsensor aktivieren, um den Besitz des Dongles nachzuweisen. Bei einem NFC-fähigen U2F-Device reicht ein Tapping des Devices.
- Die entschlüsselten Ursprungsdaten $H(Origin)$ werden mit einem Flag, das die Aktivierung des Dongles repräsentiert (touch) konateniert, sowie auch mit den gehaschten Browser-Daten $H(D_B)$, die der Browser übermittelt hat, einem Counter-Wert ctr und dem Channel-Key K_C . Der ctr -Wert zählt die Anzahl der durchgeföhrten Signieroperationen. Dies ist später für den Web-Server eine hilfreiche Information, um zu entscheiden, ob es möglicherweise einen Clone des U2F-Device gibt.

Das Ergebnis dieser Konkatenationen ist die *Response* des U2F-Device, also

$$resp = H(Origin) \mid H(D_B) \mid touch? \mid ctr \mid K_C.$$

- Die berechnete Antwort $resp$ wird abschließend mit dem privaten Schlüssel $Priv_U$ signiert, also

$$s = ECDSA_{Priv_U}(resp).$$

Ggf. muss das U2F-Device dazu erst den Key-Blob unter Nutzung seines internen U2F-Schlüssels K_{intern} entschlüsseln, um Zugriff auf den privaten Signierschlüssel $Priv_U$ zu erhalten. Der Key-Blob wird dazu vom Web-Server zusätzlich zur Challenge an den Browser übertragen, der den Blob an das Device weiterleitet.

- Das U2F-Device erstellt eine mit dem Channel-Key K_C verschlüsselte Antwortnachricht C_{res} an den Browser, indem der $touch$ -Wert, der ctr -Wert und die signierte Response s konkateniert werden, also

$$C_{res} = AES_{K_C}(touch? \mid ctr \mid s).$$

- Der Kryptotext C_{res} wird an den Browser zurück gesandt.

Da die ursprüngliche Challenge r des Web-Servers in den gehashten Browser-Daten $H(D_B)$ enthalten ist, und diese Daten Bestandteil der Antwort $resp$ sind, die das Device signiert und als s in die verschlüsselte Antwortnachricht einbindet, wird die Challenge des Servers mittels eines asymmetrischen CR-Verfahrens beantwortet.

4. Aktionen des Browsers:

- Der Browser entschlüsselt C_{res} , also

$$(touch? \mid ctr \mid s) = AES_{K_C}(C_{response})$$

Da der Browser in diesem Schritt den Schlüssel K_C verwendet, kann er auch indirekt noch einmal prüfen, ob das U2F-Device und der Web-Server mit den gleichen Daten agieren, oder ob sich ein Man-in-the-Middle dazwischen geschoben hat.

- Der Browser leitet über die gesicherte SSL/TLS Verbindung den $touch$ -Wert, den ctr -Wert, die signierte Response s sowie die Browser-Daten D_B , die er selber für diese Kommunikation gespeichert hatte, an den Server weiter, also

$$BrowserResponse = D_B, touch?, ctr, s.$$

5. Aktionen des Servers:

- Der Server validiert die Signatur s mit dem öffentlichen Schlüssel Pub_U , also $resp = ECDSA_{Pub_U}(s)$, wobei gilt:
$$resp = H(Origin) \mid H(DB) \mid touch? \mid ctr \mid K_C.$$
- Er prüft den *touch*-Wert, d.h., ob eine physische Aktivierung des Dongles erfolgt ist, und
- überprüft, ob die Browser-Daten DB in der Nachricht *BrowserResponse* mit den in der Response $resp$ abgelegten gehaschten Browser-Daten $H(DB)$ übereinstimmen. Da in die Browser-Daten DB auch die Challenge r eingegangen ist, kann der Server an dieser Stelle auch einen Frischenachweis durchführen, also dass die signierte Antwort tatsächlich eine Antwort auf seine Challenge r ist.
- In der Antwort vom U2F-Device ist, falls diese Option genutzt wird, auch die Information enthalten, ob die aktuelle TLS-Session ID K_B genutzt wurde, und der Server damit prüfen kann, dass kein Man-in-the-Middle Angriff aufgetreten ist.
- Schließlich wird auch der *ctr*-Wert überprüft. Dazu hat der Server den alten *ctr*-Wert ctr_{old} abgespeichert und vergleicht den neuen damit:

$$ctr > ?$$

Sollte der neue Wert ctr kleiner als ctr_{old} sein, dann hat ein zweites U2F-Device das gleiche Schlüsselpaar in Benutzung und es könnte sich um ein geclontes U2F-Device handeln, mit dem das Login versucht wird. In diesem Fall wird die Authentisierung abgebrochen.

11.3.3 Sicherheitsbetrachtungen

Ziel des U2F-Konzepts war es, Man-in-the-Middle-Angriffe und Phishing-Angriffe zu erschweren, aber auch eine einfache, unkomplizierte Nutzbarkeit zu gewährleisten. Der zweite Authentisierungsfaktor, das U2F-Token, ist in der ersten Role-out-Stufe sehr einfach gehalten, ohne Display oder Tastatur und auch ohne Datenspeicher für Anwenderdaten. In einer nächsten Ausbaustufe ist eine Integration in Smartphones vorgesehen.

Sicherheit

Das U2F-Konzept wird nachfolgend unter verschiedenen Blickwinkeln, insbesondere Sicherheitsblickwinkeln analysiert. Da auch der digitale Personalausweis (nPA) mit seiner eID-Funktion darauf abzielt, eine höhere Sicherheit bei der Authentisierung bei Web-Anwendungen zu ermöglichen, vgl. Abschnitt 11.2.2, versuchen wir anschließend dort, wo es sinnvoll er-

scheint, einen Vergleich zwischen den Konzepten der U2F-Authentisierung und der Online-Authentisierung mittels der eID-Funktion des nPA durchzuführen.

Authentizität

Authentizität

Die Authentisierung des U2F-Devices durch den Web-Server basiert zum einen auf einer authentischen Registrierung, bei der im Wesentlichen die Echtheit des Device anhand des Attestierungskeys des Chip-Herstellers geprüft wird. Bei der Registrierung validiert der Server die Authentizität des Device anhand der übermittelten Zertifikate und öffentlichen Attestierungsschlüssel des Chip-Herstellers. Diesem muss der Server vertrauen. Zum anderen erfolgt bei jedem Log-in eine Prüfung der signierten Antwortdaten des Devices.

Frischenachweis

Die Frischeprüfung der Antwort erfolgt über die Challenge r des Servers. Diese wird vom Browser in die Browser-Daten D_B inkludiert und gehasht, als $H(D_B)$ an das Device gesandt. Das Device signiert $H(D_B)$ zusammen mit anderen Daten in seiner signierten Antwort s . Durch den Vergleich der Hashwerte mit den Originaldaten D_B , die der Server in *BrowserResponse* erhält, kann der Server prüfen, ob es sich um eine Antwort auf seine Anfrage handelt. Ein weiterer Frischenachweis erfolgt indirekt über den in der signierten Antwort s enthaltenen Channel-Key K_C (der ist in der Datenstruktur $resp$, die signiert wird, enthalten).

Nutzerbindung

Da aber das Device und dessen Daten nicht explizit an eine Person gebunden sind, kann ein Server mit der Authentisierung des Device nicht notwendigerweise Rückschlüsse auf die Person ziehen, die dieses Device nutzt. Dies muss durch separate Maßnahmen erfolgen. Ist von Seiten des Web-Dienstes eine starke Nutzerauthentisierung erforderlich, so könnte beispielsweise eine Registrierung des Nutzers mittels authentischer nPA-Daten erfolgen, wodurch eine Kennung eingerichtet wird, die in einem nächsten Schritt durch die Device-Registrierung an ein Device gebunden wird.

Server-Authentizität

Zu beachten ist, dass sich bei der U2F-Authentisierung der Web-Server nicht explizit gegenüber dem U2F-Device authentisiert. Vielmehr ist es die Aufgabe des Browsers, Daten zu erheben, die Evidenz darüber geben, dass es sich um denjenigen Server handelt, bei dem sich der Nutzer registrieren möchte.

Bei der Registrierung reicht der Browser dazu nicht nur den öffentlichen EC-DH-Schlüssel y_S des Servers an das U2F-Device weiter, sondern verknüpft diesen mit Informationen über den Ursprung, d.h. den Domänenamen, des angefragten Servers $H(Origin)$. Diese Daten werden vom U2F übernommen und an den öffentlichen Validierungsschlüssel Pub_U geknüpft, den das

Device für dieses Nutzerkonto bei der Registrierung erzeugt. Über das für diesen Datensatz erzeugte Key-Handle H_K sind die Daten eindeutig identifizierbar. Da diese Information bei der Registrierung des Devices auch in der Registrierungs-Antwort GRM in signierter und attestierter Form an den Server gesandt wird, kann der Server beim Erhalt dieser Nachricht prüfen, ob dem U2F-Device mit $H(Origin)$ auch wirklich die korrekte Ursprungsinformation bezüglich des anfragenden Servers vorliegt, oder ob ein Man-in-the-Middle versucht hat, die Server-Identität zu spoofen.

Die Basis für eine Vertrauensbeziehung zwischen dem Browser und dem anfragenden Web-Server ist, dass Browser und Server mittels einer SSL/TLS-Verbindung kommunizieren, bei der sich der Server gegenüber dem Browser mittels eines SSL/TLS-Zertifikats ausgewiesen hat. Hierfür gilt dann natürlich das, was für alle SSL-/TLS-Zertifikate und deren CAs¹⁹ zutrifft. Da bei heutigen Standard-Browsern eine Vielzahl von CAs vorkonfiguriert sind, ist unklar, ob der Nutzer tatsächlich allen diesen CAs vertraut. Da ein Nutzer in der Regel keine manuelle Nachjustierung vornimmt, enthalten Standard-Browser auch CAs, die nicht vertrauenswürdig sind. Diese Problematik liegt jedoch out-of-scope des Google Konzeptes. Dagegen wird das Problem beim nPA explizit aufgegriffen und über die Konzepte der Berechtigungszertifikate und der Terminal-Authentisierung gelöst.

Beim Login findet ebenfalls nur eine Evidenz-basierte Prüfung der Server-Authentizität statt. Dies erfolgt zum einen durch das U2F-Device, wiederum auf der Basis von Informationen, die der Server dafür zusammenstellt, nämlich die Browser-Daten D_B und die Ursprungsdaten $H(Origin)$, und zum anderen durch den Web-Server, der prüfen kann, von welchen Daten das Device ausgeht. Beim Log-in erzeugt der Browser wie beschrieben die Browser-Daten D_B und sendet diese gehasht an das Device. Da dieses aber die Bestandteile K_B und K_S nicht kennt, kann das Device mit diesen Informationen keine Prüfung durchführen, aber es übernimmt diese Daten in die signierte Antwort $resp$, so dass der Server prüfen kann, ob sich mögliche Modifikationen ergeben haben. Für das Device sind die anderen Daten interessant, die der Browser aufbereitet versendet: das ist die mit dem Channel-Key K_C verschlüsselte Ursprungsinformation $H(Origin)$, der mit K_M verschlüsselte Channel-Key K_C und das Key-Handle H_K .

Das Device kann, wenn es über den korrekten K_M aus der Registrierung verfügt, den Channel-Key entschlüsseln und damit die Ursprungsinformation offen legen. Wenn diese Information mit der Information übereinstimmt, die im Key-Handle H_K hinterlegt ist, dann vertraut das Device darauf, dass es mit dem authentischen Server kommuniziert (Origin-Check). Ein

¹⁹ Certification Authority

Man-in-the-Middle müsste versuchen, diese Ursprungsdaten bereits bei der U2F-Registrierung dem Browser unterzuschieben. Das heißt, dass in diesem Fall der Nutzer bereits für die Anmeldung auf eine gefälschte Webseite gelockt werden müsste. Da jedoch in den Log-in-Schritten erneute Evidenz-Überprüfungen durchgeführt werden, muss der Angreifer zusätzlich die Log-ins abfangen und auf sich umleiten. Diese Angriffe sind noch möglich, erfordern aber einen deutlich höheren Aufwand als in heutigen Web-Authentisierungs-Szenarien.

Ein Man-in-the-Middle-Angriff ist zwar theoretisch auch während des Log-in-Schrittes noch immer möglich, falls der Angreifer-Server das gleiche Zertifikat von der gleichen CA wie der Original-Server besitzt und darüber hinaus den gleichen Domänennamen wie der Originalserver hat. Dann kann er sich gegenüber dem Device und dem Browser als Server ausgeben. Jedoch ist dies kein inhärentes U2F-Problem, sondern ein Problem der zugrundeliegenden PKI.

Browser

Da das Konzept keine Überprüfung der Authentizität des Browsers vor sieht, kann ein U2F-Device nicht entscheiden, ob es mit einem manipulierten Browser oder einem integren Browser kommuniziert. Ein manipulierter Browser kann dem Device absichtlich falsche Origin-Daten übergeben, so dass das Device sich bei einem gefälschten Server einloggt. Dies ist durch das U2F-Protokoll nicht ausgeschlossen. Wenn der Browser manipuliert ist, kann der Angreifer, der den Browser manipuliert hat, auch beliebige Web-Server aufsetzen und deren Domänennamen bei der Registrierung dem U2D-Device nennen. Da der Angreifer den Web-Server kontrolliert, kontrolliert er auch dessen Schlüsselgenerierung, so dass die Ende-zu-Ende Sicherheit zwischen U2F-Device und Web-Server mittels des gemeinsamen Schlüssel K_M ebenfalls gebrochen ist. Im Zweifelsfall kann also der manipulierte Browser auch die Inhalte der Registrierungsnachricht GRM, die vom Device mit dem Shared Key K_M verschlüsselt ist, entschlüsseln und bei Bedarf die Information zum Spoofen nutzen. Über einen kompromittierten Browser und einen kontrollierten Web-Server können Log-in-Informationen abgefischt und an den korrekten Web-Server eingespielt werden. Der Channel-Key K_C liegt im Browser im Klartext vor, ein manipulierter Browser könnte auch diese Information an eine andere Instanz weiter reichen. Die kann die Antworten des U2F-Devices, falls sie beobachtbar sind (z.B. über NFC), entschlüsseln. Ein manipulierter Browser kann in diesem Szenario die gesamte Authentisierung korrumpern. Derartige Szenarien sind jedoch out-of-scope des U2F-Ansatzes, da es nicht das Ziel ist, die Browsersicherheit zu erhöhen; diese Probleme sind auf anderem Weg zu beheben.

Vertraulichkeit

Die Kommunikation zwischen dem Web-Server und dem Browser erfolgt SSL/TLS-gesichert mit dem Server-Zertifikat K_S , an das aber keine weiteren Anforderungen gestellt werden.

Vertraulichkeit

Zwischen dem U2F-Device und dem Server S wird bei der Registrierung ein gemeinsamer Schlüssel K_M vereinbart und für die sichere Ende-zu-Ende-Kommunikation zwischen den beiden Endpunkten genutzt. Zur symmetrischen Verschlüsselung kommt AES zum Einsatz, wobei K_M ein 128 Bit Schlüssel ist. Das Device verschlüsselt bei der Registrierung seine Antwort an den Server mit K_M , so dass das Einschleusen von gefälschten Schlüsselmaterialien oder gefälschten Origin-Informationen mittels Man-in-the-Middle Angriffen, wie oben diskutiert, deutlich erschwert werden.

Der Schlüssel K_M wird auch beim Log-in vom Server verwendet, um den Channel Key K_C damit zu verschlüsseln. K_C wird verwendet, um den potentiell unsicheren Kanal zwischen dem Browser und dem U2F-Device abzusichern. Dies ist z.B. bei NFC-Verbindungen wichtig. Ein Man-in-the-Middle kann keine gefälschten Daten an Stelle des Devices an den Server zurückgeben, da die Antworten mit K_C verschlüsselt werden und nur der Browser, der Server und das Device diesen Schlüssel kennen. Das Konzept nutzt natürlich nichts, wenn der Browser Schadcode enthält, dann können diese Schlüssel natürlich auch abfließen und missbraucht werden.

Eine automatisierte Schlüsselerneuerung scheint nicht vorgesehen zu sein, jedoch ist es möglich, ein Dongle manuell erneut zu registrieren. Das Schlüsselmanagement erscheint in der aktuellen Version der U2F-Spezifikation noch sehr rudimentär. Derzeit ist zum Beispiel keine Art von Reset implementiert, um Schlüsselmaterial wie den Key-Handle H_K altern zu lassen, so dass er nach dem Restart nicht mehr gültig ist.

Zu beachten ist auch, dass geplant ist, die U2F-Devices in Zukunft mit einem internen Schlüssel K_{intern} auszustatten, der im sicheren Speicher des Secure Elements gespeichert wird. Der Hintergrund dafür ist, dass derzeit die Dongles noch hochpreisig mit genügend Speicherplatz auf dem Chip sind, aber in Zukunft auch Chips in einem Niedrigpreis Sektor einsetzbar sein sollen. Diese werden wenig Speicherplatz für die generierten Schlüsselpaare haben. Analog zum TPM Ansatz soll deshalb ein Export des Schlüsselmaterials als Blob auf den Web-Server möglich werden. Falls der Speicherplatz im Secure Element für die Aufnahme von privaten Schlüsseln nicht mehr ausreicht, können diese dann auch verschlüsselt als Key-Blobs ausgelagert werden:

Innerner Schlüssel

$$KeyBlob = AES_{K_{intern}}(Priv_U)$$

Key-Blob

Der interne Schlüssel ist nur für U2F-interne Ver- und Entschlüsselungsoperationen nutzbar und darf das Device nicht verlassen.

Falls in Zukunft ein solcher Schlüssel auch extern erzeugt und in das U2F-Device gebracht werden kann, eröffnen sich natürlich damit potentielle Schwachstellen. Einerseits ist sicherzustellen, dass dann nur berechtigte Instanzen einen solchen Schlüssel einbringen dürfen. Zum anderen ist sicherzustellen, dass keine Kopien dieses Schlüssels extern existieren, da damit alle privaten Schlüssel des Devices kompromittiert werden könnten.

Echtheit und Vertrauenswürdigkeit

Secure Element

Um ein ausreichendes Maß an Vertrauenswürdigkeit bereit zu stellen, erfordert das U2F-Protokoll einen sicheren Hardware-Anker, ein Secure Element (Chip), als Vertrauensbasis. Die FIDO Spezifikation enthält jedoch keine konkreten Anforderungen an das Sicherheitsniveau dieses Elements. Das Secure Element enthält bei Auslieferung den Hersteller-Attestierungsschlüssel und dessen Zertifikat. Vorgesehen ist auch (s.o.) ein interner Schlüssel, der ggf. auch bei der Herstellung erzeugt wird, oder auf das Gerät eingebracht wird. Alle weiteren Schlüssel werden durch das Device selbst generiert.

Attestierung

Der Hersteller-Attestierungsschlüssel dient dazu, die Echtheit und Standardkonformität des Devices zu bestätigen. Dazu wird bei der Registrierung die vom Device erzeugte Datenstruktur CSR mit diesem Attestierungsschlüssel signiert. Da der Schlüssel Hersteller- und nicht Geräte-spezifisch ist, kann damit auch kein Tracken und Tracen einer individuellen Device-Nutzung erfolgen. Das Testat, konkret ist das die signierte Datenstruktur *GRM*, wird mit K_M Ende-zu-Ende verschlüsselt zum Server gesandt, so dass auch der Browser keine Einblicke in die signierten Daten erhält. Der Server kann anhand des Zertifikats des Attestierungsschlüssels eines Herstellers entscheiden, ob er dem Hersteller (dessen Zertifikat) vertraut, oder nicht. Es wird vorgeschlagen, öffentliche Schlüssel der Herstellerschlüssel in öffentlichen Verzeichnissen zu verwaltet. Klar ist, dass mit dem Konzept nur attestiert wird, dass der Chip von dem spezifischen Hersteller stammt. Ob das verwendete Secure Element wirklich über die erforderlichen Sicherheitsmerkmale verfügt, wird damit nicht attestiert. Das muss durch weitere Zertifizierungen abgedeckt werden.

Eine Bindung des U2F-Devices an den Nutzer ist natürlich auch nicht attestierbar, die Devices werden nicht personalisiert ausgegeben. Dies hat den Vorteil, dass beim Verlust eines Dongles keine Rückschlüsse auf die Person gezogen werden können und auch ein Tracken und Tracen einer Person, die unterschiedliche Web-Accounts bei unterschiedlichen Servern verwendet, auch bei Nutzung des gleichen Dongles nicht möglich ist. Ein Nachteil ist natürlich, dass der Dongle somit auch keine verbindlichen

Auskunft über die Identität des Nutzers gibt. Dies ist aus Compliance-Gründen jedoch bei verschiedenen Diensten gewünscht, wie beispielsweise die nachweislich korrekte Altersangabe bei einigen Web-Angeboten.

Privatheit

Das U2F-Device generiert für jede Web-Seite ein eigenes Schlüsselpaar ($Pub_U, Priv_U$). Das U2F-Device verwendet keinen globalen Identifikator, so dass keine Web-Seiten-übergreifende Verbindung zwischen Login-Aktivitäten eines Nutzers unter Verwendung des Devices möglich ist. Falls jedoch ein Nutzer mit dem *gleichen U2F-Device* unterschiedliche Accounts auf dem *gleichen Server* anlegt, beispielsweise die Accounts $user1, user2$, dann kann der Server, falls sich der Nutzer beispielsweise mittels der Kennung $user1$ anmeldet, in der Challenge-Nachricht das Key-Handle für $user2$ senden. Anhand der korrekt signierten Antwort des U2F-Devices ist der Server dann in der Lage abzuleiten, dass die beiden Kennungen $user1$ und $user2$ das gleiche U2F-Device nutzen, also ggf. dem gleichen Endgerät zugeordnet sind, aber unterschiedliche Personen z.B. Familienmitglieder betreffen. Dies ist eine mögliche Verletzung der Privatsphäre, der derzeit wohl nur durch die Verwendung von verschiedenen U2F-Devices für verschiedenen Accounts auf dem gleichem Server begegnet werden kann, da die Nutzerkennung nicht als Datum an die genutzten Schlüssel auf dem Device gebunden ist.

Privatheit

Auf dem U2F-Gerät werden keine personenbezogenen Daten gespeichert. Die auf dem Gerät gespeicherten Schlüsseldaten sind zwar prinzipiell Nutzeraccounts zugeordnet (über Key-Handle und CSR), aber diese Zuordnung wird beim Server und nicht auf dem Device gespeichert. Es wird aber kein global gültiger Device-Identifikator verwendet, so dass eine direkte Verknüpfung der Daten mit einer natürlichen Person oder ein Tracken nicht möglich ist. Auch ein nicht vertrauenswürdiger Server kann aus den Datensätzen keine Rückschlüsse auf das Device ziehen, wenn der Attestierungsschlüssel wirklich für viele Devices des Herstellers und nicht nur für sehr wenige oder gar nur eines verwendet wird. Die vorgesehenen Maßnahmen unterstützen damit die Wahrung der Privatsphäre sehr gut.

Einfachheit, Usability und Benutzerinteraktion

Da die einfache Nutzbarkeit ein erklärtes Ziel des U2F-Konzeptes ist, sollte mit dem Ansatz ein *Just-works-Feeling* erreicht werden. Der verfolgte Ansatz erfordert keine Treiber-Installation und auch keine Software-Änderungen, so dass derartige Akzeptanzhürden, aber auch Fehlerquellen von vornherein ausgeschlossen werden.

Usability

Das Protokoll basiert auf der Nutzung der JavaScript API für die standardisierte Interaktion zwischen Web-Seite, Browser und U2F-Device. Die API-

Calls werden vom Browser interpretiert und ausgeführt. Die Browser müssen das U2F-Protokoll unterstützen. Der derzeit konzipierte USB-Dongle ist universell einsetzbar und enthält alle notwendigen Informationen für eine asymmetrische Challenge-Response-Authentisierung. Das Dongle kann als eine Art *virtueller Schlüsselbund* für Web-Schlüssel betrachtet werden.

Sowohl die Registrierung als auch der spätere Login-Vorgang sind fast völlig transparent für den Nutzer. Die Spezifikation sieht vor, dass dem Nutzer über eine Informationszeile eine Information darüber angezeigt werden kann, dass eine Registrierung, eine Schlüsselvereinbarung oder später auch eine Authentisierung zwischen dem Dongle und dem Web-Server durchgeführt wird. Beim Login sollen nur wenige Interaktionen mit dem Nutzer erforderlich sein; diese beschränken sich derzeit auf das explizite Einsticken des USB-Dongles und das physische Aktivieren des Touchsensors beim Login-Vorgang.

Der Besitznachweis über eine plug-in Präsenz-Prüfung ist für eine Nutzung mit Smartphones derzeit noch schwierig, da sie über keinen USB Anschluss verfügen. Eine Nutzung von Bluetooth und WLAN ist zwar nach dem FIDO-Standard vorgesehen, wird aus Usability-Gründen derzeit von Google aber nicht verfolgt, da dies zuviel an Nutzer-Konfigurierung erfordert und damit die Einfachheit des Konzepts nicht mehr gegeben ist.

11.3.4 U2F-Protokoll versus eID-Funktion

nPA

Der nPA ist ein hoheitliches Dokument zur Authentisierung von Personen. Die in dem Dokument gespeicherten, personenbezogenen Daten sind von einer hoheitlichen Stelle erhoben und bestätigt auf dem Ausweis abgelegt. Mit den Daten des Ausweises lassen sich somit Personen zweifelsfrei identifizieren, wenn man nicht die Anonymisierungsdienste des nPA nutzen möchte (z.B. nur Altersangabe). Der Zugriff auf diese personenbezogenen Daten im nPA muss deshalb sehr viel stärker kontrolliert werden, als bei einem U2F-Device ohne Nutzerbindung. Web-Server, die nPA-Daten auslesen möchten, benötigen ein dediziertes Berechtigungszertifikat zur Kommunikation mit dem nPA. Mittels der Terminal-Authentisierung kann der nPA die Authentizität des Servers überprüfen. Der nPA vertraut hierbei also nicht auf Daten, die der Browser oder die Ausweis-App erheben, sondern prüft explizit die Authentizität mittels eines Challenge-Response-Verfahrens im Verlauf der Schritte der Terminal-Authentisierung.

Authentisierung

Authentisierung

Der Chip im nPA authentisiert sich gegenüber dem Web-Server mittels des Chip-Authentisierungs-Protokolls, bei dem der Chip die Kenntnis des privaten Chip-Schlüssels nachweist. Diese Daten sind von der Bundes-

druckerei signiert, da die Bundesdruckerei die hoheitlichen Dokumente ausstellt. Der öffentliche Validierungsschlüssel ist vom BSI zertifiziert. Analog zur Attestierungsprüfung beim U2F-Dongle prüft der Web-Server das Zertifikat des BSI und dann die Signatur der Bundesdruckerei. Anders als bei der U2F-Authentisierung werden hier aber nicht beliebige Hersteller-Attestierungs-Zertifikate genutzt, sondern die von hoheitlichen Stellen ausgestellten Zertifikate. Freiheitsgrade und Wahlmöglichkeiten, welcher CA man vertrauen möchte, sind entsprechend beim nPA nicht gegeben. Eine solche Wahlmöglichkeit besteht demgegenüber bei der Nutzung von U2F.

Auch beim nPA-Szenario kann der Rechner des Nutzers kompromittiert sein und die PIN-Eingabe abfangen, falls diese nicht über eine abgesicherte Tastatur auf dem Lesegerät erfolgt. Ist die Eingabe jedoch abgesichert, dann wird ein Ende-zu-Ende sicherer AES-verschlüsselter Kommunikationskanal zwischen dem nPA und dem Web-Server etabliert und ein manipulierter Browser bzw. Rechner wird vollständig getunnelt.

Vertraulichkeit

Beim nPA wird ebenfalls zwischen dem eID-Provider und der Ausweis-App auf dem Nutzer-PC eine SSL/TLS-Verbindung aufgebaut und die Terminal- und Chip-Authentisierung wird innerhalb dieser so gesicherten Verbindung abgewickelt. Nach erfolgreicher Terminal- und Chip-Authentisierung generieren der Server und der RFID Chip einen gemeinsamen 128-bit AES-Schlüssel, mit dem die benötigten nPA-Daten auf dem RFID-Chip mittels AES verschlüsselt und an den Server übertragen werden.

Vertraulichkeit

Für jede Verbindung wird nach der erfolgreichen Authentisierung ein neuer Schlüssel mittels des DH-Verfahrens vereinbart, so dass keine Kommunikationsschlüssel persistent gespeichert werden müssen und eine *Perfect Forward Secrecy* unterstützt wird.

Die PIN, die zur Aktivierung des nPA vom Nutzer am Lesegerät einzugeben ist, wird nicht über Kommunikationsmedien übertragen, sondern dient als Eingabe in den PACE-Algorithmus, mit dem sich das Lesegerät und der RFID-Chip, der die PIN gespeichert hat, wechselseitig davon überzeugen, die PIN zu kennen.

Vertrauenswürdigkeit

Beim nPA kommt ebenfalls ein Secure Element, der RFID-Chip, zum Einsatz. Dieser muss ein evaluiert Hochsicherheits-Chip sein und Anforderungen der Common Criteria (CC) gemäß EAL5+ erfüllen.

Secure Element

Auf dem Chip ist ein öffentlicher und ein privater Chip-Schlüssel bei dessen Auslieferung eingebbracht und der öffentliche Schlüssel ist vom Aussteller

des Ausweises, also der Bundesdruckerei, signiert, deren Zertifikat wiederum vom BSI zertifiziert ist. Der zugehörige private Schlüssel wird bei der Chip-Authentisierung verwendet und der Server kann prüfen, dass er es mit einem nicht geclonten, echten RFID-Chip zu tun hat. Die Zertifikatskette ist wohldefiniert.

Auf dem RFID Chip ist weiterhin noch die 6-stellige Nutzer-PIN gespeichert, die benötigt wird, um den Chip zu aktivieren. Aber alle anderen Schlüssel werden nur für die Kommunikationsverschlüsselung temporär erzeugt und nach Beendigung des Datenverkehrs wieder gelöscht.

Privatheit

Privatheit

Wie bereits erläutert, ist es der Hauptzweck des nPA als Ausweisdokument Angaben zu den personenbezogenen Daten eines Nutzers zu machen. Es ist der Zweck der nPA-Authentisierung, dass der Authentisierungsvorgang entweder explizit mit der amtlich bestätigten Person, die von dem Ausweis identifiziert wird, erfolgt, oder mit anonymisierten Informationen zu der Person, wie beispielsweise eine Altersangabe, deren Korrektheit aufgrund des amtlichen Dokuments auch amtlich bestätigt ist. Der Server erhält über den nPA also verlässliche und geprüfte Daten über denjenigen, der sich bei ihm anmeldet. Dies könnte in manchen Online-Business-Szenarien erwünscht sein, um beispielsweise gesetzliche Auflagen in Bezug auf eine Altersbegrenzung zu erfüllen.

Der nPA erzeugt Pseudonyme, so dass Authentisierungen bei verschiedenen Web-Servern nicht miteinander in Beziehung gebracht werden können. Auch beim nPA wird durch die verwendeten Konzepte der Wahrung der Privatsphäre Rechnung getragen.

Einfachheit, Usability

Usability

Im Vergleich zum U2F-Dongle muss bei der Nutzung des nPA einiges mehr an Aufwand spandiert werden. Erforderlich ist ein Lesegerät, um den kontaktlosen RFID-Chip im nPA auszulesen, sowie die Installation einer Applikation auf dem Nutzer-PC, die Ausweis App, um das nPA-Protokoll abzuwickeln. Die Ausweis App steht als zertifizierte Software zum Download kostenlos zur Verfügung, während das Lesegerät gekauft werden muss.

Auch die Einbindung des Nutzers ist stärker als beim U2F-Konzept. Das ist aber auch sehr sinnvoll, da auf dem nPA, im Gegensatz zum U2F-Device, personenbezogene Daten abgelegt sind. Der Zugriff auf die Daten sollte nutzerkontrollierbar ablaufen; dies ist auch so vorgesehen. Bei der Nutzung des nPA für eine Online-Authentisierung ist der Nutzer bereits dann involviert, wenn der Server sein Berechtigungszertifikat sendet und der Nutzer entscheiden kann, welche Zugriffe auf welche nPA-Daten er

dem Server zubilligen möchte. Eine zweite wesentliche Nutzerinteraktion besteht dann darin, eine 6-stellige PIN am Lesegerät einzugeben, um zu bestätigen, dass man mit dem Auslesen der Daten für den anfragenden Server einverstanden ist.

Zudem muss natürlich der physische Besitz des nPA nachgewiesen werden, indem er auf das Lesegerät gelegt wird (oder anderweitig in Kontakt zum Lesegerät gebracht wird). Hier ist also mit der zusätzlichen PIN-Kenntnis noch ein stärkerer Besitznachweis zu führen, als lediglich durch das Aktivieren des Touchsensors. Analog zu einem über lange Zeit eingesteckten U2F-Dongle kann natürlich auch ein nPA über lange Zeit auf einem Lesegerät liegen, so dass das reine Vorhandensein des nPA oder auch des Dongles nicht für den Besitznachweis ausreicht.

Fazit

Das U2F-Protokoll stellt bei Standard-Web-Anwendungen einen deutlichen Zugewinn an Sicherheit gegen Phishing und Man-in-the-Middle Angriffe dar. Es ist klar, dass das U2F-Protokoll und der nPA sehr unterschiedliche Szenarien bedienen, so dass ein direkter Vergleich beider Konzepte und Protokolle nicht möglich ist.

Fazit

Ein U2F-Dongle ist ein einfach zu nutzender, sicherer Ersatz für z.B. OTP-Token, wie sie durch das RSA-SecureID-Token (vgl. Seite 458) weit verbreitet sind, da mit dem U2F-Protokoll eine noch stärkere Phishing-Abwehr gewährleistet werden kann. Ein U2F-Dongle sichert einen existierenden Account bei einem Web-Dienst auf einfache Weise zusätzlich ab; dies ist unabhängig von der Art und Ausrichtung der Dienstleistungen, die der entsprechende Server über den Account zur Verfügung stellt. Wird für eine Web-Authentisierung jedoch eine stärkere Personenidentifizierung oder eine bestätigte Altersangabe benötigt, so kann ein U2F-Dongle dies so nicht leisten. Der nPA besitzt die geforderte Aussagekraft, um dem Server eine spezifische Nutzer-Identität zu bestätigen. Aus Gründen der einfachen Handhabbarkeit wird beim U2F-Protokoll zudem der Server nicht mit der hohen Mechanismenstärke authentisiert, wie dies in Szenarien, in denen der nPA zum Einsatz kommt, der Fall ist. Für viele Einsatzszenarien ist die U2F-Server-Authentisierung aber völlig ausreichend und bildet einen sehr guten Kompromiss zwischen Sicherheitslevel und Nutzerfreundlichkeit.

11.4 Trusted Computing

Die Sicherheitskonzepte, die eine herkömmliche Hardware anbietet (u.a. Speicherschutz und Adressraumisolierung) sind unzureichend. So können

Ausgangssituation

beispielsweise Hardware-Komponenten wie DMA-Geräte den Speicher direkt, also unter Umgehung der Sicherheitskontrollen der CPU, nutzen und somit auf beliebige Daten lesend und schreibend zugreifen. Aber auch eine vollständige Verlagerung des Objektschutzes in das Betriebssystem und/oder auf die programmiersprachliche Ebene garantiert keinen lückenlosen Schutz. Auch hier können z.B. Hardware-Komponenten wie Soundkarte und Grafikkarte aus Performanzgründen direkt, d.h. ohne Kontrolle durch das Betriebs- oder Laufzeitsystem, agieren.

Anforderungen

Zur Etablierung einer vertrauenswürdigen Sicherheitsarchitektur benötigt man deshalb ineinander greifende, wirksame Hardware-, Firmware- und Software-Sicherheitskonzepte. Die Hardware sollte vertrauenswürdige, nicht umgehbar Basisfunktionen zur Verfügung stellen, die über entsprechende vereinheitlichte Schnittstellen dem Betriebssystem, aber auch individuellen Anwendungen zur direkten Nutzung angeboten werden. Diese können die Basisdienste zu komplexeren Diensten veredeln. Ein Beispiel wäre der in das Betriebssystem integrierte Dienst eines verschlüsselnden Objektspeichers, der Hardware-Mechanismen zur sicheren Erzeugung und Speicherung der benötigten Schlüssel verwendet. Um flexibel anpassbare Kontrollen zu ermöglichen, sollten strategische Entscheidungen (Policies) von den Basismechanismen strikt getrennt und auf der Betriebssysteme- oder Applikationsebene angesiedelt sein. Die Forderung könnte soweit gehen, dass für jedes einzelne Objekt (z.B. ein Word-Dokument) eine eigene Policy (z.B. auf welchen Rechnern ein vertrauliches Dokument im Unternehmen und unter welchen Randbedingungen gelesen oder verändert werden darf) festgelegt und mit dem Objekt assoziiert ist. Zur Durchsetzung (Enforcement) der Objekt-Policy werden dann jeweils die Dienste desjenigen Betriebssystems und der Hardware genutzt, in dessen Kontext das Objekt gerade verarbeitet wird. Mit Anwendungsdiensten wie sicherer E-Mail gibt es übrigens bereits eine sehr einfache Umsetzung dieses Gedankens, da jeder E-Mail die zu ihrer Bearbeitung benötigten Sicherheitsrichtlinien (welche Empfänger, welche Verschlüsselungsverfahren, verwendeter Schlüssel, der nur von berechtigten Empfängern zu entschlüsseln ist, etc.) beigefügt werden.

Mit der Trusted Computing Initiative wird seit einiger Zeit gemeinsam von Hardware- und Softwareherstellern sowie Anwendungsentwicklern eine Sicherheitsarchitektur entwickelt, die geeignet ist, die gestellten Anforderungen zumindest teilweise zu erfüllen.

11.4.1 Trusted Computing Platform Alliance

TCPA

Die Trusted Computing Platform Alliance (TCPA) wurde 1999 von dem Hersteller-Konsortium bestehend aus den Mitgliedern Microsoft, Intel, IBM, Compaq und HP gegründet. Deren Ziel bestand darin, Hard- und Software-

standards zu spezifizieren, um das Vertrauen (trust) in Computer-Plattformen bei e-Business Transaktionen zu erhöhen. Der Begriff Plattform umfasst alle technischen Komponenten wie das Motherboard mit CPU, die angeschlossenen Geräte, BIOS-Komponenten sowie weitere Chips (z.B. Smartcard-artige Module). Ausgehend von den TCPA-Spezifikationen sollten Hardware-Module und technische Infrastrukturen entstehen, die spezifische Sicherheitsfunktionen beinhalten, wie ein sicheres Booten, die Bestätigung der Integrität der Systemkonfiguration einer Plattform gegenüber externen Kommunikationspartnern (Attestation), sowie die sichere Generierung und Aufbewahrung kryptografischer Schlüssel. Durch ein Zusammenwirken von vertrauenswürdigen Hardware-, Firmware- und Software-Mechanismen sollen PCs und Server sicherer werden, so dass man ihnen vertrauen kann. Mit Vertrauen ist in diesem Zusammenhang stets gemeint, dass eine Komponente, ein Modul oder eine ganze Plattform so wie erwartet agiert.

Trust

Der TCPA-Allianz schlossen sich relativ schnell eine Vielzahl weiterer Mitglieder an, so dass sie im Jahre 2003 bereits über 200 Mitglieder zählte, wozu unter anderem auch Infineon, Siemens, RSA, und Nokia gehörten. Die Allianz veröffentlichte die erste Version 1.0 der TCPA Spezifikation im Jahr 2001, die durch die Versionen 1a (2002) und 1b bzw. Version 1.2 (2003) abgelöst wurde. Da in der TCPA alle Entscheidungsfindungen einstimmig gefällt werden mussten, erwies sich die Größe der Allianz sehr bald als Hemmschuh, was unter anderem Auslöser dafür war, dass AMD, HP, Intel und Microsoft die TCG²⁰ (Trusted Computing Group) gründeten, die im Jahr 2003 die Rechtsnachfolge der TCPA antrat.

Die TCG hat sich ihrerseits zum Ziel gesetzt, die Initiative der TCPA fortzuführen und offene Industriestandards für ein „Trusted Computing“ auch für Laptops und Smartphones zu entwickeln. Die Gruppe²¹ entscheidet über die Weiterentwicklung des Standards mit einer 2/3 Mehrheit, wobei nur diejenigen Mitglieder stimmberechtigt sind, die mindestens 15.000 US Dollar jährlich als Mitgliedsbeitrag bezahlen. Die TCG übernahm die bereits existierenden Spezifikationen der TCPA und entwickelte sie weiter.

TCG

11.4.2 TCG-Architektur

Die TCG Spezifikation²² besteht aus einem Hardware-bezogenen Teil, der den TCG-Chip, das so genannte Trusted Platform Module (TPM), beschreibt, einem Hardware- oder Firmware-bezogenen Teil, dem Root of Trust for Measuring Integrity Metrics (RTM), der auch vollständig durch das TPM implementiert werden kann, und einem Software-bezogenen Teil, dem

TCG-Subsystem

²⁰ siehe <http://www.trustedcomputinggroup.org>

²¹ Sie umfasst über 120 Mitglieder (Stand 2007).

²² Version v1.2, siehe <http://www.trustedcomputinggroup.org>

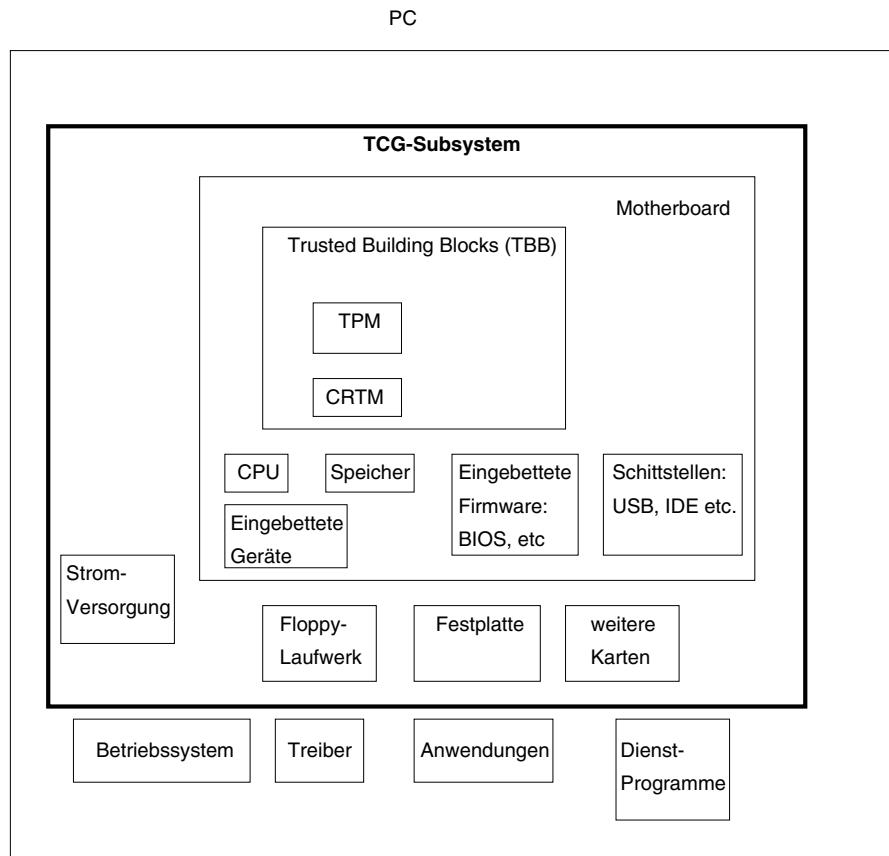


Abbildung 11.13: Einbettung des TCG-Subsystems in eine PC-Architektur

Trusted Software Stack (TSS). Der TPM, das RTM und die TSS bilden das TCG-Subsystem. Das TCG-Subsystem umfasst keine Betriebssystemdienste, sondern stellt für Betriebssysteme vertrauenswürdige Basismechanismen und Dienste als Bausteine zur Verfügung.

TCG-Subsystem

Abbildung 11.13 veranschaulicht die Komponenten des TCG-Subsystems und deren Einbettung in eine herkömmliche PC-Architektur.

TPM, RTM, TBB

Das TPM bildet den Kern der TCG-Architektur. Auf die Architektur und Dienste des TPM gehen wir in Unterabschnitt 11.4.3 noch genauer ein, wobei wir uns auf die Darstellung des TPM 1.2 konzentrieren, da der TPM 2.0 noch nicht weit ausgerollt ist. Änderungen, die mit der TPM 2.0 Spezifikation verbunden sind, werden explizit in Abschnitt 11.4.4 besprochen. Ein TPM übernimmt im Wesentlichen die Aufgaben einer Smartcard. Dazu gehören die Bereitstellung eines Hardware-unterstützten, sicheren Speichers

für Schlüssel und andere sensitive Daten, aber auch die Generierung kryptografisch starker Schlüssel. Das RTM stellt Dienste zur Verfügung, die es erlauben, beim Booten über ausgewählte Komponenten (u.a. BIOS, Bootloader) des Systems Hashwerte zu berechnen (Integrity Measurement) und diese Werte in speziellen Registern, den Platform Configuration Registern (s.u.) im TPM, aufzuzeichnen. Zum RTM gehören die in der Abbildung 11.13 als Trusted Building Blocks gekennzeichneten Komponenten und insbesondere die CRTM (Core Root of Trust Measurement). Das CRTM ist entweder auch in den TPM integriert oder Bestandteil des BIOS. Auch auf das CRTM gehen wir weiter unten noch näher ein.

Auf der Basis dieser berechneten und sicher abgelegten Integritäts-Werte können während der Laufzeit des Systems Attestierungen (Reporting) über die Systemkonfiguration erstellt werden. Dadurch lassen sich durch ein Betriebssystem, das solche Reports erstellen lässt, Änderungen an der Konfiguration aufdecken. Ferner können sich Dienste bzw. Komponenten, die mit der Plattform interagieren möchten, über diese Attestierungen Gewissheit über die gewünschte Authentizität der Plattform verschaffen. Beachtenswert hierbei ist, dass der Authentizitätsnachweis deutlich umfassender ist, als nur der Nachweis einer eindeutigen Identität, da in den Nachweis Aussagen über die Konfiguration des Systems einfließen.

Der TSS spezifiziert eine Standard-API, über die die Funktionen und Dienste des TPM zugreifbar sind. Eine objekt-orientierte Schnittstelle wird durch den TSS Service Provider zur Verfügung gestellt, so dass Anwendungen die Fähigkeiten der TCG-Plattform darüber nutzen können. Beispiele für derartige Dienste sind der Export- oder Import von Kommunikationsschlüsseln, die Erstellung von Hashwerten oder auch Aufträge zur Erstellung von Integritäts-Bescheinigungen. Das TSS beinhaltet Funktionen, die es ermöglichen, Schnittstellen für existierende Krypto-APIs zu erstellen, wie zum Beispiel die Microsoft CryptoAPI (CAPI), CDSA und PKCS#11, so dass die Applikationen, die diese APIs verwenden, darüber auch direkt die Dienste des TPM nutzen können. Über die Device Driver Library wird eine Abstraktionsschicht angeboten, so dass Anwendungen keine Kenntnis über die Hardware-Eigenschaften des spezifischen TPM Moduls der Plattform berücksichtigen müssen.

TSS

Eindeutige Identität und Besitzerkonzept

Bereits während der Herstellung des TPM werden in den Chip Basis-Daten (Endorsement Key und Zertifikat) integriert, die dafür sorgen, dass der Chip eindeutig identifizierbar ist und seine Identität und korrekte Erzeugung nach außen dokumentieren kann. Da eine derartige eindeutige Identifizierung von Geräten zu Privacy-Problemen führen kann und von Benutzern wenig akzeptiert wird (man erinnere sich an die Problematik

Identität

der Seriennummern von Intel-Prozessoren), bietet das TPM eine Möglichkeit, diese Identität gegenüber Dritten zu verschleiern. Das ist das Konzept der so genannten Attestation Identity Keys (AIK), auf das wir weiter unten noch genauer eingehen werden. Außerdem sieht die TPM-Spezifikation ab Version v1.2 vor, einen revozierbaren Endorsement-Key zu erzeugen (*TPM_CreateRevocableEK*); dies erfolgt durch den Hersteller.

Besitzerkonzept

Das TPM ist ein passiver Chip, der beim Power-on und während des Bootvorgangs durch das BIOS gestartet wird, falls der Besitzer (Owner) der Plattform das TPM aktiviert hat. Da das TPM1.2 defaultmäßig deaktiviert ausgeliefert wird²³, muss der Besitzer der TCG-Plattform ihn explizit aktivieren, falls er die Sicherheitsdienste des TPM über das TSS dem jeweiligen Betriebssystem oder anderen Diensten verfügbar machen möchte. Damit nicht ein beliebiger Benutzer das TPM durch Fernzugriff aktivieren und deaktivieren kann, ist eine explizite Inbesitznahme (Take_Ownership) der Plattform durchzuführen, um die Berechtigung zur Durchführung derartiger Operationen zu reglementieren. Bei der Ausführung der entsprechenden Kommandos muss der Besitzer Authentifizierungsdaten festlegen (Passwort), das als gemeinsames Geheimnis zwischen dem Besitzer und der Plattform etabliert wird (vgl. Seite 588). Mit der Besitzübernahme ist der Besitzer in der Lage, die Aktivierung und Deaktivierung des TPM zu kontrollieren und die Zugriffe auf Objekte im aktivierte TPM durchzuführen, die spezielle Besitzer-Privilegien, d.h. den Nachweis der Kenntnis der Authentifizierungsdaten, erfordern. Ist ein TPM durch den Besitzer deaktiviert worden, so kann es weder als Schlüsselspeicher genutzt, noch können Reports über dessen Integritätszustand erstellt werden. Der Besitzer einer Plattform kann somit darüber bestimmen, ob die Sicherheitsfunktionen angewendet werden oder nicht. Dies ist vor dem Hintergrund der kontroversen Diskussion, die im Zusammenhang mit dem TPM und der Trusted Plattform Initiative in der Öffentlichkeit entbrannt ist, beachtenswert. Wir werden auf den Einsatz einer solchen Plattform und mögliche Probleme am Ende des Abschnitts noch eingehen.

Zu beachten ist, dass es neben der Besitzer-kontrollierten Verwendung auch stets die Möglichkeit gibt, beim direkten physischen Zugriff auf die Plattform eine Deaktivierung bzw. Aktivierung ohne Kenntnis der Authentifizierungsdaten durchzuführen. Beim Intel-Board geschieht dies z.B. durch die Verwendung eines Jumpers, der den TPM zurücksetzt. Durch eine solche Deaktivierung wird aber auch das Owner-Passwort gelöscht, so dass keiner der im TPM abgelegten Schlüssel mehr nutzbar ist! Durch eine erneute Initialisierung kann man den TPM wieder nutzen, aber die Schlüssel werden dadurch nicht reaktiviert.

²³ Achtung: der TPM 2.0 ist defaultmäßig aktiviert.

Roots of Trust

Um Vertrauen in die Konfiguration einer Plattform zu etablieren, wird ein schrittweiser Prozess durchlaufen, dessen Startpunkt die Vertrauensanker, das sind die Roots of Trust, bilden. Den Vertrauensankern muss ohne weitere Prüfung vertraut werden, da deren Fehlverhalten nicht aufgedeckt werden kann. Ein TCG-Subsystem enthält drei Vertrauenswurzeln, das sind der Root of Trust for Measurement (RTM), die Root of Trust for Storage (RTS) und die Root of Trust for Reporting (RTR).

Root of Trust

Der RTM umfasst Funktionen, um auf zuverlässige Weise, die Integrität einer Konfiguration während des Boot-Vorganges bzw. beim Reset und nach dem Suspend-Modus zu bestimmen. Die entsprechenden Berechnungen beginnen mit den Operationen des Core Root of Trust for Measurement (CRTM), das meist als Erweiterung des BIOS realisiert ist und bei jedem Neustart als Erstes ausgeführt wird (siehe Abschnitt 11.4.4). Die Funktionen des RTM werden (bis auf den CRTM-Anteil) von dem TPM zur Verfügung gestellt. Das TPM implementiert die zur Hashwertberechnung benötigte Hashfunktion (SHA-1²⁴) und stellt über spezielle Register (die PCR-Register²⁵) einen sicheren Speicher für die berechneten Integritätswerte zur Verfügung.

RTM

Die Root of Trust for Storage (RTS) schützt Schlüssel und Datenobjekte, denen das TPM vertraut. Die Funktionen des RTS werden ebenfalls durch das TPM implementiert. Das RTS hat die Aufgabe, kryptografische Schlüssel zu erstellen und zu verwalten, mit denen Objekte der Plattform (z.B. kryptografische Schlüssel, Passphrasen, Cookies oder Authentifizierungs-Tickets). geschützt werden. Das RTS verwaltet die on-Chip erzeugten Schlüssel in einer Schlüsselhierarchie, wobei ein Schlüssel stets vom Elternknoten in der Hierarchie verschlüsselt wird. Nicht aktiv genutzte Schlüssel können deshalb auch in dieser verschlüsselten Form off-Chip (als verschlüsselte Key-Blobs²⁶) abgelegt werden. Die Wurzel dieser Schlüsselhierarchie bildet der Storage Root Key (SRK) (siehe Seite 588), der das TPM nicht verlassen kann und bei der Inbesitznahme automatisch generiert wird. Aktive Schlüssel, die zum Signieren oder Ver- bzw. Entschlüsseln benötigt werden, werden dazu in das TPM in dessen flüchtigen Speicher geladen.

RTS

Über das Root of Trust for Reporting (RTR) werden auf vertrauenswürdige Weise Bescheinigungen über die Integrität der Daten und Informationen, die vom RTS verwaltet werden, erstellt. Auch diese Funktionen werden durch das TPM implementiert. Hierbei handelt es sich um die Attestierungen, auf

RTR

²⁴ Der TPM 2.0 ermöglicht auch andere Verfahren.

²⁵ Platform Configuration Register

²⁶ Binary Large Object

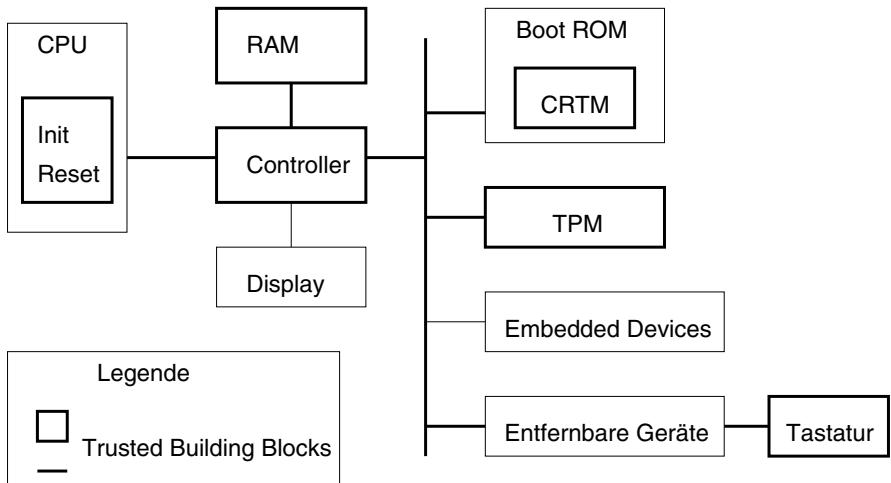


Abbildung 11.14: Generische Architektur einer Trusted Computing Plattform und TBBs

die wir weiter unten noch einmal zurückkommen, wenn die zum Verständnis notwendigen Konzepte erklärt sind.

TBB

Als Trusted Building Blocks (TBB) werden diejenigen Komponenten und Verbindungswege des TCG-Subsystems bezeichnet, die sicherheitskritischen Code enthalten bzw. sicherheitskritische Aufgaben übernehmen. Typischerweise umfassen sie Operationen zur Initialisierung des RTM bzw. des TPM (z.B. die Init- und die Reset-Funktion). Abbildung 11.14 veranschaulicht, welche TBBs in einer generischen Plattform-Architektur üblicherweise vorhanden sind.

Die Trusted Building Blocks sind die fett umrandeten Komponenten bzw. Verbindungen. Dazu gehören das TPM selber, das CRTM sowie die Verbindung des CRTM zum Motherboard, wie auch die Verbindung des TPM zum Motherboard, oder auch das Konzept, mittels dessen die physische Anwesenheit eines Operators geprüft wird, wozu z.B. eine abgesicherte Eingabe über die Tastatur erforderlich ist. Anzumerken ist, dass der sichere Eingabekanal von der Tastatur zum Motherboard häufig nicht gewährleistet werden kann, so dass zurzeit die Hersteller vielfach einfach einen Jumper verwenden, um diese physische Anwesenheit an der Plattform zu prüfen.

11.4.3 TPM 1.2

TPM

Das Trusted Platform Module ist ein Chip, der den Hardware-Teil der Trusted Computing Architecture abdeckt. Der Chip wird in manchen Kreisen

auch unter dem Spitznamen Fritz-Chip gehandelt. Der Name bezieht sich auf den US Senator Fritz Hollings aus South Carolina, der sich für Gesetzesvorschläge stark machte, die darauf abzielen, in alle Gegenstände der Unterhaltungselektronik zwangsweise Module zu integrieren, um die digitalen Rechte (Digital Rights Management) der Urheber von Audio- und Video-Daten zu schützen. Auch das TPM wurde in der Vergangenheit stark mit derartigen Bestrebungen in Zusammenhang gebracht, woraus sich diese Namensbildung ableitet.

Der Trusted Platform Module bietet im Groben die Funktionen einer Smartcard. Zu seinen wesentlichen Aufgaben gehören:

- Die Generierung von asymmetrischen und symmetrischen Schlüsseln unter Nutzung eines Hardware-basierten Zufallszahlengenerators.
- Die Signaturerstellung, Hashwertberechnung sowie asymmetrische und Chip-interne symmetrische Verschlüsselung.
- Das Verschlüsseln kryptographischer Schlüssel (Binding bzw. Wrapping), die in dieser geschützten Form auch außerhalb des TPM verwaltet werden können.
- Eine Hardware-unterstützte, sichere Speicherung von (kleinen) Objekten und von Hashwerten, die über die Konfigurationsdaten der Plattform berechnet wurden. Auf diesen geschützten Speicher (shielded Register) darf nur im Rahmen der Ausführung festgelegter Sicherheitsdienste (z.B. Signieren, Verschlüsseln) bzw. Kommandos (protected Capabilities) zugegriffen werden.
- Das Erstellen von signierten Auskünften (Reporting) über diese Werte, sofern der Plattformbesitzer diese Auskunftserteilung autorisiert hat.
- Ein spezieller Schlüssel (Endorsement Key), mit dem die Korrektheit von Identitätsschlüsseln (AIKs), die der Benutzer im TPM erzeugen kann, bestätigt wird, wobei nur vertrauenswürdigen Stellen offengelegt wird, zu welchem TPM der jeweilige Identitäts-Schlüssel gehört. Für das von IBM-Research entwickelte Konzept der Direct Anonymous Attestation (DAA²⁷) trifft dies so nicht zu, da hierbei zur Wahrung der Anonymität blinde Signaturen und Zero-Knowledge-Beweise verwendet werden.
- Funktionen, so dass der Besitzer der Plattform den Chip in Besitz nehmen kann und berechtigt ist, das TPM zu aktivieren oder auch (wieder) zu deaktivieren.

Funktionen

²⁷ Siehe <http://www.zurich.ibm.com/security/daa/>

- Ein vertrauenswürdiger Zeitgeber (timer), der z.B. für die Festlegung bzw. Prüfung der Gültigkeitsdauer von Zertifikaten genutzt werden kann.

Aktivierung

Analog zu einer Smartcard ist auch das TPM nur passiv und reagiert auf Kommandos. Das Modul wird beim Power-on durch das BIOS gestartet, falls es nicht im Zustand „deaktiviert“ ist. Beim Start führt das TPM stets als erstes einen Selbsttest durch, um zu prüfen, ob alle Funktionen korrekt funktionieren (z.B. Durchführen einer Signaturopération für einen vorgegeben Wert und Überprüfung des Resultats mit einem Testwert). Im Unterschied zu einer Smartcard ist das TPM einer Rechnerplattform zugeordnet und enthält u.a. Schlüssel (den Endorsement Key) und Informationen (das Endorsement Key Zertifikat), die es erlauben, die Plattform, der das TPM assoziiert ist, eindeutig zu identifizieren. Darüber hinaus wird über die Reporting-Funktion ein Dienst angeboten, der es anderen Komponenten ermöglicht, von dem TPM auch eine signierte Bestätigung über seine Konfiguration zu erhalten, wodurch das Vertrauen erhöht werden soll, dass die Plattform sich in einem gewünschten Zustand befindet.

Während der TPM-Chip früher häufig noch ein eigener Baustein war, der auf die Hauptplatine des Rechners gelötet wurde, fordert die Spezifikation 1.2, dass alle Spezifikations-konformen TPMs an den LPC²⁸-Bus der Southbridge angebunden sind.

Protection Profile

Seit Oktober 2002 ist ein TPM-Protection Profile gemäß der Common Criteria (siehe Abschnitt 5.4.3) veröffentlicht. Das Schutzprofil legte Anforderungen an ein TPM fest, die der Vertrauensstufe EAL3 entsprechen.

TPM 1.2-Architektur

TPM-Architektur

Abbildung 11.15 veranschaulicht die Architektur des TPM-Chips. Die Kommunikation zwischen dem Chip und der Außenwelt erfolgt über die I/O Komponente. Diese leitet Nachrichten an die internen Komponenten weiter und führt die erforderlichen Codierungen durch, so dass eine Kommunikation über externe und interne Busse möglich ist.

Krypto-Funktionen

Das TPM implementiert grundlegende kryptografische Dienste. Dazu gehört ein hardware-basierter Zufallszahlen-Generator (Random Number Generator (RGN)), der sowohl von anderen Funktionen auf dem Chip genutzt wird, z.B. bei der Schlüsselgenerierung oder bei der Generierung von Nonces, als auch von außen angesprochen werden kann. Weiterhin bietet das TPM 1.2 eine Hashfunktion (SHA-1 Engine), sowie eine Funktion zur Erzeugung von Message Authentication Codes (HMAC-Engine), wobei der MAC-Schlüssel eine Länge von 160 Bit besitzen muss. Während die HMAC-Engine nur intern nutzbar ist, ist die SHA-1-Hashfunktion mit ihrem 160-Bit Hash-

²⁸ Low-PinCount

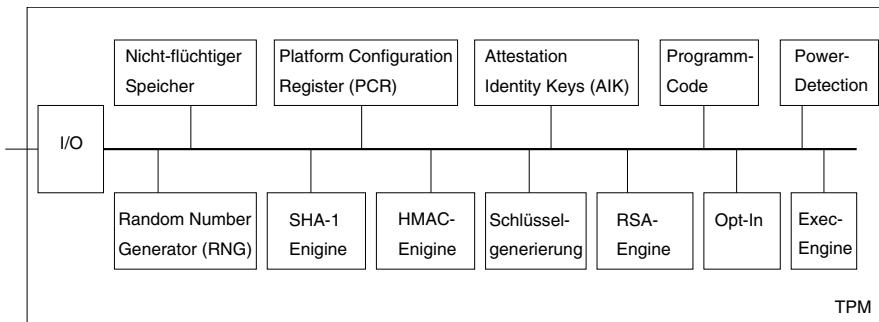


Abbildung 11.15: Architektur des TPM

wert auch von außerhalb nutzbar. Dies ist eine grundlegende Eigenschaft, da erst dadurch möglich ist, dass während des Bootprozesses (siehe Seite 597) Hashwerte von Systemkonfigurationen und Zuständen vertrauenswürdig berechnet werden können.

Das TPM enthält ferner einen Dienst zur on-Chip Erzeugung von asymmetrischen (RSA) und symmetrischen Schlüsseln (Schlüsselgenerierung), wobei bereits bei der Erzeugung festzulegen ist, für welche Aufgabe der jeweilige Schlüssel eingesetzt werden soll (signieren, verschlüsseln etc.). Weitere Dienste umfassen Funktionen zur Durchführung von RSA-Ver- und Entschlüsselungen sowie zur Erzeugung von RSA-Signaturen (RSA-Engine). Die Datenformate und Implementierungsdetails für RSA-Schlüssel auf dem TPM sind durch den PKCS#1-Standard festgelegt. Ein TPM kann neben dem obligatorischen RSA-Engine auch andere asymmetrische Verfahren verwenden. Die Spezifikation legt hierfür keine Vorgaben fest. Der TPM enthält aber keine Funktion zur Signatur-Validierung, da dies keine stark sicherheitskritische Funktion ist, so dass man sie aus Performanz- und Ressourcen-Gründen aus dem Chip ausgelagert hat.

Symmetrische Verschlüsselungsverfahren dürfen ausschließlich durch TPM-interne Operationen verwendet werden; sie werden nicht nach außen exportiert. Symmetrische Verfahren werden TPM-intern genutzt, um sensitive Daten wie beispielsweise Schlüssel zu sichern (Binding bzw. Wrapping). Beispiele hierfür sind Datei-Schlüssel K , wie sie bei einem verschlüsselnden Dateisystem benötigt werden. Da die Speicherressourcen eines TPM beschränkt sind, ist es nicht möglich, das TPM als sicheren Speicher für alle diese verschlüsselten Objekte und deren Objekt-Schlüssel K zu nutzen. In der TCG-Architektur werden Schlüssel hierarchisch verwaltet. Mit der Erzeugung eines Schlüssels im TPM muss der Erzeuger einen bereits vorhandenen Schlüssel benennen, der zur Verschlüsselung des neuen Schlüssels verwendet wird. Auf diese Weise wird schrittweise die Hierarchie aufgebaut. Die Wurzel einer derartigen Hierarchie bildet der so genannten Storage Root

symmetrische
Verfahren

Key, der im nicht flüchtigen Speicher des Chips verwaltet wird (siehe Seite 588). Die TPM-Spezifikation schreibt keine spezifischen symmetrischen Verfahren vor, so dass also z.B. der AES zum Einsatz kommen kann.

Symmetrische Verfahren werden auch zur Absicherung von Authentifizierungsinformationen (Besitzer-Passwort) im TPM verwendet. Hierbei kommt eine Vernam-Chiffre als ein One-time-Pad zum Einsatz, d.h. der zu verschlüsselnde Strom und der Vernam-Strom werden mittels XOR verknüpft.

Opt-in

Die Funktionseinheit Opt-In dient dazu, die Funktionen des Chip zu aktivieren bzw. zu deaktivieren. Durch eine Deaktivierung werden die Dienste der TCG-Plattform außer Kraft gesetzt. Ein Setzen des Opt-In Flags erfordert die Autorisierung durch den Besitzer des Geräts (Nachweis der Kenntnis der Authentifikations-Daten) oder eine physische Präsenz eines Administrators an der Plattform (z.B. Drücken einer spezifischen Taste oder Schalters). Damit soll verhindert werden, dass ohne das Wissen des Besitzers der Plattform mittels Fernzugriffen der Status des TPM verändert (aktiviert oder deaktiviert) werden kann. Standardmäßig sind die Sicherheitsdienste des TPM deaktiviert, so dass der Besitzer die Kontrolle über die Aktivierung der Dienste erhält.

PCR

Die Platform Configuration Register (PCR) sind i.d.R. Teil des flüchtigen Speichers und dienen zur vertrauenswürdigen Speicherung von 160-Bit Hashwerten, die unter anderem im Verlauf des sicheren Bootvorgangs von Teilen der Systemkonfiguration berechnet werden. Jeder TPM muss mindestens 16 derartige geschützte (shielded) Register umfassen, auf die nur im Rahmen der Ausführung von Sicherheitsdiensten zugegriffen werden darf. Damit ein gespeicherter Integritätswert nicht absichtlich oder unabsichtlich mit neuen Werten überschrieben werden kann, ermöglicht es ein TPM, eine konzeptuell unendlich lange Folge von Integritätswerten zu speichern. Dazu werden alle Update-Werte mit den bereits gespeicherten Werten verknüpft. Im Groben erfolgt dies wie folgt:

$$PCR_i\text{-neu} = \text{HASH}(PCR_i\text{-alt} \mid \text{update-Wert}).$$

Sealing

Neben der üblichen Verschlüsselung bietet das TPM auch noch die Möglichkeit zum so genannten Versiegeln (sealing) über die Funktion *TPM_Seal()*. Dieser Funktion können als Eingabewerte neben dem Schlüssel, mit dem Daten zu verschlüsseln sind und der über ein Key-Handle identifiziert wird, spezielle PCR-Inhalte übergeben werden, so dass in die Verschlüsselung deren Werte, also aktuelle Konfigurationsinformationen mit einfließen. Zur Entschlüsselung bzw. Entsiegelung werden wiederum die PCR-Werte benötigt und das Kommando schlägt fehl, wenn diese Werte nicht mehr mit den ursprünglichen übereinstimmen. Auf diese Weise wird sichergestellt, dass versiegelte Werte nur dann wieder zugreifbar sind, wenn das System sich in einem bekannten Zustand befindet. Mit dem Kommando *TPM_Seal()* ist

es möglich, PCR-Werte festzulegen, die ggf. erst später, zum Zeitpunkt des Entsiegelns vorhanden sein müssen. Auf diese Weise können also einfache Policies für den Umgang mit versiegelten Daten festgelegt werden.

Die Power Detection Komponente überwacht den Energieversorgungszustand der Plattform. Das TPM wird über alle Veränderungen bei der Energieversorgung der Plattform informiert und muss ggf. dafür sorgen, dass die Ausführung von Kommandos angehalten wird, falls eine unzureichende Energieversorgung oder andere physische Beschränkungen beobachtet werden. In einer PC-Umgebung ist dies beispielsweise in der Phase der Durchführung des Power-on self-test (POST) während des Bootens der Fall. Es werden vier wesentliche Energieversorgungszustände unterschieden. Beim initialen Power-on muss das TPM, aktiviert durch ein entsprechendes Kommando seitens des BIOS, dafür sorgen, dass alle PCR-Register bereinigt werden, um die beim nachfolgenden Booten zu berechnenden Hashwerte aufzunehmen. Geht die Plattform in den Zustand Sleep oder Hibernation über, so müssen alle Daten aus dem flüchtigen Speicher persistent gespeichert werden, so dass sie bei einem Resume wieder zur Verfügung gestellt werden können.

Power Detection

Attestation Identity Keys (AIK) sind 2048-Bit RSA-Signaturschlüssel, die dazu verwendet werden, Daten des TPM (z.B. Informationen über seinen Konfigurations-Zustand) zu signieren, so dass ein außen stehender Dritter prüfen kann, ob diese Daten tatsächlich von einem echten, d.h. entsprechend der Spezifikation hergestellten TPM stammen. Auf die Generierung von AIKs sowie deren Zertifizierung durch CAs gehen wir im folgenden Abschnitt genauer ein. AIKs müssen persistent sein, das heißt, sie sind entweder im nicht-flüchtigen Speicher des TPM zu verwalten oder sie können auch verschlüsselt auf Speichermedien außerhalb des TPM abgelegt sein.

AIK

Der Execution Engine führt Programmcode aus, um TPM-Befehle auszuführen, die über den I/O-Port an das TPM übermittelt wurden. Der nicht-flüchtige Speicher verwaltet persistente Daten, auf die wir im nächsten Abschnitt genauer eingehen. Die Programmcode-Komponente enthält Firmware mit der die Integrität der Geräte der TCG-Plattform bestimmt werden kann. Hierbei handelt es sich, falls auf dem TPM realisiert, um die Core Root of Trust for Measurement (CRTM), die aber auch im BIOS außerhalb des TPM implementiert sein kann.

TPM-Schlüssel

Abbildung 11.16 verdeutlicht, für welche Aufgaben der flüchtige (volatile) und nicht flüchtige (non volatile) Speicher des TPM genutzt wird.

Der flüchtige Speicher dient als nicht persistenter, sicherer Schlüsselspeicher (Key Slots) für die Root of Trust for Storage (RTS), die diese Schlüssel

flüchtiger Speicher

benötigt, um Ver- und Entschlüsselungen sowie Signieroperationen durchzuführen. Die Key Slots dienen somit als eine Art Schlüssel-Cache, in den Schlüssel eingelagert (*LoadKey()*) bzw. wieder ausgelagert (*EvictKey()*) werden können. Die Verwaltung der Key-Slots übernimmt ein Key-Cache Managermodul, das außerhalb des TPM implementiert ist und die Schnittstelle zu dem Speichermedium darstellt, auf dem die inaktiven Schlüssel persistent verwaltet werden. Weiterhin enthält der flüchtige Speicher die PCR-Register, die für die Root of Trust for Reporting (RTR) die erstellten Hashwerte sicher verwaltet. Das PCR-Register-Feld kann auch im nicht flüchtigen Speicher des Chips implementiert werden.

Kryptographische Funktionen	Nicht-flüchtiger Speicher	Flüchtiger Speicher
RNG	DIR0, ... (160 Bit)	Key-Slot 0 ... Key Slot9
SHA-1	Endorsement Key (2048 Bit)	
HMAC	Storage Root Key (2048 Bit)	
Schlüsselgenerierung	Owner Authoriz. Key (160 Bit)	
Ver- und Entschlüsselung	ggf. AIKs	PCR0 ... PCR 15

Abbildung 11.16: Flüchtiger und Nicht-flüchtiger TPM-Speicher

NV-Speicher

Der nicht-flüchtige Speicher (NV Storage) des TPM enthält ein²⁹ 160-Bit Data Integrity Register (DIR), in das der Besitzer sicherheitskritische Daten ablegen kann. Ein Beispiel für ein solches Datum ist das vom Hersteller des TPM signierte Platform Endorsement-Zertifikat (s.u.), das die korrekte und sichere Herstellung des Chips und seines eindeutigen Schlüssels (Endorsement Key) bestätigt. Der Speicherbereich des DIR-Speichers ist zugriffskontrolliert. Das bedeutet, dass mittels entsprechender Attributierungen festgelegt werden kann, ob ein Lese/Schreibzugriff direkt zulässig ist, ob er abhängig vom Speicherort des Datums zulässig ist oder ob spezielle Integritäts-Werte der Plattform vorzuweisen sind, bevor ein Zugriff erlaubt wird.

²⁹ In früheren Versionen konnten es auch noch mehrere DIR-Register sein.

Sicherer Schlüsselspeicher

Im nicht flüchtigen Speicher des Chips werden darüber hinaus drei sicherheitskritische Schlüssel verwaltet. Es handelt sich dabei um den Endorsement Key, den Storage Root Key sowie den Owner Authorization Key.

Der Endorsement Key (EK) ist ein RSA-Schlüsselpaar (2048 Bit), das bei der Chipherstellung erzeugt wird. Er dient dazu, gegenüber Dritten zu bestätigen, dass das TPM ein authentisches TPM ist. Der EK wird vom Hersteller des TPM-Chips generiert (entweder innerhalb des TPM mittels des Kommandos *TPM_CreateEndorsementKey* oder außerhalb und zum TPM exportiert) und der Hersteller erstellt dazu auch ein signiertes Endorsement Key Zertifikat, das die Authentizität des EK attestiert.

Endorsement Key

Der Endorsement Key ist eindeutig dem TPM zugeordnet, deshalb darf auch der private Schlüssel dieses Schlüsselpaares das TPM nie verlassen. Er wird nur zum Entschlüsseln von Daten verwendet, niemals jedoch zum Signieren.

Die Attestation Identity Key (AIK) sind ebenfalls 2048-Bit RSA-Schlüsselpaare und ein solches Paar (es können virtuell beliebig viele solcher Schlüssel auf dem TPM erzeugt werden) kann erst nach der Inbesitznahme des TPM durch einen Besitzer erzeugt werden. Der Besitzer kontrolliert die Erzeugung und Aktivierung aller AIKs. Ein AIK dient dazu, Daten, die im TPM generiert wurden (z.B. Inhalte von PCR-Registern, TPM-Statusinformationen), zu signieren. AIKs werden hauptsächlich verwendet, um auf Anfrage die Authentizität der Plattform bzw. des TPM gegenüber einer dritten Instanz zu bestätigen, ohne jedoch gleichzeitig dieser Instanz auch die tatsächliche Identität (nämlich den EK) der Plattform offen zu legen. Um dieses Ziel zu erreichen, wird dem öffentlichen AIK-Schlüssel ein AIK-Zertifikat assoziiert, das die Gültigkeit des öffentlichen AIK-Schlüssels bestätigt, ohne die Plattform-Identität preiszugeben. Der Besitzer des TPM kann eine Zertifizierungsinstanz seines Vertrauens (Privacy CA) dafür bestimmen, AIK-Zertifikate auszustellen. Für unterschiedliche Anfragen kann der Plattform-Besitzer unterschiedliche AIKs erzeugen und auch unterschiedliche CAs zu deren Zertifizierung bestimmen, um auf diese Weise zu verhindern, dass gezielt Profile erstellt werden. Die AIK-Schlüssel sind somit eine Art Pseudomisierungs-Konzept, so dass ein Besitzer einer Plattform unter verschiedenen Identitäten gegenüber Anfragern agieren kann. AIKs können aber auch über die Funktion *TPM.CertifyKey* dazu verwendet werden, nachzuweisen, dass ein bestimmter Schlüssel im TPM gespeichert ist.

AIK

Pseudonym

AIK-Schlüsselpaare werden im TPM mit dem Kommando *TPM_MakeIdentity* generiert, wobei gleichzeitig die Erstellung des Zertifikats, des AIK-Credential für den öffentlichen AIK-Schlüssel an eine Privacy CA in Auftrag gegeben wird. Dieser AIK-Request wird von dem generierten AIK signiert.

AIK-Zertifizierung

Der Request enthält u.a. auch das EK-Zertifikat, das wiederum bei der Erstellung der Antwort an den TPM durch die angefragte Privacy CA verwendet wird, um die Daten der Antwort mit dem öffentlichen EK zu verschlüsseln. Die Antwort der Privacy CA enthält das verschlüsselte AIK-Zertifikat. Der TPM benötigt nun seinen privaten EK, um die Daten, und damit das AIK-Zertifikat, das für die Attestierung erforderlich ist, zu entschlüsseln. Den öffentlichen EK-Schlüssel erhalten nur solche Stellen, die vom Besitzer des TPM als vertrauenswürdige Zertifizierungsstellen (Privacy CA) festgelegt wurden.

Der Zugriff auf den privaten EK ist nur dem Besitzer des Geräts möglich, d.h. zum Zugriff muss der Besitzer die Kenntnis der Authentifizierungsdaten nachweisen. Auch der Zugriff auf den öffentlichen Schlüsselteil unterliegt einer entsprechenden Zugriffsbeschränkung. Bis zur Version 1b war weder ein Löschen noch ein Ändern des Endorsement Keys vorgesehen, aber seit der TPM Version 1.2. sind Änderungen möglich (`TPM.CreateRevocableEK`); wobei ein Löschen laut Spezifikation nicht möglich ist. Die Möglichkeit des Änderns ist die Reaktion auf massive Vorbehalte von Kritikern, die die Verankerung eines statischen und von Herstellern bestimmmbaren Schlüssel in die TCG-Plattform als einen gravierenden Eingriff in die Privatsphäre der Benutzer einstufen.

Besitzübernahme

Nach der Auslieferung einer Plattform kann sie explizit in Besitz genommen werden. Die Besitzübernahme erfolgt durch die Ausführung des *Take_Ownership* Kommandos des TPM, das den Nachweis der physischen Anwesenheit des Besitzers am Gerät erfordert. Das *Take_Ownership*-Kommando benötigt als erstes Argument einen 160-Bit SHA-1 Hashwert eines Passwortes (`ownpass`), das der Besitzer als Authentifizierungs-Passwort festlegt. Dieser 160 Bit Wert wird im TPM mit dem öffentlichen Endorsement Key der Plattform verschlüsselt und sicher in dem nicht flüchtigen Speicher des TPM verwaltet (Owner Authentication Key). Durch diese Bindung ist sichergestellt, dass das gehaschte und signierte Authentifizierungs-Passwort des Besitzers nur auf der spezifischen Plattform gültig ist. Der Owner Authentication Key ist ein gemeinsames Geheimnis zwischen dem Besitzer und dem TPM.

Falls die Ausführung eines TPM-Kommandos die Owner-Berechtigung erfordert (z.B. der Zugriff auf den Schlüsselspeicher), so muss der Besitzer die Kenntnis dieses gemeinsamen Geheimnisses nachweisen, um die Zugriffserlaubnis zu erlangen. Falls der Chip bereits im Besitz eines Benutzers ist, bleibt die Ausführung des *Take_Ownership*-Kommandos wirkungslos. Ein Besitzerwechsel ist nur dadurch möglich, dass eine spezielle Reset-Sequenz aktiviert wird, die die physische Anwesenheit eines Benutzers erfordert.

Owner Authentication Key

Der Storage Root Key (SRK) ist ebenfalls ein 2048-Bit RSA-Schlüsselpaar. Im Unterschied zum Endorsement Key wird dieses aber erst dann

Storage Root Key

generiert, wenn ein Benutzer den Chip in Besitz nimmt, also das Kommando *TPM_TakeOwnership* ausführt. Der Storage Root Key bildet die Wurzel einer Schlüsselhierarchie, die dynamisch aufgebaut wird, sobald ein weiterer Schlüssel auf dem Chip generiert wird. Der Storage Root Key wird Passwort-gesichert im TPM-Speicher abgelegt und der private Schlüsselteil kann den Chip nicht verlassen. Das erforderliche Passwort wird bei der Ausführung der *Take_Ownership* Operation als zweiter Parameter in Form eines SHA-1-Hashwertes (srkpass) übergeben. Der öffentliche Schlüsselteil wird nur dazu verwendet, um private Schlüssel der ersten Hierarchiestufe zu verschlüsseln, so dass diese außerhalb des Chips abgelegt werden können. Storage Root Keys können vom Besitzer gelöscht werden. Mit jeder Besitzübernahme wird eine neue Schlüsselhierarchie gegründet, so dass die Schlüssel derjenigen Benutzer, in deren Besitz der Chip zuvor gewesen ist, für den neuen Besitzer nicht zugreifbar sind. Das bedeutet natürlich auch, dass diese verschlüsselten Daten verloren sind, wenn keine expliziten Recovery-Maßnahmen in der Software vorgesehen werden. Entsprechende Features sind beispielsweise im Infineon TPM realisiert.

Das TPM bietet Kommandos an, um Schlüssel on-Chip zu generieren und eine Schlüsselhierarchie ausgehend vom Storage Root Key aufzubauen. Bei der Erzeugung eines Schlüssels muss der Aufrufer gleichzeitig auch denjenigen bereits existierenden Schlüssel angeben, mit dem der neu erzeugte Schlüssel verschlüsselt werden soll. Nehmen wir an, dass dies der bereits existierende Schlüssel K_{wrap} sei. Da Schlüssel auch passwortgeschützt abgelegt werden, ist der korrekte SHA-1-Hashwert des Passwortes für den Schlüssel K_{wrap} vorzuweisen, damit K_{wrap} in den Schlüssel-Cache des TPM geladen werden kann. Die Erzeugungsfunktion benötigt zusätzlich auch ein zu wählendes Passwort, das zur zugriffskontrollierten Verwahrung des neuen Schlüssels verwendet wird.

Schlüsselhierarchie

Schlüsselattribute und -typen

Die in einem TPM erzeugten Schlüssel können migrierbar oder nicht-migrierbar sein, wobei dieses Attribut statisch mit dem Schlüssel assoziiert ist. Ist ein Schlüssel bei seiner Erzeugung als migrierbar attribuiert worden, so bedeutet das, dass er auf andere Speichermedien und insbesondere auch auf ein anderes TPM übertragen werden darf. Dies kann dazu verwendet werden, einen Schlüssel dem Benutzer auf unterschiedlichen Geräten zur Verfügung zu stellen, so dass er in der Lage ist, unabhängig von dem genutzten Gerät seine Daten zu entschlüsseln. Da migrierbare Schlüssel konzeptuell aus dem TPM entfernt werden dürfen, verlieren sie dadurch natürlich auch an Vertrauenswürdigkeit, da damit der Hardware-unterstützte Schutz des Schlüssels eingebüßt wird, falls der Schlüssel nicht gesichert von einem TPM-Schutzbereich in einen anderen Schutzbereich übertragen wird.

migrierbar

Schlüssel, die zum Signieren verwendet werden, sollten deshalb nicht als migrierbare Schlüssel attributiert werden.

nicht-migrierbar

Ein nicht-migrierbarer Schlüssel ist somit permanent mit einem eindeutig identifizierbaren TPM assoziiert. Mit anderen Worten, über nicht-migrierbare Schlüssel kann eine Identifikation bzw. Authentifikation des TPM und der Plattform erfolgen. Wie wir gesehen haben, ist dies die Aufgabe der AIK Schlüssel, so dass klar ist, dass solche Schlüssel als nicht-migrierbar zu erzeugen sind. Auch der Endorsement Key und der SRK sind nicht-migrierbare Schlüssel.

Zusammenfassend können wir festhalten, dass die TCG-Architektur 7 Schlüsseltypen definiert, wobei mit jedem Typ eine Anzahl von Nutzungs-Restriktionen verbunden ist.

- *Signaturschlüssel* sind asymmetrische Schlüssel, die zum Signieren von Anwendungsdaten sowie Nachrichten verwendbar sind. Sie können als migrierbar oder aber auch als nicht migrierbar attributiert werden.
- *Storage Keys* sind ebenfalls asymmetrische Schlüssel, die zum Verschlüsseln von anderen Schlüsseln oder von Daten, die außerhalb des TPM verwaltet werden, verwendbar sind.
- *Identity Keys* (bzw. AIK Keys) sind nicht-migrierbare Signaturschlüssel, die ausschließlich zum Signieren von solchen Daten verwendet werden dürfen, die vom TPM stammen.
- Der *Endorsement Key* (EK) ist ein asymmetrischer nicht-migrierbarer Schlüssel zum Entschlüsseln von Authentifizierungsdaten des Besitzers sowie von Nachrichten, die im Zusammenhang mit der AIK-Erzeugung generiert werden.
- *Bind Keys* werden verwendet, um kleine Datenmengen (z.B. symmetrische Schlüssel) auf einer Plattform zu ver- und auf einer anderen Plattform wieder zu entschlüsseln.
- *Legacy Keys* sind migrierbare Schlüssel, die in dem TPM erzeugt werden und zum Signieren und Verschlüsseln verwendet werden.
- *Authentication Keys* sind symmetrische Schlüssel, die für Transport Sessions³⁰ verwendet werden, um den vertraulichen und integeren Transfer von Daten aus dem TPM zu einem Betriebssystem- oder Anwendungsprozess zu gewährleisten.

Zertifikate

Um die Identität einer TCG-Plattform oder eines AIK einer Plattform zu attestieren, werden unterschiedliche Zertifikattypen (Credentials) verwendet.

³⁰ Das Konzept der Transport Sessions wurde erst an der Version TPM v1.2 eingeführt.

Die TCG definiert fünf derartige Typen, nämlich Endorsement oder EK Credential, Conformance Credential, Platform Credential, Validation Credential sowie Identity oder AIK Credential. Jeder Credential-Typ stellt nur genau die Informationen bereit, die zur Durchführung einer spezifischen Klasse von Operationen benötigt wird. TCG-Credentials sind Datenstrukturen in ASN.1 Notation, die auf X.509 Zertifikat-Strukturen basieren, aber zusätzlich noch weitere spezielle Datenfelder spezifizieren.

Das Endorsement Credential wird von dem Hersteller des TPM ausgestellt und signiert, der auch den Endorsement Key des TPM generiert. Das Credential bestätigt, dass die Schlüsselerzeugung und Übertragung in das TPM sicher gemäß der Vorgaben der Spezifikation erfolgt ist³¹. Das Endorsement Credential enthält u.a. den Namen des TPM Herstellers, eine TPM Versionsnummer und natürlich den öffentlichen Endorsement Key. Zu beachten ist, dass es sich bei dem in dem Zertifikat enthaltenen EK-Schlüssel zwar um dessen öffentlichen Anteil handelt, dies aber dennoch eine sensible Information darstellt, da der Schlüssel eindeutig ein TPM identifiziert und über die Nutzung des Schlüssels Rückschlüsse auf den Besitzer des TPM gezogen werden könnten.

Endorsement

Die Conformance Credentials können von einer beliebigen Instanz (Hersteller, Händler etc.) ausgestellt werden, solange sicher gestellt ist, dass die Instanz in der Lage ist, einen TPM zu evaluieren. Mit seiner Signatur für das Credential bestätigt der Aussteller, dass das Design und die Implementierung der Trusted Building Blocks (TBB) des TPM die Spezifikation erfüllt. Conformance Credentials sollten somit nur von dazu akkreditierten Instanzen ausgestellt bzw. akzeptiert werden.

Conformance

Das Platform Credential wird vom Hersteller der TCG-Plattform oder aber auch dem Händler bzw. einer Instanz mit genügender Vertrauenswürdigkeit ausgestellt. Das Credential identifiziert den Hersteller der Plattform sowie einige Eigenschaften der Plattform und es enthält einen Verweis auf das Endorsement-Credential des TPM sowie auf die Conformance Credentials der Plattform. Ein solcher Verweis ist einfach ein MAC-Wert des entsprechenden Credentials. Der Zweck des Platform Credentials besteht darin, zu bescheinigen, dass die Plattform ein TPM der Art und Ausprägung enthält, wie er im Endorsement Credential beschrieben ist. D.h. über das Platform Credential erfolgt eine Art bestätigte Bindung des TPM an eine spezifische Rechner-Plattform. Analog zum Conformance Credentials, das die Eigenschaften des TPM der Plattform durch einen akkreditierten Dritten bestätigt, gibt es auch Platform Conformance Credentials, die von Plattform-Evaluatoren ausgestellt und signiert sind und die Sicherheitseigenschaften der gesamten Plattform attestieren.

Platform

³¹ U.a. alle Kopien des Schlüssels sind gelöscht.

Validation

Um Herstellern die Möglichkeit zu geben, auch einzelne sicherheitskritische Komponenten nach dem erfolgreichen Testen in einer sicheren Umgebung als vertrauenswürdig (z.B. frei von Trojanern und Backdoors) zu testieren, wurde das Validation Credential eingeführt. Beispiele für Komponenten, für die ein Validation Credential ausgestellt sein kann, sind Video-, Netzwerk- oder Platten-Adapter, Prozessoren, Controller, Tastatur und Maus oder aber auch Software. Ein Validation Credential kann von jeder Instanz, die die korrekte Funktionsweise der jeweiligen Komponente geprüft hat, ausgestellt werden.

Attestation Identity

Das Attestation Identity Credential beglaubigt und identifiziert private AIK-Schlüssel, die zum Signieren von TPM-Werten verwendet wurden. Bei der Erzeugung eines AIKs sendet das TPM eine Anfrage zur Ausstellung eines AIK-Credentials an die dafür vom Benutzer ausgewählte Privacy CA (vgl. Abbildung 11.17). Diese Anfrage enthält neben dem öffentlichen AIK-Schlüssel auch das Endorsement, das Platform und das Conformance Credential. Da das Endorsement Credential den öffentlichen EK der Plattform enthält und damit die eindeutige Identifikation der Plattform ermöglicht, ist es wichtig, dass der Besitzer seine Privacy CAs, an die diese Information gesandt wird, sorgfältig auswählt. Dies ist wichtig, da diese CA eine Bindung von AIK und TPM-Endorsement Key sowie ggf. weiteren Informationen vornimmt, wodurch die Privatsphäre des Benutzers gefährdet werden könnte, falls diese Information in falsche Hände gelangt.

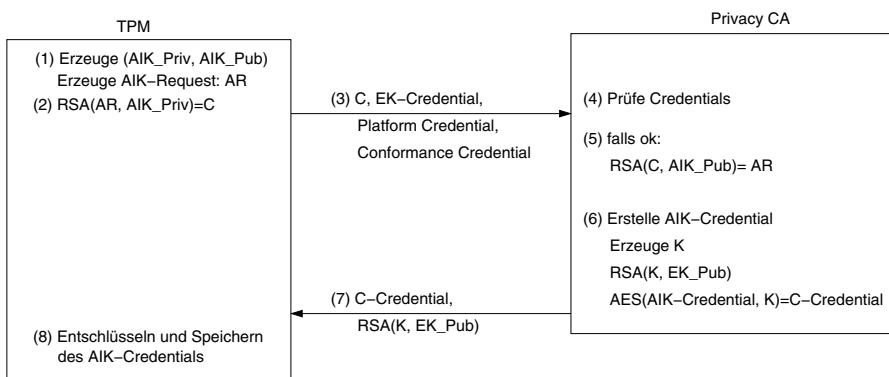


Abbildung 11.17: Stark vergrößertes Protokoll zur Ausstellung eines AIK-Credentials

Die Privacy CA prüft die Credentials bevor sie ein AIK-Credential ausstellt, das unter anderem den öffentlichen AIK-Schlüssel enthält. Mit dem Signieren eines solchen Credentials bestätigt der Aussteller, dass der AIK zu einem Spezifikations-konformen (genuine) TPM gehört und dass der AIK an gültige Endorsement, Platform und Conformance Credentials gebunden ist. Die

Privacy CA verschlüsselt das ausgestellte AIK-Credential mit einem von der CA erzeugten Session Schlüssel K . Abschließend sendet die CA das Zertifikat zusammen mit dem K , der zuvor mit dem öffentlichen EK-Schlüssel des TPM verschlüsselt wurde, an das TPM zurück. Durch die Ausführung des *TPM_ActivateIdentity* Kommandos ist das TPM in der Lage, das AIK-Credential zu prüfen, indem es mit dem zugehörigen privaten EK-Schlüssel zunächst den Session Schlüssel K wiederherstellt.

Attestierung

Diese vielfältigen Credentials dienen dazu, das Vertrauen in die Integrität und Sicherheitsfunktionen eines spezifischen Moduls oder einer ganzen Rechner-Plattform zu steigern und eine entsprechende Integritätsaussage über die Plattform auf Anfrage eines Dritten zu attestieren. Der Vorgang läuft vergröbert wie folgt ab (vgl. Abbildung 11.18).

Attestierung

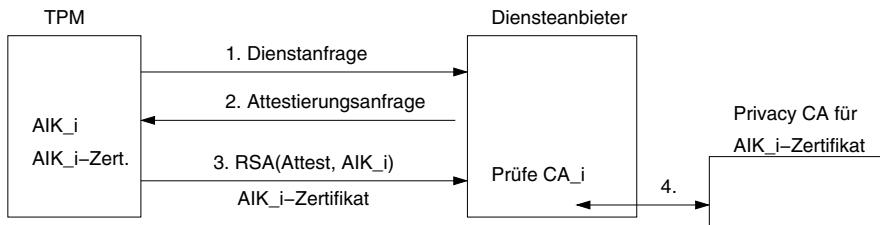


Abbildung 11.18: Attestierung eines TPM gegenüber einem Dienstanbieter

Möchte die Plattform (bzw. der Besitzer der Plattform) einen Dienst eines Dritten nutzen (z.B. Zugriff auf spezielle Datenobjekte in einem Unternehmen), so stellt er an diesen eine entsprechende Anfrage (Schritt 1). Der Dienste-Anbieter versichert sich nun darüber, ob der Anfrager über eine Plattform verfügt, deren Konfigurations-Werten er vertraut, indem er an die Plattform eine Attestierungs-Anfrage stellt (Schritt 2). Die Root of Trust for Reporting Komponente (RTR) als Teil des TPM hat die Aufgabe, entsprechende Attestierungen zu erstellen. Dazu wählt sie einen AIK aus, um damit Daten, die den Integritätszustand der Konfiguration beschreiben (z.B. Inhalte der Platform Configuration Register), mit dem privaten Teil des AIK zu signieren. Um Wiedereinspielungen zu verhindern, wird dem Attest vor dem Signieren noch eine Nonce hinzugefügt. Dieses Attest wird zusammen mit dem zugehörigen AIK Credential an den Anfrager gesendet (Schritt 3). Dieser prüft die Vertrauenswürdigkeit derjenigen CA, die das AIK Credential erstellt hat (Schritt 4). Bei einem positiven Ausgang der Überprüfung wird der Anfrager den öffentlichen AIK-Schlüssel aus dem Credential verwenden, um die Signatur des Attests zu prüfen. Falls die attestierte Konfigurations-

daten den Anforderungen des Anfragers entsprechen, wird er die angefragte Kommunikation mit der Plattform aufnehmen.

Es gibt aber bereits Angriffe, die zeigen, dass man dieses Konzept aushebeln und einen vertrauenswürdigen Zustand maskieren kann (siehe [167]).

DAA

Da bei dieser Vorgehensweise nach wie vor die Gefahr besteht, dass durch eine Zusammenarbeit zwischen der CA, die einen AIK der Plattform zertifiziert und einem Dienst, der lediglich Attestierungsanfragen stellt, ein Nutzungsprofil des Plattformbesitzers erstellbar ist, wurde in der TPM Spezifikation 1.2. das Konzept der direkten anonymen Attestierung eingeführt. Das Ziel des Direct Anonymous Attestation³² (DAA) Konzepts war es, die Privacy-Eigenschaften einer Plattform zu verstärken, indem darauf verzichtet wird, den öffentlichen Endorsement Key an CAs auszugeben. Ferner sollte generell auf die Vermittlungsfunktion derartiger CAs verzichtet werden können. Das DAA-Konzept basiert auf Zero-Knowledge Beweisen, indem ein Anfrager von der Authentizität eines TPM überzeugt wird, ohne dass Informationen, die das TPM eindeutig identifizieren, bekannt gegeben werden. Dazu werden DAA-Credentials von einem externen Aussteller erzeugt, wobei ein TPM-eindeutiges Geheimnis verwendet wird, das aber das TPM nie verlässt.

11.4.4 Sicheres Booten

Booten

Ein Ziel der TCG-Plattform ist es, ein sicheres Booten zur Verfügung zu stellen. Im Folgenden wird erklärt, was im Rahmen der TCG darunter zu verstehen ist und welche Sicherheitsziele mit den bereitgestellten Diensten gewährleistet werden können. Als Ausgangspunkt der Diskussion ist es hilfreich, sich zunächst noch einmal den Bootvorgang eines herkömmlichen PCs vor Augen zu führen. Es handelt sich hierbei um einen typischen Bootstrapping-Vorgang, d.h. einen schrittweisen Prozess (vgl. Abbildung 11.19).

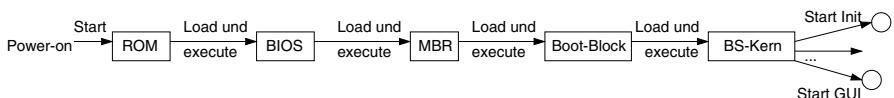


Abbildung 11.19: Bootstrapping

Bootstrapping

Bootstrapping

Beim Systemstart wird die Ausführung eines Bootloader-Programms gestartet, dessen initiale Befehle meist im ROM auf dem Motherboard abgelegt sind. Durch die Ausführung dieser sehr kurzen Startsequenz wird als erstes

³² vgl. <http://www.hpl.hp.com/techreports/2004/HPL-2004-93.pdf>

das BIOS (Basic Input/Output System), das den eigentlichen Bootloader-Code enthält, in den RAM geladen und dessen Ausführung initiiert. Das BIOS wird heutzutage normalerweise nicht mehr selber vollständig im ROM gespeichert, da dieser unveränderlich ist, so dass Änderungen am Bootstrap-Vorgang eine Änderung der Hardware erfordern würden. Als erster Schritt bei der Ausführung des BIOS-Codes wird ein Power-on-Self Test (POST) durchgeführt, um die wesentlichen Eigenschaften des Rechners zu prüfen. Anschließend wird das eigentliche Bootloader-Programm in den Hauptspeicher geladen. Der erste Teil dieses Programm befindet sich im MBR (Master Boot Record) auf der Festplatte in Platten-Sektor 0, also an einer festgelegten Stelle (vgl. Abbildung 11.20).

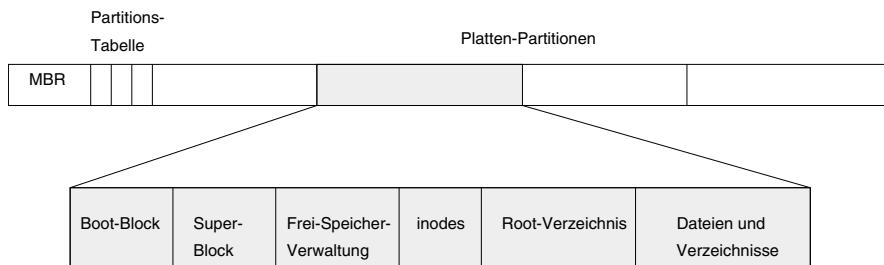


Abbildung 11.20: Festplattenlayout am Beispiel von Unix/Linux

Der MBR enthält neben dem Bootloader-Code auch eine Tabelle mit den Anfangsadressen der gültigen Plattenpartitionen. Durch die Ausführung des Codes des MBR wird im nächsten Schritt der Inhalt des Boot-Blocks der aktiven Partition von der Platte in den Hauptspeicher geladen. Der Boot-Block enthält den nächsten und hauptsächlichen Teil des Codes des Bootloaders, der durch das Laden des Boot-Blockes in den Speicher gebracht und ausgeführt wird. Durch die Ausführung des Bootloader Codes wird nun im nächsten Schritt der Betriebssystem-Kern von der Platte geladen und gestartet. Die weiteren Boot-Schritte des Betriebssystemkerns umfassen die Initialisierung des Rechners (u.a. der CPU-Register, EA-Controller, des Speichers), das Einlesen von Informationen über das Dateisystem (u.a. Superblock, Dateibeschreibungen (z.B. Unix-inodes)) und die Ausführung des ersten Prozesses (init-Prozess), der u.a. alle Gerätetreiber von angeschlossenen Geräten von der Platte lädt und weitere System-Prozesse wie u.a. die Login-GUI startet.

Sicherheitsprobleme beim Booten

Sicherheitsprobleme beim Booten ergeben sich an vielfältigen Stellen. So kann beispielsweise der MBR, der vom BIOS eingelesen wird, modifiziert worden sein und einen Boot-Sektor-Virus enthalten. Da der Code des MBR

Probleme

in den RAM geladen wird bevor das Betriebssystem geladen ist, erfolgen natürlich auch noch keinerlei Kontrollen. Der Code aus dem MBR wird somit im Kernel-Modus (privilegiert), ohne MMU-Schutz und ohne Kontrollen des Betriebssystems (auch natürlich ohne Kontrollen von Antiviren-Programmen) ausgeführt. Weiterhin kann der Boot-Block zerstört oder manipuliert sein. Das führt dazu, dass das System unter Umständen nicht mehr bootbar ist, was auf eine Variante eines Denial-of-Service Angriffs hinauslaufen würde. Weit gravierender wäre aber eine gezielte Modifikation des Boot-Blocks, so dass durch eine Veränderung der Anfangsadresse des zu ladenden System-Codes ein modifizierter Betriebssystem Kern geladen wird. Ein modifizierter Kern kann manipulierte Standardsystemdienste enthalten wie z.B. modifizierte Versionen der Dienste *login*, *ls*, *netstat*, die beliebige Schadroutine in Form von Trojanischen Pferden ausführen könnten. Weiterhin kann über einen manipulierten Betriebssystem-Code dafür gesorgt werden, dass z.B. unter Windows nicht-vertrauenswürdige Dienste als Services in der Registry registriert und automatisch mit den höchsten Privilegien bei jedem Systemstart gestartet werden. Wird der Super-Block manipuliert, so sind unter Umständen Dateien oder Verzeichnisse nicht mehr auffindbar und werden i-nodes manipuliert, so können damit die in den i-nodes verwalteten Rechtevergaben manipuliert werden. Analoges gilt auch für die Datenstrukturen anderer Betriebssysteme, die die Informationen über die Zugriffsberechtigungen beispielsweise in Zugriffskontrolllisten (ACLs) verwalten. An den skizzierten Bedrohungen ist erkennbar, dass ein abgesicherter Boot-Prozess sehr wünschenswert ist, um das Vertrauen in ein System zu erhöhen.

Sicheres Booten und TCG

sicheres Booten?

Das sichere Booten mittels der Konzepte der TCG-Plattform hat zum Ziel, Angriffe, wie sie gerade angesprochen wurden, abzuwehren und zu verhindern, dass eine manipulierte Systemkonfiguration geladen und gestartet wird. Das bedeutet, dass dafür gesorgt werden muss, dass das System von einer garantiert unveränderten Basis (Integritätsgarantie) gebootet wird. Wir beschreiben im Folgenden das Vorgehen beim Booten in TCG-Plattformen. Dabei wird man sehen, dass dieser Bootvorgang keineswegs ein sicheres Booten ist, sondern dass lediglich schrittweise der beim Booten herrschende Systemzustand im sicheren Speicher des TPM aufgezeichnet wird. Dies ist der Vorgang, der unter TCG als *integrity measurement* bezeichnet wird. Das heißt, es werden nur Reports über den Systemzustand erstellt. Die Auswertung dieser Daten und deren Abgleich mit vorgegebenen Werten sowie die Spezifikation eines Regelwerks, das festlegt, wie im Falle eines Abweichens von den vorgegebenen Werten zu verfahren ist, ist nicht Gegenstand der TCG-Spezifikationen. Das bedeutet, dass wenn nicht weitere Maßnahmen – in der Regel wird das dann die Aufgabe des Betriebssystems sein –

ergriffen werden, auch unter TCG ein manipuliertes System gebootet werden kann, so dass der Begriff sicheres Booting, wie er in diesem Kontext häufig verwendet wird, nicht nur irreführend, sondern sogar falsch ist.

Aufbau einer Vertrauenskette

Der Bootvorgang und das schrittweise Aufzeichnen von Konfigurationszuständen beginnt unter TCG bei der vertrauenswürdigen, unveränderlichen Basis, die durch das CRTM (Core Root of Trust Measurement) geliefert wird. Das CRTM kann entweder Bestandteil des TPM oder eine BIOS-Erweiterung bzw. ggf. das gesamte BIOS sein. Um BIOS-Herstellern jedoch die Flexibilität zu erhalten, BIOS-Updates zur Verfügung zu stellen, wird das CRTM häufig aus dem BIOS ausgelagert und in Firmware in dem TPM realisiert.

Vertrauenskette

Bei einem Reset bzw. Neustart eines Systems wird stets zuerst der Code des CRTM in den Speicher geladen und ausgeführt. Der Hersteller des TPM bzw. des Motherboards, falls das CRTM Bestandteil des BIOS ist, ist verantwortlich dafür, die Integrität des CRTM-Codes zu garantieren, d.h. erforderliche Updates und Code-Modifikationen müssen unter der Kontrolle des Herstellers erfolgen. Ausgehend von der Vertrauenswurzel des CRTM wird schrittweise eine Vertrauenskette aufgebaut, die das BIOS, den Bootloader bis hin zum Betriebssystemkern umfasst.

Abbildung 11.21 veranschaulicht den Bootvorgang unter TCG.

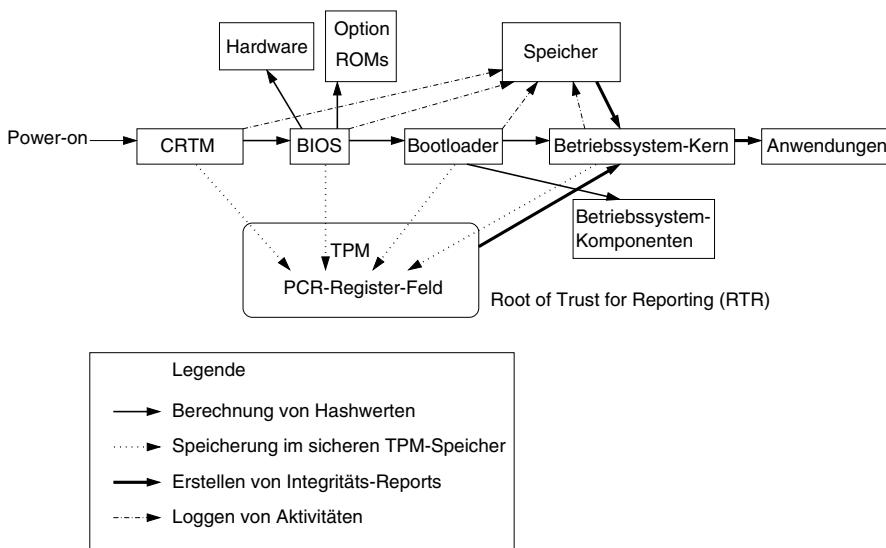


Abbildung 11.21: Berechnen von Integritätswerten beim Booten

BIOS

Als erstes berechnet das CRTM einen Hashwert über sich selbst und speichert diesen in das Register PCR[0] bevor der CRTM-Code in den Speicher geladen wird. Dieser initiale Hashwert wird in der weiter oben beschriebenen Weise fortgeschrieben, indem der Hashwert des BIOS sowie Hashwerte über die Firmware auf dem Motherboard berechnet und schrittweise mit den bereits bestehenden Werten verknüpft werden. Als nächstes wird der Zustand des Motherboards (u.a. welche Hardwarekomponenten gehören dazu, wie sind sie konfiguriert, mit welchen Werten sind wichtige Flag-Bits belegt) gehasht und im PCR[1] abgelegt. Die Überprüfbarkeit der Integrität derartiger Hard- und Firmware ist ein wichtiger Schritt, um eine vertrauenswürdige Basis zu schaffen, da z.B. Busmaster-fähige Graphikkarten direkt auf den Speicher zugreifen können und so alle Sicherheitsvorkehrungen der CPU (wie z.B. den Speicherschutz) oder des Betriebssystems umgehen können. Anschließend werden die Hashwerte von Option ROMs vom BIOS berechnet und in die Register PCR[2] und PCR[3] gespeichert, bevor diese Option ROMs geladen und ausgeführt werden.

Bootloader

Als nächstes (vgl. Abbildung 11.19) muss der MBR geladen werden. Bevor dies geschieht, wird dessen Hashwert berechnet und in dem PCR[4] abgelegt. Die nächsten beiden PCR-Register nehmen Hashwerte u.a. von solchen Daten auf, die im MBR gespeichert sind, wie Informationen über das Plattenlayout. In das Register PCR[6] werden Hashwerte von gelogten Zustandsübergängen abgelegt, wie beispielsweise der Kontrolltransfer vom BIOS zu einem der Option ROMs und auch der entsprechende Kontroll-Return. Im weiteren Verlauf der Ausführung werden in diesem Register auch Aktivitäten wie die korrekte Eingabe des Administrator-Passwortes oder die Angabe der korrekten Besitzer-Authentifikationsdaten festgehalten. Nach der Berechnung des MBR-Hashwertes wird der restliche Code des Bootloaders ebenfalls gehasht bevor er schrittweise, wie oben erläutert, in den Hauptspeicher geladen und ausgeführt wird.

Betriebssystem

Anschließend wird der Code des zu ladenden Betriebssystemkerns wiederum zunächst gehasht und dann ebenfalls in PCR[4] abgelegt. Erst danach wird der Betriebssystem-Code in den Speicher geladen. Das Betriebssystem kann nun das TPM im weiteren Verlauf verwenden, um Hashwerte von Anwendungen zu erstellen und abzulegen oder Anfragen nach Reports über die Systemkonfiguration zu beantworten. Weiterhin muss auch im laufenden Betrieb dafür gesorgt werden, dass alle sicherheitskritischen Konfigurationsänderungen der Plattform in den PCR-Registern festgehalten werden. Das Konzept des schrittweisen Messens der Integritätswerte ermöglicht es auch, den Code des Betriebssystem-Kerns verschlüsselt auf der Festplatte abzulegen, wodurch eine gezielte Manipulation des Codes verhindert werden kann. Zur Entschlüsselung des Codes kann ein symmetrischer Schlüssel verwendet werden, der im sicheren Speicher des TPM verwaltet wird.

Damit die berechneten Hashwerte nicht nur aufgezeichnet, sondern auch überprüft werden können, müssen Referenzwerte von gültigen Hashwerten, die vertrauenswürdigen Konfigurationen entsprechen, zu Rate gezogen werden. Entsprechende Referenzwerte müssen dem System über einen vertrauenswürdigen Kanal zugreifbar sein (z.B. von einer vertrauenswürdigen URL abfragen) und es müssen Regeln (Policies) für die durchzuführenden Kontrollen festgelegt werden. Dies ist jedoch nicht mehr Gegenstand der TCG-Spezifikation.

Während des laufenden Betriebs kann das Betriebssystem selbst die Dienste der Plattform zur Berechnung bzw. zum Erstellen von Berichten über Konfigurationen nutzen. So ist es beispielsweise möglich, vor dem Laden eines Programms (z.B. eines Treibers), dessen Hashwert zu berechnen und mit vorab gespeicherten Werten zu vergleichen. Andererseits ist es auch möglich, dass eine Anwendung selbst eine Policy spezifiziert, die besagt, dass sie nur auf bestimmten Plattformen ausführbar ist, so dass die Plattform zunächst nachweisen muss, dass sie über die geforderten Eigenschaften (z.B. Konfigurationen, Zertifikate, Rechte) verfügt.

Anwendungsfelder für Trusted Computing

Ein interessantes Anwendungsfeld für die Trusted-Computing Konzepte liegt im Bereich der betrieblichen Geschäftsprozesse. In Unternehmensszenarien ist der Besitzer der Plattform das Unternehmen, das unter Nutzung der TCG-Mechanismen unternehmensweite Policies zum Umgang mit Dokumenten (welche Hard- und Software muss installiert sein, welche Software darf nicht installiert sein, etc.) bzw. zum Austausch von Dokumenten festlegen kann. Die TCG-Mechanismen können dazu eingesetzt werden, mandatorische Regelungen eines Unternehmens, wie das generelle Verbot für Mitarbeiter, Software auf den Rechnern des Unternehmens selbsttätig zu installieren, zu formulieren und automatisch durchzusetzen.

Anwendungs-szenarien

Unternehmensübergreifende Anwendungsbereiche für TCG könnten sich beispielsweise im Zusammenhang mit dem Cloud-Computing oder auch dem Peer-to-Peer Computing ergeben. Diese Anwendungen sind auf eine dezentrale Zusammenarbeit ausgerichtet, wobei zwischen den Entitäten ausgetauscht werden, aber auch, wie im Falle des Cloud-Computings, Rechenleistungen und Services zur Verfügung gestellt werden. Das Attestierungskonzept zusammen mit dem Konzept des Sealings könnte dazu genutzt werden, dafür zu sorgen, dass die eigenen Daten nur auf solche Peers bzw. in solche Cloud-Plattformen transferiert und bearbeitet werden, die in der versiegelten Beschreibung spezifizierten Voraussetzung an die Plattformkonfiguration erfüllen. Andererseits könnten Dienstleister bzw. Anbieter von Ressourcen ihrerseits Anforderungen spezifizieren, die die

unternehmens-übergreifend

Software bzw. die der Code erfüllen muss, dem die eigenen Ressourcen zur Nutzung zur Verfügung gestellt werden.

sichere Anwendungen

Spezielle Dienste des TPM wie der manipulationssichere Timer können von Anwendungen zur Überprüfung von Gültigkeitsdauern sowohl von Zertifikaten als auch von sonstigen Tickets (z.B. Kerberos Ticket) verwendet werden. Der sichere Speicher bzw. die Schlüsselhierarchie lässt sich zur sicheren Verwaltung von Objektschlüsseln z.B. für ein verschlüsselndes Datei- oder Dokumentenmanagementsystem, aber auch für sichere E-Mail-Systeme nutzen.

Fazit TPM 1.2

Sicherheitsanforderungen

Das TPM und der TCG Software Stack (TSS) stellen wichtige Basismechanismen zur Verfügung, um die Systemsicherheit durch hardwarebasierte Schutzmaßnahmen wirksam zu erhöhen, so dass sie einen deutlichen Fortschritt für die Absicherung von IT-Infrastrukturen darstellen. Das Protection Profile für das TPM zielt auf eine Vertrauensstufe EAL3 ab, womit klar ist, dass der Chip keine extrem hohen Sicherheitsanforderungen erfüllen muss, da dies seine Herstellungskosten sehr erhöht und die Akzeptanz verringert hätte. Vom Chip wird zwar erwartet, dass er Funktionen bereitstellt, damit nachträgliche Manipulationen erkennbar sind (tamper evidence), jedoch ist kein physischer Schutz (tamper resistance) gefordert.

Mechanismus

Die Eingangs geforderte Trennung von Mechanismus und Strategie wird in der TC-Architektur umgesetzt. Dies hat die bereits erwähnten Vorteile, birgt aber natürlich gleichzeitig die Gefahr der trügerischen Sicherheit, wenn Policies fehlerhaft konfiguriert sind oder ganz fehlen. Die Problematik soll am Beispiel des Bootvorgangs verdeutlicht werden. Durch den TC-Bootvorgang soll das Vertrauen in die Integrität der Konfiguration der Plattform erhöht werden. Dies ist aber sicherlich allein durch das Ablegen von berechneten Hashwerten in einem geschützten Speicher nicht zu erreichen, sondern es werden Regelwerke benötigt, die schrittweise die berechneten Werte mit vorgegebenen Werten abgleichen und bei Unstimmigkeiten einen Abbruch des Bootvorgangs herbeiführen. Derartige Policies sind in der Spezifikation des TCG-Subsystems nicht vorgesehen. Das TPM und die Funktionen der TBBs stellen lediglich Sicherheitsmechanismen zur Verfügung. Die regelbasierte Nutzung dieser Konzepte für ein Sicherheitsmanagement bleibt dann dem Betriebssystem sowie den auf der Plattform zur Ausführung kommenden Applikationen überlassen. Fehlen derartige Kontrollen, so ist es trotz der durchgeführten Integritätswerte-Berechnung nach wie vor möglich, beliebigen Code zu laden und zur Ausführung zu bringen.

Policy

Möglich wäre eine Überprüfung durch ein entsprechend vorbereitetes Betriebssystem, indem dieses z.B. vor dem Übergang in den normalen Nutzungsbetrieb Integritätskontrollen vornimmt. So könnte das Betriebssystem

eine gesicherte Verbindung zu einem vertrauenswürdigen Server aufbauen und sich Listen mit Hashwerten für vertrauenswürdige Komponenten laden. Auf diese Weise ließe sich beispielsweise sicherstellen, dass nur ein zertifiziertes BIOS geladen, zertifizierte DMA-Geräte ihre Privilegien nicht für unerwünschte RAM-Zugriffe missbrauchen oder dass kein Kernel-level Debugger geladen ist (um z.B. durch ein Debugging Schlüsselinformationen zu extrahieren). Document Revocation bzw. Serial Number Revocation Listen könnten verwendet werden, um das Laden unerwünschter Applikationen zu unterbinden oder auch um zum Beispiel das Ablaufen von Lizenzen zu kontrollieren. Offen bleibt dabei zurzeit die Frage, welche Einfluss- und Kontrollmöglichkeiten dem Benutzer hierbei verbleiben dürfen oder auch sollen und wie diese Kontrolle durchgesetzt werden kann.

Lückenlose Vertrauenskette?

Auch der beschriebene Aufbau der Vertrauenskette muss kritisch betrachtet werden. Die Berechnung von Integritätswerten wird für viele sicherheitskritische Bereiche und Komponenten durchgeführt, was sicherlich sehr positiv ist. Jedoch wird damit gleichzeitig auch der fatale Eindruck erweckt, dass mit dem sicheren Bootvorgang tatsächlich alle sicherheitskritischen Komponenten erfasst sind. Dazu gehören aber neben dem eigentlichen Betriebssystemkern insbesondere auch sämtliche Gerätetreiber, die zu großen Teilen heutzutage dynamisch, on-demand nachgeladen werden, sowie auch alle Betriebssystemmodule, die erst zur Laufzeit in den Kern eingelagert werden (erweiterbare Betriebssystemkerne) und dann ebenfalls über privilegierte Zugriffsberechtigungen verfügen. Zu nennen sind natürlich auch Anwendungen, die Administrator-Berechtigungen besitzen, deren Missbrauch die Sicherheit des Systems und damit das Vertrauen in dessen Konfiguration erschüttern kann. Die Frage, welche Komponenten einer komplexen Plattform-Konfiguration als sicherheitskritisch einzustufen und kontinuierlich zu überwachen sind, ist weder verlässlich zu beantworten noch ist eine Ausweitung des gesamten Vorganges der Integritätsmessung auf wirklich alle Komponenten des Systems praktisch realisierbar. Lücken in der Vertrauenskette sind somit nicht vermeidbar und es muss in weiteren Forschungsarbeiten geklärt werden, wie die erklärten Konzepte so in eine Gesamtsystemarchitektur eingebettet werden können, dass die Lücken tolerabel und die damit einhergehenden Risiken beherrschbar sind. Ein vielversprechender Ansatz ist die Nutzung von Virtualisierungskonzepten (u.a. [168]), um kritische und nicht-kritische Systembereiche in virtuellen Maschinen zu kapseln und zu isolieren.

Vertrauenskette

Viren, Würmer und Trojanische Pferde?

Treten Lücken in der Vertrauenskette auf, so können Viren, Würmer oder Trojaner, die sich Anwendungen bedienen, die nicht durch den Prüfvorgang erfasst werden (z.B. Startscripte), nach wie vor ungehindert in das System

Viren, Würmer

eindringen. Betrachten wir zur Veranschaulichung hierzu den in Beispiel 2.4 erklärten Lovesan-Wurm, der über den Port 135 eine Verbindung zum Opfer-Rechner aufbaut, einen Buffer-Overflow hervorruft, durch den der Port 4444 des Opfer-Rechners geöffnet und zum Starten einer Remote Shell genutzt werden kann. Die Kommandos, die der Wurm hierzu ausführt, sind *ftp.exe* und das Starten von *cmd.exe*. Diese Kommandos sind zulässige Kommandos, deren Ausführung nicht der Integritätsüberwachung durch das RTM des TCG-Subsystems unterliegen wird, so dass der Wurm eindringen könnte. Wie wir bei der Erklärung der Vorgehensweise des Lovesan-Wurms erläutert haben, versucht der Wurm sich in die Windows-Registry einzutragen. Diese Systemdatenbank ist eine sicherheitskritische Komponente, deren Zustandsänderungen tunlichst protokolliert werden sollten, so dass beim nächsten Booten eine Abweichung der Konfiguration erkennbar und zumindest dieser „Einnistversuch“ abwehrbar wäre.

Backup und Attestierung

Backup

Bei der Erklärung der Schlüsselhierarchie mit dem Storage Root Key (SRK) als deren Wurzel ist bereits die Problematik des Schlüssel-Recovery und Backups angeklungen. Da das TPM konzeptuell nicht-migrierbare Schlüssel enthält, wozu neben dem SRK auch der EK und die AIK gehören, besteht natürlich die Gefahr, dass bei einem Fehler, der zu einer Zerstörung des Chips oder der entsprechenden Speicherbereiche führt, diese Daten verloren gehen. Damit gehen dann aber auch alle Daten, die mit diesen Schlüsseln abgesichert sind, unwiederbringlich verloren. Benötigt werden Recovery- bzw. Backup-Maßnahmen auch für derartige Schlüssel, an die jedoch hohe Anforderungen zu stellen sind. So darf durch die Maßnahmen das Vertrauen in die Sicherheit der Schlüssel nicht verloren gehen, d.h. der hardware-basierte Schutz muss beibehalten bleiben. Ferner muss aber auch gewährleistet sein, dass die Maßnahmen nicht zur Beeinträchtigung der Privatsphäre des Plattform-Besitzers bzw. -Benutzers führt, da durch ein Backup- oder Recovery-Konzept ggf. Kopien von sensiven Schlüsseln existieren, die vom Besitzer der Plattform unter Umständen nicht mehr kontrollierbar sind. Die Entwicklung und Integration von derartigen Verfahren ist ein noch nicht zufriedenstellend gelöstes Problem.

Attestierung

Auch das Attestierungskonzept birgt eine Reihe von Problemen. Neben der Frage, inwieweit ein Endbenutzer noch selbstbestimmend über seinen eigenen Rechner verfügen kann (siehe DRM-Problematik), besteht die Frage, wie mit der nicht enden wollenden Flut an Patches umzugehen ist. Einerseits sollen ja Integritätskontrollen darauf basieren, dass eine Anwendung, z.B. in Form von Sealings, Vorgaben macht, welche Konfiguration auf einer Plattform herrschen muss. Andererseits unterliegen heutige Systeme im laufenden Betrieb einer sehr großen Veränderungsdynamik, so dass

aktuell erstellte Atteste bereits neue Patches umfassen könnten und die Integritätsüberprüfung fehlschlagen würde. Akzeptanzprobleme bis hin zum Deaktivieren des TPM könnten die Folge sein, falls ein Benutzer zu häufig mit unerwarteten Fehlermeldungen und der Verweigerung von Dienstleistungen konfrontiert wird.

TPM 2.0

Derzeit ist der TPM 1.2 sehr weit verbreitet und in vielen Geräten im Einsatz. Mit der Version TPM 2.0 [8] wurden wesentliche neue Konzepte und Ansätze in die TPM-Architektur integriert. So unterstützt der TPM 2.0 nicht nur einen symmetrischen und einen asymmetrischen Verschlüsselungsalgorithmus wie beim TPM 1.2 (AES und RSA), sondern ermöglicht die Implementierung unterschiedlicher kryptografischer Algorithmen sowie Hashfunktionen. Dadurch bietet er eine gewisse Krypto-Agilität, die über verschiedene Krypto-Engines, die in der Architektur verankert sind, umgesetzt ist. So unterstützt der TPM 2.0 auch elliptische Kurven Kryptografie (ECC), aber beispielsweise auch SHA-2 mit unterschiedlichen Schlüssellängen. Zudem ist es möglich, symmetrische Verfahren, wie den AES, auch außerhalb des TPMs zu nutzen.

TPM 2.0

Der TPM 2.0 definiert auch zusätzliche Module und Dienste, wie die erweiterte Autorisierung (enhanced authorisation). Während die Autorisierung für den Zugriff auf sensitive Daten unter TPM 1.2 allein Passwortbasiert erfolgt, ermöglicht der TPM 2.0 eine Policy-basierte Autorisierung. Der Zugriff auf beliebige Entitäten im TPM, wie kryptografische Schlüssel, Speicherbereiche oder TPM-Funktionen, kann durch eine solche Policy kontrolliert werden. Policy-Regeln spezifizieren die Zugriffsbedingungen und Operationen, die vor einem zulässigen Zugriff ausgeführt werden müssen. Eine TPM-Policy wird als Hashwert repräsentiert. Analog zu der Vorgehensweise bei den PCR-Registern, wird eine Policy schrittweise durch das Hinzufügen neuer Regeln aufgebaut und gehasht. Eine Policy kann zudem alternative Autorisierungspfade spezifizieren.

Der TPM 2.0 ermöglicht es zudem, einen Privacy Administrator einzurichten. Dies ist normalerweise der Besitzer des TPM-Geräts. Der Privacy Administrator kann eine Menge von EK-Schlüsseln nutzen, um unter unterschiedlichen Identitäten aktiv zu sein. Durch dieses Konzept wird das frühere Konzept der Attestierungs-Schlüssel hinfällig. Unter dem TPM 2.0 kann auch das RTR, das Root of Trust for Reporting, flexibler genutzt werden, so dass beispielsweise auch Audit-Log-Einträge und Eigenschaften kryptografischer Schlüssel damit zertifiziert werden können.

Für eine ausführlichere Darstellung der neuen TPM 2.0 Konzepte sei auf die Spezifikation verwiesen³³.

Abschließend sei noch einmal darauf hingewiesen, dass das Ziel der Trusted Computing-Initiative darin besteht, mit entsprechenden Maßnahmen das Vertrauen in die Integrität und Authentizität einer Plattform bzw. von Komponenten einer Rechnerplattform zu erhöhen. Ob diese tatsächlich vertrauenswürdig (trustworthy) ist, lässt sich aber mit den verwendeten Maßnahmen nicht sicherstellen, da keine Beweise der korrekten Funktionalität bzw. Einhaltung spezifizierter Sicherheitseigenschaften gefordert werden. Auch wenn mit der Trusted Computing Initiative noch eine Reihe von offenen Fragen verbunden ist, so ist sie dennoch ein begrüßenswerter Schritt, um die System-Sicherheit zu erhöhen.

11.5 Physically Unclonable Functions (PUF)

Eingebettete
Systeme

Vernetzte eingebettete Systeme, vernetzte Sensoren und Akteure sind bereits heute allgegenwärtig. Im Internet der Dinge (Internet of Things) kommuniziert und kooperiert eine Vielzahl von Objekten. Sie erfassen Daten, verarbeiten und speichern sie und tauschen über unterschiedliche meist drahtlose Kommunikationskanäle Daten aus. Vernetzte eingebettete Systeme werden häufig in sicherheitskritischen Umgebungen zur Steuerung, Überwachung und zum Betrieb von Anlagen und Systemen verwendet, so dass der Sicherheit dieser vernetzten Objekte eine große Bedeutung zukommt. Es werden preisgünstige, sichere und auch skalierende Techniken benötigt, um die Identität von Objekten überprüfen zu können, um die Objekte vor Manipulation zu schützen oder auch um die Vertraulichkeit der Daten zu gewährleisten.

PUF

Physical Unclonable Function

Mit den Physically Unclonable Functions (PUFs) bzw. Physical One-Way Funktionen werden bereits seit 1983 [11] und insbesondere seit 2001 bzw. 2002 mit den Arbeiten von Pappu und Gassend ([147, 69]) neuartige, Hardware-basierte Techniken entwickelt, um Objekte anhand von physikalischen Eigenschaften eindeutig zu charakterisieren. Eine PUF ist eine Funktion, die als ein physikalisches Objekt, z.B. eine integrierte Schaltung, implementiert wird. Charakteristisch für eine PUF ist, dass sie im Verlauf ihrer Herstellung durch Variationen im Produktionsprozess eindeutige physikalische Merkmale erhält. Häufig wird auch der Begriff Physical Unclonable Function verwendet, um damit physikalische Funktionen, die unklonbar sind, zu charakterisieren, während wir im Folgenden aber vordringlich phy-

³³ <https://trustedcomputinggroup.org/wp-content/uploads/TPM-Rev-2.0-Part-1-Architecture-01.38.pdf>

sikalisch unklonbare Funktionen meinen und deshalb die Begrifflichkeit der Physically Unclonable Function verwenden.

PUFs können in Zukunft wichtige Bestandteile von Sicherheitsprotokollen und -Diensten werden. Analog zu biometrischen Techniken, die zur eindeutigen Identifizierung von Personen eingesetzt werden, können PUFs zur Objekt-Identifikation und -Authentifikation genutzt werden. Der Einsatz von PUFs ist aber nicht auf den Bereich der Objekt-Identifizierung beschränkt, sondern sie können auch als Basis für weitere Sicherheitsdienste dienen. Mit einer PUF kann beispielsweise eine Bindung von Software bzw. Firmware an eine Hardware-Plattform erfolgen, um ein Reverse-Engineering der Plattform ganz erheblich zu erschweren. PUFs können darüber hinaus verwendet werden, um Manipulationen an den durch sie geschützten eingebetteten Objekten zu erkennen und den Zugriff auf gespeicherte, sensitive Informationen zu schützen. Damit sind sie auch eine pragmatische Lösung für den Bereich des Know-How-Schutzes bei eingebetteten Systemen. Eine weitere sehr interessante Verwendung von PUFs ist die Möglichkeit, aus den charakteristischen physikalischen Eigenschaften der PUF kryptographisches Schlüsselmaterial abzuleiten. Das Besondere hierbei ist, dass dieses Schlüsselmaterial nur temporär gespeichert werden muss und der Schlüssel bei Bedarf immer wieder direkt aus den Eigenschaften der PUF generiert wird. In Abschnitt 11.5.2 gehen wir auf die Nutzung von PUFs in Sicherheitsprotokollen noch etwas mehr ein.

PUF als Sicherheitsbaustein

11.5.1 Einführung

Eine PUF charakterisiert eine Funktion, die durch eine Eingabe, das ist ein Stimulus, der auch als Challenge bezeichnet wird, angeregt wird und eine Ausgabe, die Response, erzeugt. Diese Response ist ein digitales oder auch analoges Signal, das nach Außen zufällig wirkt, aber reproduzierbar ist. Das heißt, beim Vorliegen der gleichen Eingabe wird eine korrekte PUF mit der gleichen bzw. aufgrund nicht zu vermeidender Rauscheffekte zumindest mit einer sehr ähnlichen Response antworten. Das Verhalten einer PUF wird durch Challenge-Response-Paare charakterisiert. Bei einer idealen PUF kann dieses Verhalten nicht vorhergesagt und das Verhalten kann auch nicht durch eine physikalisch geklonte PUF nachgebaut oder in Software modelliert werden; das Verhalten ist eindeutig und nicht fälschbar. Nachfolgend werden bekannte Beispiele für PUFs kurz vorgestellt.

Challenge-Response

Beispiele für PUF-Implementierungen

Optische Tags, die auch als Physical One-way Functions [138] bekannt sind, gehören zu den ersten PUF-Implementierungen. Ein solche optische

optische PUF

PUF besteht aus einer durchsichtigen Glas- oder Kunststoffmasse, in die reflektierende Partikel ($500\mu\text{m}$) eingebettet sind. Die Partikel werden beim Produktionsprozess zufällig im Objekt verteilt. Wird diese optische PUF nun durch eine Challenge angeregt, indem sie von einem Laser in einem bestimmten Winkel angestrahlt wird, so wird der Laserstrahl durch die Partikel reflektiert und die PUF erzeugt ein zufälliges Reflexionsmuster. Dieses Muster ist die Response, die weiter verarbeitet wird, indem sie zunächst digitalisiert und dann häufig noch gehasht wird. Das Set-up einer optischen PUF ist jedoch sehr aufwändig und teuer, da sowohl ein Laser als auch komplizierte Positionierungsmechanismen benötigt werden. Die Messapparatur ist teuer, die eigentliche PUF dagegen ist preiswert.

Kostengünstige PUF-Lösungen bieten physikalische Strukturen, die mit CMOS Schaltkreisen umzusetzen oder auch auf FPGAs (Field-Programmable Gate Arrays) einsetzbar sind. Die drei wichtigsten PUF-Varianten sind die Arbiter-, die Ring-Oszillatoren- und die SRAM-PUF. Die physikalischen Charakteristika, die sich aus Varianten beim Herstellungsprozess ergeben, basieren hierbei auf unterschiedlichen Signallaufzeiten in integrierten Schaltungen, wie dies bei der Arbiter- oder der Ring Oszillatoren-PUF der Fall ist, oder auf uninitialisierten Speicherzellen, wie bei der SRAM-PUF.

Arbiter-PUF

Eine Arbiter-PUF (vgl. Abbildung 11.22) wird aus CMOS-Gattern aufgebaut und besteht aus zwei Pfaden von Verzögerungsgliedern (Multiplexern). Aufgrund von Unterschieden beim Herstellungsprozess weisen die Multiplexer unterschiedliche Signallaufzeiten auf. Durch Anlegen einer Challenge (z.B. 128 Bit) wird eine Art Wettrennen zwischen den beiden Pfaden ausgelöst und das letzte Glied der Schaltung bestehend aus einem Arbiter (latch) entscheidet, welcher Pfad schneller war und liefert eine 1 Bit Ausgabe. Diese einfachen PUFs weisen aber Schwachstellen auf, da man ihr Ausgabeverhalten modellieren und mittels maschineller Lerntechniken mit hoher Treffergenauigkeit für beliebige Challenges die von der PUF erzeugte Response vorhersagen kann (vgl. [155]). Die Arbiter-PUF ist zwar nicht physikalisch fälschbar, sie erfüllt also die Anforderung an eine Physically Unclonable Function, aber da man sie in Software modellieren kann, so dass ihr Verhalten vorhersagbar ist, ist sie keine physikalische Funktion, deren Verhalten generell nicht fälschbar ist. D.h. sie ist keine Physical Unclonable Function.

Ring-Oszillatoren-PUF

Eine Ring-Oszillatoren-PUF (vgl. Abbildung 11.23) basiert auf speziellen Schaltungen, die mit einer gewissen Frequenz oszillieren. Diese Schaltungen sind als eine Matrix von Ring-Oszillatoren angelegt. Während des Herstellungsprozesses wird die Frequenz der Oszillatoren zufällig beeinflusst und

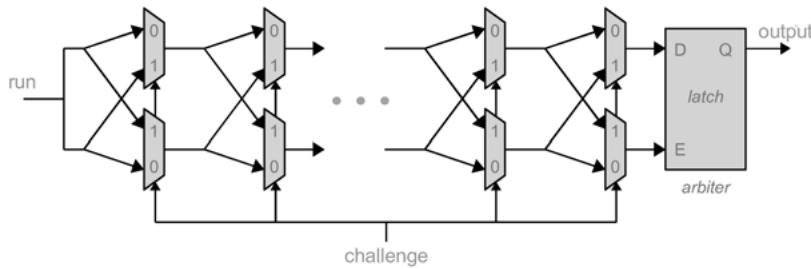


Abbildung 11.22: Arbiter-PUF

bildet damit die Basis für das einzigartige PUF-Verhalten. Um Störeinflüsse, wie z. B. Temperaturänderungen, zu kompensieren, wird das Ausgangsbit durch einen relativen Vergleich der Frequenz zweier auf dem gleichen Chip angebrachter Ring-Oszillatoren erzeugt. Abhängig davon, welcher Oszillator die höhere Frequenz hat, wird eine 1 oder eine 0 ausgegeben.

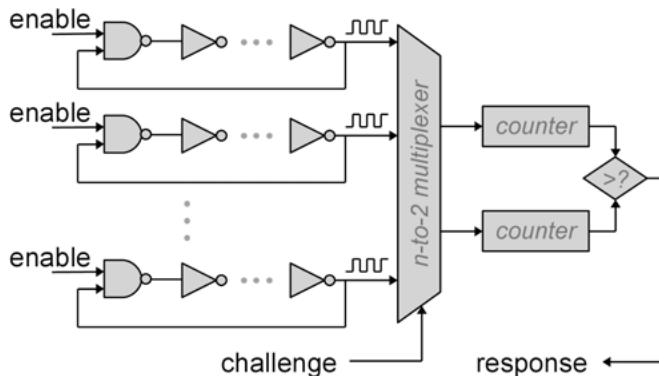


Abbildung 11.23: Ring-Oszillatoren-PUF

Eine SRAM-PUF nutzt als physikalische Charakteristik das Einschaltverhalten von SRAM Zellen (vgl. Abbildung 11.24) aus. Beim Anlegen der Versorgungsspannung geht die SRAM Zelle dann, abhängig von der physikalischen Zellenbeschaffenheit, wie beispielsweise der Schwellspannung der Transistoren, in einen, für die PUF charakteristischen Anfangszustand über. Die physikalischen Zellen-Charakteristika entstehen bei der Produktion und haben zur Folge, dass manche Zellen bevorzugt in Richtung 0 und manche in Richtung 1 kippen.

SRAM-PUF

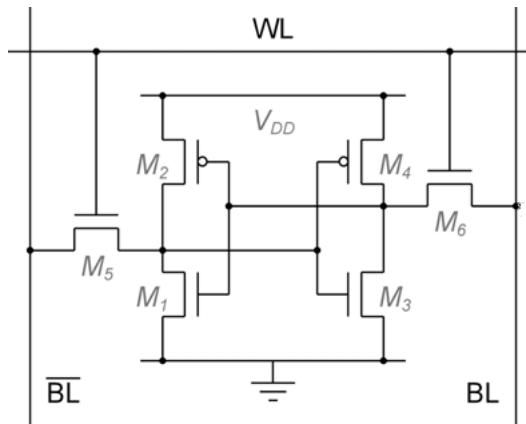


Abbildung 11.24: 6 Transistor SRAM Zelle, die als PUF verwendbar ist.

Ein ausführlicher Überblick über weitere PUF-Varianten findet sich u.a. in [112].

PUF-Eigenschaften

Aus dem bislang Gesagten wird deutlich, dass PUF-Strukturen bestimmte Eigenschaften besitzen müssen, damit sie für die oben genannten Sicherheitsdienste der Authentisierung oder Schlüsselerzeugung verwendet werden können. Abhängig davon, welche Eigenschaften eine PUF erfüllt, unterscheidet man zwischen Strong (starken), Controlled (kontrollierten) und Weak (schwachen) PUFs.

Robustheit

Damit eine PUF für die Authentisierung oder Schlüsselgenerierung nutzbar ist, muss sie einen hohen Grad an Robustheit aufweisen. Das bedeutet, dass die PUF, wenn sie mehrfach mit der gleichen Challenge stimuliert wird, stets ein ähnliches Antwortverhalten zeigen sollte. Damit unterscheidet sich die PUF gravierend von einem Zufallszahlengenerator. Wie bereits gesagt, basiert der PUF-Ansatz auf dem Ausnutzen spezifischer physikalischer Phänomene, die dazu führen, dass eine Antwort einer PUF stets verrauscht ist. Wichtig für die Robustheit einer PUF ist deshalb, dass dieses Rauschen, das nach der Digitalisierung als Bitfehler erkennbar ist, durch fehlerkorrigierende Codes beseitigt werden kann. Das bedeutet, dass eine PUF eine möglichst geringe Bitfehlerrate besitzen sollte. Zusätzlich fordert man von PUFs, dass sie auch bei sich ändernden Umweltgegebenheiten, wie beispielsweise der Temperatur, keine hohen Bitfehlerraten aufweisen, also auch dann noch robust sind.

Unclonability

Eine zentrale PUF-Eigenschaft ist die Unkopierbarkeit, (engl. *unclonability*). Sie fordert, dass es nicht möglich sein darf, eine physikalische oder

simulierte (digitale) Kopie der PUF zu produzieren, die das gleiche Verhalten wie die Original-PUF aufweist. Unkopierbare PUFs müssen ein sehr komplexes Challenge-Response-Verhalten aufweisen, also sehr viele Challenge-Response Paare ermöglichen, so dass es für einen Angreifer unmöglich ist, das Verhalten der PUF aufzuzeichnen und dann in Software zu simulieren.

Eng verwandt mit der Unkopierbarkeits-Eigenschaft ist die Forderung nach Unvorhersagbarkeit des Antwortverhaltens der PUF. Das bedeutet, dass ein Angreifer, auch wenn er das Verhalten der PUF bezüglich einer Menge von Challenges bereits kennt, nicht in der Lage sein darf, das Antwortverhalten der PUF für eine konkrete neue Challenge korrekt vorherzusagen. Um den Grad der Unvorhersagbarkeit einer Antwort quantitativ zu bestimmen, wird die Entropie der Antwort bestimmt. PUFs, die zur Erzeugung kryptographischer Schlüssel eingesetzt werden, müssen eine sehr hohe Entropie der Antworten liefern. Eine analoge Forderung stellt man ja auch an kryptographische Schlüssel, die über Zufallszahlengeneratoren erzeugt werden. Die Eigenschaft der Unvorhersagbarkeit wird bei PUFs, die eine interne Nachbearbeitung der Response vorsehen, wie dies z.B. bei den schwachen PUFs (s.u.) der Fall ist, durch das Anwenden von Hashfunktionen gewährleistet. Dadurch wird erreicht, dass der Schlüssel uniform zufällig ist und keine statistischen Muster aufweist.

Unvorhersagbarkeit

Strong, Controlled und Weak PUF

Man spricht von einer strong PUF (starken PUF), wenn sie unkopierbar und ihr Verhalten nicht vorhersagbar ist. Dies muss auch dann gelten, wenn es einem Angreifer möglich ist, beliebige und auch beliebig viele Challenges an die PUF zu senden und deren Antwort aufzuzeichnen. Strong PUFs ermöglichen eine Konstruktion, bei der fehlerkorrigierende Codes, die zur Behebung von Rauschen erforderlich sind, auch außerhalb des Chips durchgeführt werden können. Optische PUFs sind Beispiele für strong PUFs, während eine Arbiter-PUF diese Eigenschaft nicht erfüllt.

Strong PUF

PUFs, die für eine Authentisierung genutzt werden, müssen strong PUFs sein, damit bei jeder Authentisierung eine neue Challenge genutzt und damit mögliche Man-in-the Middle-Angriffe unterbunden werden.

Eine controlled PUF (kontrollierte PUF) basiert auf einer strong PUF und verknüpft diese mit zusätzlicher Kontroll-Logik, die verhindert, dass Challenges beliebig an die PUF gesandt und die Antworten der PUF direkt ausgelesen werden können. Das bedeutet, dass der Zugriff auf die PUF kontrolliert wird. Mit dieser zusätzlichen Logik, die zum Beispiel als ei-

Controlled PUF

ne Hashfunktion implementiert ist, werden Modellierungs-Angriffe auf die PUF abgewehrt, wenn sichergestellt ist, dass das Ausgabeverhalten der PUF nicht direkt abgelesen werden kann.

Controlled PUFs können zur Erkennung von Manipulationen an Schaltungen eingesetzt werden.

Weak PUF

Eine weak PUF (schwache PUF) verfügt im Gegensatz zur starken PUF nur über sehr wenige Challenge-Response Paare, im Extremfall sogar nur über ein einziges Paar und damit über lediglich eine Challenge. Weak PUFs werden häufig zur Erzeugung von kryptographischem Schlüsselmaterial verwendet (siehe unten), wobei eine weak PUF ihre Antworten nie nach außen weiterleitet (und natürlich auch nicht abgehört werden darf.) Eine SRAM-PUF ist ein Beispiel für eine weak PUF.

Weak PUFs können auch durch integrierte strong PUFs realisiert werden. Diese werden dann auch häufig als Physically Obfuscated Keys (POK) bezeichnet, da sie anders als die digitalen Schlüssel, auf physikalische Weise permanent gespeichert sind. Durch invasive Angriffe werden zudem diese Schlüssel zerstört, da sich die physikalischen Gegebenheiten der PUF ändern und damit die PUF auf die Challenge nicht mehr mit der korrekten Antwort reagiert.

11.5.2 Einsatz von PUFs in Sicherheitsprotokollen

Die wichtigsten Einsatzbereiche für PUF-Implementierungen sind die Objekt-Authentisierung, die Schlüsselgenerierung inklusive dessen Speicherung in physikalischen Strukturen und die Manipulationserkennung bzw. der Know-How-Schutz.

PUFs zur Objekt-Identifikation und Authentisierung

Challenge- Response

Analog zu biometrischen Erkennungsverfahren unterscheidet man auch bei PUFs zwei Phasen: (1) die Enrollment- und (2) die Verification-Phase. Während der Enrollment-Phase wird die PUF durch eine Menge von Challenges stimuliert und sie erzeugt die zugehörige Antworten (Responses). Diese CR-Paare werden in einer Datenbank als Referenzwerte abgelegt.

In der Verification-Phase erfolgt die Auswahl einer Challenge aus der Datenbank. Die PUF wird damit angeregt, erzeugt eine Antwort, die mit dem hinterlegten Referenzwert in der Datenbank verglichen wird. Aufgrund des auftretenden Rauscheffektes bei PUFs, durch den es bei den Antwort-Bitstrings zu Bitfehlern kommt, ist eine Gleichheit zwischen dem in der Datenbank gespeicherten Referenzwert und der aktuellen Response unwahrscheinlich. Dies ist analog zu den biometrischen Erkennungstechniken, bei

denen auch entsprechende Ungenauigkeiten auftreten, die zu den bekannten Fehlersituationen der False-Acceptance bzw. False-Rejection führen können, falls die Toleranzwerte für die Akzeptanz einer abweichenden Antwort zu hoch bzw. niedrig eingestellt ist. Analog benötigt man auch bei PUFs ein Ähnlichkeitsmaß, mit dem man entscheidet, ob eine Antwort ähnlich genug ist. Sind die Antworten Bitstrings, so wird hierfür häufig eine Distanzmetrik, wie die Hamming-Distanz, verwendet. Offensichtlich ist es wünschenswert, dass zwei Antworten, denen die gleiche Challenge der gleichen PUF zugrunde liegt, eine kleine Distanz besitzen. Ebenso ist es wünschenswert, dass die Antworten bei unterschiedlichen PUFs, die mit der gleichen Challenge stimuliert wurden, einen großen Abstand besitzen.

Neben den unvermeidbaren Abweichungen durch physikalische Rauscheffekte, können auch systematische Einflüsse der Umwelt das Antwortverhalten einer PUF beeinflussen. Dazu gehören Temperatur-Unterschiede, Stromschwankungen oder aber auch Änderungen von Materialbeschaffenheit u.a. in Folge von Alterung. Falls die Einflüsse die gesamte PUF gleichermaßen betreffen und zumindest teilweise linear sind, kann man durch Kompensationsmaßnahmen wie fehlerkorrigierende Codes die aufgetretenen Fehler tolerieren. Man kann auch diese Umwelteinflüsse versuchen separat zu messen und bei der Interpretation der ermittelten Responses berücksichtigen.

Schlüsselgenerierung

Schwache PUFs sind, wie oben bereits angesprochen, geeignet, um Chip-individuelle, kryptografische Schlüssel zu erzeugen. Wesentlich dabei ist, dass die ermittelte Response der PUF nicht nach aussen weitergereicht wird. Zur Erzeugung kryptografischer Schlüssel werden die Bitfehler in der PUF-Antwort zunächst mittels fehlerkorrigierender Codes korrigiert. Um Schlüsselmaterial mit einer ausreichend hohen Entropie zu erzeugen, wendet man in der Regel noch eine Hashfunktion auf die PUF-Response an, um die für das zu nutzende Verschlüsselungsverfahren notwendige Anzahl von Schlüsselbits (z.B. 128 oder 256 Bit beim AES) zu erzeugen. Schwache PUFs sind damit spezielle Ausprägungen von nicht-flüchtigen Schlüssel-Speichern, die aber deutlich schwieriger auszulesen sind, als ein klassischer nicht-flüchtiger EEPROM.

Da die Antworten einer schwachen PUF nur intern als geheimer Schlüssel weiter verarbeitet werden und nicht nach draußen dringen, müssen die fehlerkorrigierenden Maßnahmen direkt ön-Chip und mit hoher Genauigkeit erfolgen. Hierzu werden in der Regel so genannte Hilfsdaten (Helper-Data) im nicht-flüchtigen Speicher auf dem Chip abgelegt. Helper Data werden durch entsprechende Algorithmen, den Fuzzy-Extraktoren [58], während einer Enrollment-Phase erzeugt, und dann bei der Schlüsselgenerierung in der

Schlüsselspeicher

Helper Data

Schlüssel-Rekonstruktions-Phase verwendet. Die Hilfsdaten ergeben sich aus einer XOR-Verknüpfung der PUF-Bits mit den Bits eines Codeworts eines fehlerkorrigierenden Codes. Dodis hat dazu in [58] die Konstruktion eines Code-Offset Fuzzy Extractors (COFEs) vorgeschlagen, auf dessen Konstruktionsprinzip nachfolgend noch kurz eingegangen wird.

Schlüssel- erzeugung

Wie bereits erwähnt, werden zur Generierung eines kryptografischen Schlüssels K aus einer Response R sowohl ein fehlerkorrigierender Code C als auch eine Hashfunktion h zur Erzeugung von Entropie für K benötigt. In einem ersten Schritt wird ein Code-Wort C_s des Codes C zufällig gewählt. Sei R die Response der PUF auf eine Challenge C . Dann wird ein Helper-Data Vektor W wie folgt erzeugt: $W = C_s \text{ xor } R$. Der kryptografische Schlüssel K wird unter Nutzung der Hashfunktion erzeugt: $K = h(R)$.

Rekonstruktion

In der Phase der Schlüsselrekonstruktion wird die PUF mit der gleichen Challenge wie in der Enrollment-Phase stimuliert. Aus der aktuellen, verrauschten Response \bar{R} der PUF wird der Schlüssel K wieder hergestellt, indem zunächst unter Nutzung der Helper-Data die Antwort \bar{R} korrigiert wird: $C_s = \text{decode}_s(W \text{ xor } \bar{R})$. Die korrigierte Antwort R erhält man mit: $R = W \text{ xor } C_s$.

Mit der Hashfunktion h wird anschließend der kryptografische Schlüssel K erzeugt: $K = h(R)$.

Manipulations- und Know-How-Schutz

PUFs können auch elektronische Bauteile vor unautorisiertem Auslesen von Speicherinhalten und vor Manipulation schützen. Dies wird beispielsweise durch die Schutzfolie PEP³⁴ realisiert, die einen solchen Manipulations- schutz für elektronische Bauteile bereitstellt. Das Bauteil, bzw. zumindest die sicherheitskritischen Baugruppen, werden mit der Schutzfolie abgedeckt.

Die Folie selbst implementiert dabei eine PUF, die zunächst zur Schlüsselerzeugung wie oben beschrieben, genutzt wird. In einem zusätzlichen Schritt wird der aus den charakteristischen Eigenschaften der Folie gewonnene Schlüssel genutzt, um die Firmware des zu schützenden elektronischen Bauteils zu verschlüsseln. Bei einem Manipulationsversuch an der Folie, also einem invasiven Angriff, ändern sich die Materialeigenschaften der Folie und damit das Antwortverhalten der PUF. Die Änderungen lassen sich nicht mehr durch die Hilfsdaten kompensieren und der korrekte Schlüssel kann nicht mehr rekonstruiert werden. Die Firmware bleibt verschlüsselt bzw. wird in

³⁴ vgl. <http://ais.ec/pep>

diesem Fall sogar automatisch zerstört, um Know-How Abfluss konsequent zu verhindern.

11.5.3 Sicherheitsuntersuchungen von PUFs

Eine vergleichende, experimentelle Untersuchung in [99] hat gezeigt, dass alle untersuchten Speicher-basierten und die Arbiter- bzw. Ring-Oszillatoren-PUFs robust und deren Bitfehler mit bekannten fehlerkorrigierenden Codes korrigierbar sind. Um die Unvorhersagbarkeit der untersuchten PUFs zu bestimmen, wurde die bedingte Entropie der PUF-Antworten gemessen, wobei davon ausgegangen wurde, dass das Antwortverhalten der PUF auf ähnliche Challenges (z.B. benachbarte Challenges) bereits bekannt ist. Die Experimente haben gezeigt, dass z.B. die Antworten von SRAM-PUFs sehr schwer vorherzusagen sind, während die Antworten von Arbiter-PUFs nur über eine geringe Entropie verfügen und durch den Einsatz von Methoden aus dem Bereich des maschinellen Lernens vorhersagbar sind [155]. Die experimentellen Untersuchungen, die aber keine invasiven Angriffe oder Seitenkanalangriffe umfassten, haben ergeben, dass die SRAM- und Ring Oszillatoren-PUFs über sehr gut ausgeprägte PUF-Eigenschaften verfügen.

Experimentelle
Untersuchung

Physically Unclonable Functions sind ein sehr interessantes Konzept, um Hardware-basierte Sicherheitsbausteine für eingebettete Systeme zur Verfügung zu stellen, so dass eine Identifikation und Authentisierung von eingebetteten Objekten, eine sichere Schlüsselspeicherung und auch eine zuverlässige Manipulationserkennung damit möglich ist. Forschungsergebnisse zeigen jedoch auch, dass PUF-Implementierungen anfällig sind gegen invasive und semi-invasive Angriffs-Techniken und insbesondere auch gegen passive Seitenkanalangriffe [121]. Um PUFs im zukünftigen Internet der Dinge als stabile, preiswerte und vertrauenswürdige Sicherheitsbausteine einsetzen zu können, sind noch weitere Forschungsarbeiten notwendig, die auch das Härteln der PUF-Implementierungen umfassen müssen.

Fazit

12 Zugriffskontrolle

Das Kapitel stellt die wichtigsten Mechanismen und Verfahren zur Rechteverwaltung und Zugriffskontrolle vor. Nach einer kurzen Einleitung beschäftigt sich Abschnitt 12.2 mit hardwarebasierten Kontrollen, die effizient durchführbar sind und nicht ohne weiteres umgangen werden können. In den Abschnitten 12.3 und 12.5 führen wir die in der Praxis am weitesten verbreiteten softwarebasierten Kontrollmechanismen ein, die vom Betriebssystemkern bzw. von Serverprozessen in Client-Server-Architekturen zur Verfügung gestellt werden. Als Fallbeispiel dient erneut das Unix Betriebssystem, dessen Zugriffskontrolle in 12.4 ausführlich erläutert wird.

Das Dienste-Paradigma ist in heutigen vernetzten und verteilten Systemen etabliert. Dienste werden spezifiziert und über das Internet zur Nutzung angeboten. Die Zugriffe auf solche Dienste unterliegen Beschränkungen, die festzulegen und zu kontrollieren sind. In Abschnitt 12.6 gehen wir deshalb abschließend auf Dienste-orientierte Architekturen (SOA) ein und stellen mit den Web-Services und deren Sicherheitsstandards eine weit verbreitete Umsetzung von SOA vor.

12.1 Einleitung

Die Aufgaben der Rechteverwaltung und der Zugriffskontrolle eines IT-Systems bestehen darin, Mechanismen zur Vergabe von Zugriffsrechten bereitzustellen sowie bei Zugriffen auf zu schützende Objekte die Autorisierung zu prüfen. Aus den Kriterienkatalogen zur Bewertung der Sicherheit von Systemen (vgl. Kapitel 5) lassen sich die folgenden Präzisierungen dieser allgemeinen Aufgabenstellungen ableiten.

Die Rechteverwaltung hat zu gewährleisten, dass alle Subjekte und Objekte eindeutig und fälschungssicher identifiziert werden und dass jedes Objekt, für das Zugriffsbeschränkungen festgelegt sind, auch wirklich von der Rechteverwaltung erfasst wird. Mit geeigneten Überprüfungen sollten Widersprüche bei der Rechtevergabe frühzeitig erkannt und dem Benutzer angezeigt werden. Dadurch soll dieser in die Lage versetzt werden, aufgetretene Konflikte, so weit wie möglich maschinell unterstützt, frühzeitig aufzulösen. Eine frühzeitige Erkennung verhindert, dass das System

im Verlauf seiner Ausführung in einen inkonsistenten Rechtezustand eintritt, aus dem sich Sicherheitsprobleme und Informationsverluste durch Prozessabbrüche ergeben können. Jede Rechteänderung an einem Objekt, die nicht durch ein vertrauenswürdiges Subjekt durchgeführt wird, ist dem Eigentümer des Objekts unmittelbar anzusehen. Die Informationen, die für die Zugriffskontrolle verwendet werden, insbesondere auch die Zugriffsrechte, müssen vor unautorisierter Manipulation besonders geschützt und fälschungssicher gespeichert werden.

Zugriffskontrolle

Von der Zugriffskontrolle wird gefordert, dass sie jeden Zugriffsversuch kontrolliert und dass diese Kontrolle nicht umgangen werden kann. Zwischen der Rechteüberprüfung und der Ausübung der Rechte sollte keine Aktion möglich sein, die den Rechteentzug zur Folge hat, da sonst unzulässige Rechtezustände eintreten können. Ist es dennoch möglich, dass ein Subjekt weiterhin Rechte wahrnehmen kann, obwohl ihm diese zwischenzeitlich entzogen worden sind, so ist festzulegen, wie lange diese Rechtewahrnehmung dem Subjekt noch möglich sein darf.

Die korrekte und vollständige Umsetzung aller dieser Anforderungen ist aufwändig und erfordert ein Zusammenspiel zwischen den hardwarebasierten Kontrollen, den Kontrolldiensten des Betriebssystems, sowie ggf. sprachbasierten Kontrollen, die durch den Übersetzer und Binder oder von Komponenten des Laufzeitsystems durchgeführt werden, bis hin zu dedizierten Kontrollen, die in die Anwendungen selber integriert sind, wie beispielsweise Datenbanken oder Web-Servern.

12.2 Speicherschutz

Betriebssysteme, die Anwendungen einen direkten Zugriff auf Hardware-Ressourcen ermöglichen, eröffnen der Anwendungsebene einen unkontrollierbaren Zugang zu den Systemressourcen. Um diese direkten Zugriffe auf die Hardware zu reglementieren, sind in heutigen Architekturen solche Zugriffe in der Regel nur über die Ausführung spezifischer Maschinenbefehle¹ möglich, zu deren Nutzung nur spezielle Systemkomponenten berechtigt sind. Zur Realisierung der verschiedenen Berechtigungsstufen stellen heutige Prozessoren unterschiedliche Betriebsmodi zur Verfügung.

12.2.1 Betriebsmodi und Adressräume

Die meisten Hardware-Architekturen beschränken sich auf zwei Modi, nämlich den System- bzw. Kernmodus und den Benutzermodus. Diese beiden

¹ Befehle im privilegierten Modus

Modi besitzen eine Entsprechung in privilegierten und nicht privilegierten Befehlssätzen eines Prozessors.

Benutzerprozesse werden generell im nicht privilegierten Modus ausgeführt, so dass sie keine privilegierten Hardwarebefehle ausführen und insbesondere nicht direkt auf die Hardware zugreifen können. Entsprechende Versuche werden von der Hardware erkannt. Sie löst aufgrund der Privilegienverletzung eine synchrone Unterbrechung (engl. *interrupt*) aus, die das Betriebssystem behandelt. Die Verwaltung der Hardware ist die Aufgabe des Betriebssystemkerns, der im privilegierten Modus abläuft, so dass die privilegierten Befehle zum Zugriff auf die Hardware ausführbar sind.

Mit den Modi wird somit eine Trennlinie zwischen Betriebssystemkern und Benutzerebene festgelegt, die nur durch Systemaufrufe (engl. *trap*) kontrolliert überschritten werden kann. Ein Systemaufruf bewirkt eine Unterbrechung des Benutzerprozesses, gefolgt von einem Einsprung in den Betriebssystemkern, der mit einem Moduswechsel einhergeht. Unter der Kontrolle der Hardware erfolgt hierbei also ein Berechtigungswechsel des Prozesses. Gleichzeitig werden die Registerinhalte und Kellerbereiche des Benutzerprozesses gerettet und der Systemkern zur Ausführung des Systemdienstes wird mit seinen eigenen Daten sowie einem eigenen Kellerbereich ausgestattet, so dass keine Interferenzen zwischen Benutzer- und Systemdaten auftreten können. Ein Wechsel vom privilegierten Modus zurück in den Benutzermodus wird durch die Ausführung spezieller Befehle oder durch das Setzen von Flags im Prozessorstatuswort erreicht.

Heutige Betriebssysteme sind in der Regel mehrbenutzer- und mehrprogrammfähig (engl. *multiuser* und *multitasking*), d.h. dass mehrere Benutzer- bzw. Systemprozesse nebenläufig aktiv sein können. Da die Prozesse unterschiedliche Anwendungsprogramme ausführen, sind deren Daten- und Codesegmente, die gleichzeitig im Speicher gehalten werden, vor absichtlichen oder unabsichtlichen Fehlzugriffen anderer Benutzerprogramme zu schützen. Zur Verhinderung solcher unautorisierter Zugriffe auf fremde Daten im Arbeitsspeicher setzt man eine Indirektionstechnik ein. Das bedeutet, dass in Benutzerprogrammen keine Maschinenadressen, die einen direkten Speicherzugriff ermöglichen, verwendet werden dürfen, sondern dass jedem Programm ein eigener Raum mit Programmadressen, den virtuellen Adressen, zugeordnet wird. Diese werden durch Speichertabellen des Betriebssystems auf Maschinenadressen abgebildet. Der damit verbundene Indirektionsschritt ermöglicht es, bei jeder Adressumrechnung einfache, hardwarebasierte Kontrollen durchzuführen. Auf diese gehen wir weiter unten noch näher ein. Da die Speichertabellen nur im privilegierten Modus verändert werden dürfen, sind sie vor unautorisierten Manipulationen durch Benutzerprogramme geschützt.

Benutzermodus

Systemmodus

Moduswechsel

Programmadressen

Maschinenadressen

Der direkte Zugriff auf den realen Adressraum mittels Maschinenadressen bleibt dem Betriebssystem vorbehalten. Bei Systemen, die von dieser Vorgehensweise abweichen, handelt es sich meist um Spezialprozessoren, wozu auch Chipkartenbetriebssysteme und eingebettete Systeme gehören, in denen vornehmlich dedizierte Systemprozesse ausgeführt werden, so dass die Isolierung von Benutzerprozessen nicht erforderlich ist. Das absichtliche oder unabsichtliche Überschreiben von Speicherbereichen, die von anderen Anwendungsprogrammen genutzt werden, führte zu den bekannten Laufzeitfehlern und Systemabstürzen, die zum Teil auch gravierende Datenverluste zur Folge hatten. Die hardwarebasierte Isolierung von Prozessaddressräumen mittels des Konzepts der virtuellen Adressen ist somit eine erste wesentliche Schutzmaßnahme, um eine wirkungsvolle Zugriffskontrolle zu realisieren.

12.2.2 Virtueller Speicher

Es ist nicht das Ziel dieses Buches, klassische Betriebssystemaufgaben, wozu auch die virtuelle Speicherverwaltung gehört, zu erklären. Der Leser sei hierzu auf die reichhaltige Fachliteratur zum Thema Betriebssysteme (u.a. [173]) verwiesen. Wir beschränken uns hier vielmehr auf die für die Sicherheit eines Systems wichtigen Aspekte.

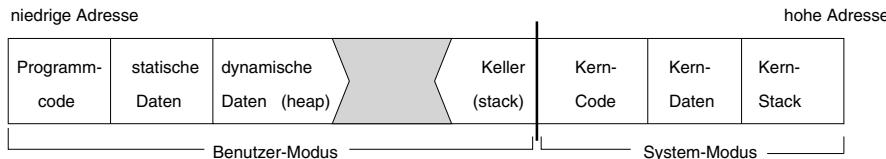
virtuelle Adressen

Ein Benutzerprogramm wird in einem Prozess ausgeführt, der den virtuellen Adressraum des Programms definiert. Jeder Prozess besitzt somit einen separaten Adressraum mit üblicherweise 32- oder 64-Bit Adressen, so dass einem Anwendungsprogramm ein 4GByte bzw. bei 64-Bit Adressen sogar ein $18 \cdot 10^{18}$ Byte großer Adressraum zur Verfügung steht. Die prozessspezifischen Adressräume ermöglichen die isolierte Ausführung der Benutzerprogramme, so dass sie vor absichtlichen und unabsichtlichen Fehlern anderer Anwendungen geschützt werden.

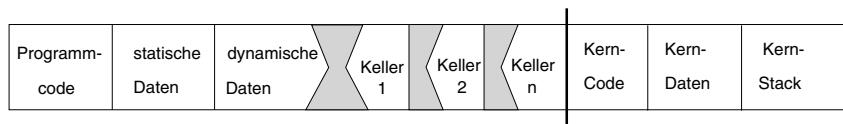
Adressraum-organisation

Abbildung 12.1 (a) zeigt eine typische Adressraumaufteilung heutiger Prozesse. Der Raum ist in einen Benutzer- und Systembereich aufgeteilt. Solange der Prozess im Benutzermodus arbeitet, darf er nur auf die Adressen des Benutzerbereichs zugreifen. Nach einem Moduswechsel stehen dann die Adressen des Systembereichs zur Verfügung. Der grau unterlegte Bereich symbolisiert, dass der dynamische Speicherbereich (engl. *heap*) sowie der Bereich des Laufzeitkellers (engl. *stack*) eines Prozesses aufeinander zu wachsen. Die handelsüblichen Prozessoren kontrollieren mögliche Überschneidungen dieser beiden Bereiche. Dazu wird in der Regel eine spezielle Sperrseite zur Trennung von Keller- und Haldenspeicher festgelegt. Wird eine Adresse dieser Sperrseite adressiert, so bedeutet dies, dass ein Keller- bzw. Heapüberlauf aufgetreten ist. Zugriffe auf Adressen der Sperrseite sind somit unmittelbar von der Hardware als Überläufe erkennbar. Auf diese

Weise wird sichergestellt, dass keine unautorisierten Zugriffe infolge eines Halden- oder Kellerüberlaufs auftreten können.



(a) Adressraumorganisation eines Prozesses



(b) Adressraumorganisation eines Multi-Threaded Prozesses

Abbildung 12.1: Adressraumorganisation eines Prozesses

Thread-Konzept

Zur Steigerung des Parallelitätsgrades setzen heutige Systeme zunehmend das Konzept der leichtgewichtigen Prozesse (engl. *thread*) ein. Ein Thread definiert einen eigenständigen, sequentiellen Ausführungspfad innerhalb des virtuellen Adressraumes eines Prozesses. Jeder Thread verfügt über sein eigenes Datensegment sowie einen eigenen Laufzeitkeller und darf die Ressourcen des umgebenden Prozesses, wie zum Beispiel das Codesegment und globale Daten, gemeinsam mit den anderen Threads des Prozesses nutzen.

Threads

Abbildung 12.1 (b) veranschaulicht die gemeinsame Nutzung eines Prozessraumes durch leichtgewichtige Prozesse. Beherbergt ein Prozess mehrere leichtgewichtige Ausführungspfade (Multi-Threading), so sieht man anhand der Adressraumorganisation von Abbildung 12.1 (b) deutlich, dass jeder Thread des Prozesses zwar seinen eigenen Kellerbereich besitzt, dass aber alle diese Threads die prozessglobalen Daten und den Programmcode gemeinsam und ohne weitere Schutzwälle nutzen. Darauf hinaus wird das Erkennen von Überläufen von Thread-Kellern in heutigen Prozessoren in der Regel nicht unterstützt, so dass durch das absichtliche oder unabsichtliche Überschreiten von Kellergrenzen Sicherheitsprobleme auftreten können. Zu deren Vermeidung wären spezielle Adressierungstechniken notwendig oder der Übersetzer müsste einen speziellen Kontrollcode generieren, der beim Belegen von Speicher überprüft, dass dadurch die zulässige Thread-Kellergröße nicht überschritten wird. In heutigen Systemen sind jedoch

fehlende Isolierung

weder die hardwareunterstützte Adressierung noch die Übersetzerunterstützung vorhanden.

Da Prozessadressräume sehr grobgranulare Einheiten zur Durchführung von Zugriffskontrollen sind, werden Programme häufig in logische (Segmente) oder physikalische (Seiten) Einheiten aufgeteilt, die als gemeinsam nutzbare Komponenten verwaltet und mit Zugriffsrechten versehen werden.

Segmentierung

Segment

Ein Segment ist eine logische Programmeinheit wie beispielsweise der Code einer Prozedur oder die Daten eines Feldes. Prozesse können Rechte zum Zugriff auf Segmente besitzen, so dass diese gemeinsam, aber dennoch mit unterschiedlichen Berechtigungen, nutzbar sind. Über die Zugriffsrechte kann man festlegen, ob auf ein Segment nur lesend (z.B. Datensegment mit Konstanten), nur ausführend (z.B. das Codesegment) oder ob schreibend darauf zugegriffen werden darf. Durch eine gezielte Berechtigungsvergabe lassen sich dann a priori Fehlzugriffe einschränken bzw. abwehren. So verhindert beispielsweise die Festlegung, auf ein Codesegment ausschließlich ausführend zugreifen zu können, eine unerlaubte Modifikation des Codes und beschränkt damit die Möglichkeiten, unbemerkt Viren einzuschleusen. Im Kapitel 2.2 haben wir des Weiteren bereits gesehen, dass Buffer-Overflow Probleme eingedämmt werden können, wenn man keinen ausführenden Zugriff auf das Segment des Laufzeitkellers erlaubt.

Rechte

Die Daten innerhalb eines Segments werden über eine virtuelle Adresse v , bestehend aus einem Paar (Segmentnummer i , Offset k), adressiert und für jeden Prozess verwaltet das Betriebssystem eine prozessspezifische Segmenttabelle. Diese enthält für jedes vom Prozess genutzte Segment eine interne Beschreibung, einen Segmentdeskriptor. Bestandteile des Deskriptors sind neben segmentspezifischen Informationen insbesondere die Zugriffsrechte, die der Prozess an dem Segment besitzt.

Verwaltung

Für jeden Segmentzugriff ist eine hardwareunterstützte Adressrechnung durchzuführen, wobei geprüft wird, ob für das in der virtuellen Adresse $v = (i, k)$ angegebene Segment mit der Identifikation i ein Deskriptor in der Segmenttabelle existiert, ob der angegebene Offset k eine gültige Segmentadresse bezeichnet und ob der zugreifende Prozess die für den Zugriff auf das Segment i benötigten Rechte besitzt. Die Adressrechnung ist in Abbildung 12.2 skizziert.

Adressrechnung

Wesentlich an der beschriebenen Vorgehensweise ist, dass für jeden Zugriff eine hardwareunterstützte Schutzüberprüfung durchgeführt wird, die nicht umgangen werden kann. Dadurch ist es für Benutzer nicht möglich, unerlaubt auf ein Segment bzw. auf ein Datum innerhalb des Segments zuzugreifen. Ein differenzierter Schutz der Objekte innerhalb eines Segmentes ist auf

Hardwarebasierte Kontrolle

$v = (i, k)$ Segmentadresse des Datenobjekts o

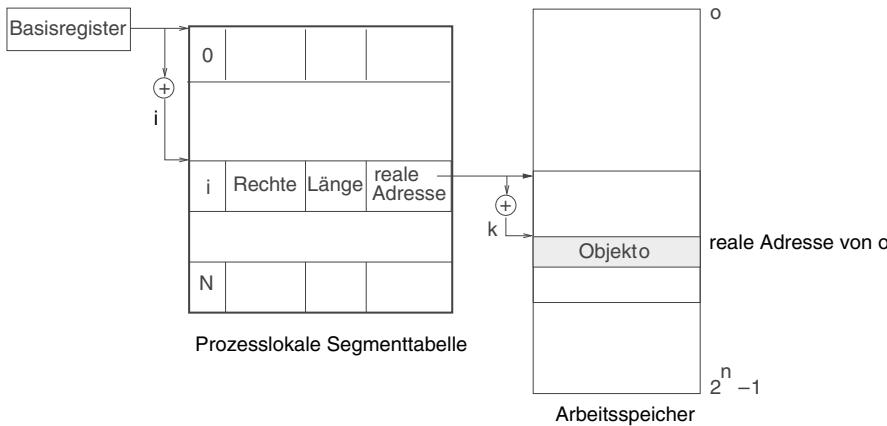


Abbildung 12.2: Adressübersetzung in segmentierten Speichern

diese Weise jedoch nicht möglich. Die hardwarebasierten Kontrollen auf der Basis von Segmenten sind somit durch weitere Maßnahmen zu verfeinern.

Paging

In einem System mit Seiteneinteilung wird ein Programm in gleich große Einheiten, die Seiten (engl. *page*), aufgeteilt, deren Größe stets eine Zweierpotenz ist. Der Arbeitsspeicher besteht ebenfalls aus gleich großen Einheiten, den Seitenrahmen (engl. *page frame*), deren Größe der Seitengröße entspricht oder ein Vielfaches davon ist. Eine virtuelle Adresse wird von der MMU (Memory Management Unit) als ein Paar (i, k) interpretiert, das aus einem Identifikator i für eine Seite und einem Offset k (meist die k niederwertigsten Bits der Adresse) besteht. Das Betriebssystem verwaltet im einfachsten Fall für jeden Prozess eine Seitentabelle und überprüft hardware-unterstützt bei jedem Zugriff, ob die Seite existiert und ob der Offset eine gültige Seitenadresse bezeichnet. Eine Zugriffsverletzung löst eine Ausnahmebehandlung im Prozessor aus und die referenzierte Adresse wird von der MMU nicht weiter aufgelöst. Da eine Seite Datenobjekte von unterschiedlichen logischen Einheiten (z.B. prozesslokale Daten und Code) beinhalten kann, ist eine gezielte und differenzierte Rechtevergabe schwierig.

Um die Vorteile der Seiteneinteilung (u.a. sehr einfache Verwaltung) sowie die Differenzierungsmöglichkeiten einer Segmentierung nutzen zu können, werden häufig beide Vorgehensweisen kombiniert. Ein Programm wird dazu zunächst in logische Einheiten (z.B. Code-, Daten- und Kellersegment) und diese dann in Seiten unterteilt, für die Schutzrechte, üblicherweise sind dies lesen, schreiben und ausführen, vergeben werden können. Auf

Seiten

Kombination

diese Weise ist auch ein großes Datensegment weiter unterteilbar und die einzelnen Datenbereiche sind mit unterschiedlichen Zugriffsrechten effizient verwaltbar.

grobe Granularität

Seiten sind jedoch grobgranulare Schutzeinheiten. Übliche Seitengrößen bewegen sich zwischen 4 und 8 KByte, so dass ein Schutz einzelner Datenobjekte mit einer feinen Granularität wie z.B. Records in Datenbanken nur möglich ist, wenn für jedes Objekt eine eigene Seite reserviert wird. Da dies natürlich unter Effizienzgesichtspunkten völlig inakzeptabel ist, stößt man hier an die Grenzen der hardwareunterstützten Kontrollen, die mit handelsüblichen Architekturen durchführbar sind. Feingranularere Zugriffskontrollen erfordern entweder eine dedizierte Hardwareunterstützung wie zum Beispiel in Form von Capability-Architekturen (siehe Seite 628) oder softwarebasierte Mechanismen zur Realisierung eines Objektschutzes.

12.3 Objektschutz

Objektverwaltung

Der Speicherschutz ist ein Spezialfall des allgemeinen Problems eines Objektschutzes. Für die Zugriffe auf Objekte können unterschiedlich komplexe Operationen bzw. Methoden zusammen mit Bedingungen, die den Zugriff objektspezifisch reglementieren, festgelegt sein. Aus diesem Grund lassen sich Zugriffskontrollen nicht mehr nur auf die einfache Kontrolle von Lese-, Schreib- oder Ausführungsrechten reduzieren. Erforderlich sind vielmehr individuelle, objekt- bzw. typspezifische Kontrollen. Dies hat zur Folge, dass eine zentrale Hardwarekontrolle nicht ausreicht, um die unterschiedlichen Prüfungen durchzuführen. Die benötigten Kontrollen werden von der Objektverwaltung eines Objekttyps, dem Objektmonitor, realisiert. In heutigen Systemen sind die zu verwaltenden Objekte hauptsächlich Dateien bzw. Verzeichnisse. Die Aufgaben der Objektverwaltung sind in diesen Systemen somit Bestandteile der Dateiverwaltung, die im Betriebssystemkern oder über Serverprozesse außerhalb des Kerns realisiert sein kann. Die bereits eingeführten Segmente und Seiten kann man ebenfalls als Objekte auffassen, wobei die Hardware die Aufgabe der Objektverwaltung wahrnimmt.

Zugriffsmatrix

Der formalisierte Ausgangspunkt für die Realisierung eines Objektschutzes ist eine Spezifikation der Sicherheitsstrategie mittels eines Zugriffsmatrix-Modells (vgl. Kapitel 6.2.1). Die Zugriffsmatrix ist in der Regel nur dünn besetzt (engl. *sparse*), so dass eine zweidimensionale Implementierung ineffizient ist. In heutigen IT-Systemen werden im Wesentlichen zwei Konzepte, bzw. auch Kombinationen aus diesen, zur Implementierung einer Zugriffsmatrix eingesetzt, nämlich Zugriffskontrolllisten (engl. *access control list* (ACL)) und Zugriffsausweise (engl. *capability*). Mit Zugriffskontrolllisten wird eine objektbezogene Sicht auf die Zugriffsmatrix realisiert, während

Zugriffsausweise eine subjektbezogene Sicht einnehmen. Abbildung 12.3 spiegelt diese beiden Sichten wider.

Subjekte	Objekte			
	Datei 1	Datei 2
Bill	owner, r,w,x	-		
Joe	r, x	w		
Anne	-	owner, r,x		

↑
Objekt-Sicht

← Subjekt-Sicht

Abbildung 12.3: Realisierungssichten einer Zugriffsmatrix

12.3.1 Zugriffskontrolllisten

Die Abbildung 12.3 verdeutlicht, dass Zugriffskontrolllisten eine Zugriffsmatrix spaltenweise implementieren. Wie der Name schon sagt, ist eine Zugriffskontrollliste (ACL) eine listenartig organisierte Datenstruktur, die einem zu schützenden Objekt zugeordnet wird. Jeder Listeneintrag identifiziert ein Subjekt, meist einen Benutzer oder eine Benutzergruppe, sowie die Rechte, die dem Subjekt an dem Objekt eingeräumt werden. Ein Eintrag enthält also mindestens die Felder `Subjekt_Name` und/oder `Gruppen_Name` sowie `Zugriffsrechte`. Zur Vereinfachung der Notation ist es in der Regel möglich, Wild Cards als Platzhalter zu verwenden, so dass damit einem beliebigen Subjekt oder einer beliebigen Gruppe Rechte erteilt werden können. So bedeutet zum Beispiel der ACL-Eintrag `Joe, *, rw`, dass der Benutzer `Joe`, egal in welcher Gruppe er gerade tätig ist, das Lese- und Schreibrecht an dem der ACL assoziierten Objekt besitzt. Der ACL-Eintrag `*, stud, r` besagt, dass alle Mitglieder der Gruppe `stud` ein Leserecht besitzen.

ACL

Zugriffskontrolle

Zur internen Repräsentation und Verwaltung der Objekte und Subjekte verwendet das Betriebssystem Deskriptoren sowie Identifikatoren zur eindeutigen Identifikation. Ein Objektdeskriptor (z.B. der `inode` in Unix-Systemen, siehe dazu auch Abschnitt 12.4) ist eine Datenstruktur, die eine Beschreibung des Objekts enthält. Dazu gehören, neben dem eindeutigen Objektidentifikator, Informationen, die für eine effiziente Verwaltung des Objekts benötigt werden, wie u.a. die Speicherabbildungstabellen, falls es

Deskriptor

sich um ein Prozessobjekt handelt, oder Dateikontrollblöcke mit Verweisen auf die Hintergrundspeicheradressen der Datenblöcke, falls es sich um ein Dateiobjekt handelt. Die zur Durchführung von Sicherheitskontrollen wichtigen Informationen eines Objektes umfassen den Namen des Eigentümers, den Objekttyp, den Zeitpunkt des letzten modifizierenden Zugriffs, die Länge des Objekts und insbesondere auch seine Zugriffskontrollliste. Die Kontrolle der Zugriffe auf ein Objekt ist die Aufgabe der Objektverwaltung, der das Objekt zugeordnet ist. Die Angaben über die Objektlänge und die Modifikationszeitpunkte werden übrigens auch von Virenerkennungsprogrammen benötigt, um potentiell unautorisierte Objektmodifikationen zu erkennen.

fälschungssichere Verwaltung

Die Objektverwaltung hat die Aufgabe, die Objektdeskriptoren fälschungssicher zu verwalten, damit die Zugriffskontrolllisten nicht unautorisiert modifiziert werden können. Ist diese Verwaltung Bestandteil des Betriebssystemkerns, so sind Zugriffe auf Deskriptoren nur im privilegierten Modus zulässig. Der Betriebssystemkern gewährleistet somit die fälschungssichere Objektverwaltung, solange diese Deskriptoren im Arbeitsspeicher verwaltet werden. Dies ist jedoch erst nach dem Booten des Betriebssystems bzw. nach dem Laden der Deskriptoren in den Arbeitsspeicher der Fall. Da diese Daten persistent auf der Festplatte abgelegt sind, besteht nach wie vor die Gefahr, dass auf diese Daten direkt, d.h. unter Umgehung der Kontrollen des Betriebssystems zugegriffen wird. Verschlüsselnde Dateisysteme oder zumindest eine Festplattenverschlüsselung schützt vor solchen Manipulationsversuchen.

Berechtigungs- und Zulässigkeitskontrolle

In Client-Server-artig strukturierten Systemen erfolgt die Objektverwaltung durch Serverprozesse. Da diese meist auch im Benutzermodus ausgeführt werden, sind spezifische Maßnahmen zur Gewährleistung der sicheren Rechteverwaltung notwendig. Eine in der Praxis häufig anzutreffende Realisierungsmaßnahme besteht darin, die durchzuführenden Überprüfungen auf eine Berechtigungs- und eine Zulässigkeitskontrolle aufzuteilen. Die Berechtigungskontrolle entspricht hierbei einem Policy-Decision Point (PDP), bei dem eine Entscheidung, hier eine Zugriffsentscheidung getroffen und hierüber eine Bescheinigung ausgestellt wird. Diese Bescheinigung ist beim tatsächlichen Zugriff vorzulegen und wird durch den Policy Enforcement Point (PEP) auf Zulässigkeit geprüft.

Berechtigungs- kontrolle

Die Berechtigungskontrolle erfolgt beim erstmaligen Zugriff auf ein Objekt und wird von vertrauenswürdigen Systemdiensten durchgeführt. Diese haben auch die Aufgabe, die Zugriffskontrollisten zu verwalten. Die vertrauenswürdigen Berechtigungskontrolleure stellen bei einer erfolgreichen Überprüfung eine Berechtigungsbescheinigung an das zugreifende Subjekt

aus, die für nachfolgende Zugriffe zu verwenden ist. Die dezentralen Objektverwaltungen, die durch verschiedene Server auf der Anwendungsebene realisiert sein können, haben dann lediglich die Aufgabe, bei einem Zugriff auf ein von ihnen verwaltetes Objekt die Gültigkeit der Bescheinigung zu überprüfen. Ein Zugriff auf die sicherheitskritischen Berechtigungsinformationen, also auf die Kontrolllisten, ist hierfür jedoch nicht erforderlich. Eine Zulässigkeitskontrolle ist somit auch von nicht vertrauenswürdigen Servern durchführbar, ohne die Sicherheit des Gesamtsystems zu gefährden. Eine derartige Aufteilung der Aufgaben zwischen vertrauenswürdigen zentralen und sonstigen Servern haben wir bereits beim Kerberos Authentifikationsdienst (vgl. Kapitel 10.4.2) kennen gelernt. Die Berechtigungsbescheinigungen werden dort vom vertrauenswürdigen Kerberos-Server in Form von Tickets ausgestellt, deren Authentizität und Gültigkeit von den beteiligten Servern auf einfache Weise überprüfbar sind.

Zulässigkeit

Zur sicheren Speicherung der Berechtigungsinformationen sind kryptografische Mechanismen einzusetzen. Der autorisierte Zugriff auf die Berechtigungsinformationen erfordert die Kenntnis der verwendeten Schlüssel, die zum Beispiel auf einer Chipkarte gespeichert sein können. Durch die Verwendung kryptografischer Hashfunktionen wird zusätzlich die Erkennung unautorisierter Veränderungen ermöglicht und mit digitalen Signaturen lassen sich durchgeführte Zugriffe eindeutig einem Subjekt zuordnen. Ein derartiger Schutz der Berechtigungsinformation ist aber in den gängigen Betriebssystemen nicht vorgesehen. Hier verlässt man sich vielmehr darauf, dass nur das privilegierte Betriebssystem oder besonders berechtigte Benutzer Zugriff auf die Daten erlangen können. Angesichts der Offenheit und Verwundbarkeit heutiger Systeme ist diese Annahme aber unrealistisch. Mit verschlüsselnden Dateisystemen wird versucht, dem unberechtigten Zugriff auch noch auf eine andere Art als lediglich durch eine Rechtevergabe zu begegnen.

Verschlüsselung

Zugriffskontrolllisten werden in fast allen im Einsatz befindlichen Betriebssystemen, wie Unix [9] (siehe Abschnitt 12.4) und Linux oder Windows (siehe auch Abschnitt 13.2) verwendet.

Vor- und Nachteile von ACLs

Die Vorteile des Konzepts der Zugriffskontrollisten liegen in der einfachen Verwaltung der Zugriffsrechte, insbesondere in der einfachen und effizienten Realisierung einer Rechterücknahme. Dazu müssen nur die entsprechenden Einträge in der Zugriffskontrollliste aktualisiert werden. Außerdem ist es sehr einfach, für ein spezifisches Objekt zu bestimmen, welche Subjekte welche Zugriffsrechte an dem Objekt besitzen. Demgegenüber ist es aus dem Blickwinkel eines Subjekts, zum Beispiel eines Benutzers, sehr aufwändig, die Menge seiner aktuellen Rechte zu ermitteln.

einfache
Verwaltung

unvollständige
Kontrollen

Problematisch ist, dass die Zugriffskontrolle bei langen Listen aufwändig und ineffizient ist, da bei jedem Zugriff auf ein Objekt dessen Zugriffskontrolliste zu durchsuchen ist. Dies hat zur Folge, dass die meisten Systeme (u.a. auch Unix und Windows-Betriebssysteme) nur sehr einfache Listenstrukturen zulassen und/oder nicht jeden Zugriff kontrollieren. Vielmehr beschränken sie sich darauf, die Berechtigung beim erstmaligen Zugriff, zum Beispiel beim Öffnen einer Datei, zu überprüfen und für nachfolgende Zugriffe eine Berechtigungsbescheinigung (z.B. den File-Descriptor in Unix bzw. das Object Handle unter Windows) auszustellen. Deren Überprüfung erfordert dann aber keine aufwändigen Kontrollen mehr. Das Problem bei dieser in der Praxis weit verbreiteten Vorgehensweise ist, dass Rechteänderungen, insbesondere Rechterücknahmen, nicht unmittelbar wirksam werden. Ein Subjekt kann somit weiterhin auf ein Objekt zugreifen, auch wenn ihm in der Zwischenzeit die Berechtigungen dafür vom Objekteigentümer bereits entzogen worden sind.

nicht fälschungs-
sicher

Weiterhin wird die Forderung nach einer fälschungssicheren Verwaltung der ACLs in herkömmlichen Systemen nur unzureichend erfüllt. Ist die ACL eines Objekts zusammen mit dessen Deskriptor in den Arbeitsspeicher geladen und werden diese Datenstrukturen vom Betriebssystemkern verwaltet, so sind sie geschützt, da, wie wir erläutert haben, ein direkter Zugriff darauf nur privilegierten Prozessen möglich ist. Zu beachten ist aber, dass die Zugriffslisten bzw. Objekt-Deskriptoren in der Regel zusätzlich noch im Klartext auf der Festplatte oder auf anderen Hintergrundspeichermedien liegen. Erlangt ein Angreifer unberechtigten Schreibzugriff auf ein solches Medium, so kann die Zugriffskontrolle unterlaufen werden. Der Angreifer benötigt also lediglich den physischen Zugriff auf das Gerät. Dies stellt im Zeitalter der mobilen Endgeräte immer weniger ein Problem dar. Abhilfe könnte hier eine verschlüsselte Speicherung der Autorisierungsdaten auf dem Hintergrundspeicher bzw. verschlüsselnde Dateisysteme (vgl. Abschnitt 13.2.5) schaffen.

schlechte
Skalierbarkeit

Für den Einsatz in verteilten Systemen mit einer Vielzahl und einer dynamisch sich ändernden Menge von Subjekten, die unterschiedliche Rechte an Objekten besitzen können, ist das ACL-Konzept aufgrund seiner schlechten Skalierbarkeit eher ungeeignet. Dies gilt besonders dann, wenn, was ja eigentlich wünschenswert ist, differenzierte Berechtigungen ohne Beschränkungen für die Anzahl der Listeneinträge vergeben werden können.

Rollen

Eine Lösung für diesen Problembereich bieten rollenbasierte Zugriffsmodelle (RBAC siehe Kapitel 6.2.2). Rollen beschreiben auf einer abstrakten Ebene Aufgabenprofile von Benutzern, so dass Rechte an solche Rollen und nicht an einzelne Benutzer vergeben werden. Rollen können als spezielle Ausprägung von Gruppen betrachtet und implementiert werden. Das

heißt, dass man intern eine eindeutige Beschreibung einer Rolle benötigt und jedem Benutzer bzw. jedem seiner Prozesse den Rollenidentifikator derjenigen Rolle assoziiert, in der er aktuell im System tätig ist. Werden hierarchische Rollenmodelle eingesetzt, so erfordert dies die Realisierung eines Vererbungskonzepts, damit gewährleistet wird, dass mit einer Rollenmitgliedschaft automatisch auch die Mitgliedschaft in allen hierarchisch niedrigeren Rollen verbunden ist. Die Zuordnung von Benutzern zu Rollen, die Aufnahme neuer Rollenmitglieder oder das Löschen einer Mitgliedschaft hat keinen Einfluss auf die vergebenen Rechte an der Rolle. Das heißt, dass die Rechteeinträge in der ACL nicht modifiziert werden müssen. Die Beibehaltung von ACLs als Realisierungskonzept bei gleichzeitiger Verwendung von Rollen zur effizienten und aufgabenorientierten Berechtigungsvergabe hat zudem den Vorteil, sehr gut für verteilte Anwendungen geeignet zu sein, die ebenfalls in erster Linie auf Zugriffskontrolllisten als Realisierungskonzept zurückgreifen.

12.3.2 Zugriffsausweise

Zugriffsausweise (engl. *capability*) sind unverfälschbare Tickets, die den Besitzer zum Zugriff auf das in dem Ticket benannte Objekt berechtigen. Neben dem Objektnamen enthält ein Zugriffsausweis auch die Berechtigungen, die dem Besitzer des Ausweises an dem Objekt eingeräumt werden. Das Konzept der Zugriffsausweise ermöglicht die zeilenweise Realisierung einer Zugriffsmatrix. Dazu wird jedem Subjekt s eine Liste von Paaren (Objekt_Identifikator, Zugriffsrechte) zugeordnet. Diese Liste, genannt Capabilityliste (C-List), spiegelt die Einträge in der dem Subjekt s zugeordneten Zeile der Matrix wider.

Capability

Capability-Realisierung

Zugriffsausweise können in Software oder in Hardware realisiert sein. Software-realisierte Capabilities sind Datenstrukturen, die meist durch das Betriebssystem fälschungssicher erzeugt und verwaltet werden. In diesem Fall fungiert das Betriebssystem als vertrauenswürdiger Capabilitymanager, der eine Modifikation oder Erzeugung einer Capability nur Prozessen im privilegierten Modus erlaubt. Mit dem Übergang auf Client-Server-Architekturen und der damit verbundenen Auslagerung von Betriebssystemkerndiensten auf Serverprozesse werden Zugriffsausweise zunehmend von Serverprozessen im Benutzermodus erzeugt und manipuliert. Es sind somit zusätzliche Maßnahmen erforderlich, um das Fälschen von Zugriffsausweisen zu verhindern. Eine Möglichkeit besteht darin, Verschlüsselungstechniken einzusetzen, so dass die Berechtigungen in den Ausweisen verschlüsselt abgelegt werden und nur vertrauenswürdige Server, die im

Software-Capabilities

Besitz des passenden Schlüssels sind, eine korrekt verschlüsselte Berechtigungsvergabe durchführen können.

Hardware-unterstützung

Zur hardwareunterstützten, sicheren Verwaltung von Zugriffsausweisen werden im Wesentlichen zwei Ansätze verfolgt, nämlich die Verwendung von Capabilitysegmenten und der Einsatz getageter Architekturen. Ein Capabilitysegment enthält nur Capabilities und der Zugriff darauf erfordert spezielle Privilegien (Capabilities), die dem Betriebssystem bzw. den zuständigen Servern erteilt werden. In getagten Architekturen benötigt man eine spezielle Hardware-Unterstützung, das Tag-Bit. Über das Tag-Bit wird für jedes Speicherwort festgelegt, ob es sich um eine Capability oder um ein normales Datenwort handelt.

Capability-adressierung

Werden Capabilities direkt vom Prozessor als Adressierungstechnik angeboten, so spricht man von einer Capabilityadressierung. Diese wurde von Dennis und Van Horn bereits 1966 in [48] eingeführt. Hierbei werden die Capabilities bei der Programmausführung durch spezielle Maschinenbefehle aus der Capabilityliste des in Ausführung befindlichen Prozesses in die Capabilityregister des Prozessors geladen. Diese Register definieren die aktuell gültige Rechteumgebung eines Prozesses. Die capabilitybasierte Adressierung ist in Abbildung 12.4 skizziert. Eine virtuelle Adresse besteht aus einem Paar (c, w) , wobei c eine Capability und w ein Offset in dem durch c identifizierten Objekt o ist. Bei einer Capabilityadressierung kann auf die

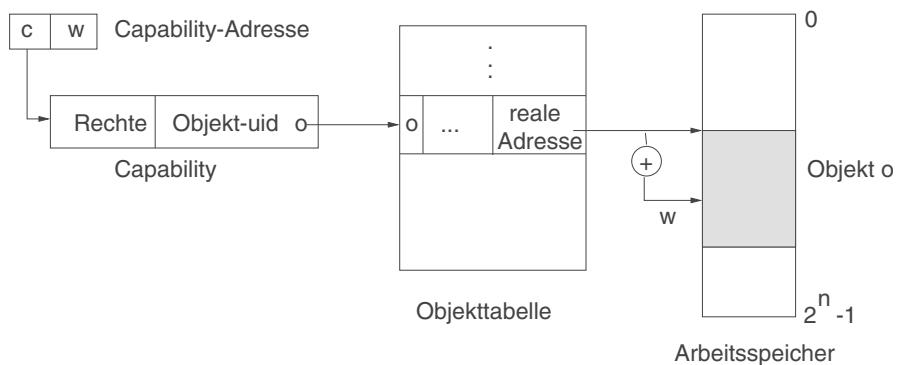


Abbildung 12.4: Capabilitybasierte Adressierung

Verwendung von virtuellen Adressräumen als Schutzeinheit verzichtet und eine hardwarebasierte Kontrolle wesentlich flexibler durchgeführt werden, da die zu schützenden Objekte mit beliebiger Granularität festgelegt werden können und keiner Seitenrasterung unterliegen. Capabilityarchitekturen sind jedoch unter den Prozessoren des heutigen Massenmarktes nicht mehr

anzutreffen. Diese bieten vornehmlich seiten- und segmentbasierte Schutzkonzepte an.

Vor- und Nachteile von Capabilities

Im Vergleich zu Zugriffslisten bietet das Konzept der Capabilities eine deutlich größere Flexibilität. Beim Ausstellen einer Capability werden die für das Subjekt zum Ausstellungszeitpunkt geltenden Rechte an dem betreffenden Objekt in die Capability eingetragen. Ein Subjekt darf stets dann auf ein Objekt zugreifen, wenn es im Besitz einer gültigen Capability dafür ist. Dadurch können die Zugriffskontrollen wesentlich vereinfacht werden, da nur eine Zulässigkeitskontrolle durchgeführt werden muss. Das unter Umständen aufwändige Durchsuchen einer langen Kontrollliste bzw. das Befragen eines entfernten, vertrauenswürdigen Zugriffsservers entfällt.

große Flexibilität

Weiterhin ermöglicht das Capabilitykonzept ein dezentrales Sicherheitsmanagement, da die Verwaltung der sicherheitskritischen Informationen und die zur Ausstellung von fälschungssicheren Capabilities notwendigen, vertrauenswürdigen Managementkomponenten von den Komponenten, die die Zulässigkeitskontrollen durchführen, getrennt werden können. Letztere lassen sich als Bestandteile von Servern der Anwendungsebene implementieren. Zugriffsausweise sind somit als Realisierungskonzept auch besonders für dezentrale Client-Server-Architekturen geeignet. Capabilities sind damit in gewisser Weise als Zugriffsberechtigungs-Analogon zu Tickets zu betrachten, die vom Kerberos-KDC für Principals ausgestellt und dezentral durch die Clients verwaltet werden. Auch die Autorisierungs-Assertions von SAML (vgl. Seite 656) kann man als eine erweiterte Form des Capability-Konzepts auffassen.

dezentrale Verwaltung

Da Zugriffsausweise nicht an Subjekte gebunden sind (sie enthalten keinen Subjektkennwert), können sie auf einfache Weise, ggf. mit gezielten Einschränkungen, an andere Subjekte weitergegeben werden. Eine kontrollierte Weitergabe von Berechtigungen im Sinne einer Delegation von Rechten wird insbesondere in Client-Server-Architekturen benötigt, wenn der Server im Auftrag eines Clients aktiv sein soll. Ferner ist es durch eine einfache Erweiterung der Capabilitydatenstruktur möglich, die Gültigkeitsdauer eines Zugriffsausweises zu beschränken oder einen Zugriff nur unter Vorlage einer Kombination aus mehreren Ausweisen zu erlauben. Damit lassen sich Sicherheitsanforderungen erfassen, die man mit dem Stichwort des „Mehraugenprinzips“ charakterisieren kann. Das heißt, dass ein erfolgreicher Zugriff auf ein derartig geschütztes Objekt gegebenenfalls das koordinierte Zusammenwirken von Subjekten erfordert, die nur jeweils einen Teil der benötigten Berechtigungen besitzen.

Rechte delegation

Schutzdomäne

Schutzdomäne

Ein weiterer wesentlicher Vorteil des Konzepts der Zugriffsausweise ist die Unterstützung differenzierter Schutzdomänen (engl. *protection domain*). Dies ermöglicht eine Rechtevergabe gemäß dem Prinzip der minimalen Rechte. So kann jedes Modul oder jede Prozedur bzw. Methode als eigenständiges Subjekt mit einer eigenen Capabilityliste behandelt werden. Diese C-Liste legt die Rechteumgebung für die Ausführung des assoziierten Codes fest. Bei jedem Prozedur- bzw. Methodenaufruf findet ein Schutzbereichswechsel statt, wobei es meist auch möglich ist, Capabilities als Parameter zu übergeben und damit die Rechteumgebung des aufgerufenen Codes zu erweitern. Diese differenzierten Schutzumgebungen sind ein wichtiger konzeptioneller Fortschritt gegenüber dem Konzept der Zugriffslisten, die als Subjekte in der Regel nur Benutzer oder Gruppen zulassen. Das hat zur Folge, dass ein Prozess, der im Auftrag eines Benutzers aktiv ist, zu jedem Zeitpunkt stets alle Rechte seines Benutzers besitzt und nicht nur diejenigen, die er zur Durchführung seiner jeweiligen Aufgabe tatsächlich benötigt. Fasst man die Prozeduren als Implementierungen von Teilaufgaben auf, so erkennt man unmittelbar, dass mit den Capabilities die zur Erfüllung der Teilaufgaben notwendigen, minimalen Rechteumgebungen zur Verfügung gestellt werden.

Beispiel 12.1 (Schutzdomänen auf der Basis von Capabilities)

Prozedurspezifischer Schutz

Gegeben seien zwei Prozeduren P und Q mit den in Abbildung 12.5 angegebenen assoziierten Capabilitylisten. Jede der Listen enthält unter anderem die Berechtigung zum Zugriff auf das jeweilige Datensegment D_P bzw. D_Q . Die Capabilityliste der Prozedur P enthält keine Capability für das Datensegment D_Q von Q, so dass ein Versuch, bei Ausführung des Codes von P auf das Datensegment D_Q zuzugreifen, als unzulässig zurückgewiesen würde. Die Prozedur P besitzt jedoch einen speziellen Zugriffsausweis für den Zugriff auf Q, eine so genannte *enter-Capability*, die P dazu berechtigt, die Prozedur Q aufzurufen. Beim Aufruf von Q durch P erfolgt ein Schutzbereichswechsel, so dass nunmehr die Berechtigungen der Capabilityliste von Q gelten. Das bedeutet, dass P für die Dauer der Ausführung des Codes von Q eine Capability mit Lese- und Schreibrechten für das Datensegment D_Q besitzt, dafür aber nicht mehr auf seine eigenen Segmente zugreifen darf. Nach Beendigung der Ausführung von Q erfolgt erneut ein Schutzbereichswechsel mit der Konsequenz, dass danach für die Prozedur P wieder nur ihre eigenen Capabilities nutzbar sind.



Rechterücknahmeproblem

Zu den Hauptproblemen capabilitybasierter Systeme gehört die dynamische Rechterücknahme. Dazu müssen die ausgegebenen Zugriffsausweise entwe-

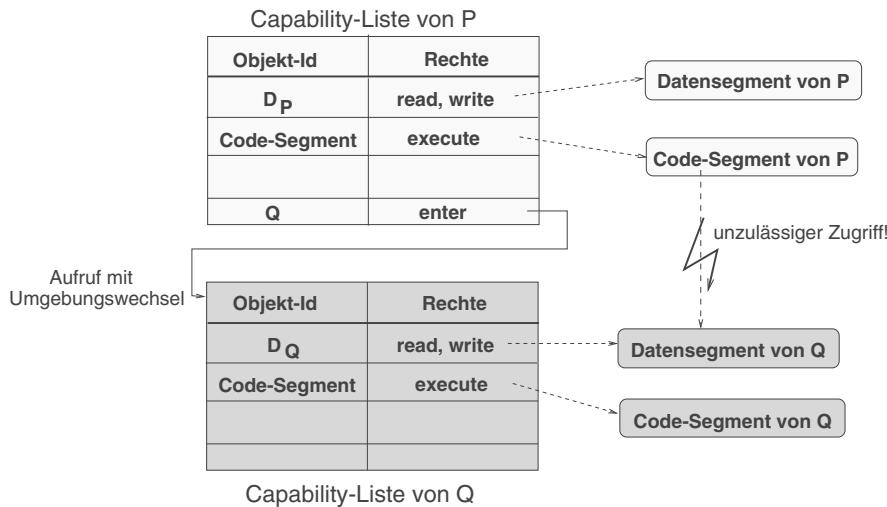


Abbildung 12.5: Schutzumgebungen von Prozeduren

der zurückgefördert oder ungültig gemacht werden. Die erste Lösung ist jedoch meist nicht praktikabel, da Capabilities unbeschränkt kopiert werden können, so dass eine solche Rückrufaktion, insbesondere in einem verteilten System, einen inakzeptablen Aufwand erforderte. Capabilities werden beispielsweise kopiert, wenn ein Prozess Kind-Prozesse erzeugt und diese die Ressourcen des Vater-Prozesses mit benutzen sollen. Eine in der Praxis eingesetzte Technik, Capabilities ungültig zu machen, besteht darin, dass diejenige Instanz, die die Zugriffsausweise erzeugt, eine Tabelle mit aktuell gültigen Ausweisen verwaltet. Bei der Vorlage einer Capability durch ein Subjekt kann dann deren Gültigkeit überprüft werden. Eine Alternative hierzu besteht darin, Capabilities mit einer Gültigkeitsdauer zu versehen, analog zu den Kerberos-Tickets, so dass sie nach Ablauf ihrer Gültigkeit nicht mehr benutzbar sind. Hierdurch wird zwar eine Rechterücknahme nicht direkt wirksam, es schränkt jedoch den Zeitraum eines möglichen Missbrauches ein.

Kombinierte Techniken

Eine allgemeinere und flexiblere Lösung für das Rechterücknahmeproblem liefert das Lock/Key-Verfahren, das eine Kombination aus Zugriffslisten und Zugriffsausweisen ist. Jedem Subjekt s wird eine Capabilityliste zugeordnet, die Paare der Form (o, K) enthält. Dabei bezeichnet o ein Objekt, auf das s unter Anwendung des Schlüssels K zugreifen darf. Jedes Objekt o besitzt seinerseits eine Zugriffskontrollliste, die Einträge der Form (L, α) enthält, wobei L ein Schloss ist und α diejenigen Zugriffsrechte sind, die den Besitzern des zum Schloss L passenden Schlüssels K eingeräumt werden.

Lock/Key

Zugriffskontrolle

Möchte nun ein Subjekt s gemäß der Rechte β auf das Objekt o zugreifen, so legt es der Zugriffskontrolle von o seine Capability (o, K) vor. Ein Zugriff ist zulässig, wenn zum einen der Schlüssel K in ein Schloss L der Zugriffsliste von o passt, d.h. wenn es einen Eintrag (L, α) gibt, mit $K = L$. Zum anderen müssen auch die von dem Subjekt gewünschten Zugriffe zulässig sein, also $\beta \subseteq \alpha$.

Rechterücknahme

Eine Rechterücknahme ist einfach und effizient zu realisieren, da lediglich in der Zugriffskontrollliste des Objekts das Schloss L verändert werden muss, so dass der in den Capabilities enthaltene Schlüssel K nicht mehr passt. Zur Rechterücknahme sind keine Kenntnisse über den oder die aktuellen Besitzer vergebener oder weitergegebener Capabilities notwendig. Wird der Zugriffsversuch eines Subjekts aufgrund einer Schlossänderung abgewiesen, so kann es von der zuständigen Objektverwaltung die Ausstellung einer neuen, aktuellen Capability beantragen.

Vereinfachungen

Schlüssel/Schlossverfahren werden nur selten in dieser Reinform in der Praxis eingesetzt. Die meisten Betriebssysteme realisieren vielmehr eine stark vereinfachte Kombination aus Capabilities und ACLs. In den folgenden beiden Abschnitten stellen wir mit den Zugriffskontroll-Diensten von Unix und Windows zwei typische Vertreter gängiger Betriebssystemfamilien vor.

12.4 Zugriffskontrolle in Unix

Als Fallbeispiel für den Bereich der Zugriffskontrolle klassischer Betriebssystemarchitekturen dient uns wieder das Unix Betriebssystem (vgl. Kapitel 10.2.2). In Unix sind Dateien sowie Verzeichnisse die zu schützenden Objekte und Benutzer, Benutzergruppen sowie Prozesse die Subjekte. Alle externen Geräte wie beispielsweise Bildschirme, Drucker, Modems, sowie Massenspeicher wie Festplatten, CD-ROM Laufwerke, Bänder oder auch der Arbeitsspeicherbereich des Betriebssystemkerns (`/dev/kmem`) werden ebenfalls als Dateien modelliert. Aus diesem Grund bedeutet die Kopplung der Zugriffskontrolle an das Dateikonzept keine so große Einschränkung, wie es auf den ersten Blick erscheint. Dennoch treffen natürlich auch hier die allgemeinen Kritikpunkte zu, die wir bereits an anderer Stelle (siehe Seite 240 ff) zum Dateikonzept als Basis für Schutzfestlegungen angeführt haben.

12.4.1 Identifikation

Identifikation

Das Unix Dateisystem ist baumartig strukturiert. Die internen Knoten des Baumes sind die Verzeichnisse, die ihrerseits Dateien (Blätter des Baumes) oder weitere Unterverzeichnisse enthalten können. Zur Benennung einer Datei, also eines zu schützenden Objekts, gibt man deren Pfadnamen an, der mit

dem Namen des Wurzelverzeichnisses des Baumes beginnt und alle Namen der auf dem Pfad liegenden Verzeichnisse enthält. Zur Benennung von Prozessen werden vom Betriebssystemkern eindeutige Prozessidentifikatoren, die pids, vergeben und die Benutzer werden ebenfalls, wie wir bereits bei der Behandlung der Identifikations- und Authentifikationsdienste (siehe Seite 460) gesehen haben, durch eindeutige Identifikatoren, die uids und die Gruppenidentifikation guid, repräsentiert. Eine uid war ursprünglich eine 16-Bit Zahl, ist aber in den aktuellen Unix-Systemen nunmehr meist ein 32-Bit Wert. uid-Werte zwischen 0 und 99 sind in der Regel für Systemprozesse reserviert und die uid = 0 ist dem Superuser vorbehalten (siehe Seite 460).

Intern unterscheidet man zusätzlich noch zwischen der realen und der effektiven uid bzw. guid. Die aktuellen Zugriffsberechtigungen hängen von der effektiven Identifikation ab, die sich dynamisch im Verlauf der Benutzeraktivitäten ändern kann. Dies geschieht zum Beispiel durch Nutzung des setuid-Konzepts. Das heißt, dass eine temporäre Änderung der Rechteumgebung möglich ist. Anders als bei den Schutzumgebungswechseln, wie wir sie im Zusammenhang mit den Zugriffsausweisen kennen gelernt haben, sind jedoch unter Unix Wechsel der Rechteumgebung eher selten, so dass in den meisten Fällen die reale und die effektive uid übereinstimmen.

effektive uid

Anzumerken ist, dass für die Rechteüberprüfung die uid und nicht die Kennung ausschlaggebend ist. Das bedeutet, wenn ein Benutzer die Kennung *root* und die uid= 150 besitzt, so ist er damit keineswegs Superuser, sondern nur dann, wenn seine uid=0 ist, unabhängig davon, welche Kennung für ihn eingetragen ist. Als Systemverwalter sollte man sich jedoch an die Konvention halten und nur derjenigen Kennung den Namen *root* geben, die auch Superuser-Rechte besitzt. Die Zuordnung zwischen der Benutzer-Kennung und der uid ist in der /etc/passwd festgehalten, während die analoge Zuordnung zwischen den Gruppen und der guid in der Datei /etc/group zu finden ist.

In den meisten Unix-Systemen wird eine Gruppe namens *wheel* verwendet, die alle Systemadministratoren auflistet. Dieser Gruppenname ist in vielen Unix-Systemen von Bedeutung, da häufig die Berechtigung zur Ausführung des Kommandos *su*, mit dem man Superuser-Rechte erlangen kann, auf die Mitglieder der Gruppe *wheel* beschränkt ist.

12.4.2 Rechtevergabe

Als Zugriffsrechte sind unter Unix im Wesentlichen lesen *r*, schreiben *w* und ausführen *x* festgelegt. Nur der Eigentümer eines Objekts *o* sowie die Benutzer mit Root-Rechten haben das Recht zur Weitergabe und Rücknahme von

Rechte

Rechten an dem Objekt o . Jedem Objekt ist eine sehr einfache Variante einer Zugriffskontrollliste zugeordnet, in der festgehalten wird, welche Rechte ein Benutzer an dem Objekt besitzt. Aus der Sicht eines Objekts zerfällt die Menge der autorisierten Benutzer in drei Klassen, nämlich in die einelementige Menge, die den Objekteigentümer enthält (Owner-Rechte), in die Klasse der Benutzer, die der Gruppe, zu der das Objekt gehört, angehören (Gruppenrechte) und in die Klasse, die alle anderen Benutzer umfasst (World-Rechte). Eine Unix-Zugriffskontrollliste hat damit die in Abbildung 12.6 angegebene, allgemeine Struktur.

Unix-ACL



Abbildung 12.6: Struktur einer ACL in Unix Systemen

Dateityp

An Dateitypen unterscheidet man:

- einfache Datei (regular file)
- d Verzeichnis (directory)
- l Verweis (link)
- c zeichenorientiertes Gerät (z.B. Terminal, Drucker)
- b blockorientiertes Gerät (z.B. Band)

Verzeichnisrechte

Handelt es sich bei der Datei um ein Verzeichnis (Typinformation d), so sind die in Abbildung 12.6 angegebenen Rechte wie in Tabelle 12.1 zu interpretieren.

Recht	Semantik für Verzeichnisse
r	Leserecht: es erlaubt, die Dateien des Verzeichnisses mit dem ls Kommando zu listen.
x	Sucherecht: es erlaubt, das Verzeichnis als Teil eines Pfadnamens zu durchlaufen bzw. mit dem Kommando cd das Verzeichnis „zu betreten“ und darin befindliche Dateien zu öffnen.
w	Schreib-Recht: es erlaubt das Hinzufügen oder Entfernen von Dateinamen oder Links aus dem Verzeichnis.

Tabelle 12.1: Verzeichnisrechte in Unix

Beispiel 12.2 (Unix-ACL)

Gegeben sei die Zugriffskontrollliste (drwx r- - -). Sie besagt, dass das zu schützende Objekt ein Verzeichnis ist, dessen Eigentümer das Lese-, Schreib- und Ausführrecht besitzt. Die Mitglieder der Gruppe des Verzeichnisses haben lediglich das Recht, das Verzeichnis zu lesen, also sich die lokalen Dateien auflisten zu lassen. Allen anderen Benutzern des Systems sind jegliche Rechte an dem Verzeichnis verwehrt.



Um auf eine Datei zugreifen zu können, muss das zugreifende Subjekt, also der Benutzer, für alle Verzeichnisse des Pfadnamens der Datei das Sucherecht besitzen. Mit dem Sucherecht ist jedoch nicht gleichzeitig ein Lese-Recht verknüpft, so dass allein mit dem x-Recht ein Auflisten der verzeichnislokalen Dateien nicht zulässig ist. Dies wird manchmal ausgenutzt, um Dateien „zu verstecken“, da der Benutzer nur dann auf die Datei zugreifen kann, wenn er deren Namen kennt. Natürlich ist das nur eine sehr rudimentäre Art des Schutzes (security by obscurity).

x-Recht

Beispiel 12.3 (Pfadname)

Es sei folgender Pfadname gegeben: /usr/local/bin/latex. Um die Datei `latex` auszuführen, also auf diese zugreifen zu dürfen, müssen die Verzeichnisse `/usr`, `/local` und `/bin` durchlaufen werden, wofür jeweils das x-Recht benötigt wird.



Zu beachten ist ferner, dass mit den w- und x-Rechten für ein Verzeichnis das Recht zum Entfernen von Dateinamen erteilt wird, unabhängig davon, welche Rechte individuell für die verzeichnislokalen Dateien festgelegt sind und unabhängig davon, welche Benutzer die Eigentümer der Dateien sind!

w-Recht

Beispiel 12.4 (Semantik des w-Rechtes)

Gegeben seien ein Verzeichnis d und eine Datei f , die in dem Verzeichnis d liegt. Die Zugriffskontrolllisten der beiden Objekte seien wie folgt festgelegt:

$$acl_d = 730 = (rwx-wx---)$$

$$acl_f = 640 = (rw-r-----).$$

Entsprechend dieser Festlegungen besitzen alle Benutzer, die zur Gruppe des Verzeichnisses d gehören, die Schreiberlaubnis für das Verzeichnis d , aber sie besitzen nicht das Recht, auf die Datei f schreibend zuzugreifen. Da jedoch das Schreibrecht für ein Verzeichnis das Recht zum Löschen einer beliebigen Datei innerhalb des Verzeichnisses impliziert, kann ein Gruppenmitglied die Datei f aus dem Verzeichnis d entfernen und durch eine andere

Datei f' ersetzen. Diese Zugriffe sind gemäß der Zugriffsbeschränkungen des Objekts d erlaubt, widersprechen jedoch den mit der Rechtevergabe intendierten Festlegungen bezüglich der Datei f . Kenntnisse über die Semantik der zugrunde liegenden Rechteverwaltung sind unabdingbare Voraussetzungen zur Vermeidung inkonsistenter Zustände, die ja von einem Unix System nicht angezeigt werden.



Sonderrechte

Spezielle Rechte können über das Setzen des so genannten **sticky-Bits** und des **suid-** bzw. des **sgid-Bits** vergeben werden. Das **sticky-Bit** (t-Bit) darf nur vom Superuser gesetzt werden. Eine entsprechende Rechtevergabe kann man sich mittels des `ls`-Kommandos anzeigen lassen. Ist das Bit gesetzt, so wird der letzte x-Eintrag der ACL in ein t umgewandelt. Ohne gesetztes t-Bit hat in einem Verzeichnis jeder Benutzer, der Schreib-Berechtigung für das Verzeichnis besitzt, auch die Berechtigung, jede Datei in diesem Verzeichnis zu löschen, also auch diejenigen Dateien, deren Eigentümer er gar nicht ist. Das für ein Verzeichnis gesetzte Bit bedeutet, dass nur der Eigentümer einer Datei diese aus dem Verzeichnis löschen darf. Dies ist unter anderem für das Verzeichnis `/tmp` sinnvoll, für das alle Benutzer Schreib- und Ausführrechte besitzen. Das für `/tmp` gesetzte **sticky-Bit** stellt sicher, dass nur der jeweilige Eigentümer und der Superuser Dateien aus `/tmp` löschen oder mittels des `move`-Befehls an eine andere Stelle transferieren dürfen. Für eine Datei bedeutet das gesetzte **sticky-Bit**, dass sie vom Betriebssystem permanent im Speicher zu halten ist, also von der Speicherverwaltung nicht auf Hintergrundspeichermedien ausgelagert werden darf. In heutigen Unix-Systemen hat das t-Bit nur noch in Verbindung mit Verzeichnissen eine Bedeutung.

sticky-Bit

suid-Recht

Das **suid**-Konzept ermöglicht die temporäre Weitergabe von Rechten eines Benutzers oder einer Gruppe. Führt ein Benutzer eine Programmdatei aus, für die das **suid**-Bit gesetzt ist, so wird die effektive **uid** des ausführenden Prozesses durch die **uid** desjenigen Benutzers ersetzt, der der Eigentümer der Programmdatei ist. Das bedeutet, dass der Prozess das Programm mit den Rechten des Eigentümers und nicht mit seinen eigenen ausführt. Das **suid**-Bit wird durch den Buchstaben **s** bei den Eigentümerrechten gekennzeichnet und ersetzt den x-Eintrag, wenn man sich die Datei-Attribute mit dem `ls`-Kommando anzeigen lässt. Im Gegensatz zum **sticky**-Bit können die **s**-Rechte sowohl vom Eigentümer einer Datei als auch vom Superuser mit dem normalen `chmod`- Kommando gesetzt werden, also z.B. durch `chmod u+s Dateiname`.

Beispiel 12.5 (suid-Programme)

In Unix Systemen ist das **suid**-Bit unter anderem für das `/bin/passwd`-Kommando gesetzt. Dadurch ist es möglich, dass ein Benutzer sein Passwort ändern kann, obwohl nur der Superuser, also die Kennung `root`, Schreibrechte auf die Datei `/etc/passwd` besitzt, für die in der Regel die Zugriffsrechte `-rw- r-- r--` vergeben sind. Das heißt, dass nur der Eigentümer der Datei, das ist Root, Schreibrechte an der Datei besitzt. Bei der Ausführung des `/bin/passwd`-Programms wird auf die Datei `/etc/passwd` zugegriffen. Durch das gesetzte **suid**-Bit erhält der Benutzer bei der Ausführung des `/bin/passwd`-Programms für die Dauer dieser Ausführung Superuserrechte. Die ACL des `/bin/passwd`-Programms könnte beispielsweise wie folgt gegeben sein:

```
-r-sr-xr-x 1 root kmem 24576 Nov 30 1992 /bin/passwd .
```

Ein weiteres Beispiel für die nützliche Verwendung des **suid**-Konzepts liefert das `sendmail`-Programm, dessen Eigentümer ebenfalls die Kennung `root` ist. Das Setzen des **suid**-Bits gewährleistet, dass ein Benutzer seine Mails in die Warteschlange des Mailers eintragen kann und damit temporär berechtigt ist, schreibend auf Datenstrukturen des Kerns zuzugreifen.



In analoger Weise können auch temporär Gruppenberechtigungen erteilt werden. Hierzu dient das **sgid**-Bit. Ein für eine Datei gesetztes **sgid**-Bit besagt, dass für die Dauer der Dateiausführung die effektive Gruppenidentifikation des ausführenden Prozesses die Gruppenidentifikation der **sgid**-Datei annimmt, also die entsprechenden Gruppenrechte wahrnehmbar sind.

`passwd`

Sicherheitsprobleme mit **suid**

Der **suid**-Mechanismus ist nützlich, birgt jedoch bei unsachgemäßem Einsatz große Sicherheitsrisiken, wie folgende Beispiele exemplarisch verdeutlichen sollen.

`sendmail`

Beispiel 12.6 (suid-Risiken)

Gegeben sei eine Datei namens `beispieldat` mit gesetztem **suid**-Bit. Der Eigentümer der Datei sei der Benutzer mit der Kennung `Joe` und die Rechte der Datei seien wie folgt vergeben:

```
-rwsr-rw- 1 joe stud 35373 Nov 20 1999 beispieldat
```

Das heißt, nicht nur der Eigentümer der Datei, sondern auch beliebige andere Benutzer des Systems besitzen an ihr sowohl Lese- als auch Schreibrechte. Diese Schreibberechtigung ermöglicht es einem Angreifer, die Datei

`sgid-Recht`

`Sicherheitsproblem`

`Trojanisches Pferd`

beispieldat mit einer eigenen Datei zu überschreiben und diese dann unter Ausnutzung des ja nach wie vor gesetzten *suid*-Bits mit den Rechten des Benutzers Joe auszuführen. Unsachgemäß verwaltete und erstellte *suid*-Programme eröffnen Tür und Tor zum Einschleusen Trojanischer Pferde mit beliebigen Schadensfunktionen!

Root-Shell

Ein weiteres Beispiel zeigt, wie man auf einfache Weise eine Shell mit Root-Berechtigungen erlangen kann. Dazu benötigt man einen Zugriff auf einen Rechner, an dem ein Benutzer mit Root-Privilegien eingeloggt ist und der diesen Rechner ohne ihn zu sperren einige Zeit unbeaufsichtigt lässt. Nun kann ein Angreifer folgende einfache Schritte durchführen²:

```
cp /bin/sh /tmp/endlich_root
chmod 4755 /tmp/endlich_root
```

Auf diese Weise hat sich der Angreifer eine Shell mit gesetztem *suid*-Bit erzeugt, die ihm bei der Ausführung Root-Rechte verschafft. Natürlich würde ein Angreifer eine solche Shell nicht so offensichtlich benennen und ggf. auch in ein verstecktes Verzeichnis kopieren.

Suche nach *suid*

Um alle Programme und Dateien mit gesetztem *suid* bzw. *sgid*-Recht auf einem System zu identifizieren, kann man unter UNIX das Kommando *find* verwenden:

```
find / \(-perm -00400 -o -perm -00200 \) -type f -ls
```

Mit diesem Kommando wird der Dateibaum beginnend bei der Wurzel / durchsucht. Es wird nach *sgid*-Rechten (00400) und *suid*-Rechten (00200) gesucht, wobei sich die Suche auf Dateien beschränkt (-type f).

Wie das Beispiel des Erlangens einer Root-Shell lehrt, sollte ein Systemadministrator regelmäßig sein System nach Programmen mit gesetztem *suid*-Bit absuchen, um auffällige Programme zu erkennen.



Tools

Die Darstellung der Unix Rechtvergabe sollte zusammen mit den angegebenen Beispielen klar gemacht haben, dass sich durch eine unsachgemäße Rechtevergabe große Sicherheitslücken ergeben können. Im Unix-Umfeld existieren vielfältige, frei verfügbare Werkzeuge, mit deren Hilfe die Rechtevergabe analysiert werden kann und die Hinweise auf mögliche Sicherheitsprobleme liefern.

12.4.3 Zugriffskontrolle

Zur Erklärung der Realisierung der Unix Zugriffskontrolle sind zumindest rudimentäre Kenntnisse über das Unix Dateisystem erforderlich, so dass wir die notwendigen Grundlagen im Folgenden knapp einführen. Eine Unix

² Unter Linux-Konfigurationen können auch dash oder zsh als Shells verwendet werden.

Datei wird auf dem Hintergrundspeicher in (logischen) Blöcken realisiert und intern durch einen so genannten *inode* (Index node) von 64 Byte repräsentiert. Ein *inode* ist also ein Dateideskriptor. *Inodes* enthalten die Attribute der Datei (u.a. den Namen des Dateieigentümers, die Dateigröße, die ACL) sowie Informationen darüber, in welchen Hintergrundspeicherblöcken die Datei realisiert ist. Abbildung 12.8 zeigt einen Ausschnitt aus einem *inode*.

Die *inodes* werden auf dem Hintergrundspeicher in einer *inode*-Liste verwaltet und darin über ihren Identifikator, die *inode*-Number, identifiziert. Abbildung 12.7 zeigt das Layout einer Festplatte in traditionellen Unix Systemen. Der Bootblock enthält üblicherweise Code zum Booten des Sys-

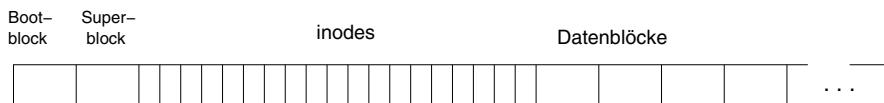


Abbildung 12.7: Layout einer Festplatte in Unix Systemen

tems und der Superblock enthält Informationen über das Plattenlayout, wie die Anzahl der *inodes*, die Anzahl der Plattenblöcke und die Anfangsadresse der Liste, die die freien Plattenblöcke verwaltet. Dem Superblock folgen die von 1 an aufwärts durchnummerierten *inodes*.

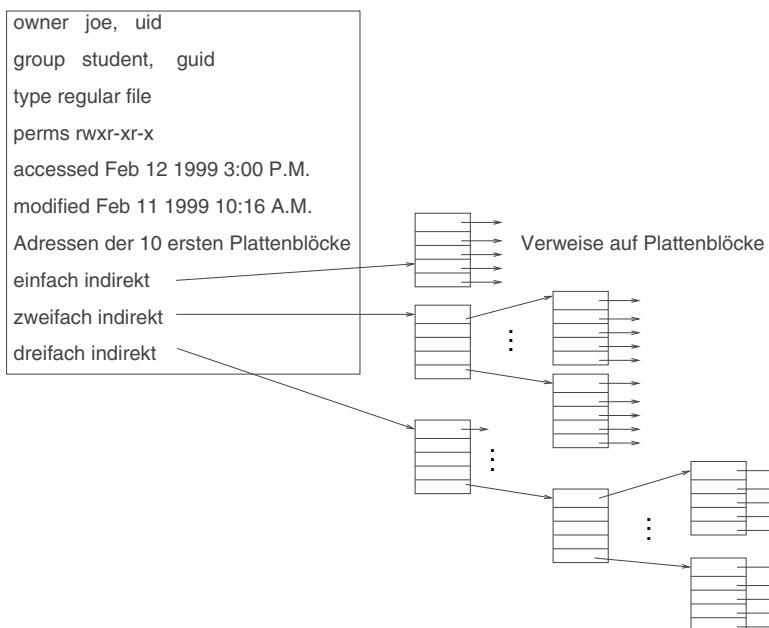


Abbildung 12.8: Unix *inode*

Zugriffskontrolle

Wenn ein Prozess auf eine Datei zugreifen will, muss er diese zunächst öffnen (open-System Call). Dabei sind als Parameter der Pfadname der Datei sowie die Operationen, die der Prozess ausführen möchte, anzugeben. Der Pfadname wird dann vom System auf den inode der gewünschten Datei abgebildet. Dafür ist es notwendig, jeden Dateinamen auf dem Pfad durchlaufen zu können, wozu der zugreifende Prozess, wie weiter oben bereits erwähnt, die x-Rechte benötigt. Der inode der gesuchten Datei wird bei einer erfolgreichen Suche in die inode Tabelle des Betriebssystemkerns (vgl. Abbildung 12.9) geladen, falls er nicht schon in den Arbeitsspeicher eingelagert ist. Der Kern überprüft anschließend anhand der in dem inode enthaltenen uid sowie der ACL, ob der Prozess zur Ausführung der gewünschten Operation berechtigt ist. Die benötigten Informationen über den Prozess, nämlich die uid seines assoziierten Benutzers sowie dessen guid, sind in der Prozessbeschreibung enthalten, auf die der Kern natürlich zugreifen darf.

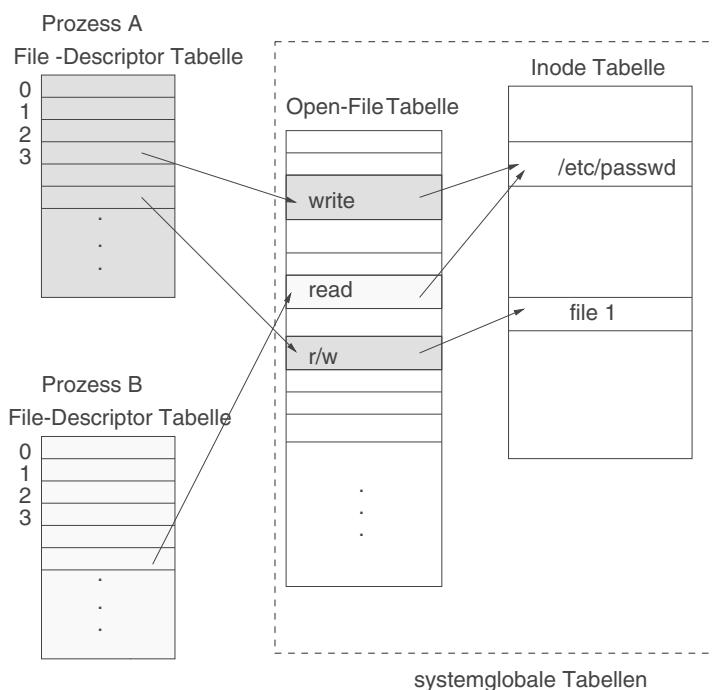


Abbildung 12.9: Zusammenhang zwischen den Datenstrukturen im Kern

File-Descriptor

Ist der Zugriff zulässig, so erzeugt der Kern für den Prozess einen so genannten File-Descriptor. Dieser Deskriptor wird in die prozesslokale File-Descriptor-Tabelle eingetragen. Diese Tabelle ist vergleichbar mit einer, dem

Prozess assoziierten Capabilityliste, da ein Prozess mittels seiner Deskriptoren auf die von ihm geöffneten Dateien zugreifen darf, ohne dass dabei eine erneute Überprüfung der ACL erfolgt. Die Aufgabe eines File-Descriptors bei der Durchführung der Zugriffskontrolle entspricht einem Object Handle von Windows.

Für jeden generierten File-Descriptor erzeugt der Kern einen Eintrag in der systemweiten Open-File-Tabelle, in der vermerkt wird, für welche Operationen die Datei von dem Prozess geöffnet worden ist. Jeder Eintrag in der File-Descriptor-Tabelle eines Prozesses verweist auf einen eindeutigen Eintrag in der Open-File-Tabelle des Kerns. Damit kann bei jedem Zugriff auf eine geöffnete Datei anhand des vom zugreifenden Prozesses vorzuweisenden File-Descriptors überprüft werden, ob der Zugriff gemäß der beim open-Aufruf angegebenen Zugriffsoperationen zulässig ist. Der Zusammenhang zwischen den angesprochenen Datenstrukturen ist in Abbildung 12.9 skizziert. Die Abbildung verdeutlicht, dass Prozesse unterschiedliche Zugriffsrechte auf eine geöffnete Datei, deren Repräsentant aber nur genau einmal im Hauptspeicher in der `inode` Tabelle vorhanden ist, besitzen können. Die differenzierten Zugriffsberechtigungen sind aus den Einträgen in der Open-File-Tabelle zu entnehmen. So hat in der in Abbildung 12.9 skizzierten Situation der Prozess B die Datei `/etc/passwd` nur zum Lesen, der Prozess A dagegen nur zum Schreiben geöffnet.

Zulässigkeit

Fazit

Unix realisiert ein einfaches Modell der benutzerbestimmten Zugriffskontrolle mit einer Kombination aus objektspezifischen Zugriffskontrolllisten und subjektspezifischen Zugriffsausweisen. Für Unix Systeme gilt, dass eine Rechterücknahme bezüglich einer Datei keine unmittelbare Auswirkung auf die Prozesse hat, die diese Datei in geöffnetem Zustand halten.

Kombinierte Mechanismen

Weiterhin haben wir gesehen, dass die verwendeten ACLs nur eine geringe Differenzierung bei der Rechtevergabe ermöglichen, so dass in der Regel zu viele Rechte vergeben werden. Besonders offensichtlich zeigt sich dies beim Superuserkonzept, da die Kennung, die die Superrechte besitzt, keinerlei Zugriffsbeschränkungen unterliegt. Mit dem `suid`-Konzept steht eine einfache Technik zur temporären Erweiterung der aktuellen Rechteumgebung eines Prozesses zur Verfügung, die jedoch bei einer unsachgemäßen Nutzung, jeder Benutzer darf das Konzept ja verwenden, erhebliche Sicherheitsprobleme hervorrufen kann.

Rechte

Eine wichtige Designentscheidung ist, dass die `inode`, die die sicherheitskritischen Berechtigungsinformationen enthalten, nur von Prozessen im Systemmodus modifizierbar sind. Dadurch wird ein Schutz vor deren unautorisierten Manipulationen durch Benutzerprozesse erzielt. Diese De-

inode-Speicherung

signentscheidung greift aber zusammen mit ihrer Implementierung zu kurz und der gewährleistete Schutz ist unzureichend. Wie wir in Abbildung 12.7 gesehen haben, liegen nämlich die `inode` im Klartext auf dem Hintergrundspeicher, so dass sie dort Angriffen ungehindert ausgesetzt sind.

Die unter Standard-Unix zur Zugriffskontrolle realisierten Maßnahmen sind pragmatisch und effizient; sie erfüllen jedoch keine hohen Sicherheitsanforderungen.

12.5 Systembestimmte Zugriffskontrolle

Mit systembestimmten Zugriffskontrollen (engl. *mandatory access Control*) lassen sich Regelwerke, die über die benutzerbestimmbare Berechtigungsvergabe hinausgehen, implementieren. Ein Beispiel ist die Kontrolle zulässiger Informationsflüsse. Mit dem Bell-LaPadula Modell haben wir in Kapitel 6.2.4 das in der Praxis am häufigsten eingesetzte Modell zur Festlegung solcher systembestimmten Zugriffsbeschränkungen kennen gelernt.

Sicherheitsklassen

Sicherheitsklassen

Charakteristisch für ein System mit einem mandatorischen Regelwerk ist, dass sowohl Objekten als auch Subjekten eine Sicherheitsklasse zugeordnet wird (engl. *labeling*). Die Klassifikation gibt an, welche Klassen von sensibler Information in dem Objekt gespeichert bzw. dem Subjekt zugänglich sein dürfen. Um Informationsflussbeschränkungen zu implementieren, werden die Sicherheitsklassen partiell geordnet und mittels systemglobaler Regeln wird der zulässige Informationsfluss zwischen den Klassen spezifiziert. Im Bell-LaPadula-Modell waren dies die no-read-up und no-write-down Regeln (siehe Seite 266), die festlegen, dass ein Informationsfluss höchstens von unten nach oben entlang der partiellen Ordnung der Klassen erfolgen darf.

Benutzerclearance

Für eine Implementierung sind Objekte und Subjekte mit ihren Sicherheitsklassifikationen zu versehen und die Zugriffskontrolle ist so zu erweitern, dass sie die Einhaltung der Flussregeln garantiert. Die Sicherheitsklassifikation eines Subjekts (engl. *clearance*) ist als Erweiterung der Subjektbeschreibung festzuhalten. Das heißt, dass beim Einrichten einer Benutzerkennung eine maximale Clearance zu vergeben ist und diese in der Benutzerbeschreibung, zum Beispiel in der `/etc/passwd` unter Unix oder im Access Token unter Windows, registriert werden muss. Die Clearance darf nur von einer vertrauenswürdigen Instanz, das ist in der Regel der Systemadministrator, vergeben werden und die gespeicherten Labels sind zu schützen, damit ein Benutzer seine Clearance nicht unautorisiert erhöhen kann.

Login

Falls das System das Agieren unter einer geringeren Clearance als der maximalen erlaubt, so muss auch der Systemdienst zur Durchführung der Zu-

gangskontrolle erweitert werden, damit ein Benutzer eine Sicherheitsklassifikation wählen und zusammen mit seiner Kennung beim Login angeben kann. Die Zugangskontrolle hat in diesem Fall zu prüfen, ob die gewählte Clearance für den Benutzer zulässig ist, also nicht dessen maximale Clearance übersteigt. Systeme, die eine mandatorische Strategie realisieren, sorgen meist zusätzlich noch dafür, dass der Systemzugang über einen vertrauenswürdigen Pfad (engl. *trusted path*) erfolgt, indem zum Login eine spezielle Tastenkombination (z.B. CTRL-ALT-DEL unter Windows) zu betätigen oder eine spezielle Kontrollsequenz zu senden ist. Der vertrauenswürdige Pfad stellt die Authentizität des Login-Programms sicher und gewährleistet, dass nicht bereits beim Systemzugang unzulässige Informationsflüsse auftreten.

Nach einer erfolgreichen Authentifikation assoziiert das mandatorische System jedem Prozess, den der Benutzer im Verlauf seiner Login-Sitzung erzeugt, die aktuelle Benutzerclearance. Damit erbt ein im Benutzerauftrag tätiger Prozess oder auch Thread die Sicherheitseinstufung seines Benutzers. Die Prozessclearance wird in der Regel in der internen Beschreibung von Prozessen, dem Prozess-Deskriptor bzw. dem Prozesskontrollblock (z.B. die `uarea` und `proc_struct` von Unix-Prozessen), festgehalten. Da der Schutz dieser Klassifikationsinformation für die Sicherheit des Systems von großer Bedeutung ist, dürfen nur speziell dazu berechtigte Prozesse im privilegierten Systemmodus modifizierend auf einen Prozessdeskriptor zugreifen oder einen solchen erzeugen. Die Prozessdeskriptoren sind wie die Dateideeskriptoren Datenstrukturen des Betriebssystemkerns.

Prozessclearance

Zugriffskontrolle

Die Sicherheitsklassifikation eines Objekts wird als zusätzliches Attribut in der Objektbeschreibung (z.B. der `inode` in Unix) festgehalten. Bei einem Objektzugriff hat der zuständige Objektmonitor die Aufgabe, die systembestimmten Regeln zu überprüfen. Das bedeutet, dass diese in ausführbaren Kontrollcode zu transformieren und alle Dienste um diesen Code zu erweitern sind. Seien beispielsweise die Regeln des Bell-LaPadula-Modells festgelegt. Dann ist für einen Lesezugriff auf eine Datei, `read(file)`, zu prüfen, ob die Clearance des zugreifenden Prozesses größer ist als die Klassifikation der Datei `file`. Der erforderliche Kontrollcode besteht somit aus dieser zu prüfenden Bedingung. Die Erweiterung von Systemaufrufen um einen solchen Kontrollcode für die Bell-LaPadula-Regeln haben wir bereits in Tabelle 6.3 für Unix MLS kennen gelernt.

Mandatorische Kontrolle

Falls die Objektverwaltung eine Informationsflussverletzung aufdeckt, ist der Zugriff auf das Objekt zu verweigern und eine geeignete Fehlermeldung zu generieren. Da Fehlermeldungen ihrerseits Informationen über Systemobjekte beinhalten, sind sie in mandatorischen Systemen mit großer Vorsicht zu wählen. So darf es nicht möglich sein, dass ein Benutzer über eine Feh-

Fehlermeldung

Iermeldung unautorisiert Kenntnisse über das Vorhandensein bzw. das nicht Vorhandensein oder über die Klassifikation von Objekten erlangt.

SE-Linux

Ein bekanntes Beispiel eines Betriebssystems, das neben rollenbasierter Zugriffskontrolle und dem Domain-Type-Enforcement auch eine Multi-level, mandatorische Zugriffskontrolle, die die Strategie des Bell-LaPadula-Modells realisiert, implementiert, ist das Open-Source SE-Linux³ System.

12.6 Service-orientierte Architektur

Unter dem Stichwort SOA (Service-orientierte Architektur) versteht man ein Architektur-Modell für Software, bei dem die Funktionen der Software als lose-gekoppelte, unabhängige Dienste (engl. *service*) definiert werden. Die Dienste verfügen über wohldefinierte Schnittstellen, über die sie aufgerufen werden können. Services sind Plattform- und Sprachen-neutrale, wieder verwendbare Einheiten, die zu komplexen Anwendungen bzw. Geschäftsprozessen komponiert werden können.

12.6.1 Konzepte und Sicherheitsanforderungen

Paradigma

Die Konzepte und Paradigmen von SOA haben zum Ziel, organisatorische Prozesse zu flexibilisieren und die IT so zu gestalten, dass sie an den unternehmerischen Geschäftsabläufen orientiert ist. Diese Top-down orientierte Vorgehensweise steht im Gegensatz zu der herkömmlichen IT-getriebenen, bottom-up orientierten Prozessgestaltung in Unternehmen. Durch den Top-down Ansatz wird eine Trennung zwischen der Business-Logik der Prozesse und der zugrundeliegenden technischen Infrastruktur ermöglicht. Aber auch die Nutzungsschnittstelle wird von der Business-Logik der Services getrennt, so dass SOA eine einfache Anpassung von Nutzungs-Views (z.B. professioneller Nutzer, Gelegenheitsnutzer) an verschiedene Bedürfnisse und Anforderungen von Benutzern ermöglicht. Flexibilisierung von Prozessen, einfache Anpassbarkeit, Wiederverwendung von Diensten sowie eine domänenübergreifende Zusammenarbeit zwischen Service-Providern und Service-Konsumenten sind wichtige Merkmale von SOA. Diese Merkmale werfen aber gleichzeitig eine Vielzahl von bekannten sowie neuen Sicherheitsfragen auf.

Services

Service

Der zentrale Baustein einer SOA ist der Service. Services realisieren betriebswirtschaftlich sinnvolle Funktionen in einer technologie-, plattform- und sprachenneutralen Weise. Services sind wieder verwendbare Einhei-

³ <http://www.nsa.gov/research/selinux/docs.shtml>

ten, die über eine wohldefinierte Schnittstelle genutzt werden können. Sie sind lose gekoppelte Softwarekomponenten, die über diese wohldefinierten Service-Schnittstellen miteinander kommunizieren. Services lassen sich flexibel zu komplexeren Einheiten komponieren bzw. auch schnell an geänderte Anforderungen anpassen. Wird die Ausführung komponierter Services durch eine zentrale Einheit (eine Engine) gesteuert und kontrolliert, so spricht man von der Orchestrierung von Diensten.

Services in einer SO-Architektur können von unterschiedlichen Anbietern (Providern), wie zum Beispiel einem Zulieferer, einem Kunden, einem Geschäftspartner definiert, bereit gestellt und verwaltet werden. Falls Service-Provider unterschiedlichen administrativen Domänen angehören, so können komponierte Services unterschiedlichen Sicherheitsrichtlinien (Policies) unterliegen. Es besteht dann die Aufgabe, diese Policies aufeinander abzustimmen. Durch die potentielle Dezentralität der Service-Anbieter ergeben sich dezentrale Verantwortlichkeiten für die Erbringung von Services. Eine vertragliche Absicherung, dass ein Service entsprechend seiner spezifizierten Schnittstelle auch korrekt erbracht wird, erfolgt in der Regel über einen Servicevertrag, der zum Beispiel in Form eines Service Level Agreements (SLA) ausgeprägt sein kann. Services werden durch Consumer genutzt. Damit ein Konsument einen Service mit den Fähigkeiten, wie er sie benötigt, finden kann, werden Services mit ihren Schnittstellenbeschreibungen in einem Repository inventarisiert. Eine Service-orientierte Architektur bietet Funktionen, so dass gezielt nach Diensten gesucht und Dienste in das Repository eingestellt werden können. Auf Web-Services, die eine spezifische Technologie zur Umsetzung des SOA-Paradigmas darstellen, gehen wir weiter unten noch etwas genauer ein.

Policies

Sicherheitsanforderungen

In offenen SO-Architekturen stellen eine Vielzahl von Anbietern aus unterschiedlichen administrativen Domänen ihre Dienste bereit. Dienste werden flexibel kombiniert und orchestriert, ändern sich unter Umständen dynamisch und werden von unterschiedlichen Nutzern, mit unterschiedlichen Sicherheitsanforderungen genutzt. Darüber hinaus können Dienste über unterschiedliche technologische Plattformen angeboten werden. Aus diesen Charakteristika ergeben sich spezifische, zum Teil neue Anforderungen an die Sicherheit von SO-Architekturen. Durch den Einsatz von SO-Architekturen werden Unternehmensdaten, Geschäftsanwendungen und ganze Geschäftsprozesse über offene Netze nutzbar gemacht. Auch für SO-Anwendungen gelten die klassischen Schutzziele, wie Vertraulichkeit, Integrität, Authentizität und Nicht-Zurückweisbarkeit. Aufgrund der SOA-Charakteristika (Dynamik, Heterogenität, Offenheit, wechselnde Partner) ergeben sich jedoch besondere Anforderungen in Bezug auf deren Umsetzung.

Anforderungen

Eine autorisierte Nutzung der Dienste ist domänenübergreifend zu gewährleisten und gleichzeitig sind unautorisierte Zugriffe domänenübergreifend zu verhindern. Dies erfordert eine verteilte Authentifizierung sowie ein Identitäts- und Rechtemanagement, das über die jeweiligen administrativen Domänen hinaus wirksam ist. Anbieter und Nutzer besitzen unterschiedliche Sicherheitsbedürfnisse, die über geeignete Sicherheitsregelwerke zu formulieren und die bei einer Dienstenutzung in Einklang zu bringen sind.

Anwendungsprobleme werden angepasst an die individuellen Anforderungen von Kunden gelöst, indem Dienste von unterschiedlichen Anbietern flexibel zu komplexen Diensten komponiert werden. Es ist sicherzustellen, dass auch in dem zusammengesetzten Dienst ein gefordertes Sicherheitsniveau durchgehend von allen beteiligten Diensteanbietern gewährleistet wird, und nicht beispielsweise sensitive Informationen unberechtigt weiter gegeben werden.

SO-Architekturen sind offene Architekturen, deren Services sich dynamisch ändern und in denen auch die Service-Anbieter einer hohen Dynamik unterliegen können (wechselnde Partner). Trotz dieser hohen Dynamik muss gewährleistet werden, dass die Sicherheitsvorgaben geltender Gesetze, Standards und Richtlinien (u.a. SOX⁴, Basel II, KontraG, SLAs, Unternehmensrichtlinien) verlässlich und nachprüfbar erfüllt werden.

Service-orientierte Architekturen werden durch sehr unterschiedliche technologische Plattformen (u.a. Netweaver, WebShere, .NET) realisiert. Eine durchgehende, plattformübergreifende Sicherheit von Diensten erfordert damit Maßnahmen zur sicheren Interoperation zwischen diesen Plattformen.

Eine spezielle Anforderung ergibt sich, falls über eine SOA-Plattform personalisierte Dienste angeboten werden. Zu den bereits genannten Schutzzieilen kommt dann noch die Forderung nach Wahrung der Privatsphäre hinzu, da personalisierte Dienste mit Informationen über spezifische Vorlieben, Orts- und Kontextinformationen eine Profilbildung von Service-Nutzern ermöglichen. Dies mag für dazu berechtigte Instanzen zulässig sein, muss aber für unautorisierte Dritte verhindert werden.

Umsetzung

Umsetzung

SOA-Sicherheit kann durch Maßnahmen auf verschiedenen Ebenen erzielt werden. Services können individuell so gehärtet werden, dass sie die geforderten Schutzziele erfüllen. Dies erfordert eine Service-spezifische Umsetzung von Sicherheitsvorgaben, was aufwändig ist, dafür aber eine auf die Bedürfnisse des Services zugeschnittene Lösung ermöglicht. Auf einer tieferen Ebene einer SOA-Implementierung können Basis-Sicherheitsmechanismen

⁴ Sarbanes-Oxley Act

in Form von Bibliotheken bereit gestellt werden, um generische Konzepte als wieder verwendbare Einheiten anzubieten. Verallgemeinert man diesen Ansatz, so kann man gemäß dem Service-orientierten Paradigma auch Sicherheit als Service spezifizieren. Ein Beispiel für einen solchen Sicherheits-Service ist ein Authentifizierungs-Dienst, der insbesondere Authentifizierungskonzepte mit unterschiedlichen Sicherheitsniveaus aufeinander abbildet und einen Niveau-Ausgleich bewerkstellt, wie z.B. die Abbilden eines Passwort-basierten Zugangs auf einen Zertifikat-basierten Zugang bei einem domänenübergreifenden Dienst. Ein weiteres Beispiel wäre die Umsetzung des Prinzips des Separation of Duty in einem Service.

12.6.2 Web-Services

Die Web-Services Standards der OASIS⁵ und des World Wide Web Consortiums (W3C)⁶ zählen zu den heute am weitesten verbreiteten Technologien zur technischen Umsetzung des SOA-Paradigmas. In einer Web-Service Architektur bietet eine Service Provider Dienste (Services) an, die von einem Service-Konsumenten genutzt werden können. Zum Austausch von Daten verwenden Dienste-Anbieter und Nutzer das standardisierte Datenaustauschformat XML (eXtensible Markup Language) und das SOAP-Protokoll. XML ist eine Auszeichnungssprache, d.h. es werden so genannte Tags verwendet, um ein XML-Dokument baumartig zu strukturieren. XML Tags sind frei wählbar und können mit der Definition von XML Schemata typisiert werden.

Web-Services

SOAP-Protokoll

SOAP⁷ ist ein einfaches Protokoll zum Austausch von strukturierten, getypten Informationen in einer verteilten Umgebung (vgl. <http://www.w3.org/TR>). SOAP kann auf einem beliebigen Übertragungsprotokoll auf unterschiedlichen OSI Schichten (u.a. HTTP, SMTP, TCP) aufsetzen. Im Zusammenhang mit Web-Services setzt SOAP standardmäßig auf HTTP auf. Auf diese Weise werden SOAP Nachrichten in der Regel zugestellt, da der http-Port 80 von den meisten Firewalls nicht blockiert wird. SOAP-Nachrichten sind als XML-Nachrichten definiert und bestehen aus drei Teilen: (1) dem SOAP-Envelope, der die gesamte Nachricht umschließt, dem (2) SOAP-Header (optional), über den eine SOAP Nachricht um weitere Funktionalität erweitert werden kann, wie z.B. um ein actor-Attribut, das Absender und URI (Ziel) der Nachricht definiert. Der (3) SOAP-Body enthält Informationen für den Empfänger, wie das aufzurufende Objekt mit aufzurufender Methode.

SOAP

⁵ Organization for the Advancement of Structured Information Standards

⁶ vgl. <http://docs.oasis-open.org/wss/v1.1/wss-v1.1-spec-os-SOAPMessageSecurity.pdf>

⁷ Früher war SOAP die Abkürzung von Simple Object Access Protocol

WSDL und UDDI

WSDL
Ein Service-Anbieter definiert die Service-Schnittstelle mittels der Web Service Description Language (WSDL) (vgl. <http://www.w3.org/TR/>). Eine WSDL-Spezifikation ist ein XML-Dokument, das die URL des Services spezifiziert und die Operationen/Methoden, die der Service anbietet, beschreibt.

UDDI
Ein Dienste-Anbieter veröffentlicht seine Dienstbeschreibung in der UDDI (Universal Description, Discovery, and Integration) Registry⁸. Dienstnutzer können in der UDDI Registry nach passenden Diensten für ihre Anforderungen suchen. Hat der Nutzer einen Dienst mit passenden Attributen gefunden, folgt er einer URL, die er von der Registry erhält und fordert die genaue Dienstbeschreibung vom Dienstanbieter an. Im nächsten Schritt nutzt er den Dienst des Anbieters wie er in der Dienstbeschreibung spezifiziert ist. Abbildung 12.10 veranschaulicht das klassische Web-Service Kommunikationsdreieck.

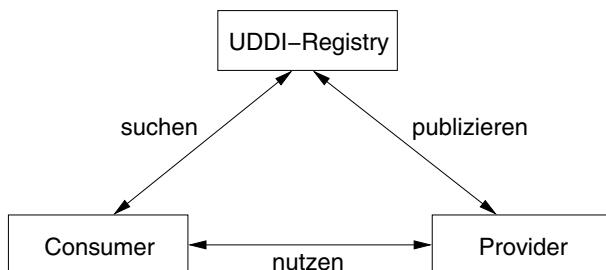


Abbildung 12.10: Web-Service Kommunikationsdreieck

Sicherheitsprobleme beim Einsatz von Web-Services

Sicherheitsprobleme
Web-Services sind einer Vielzahl von Sicherheitsbedrohungen ausgesetzt. Das SOAP-Protokoll bietet keine Sicherheitsmaßnahmen, so dass alle übertragenen Daten den üblichen Bedrohungen bei der Kommunikation ausgesetzt sind. Beispiel für solche übertragenen Daten sind die Schnittstellenbeschreibungen mittels WSDL-Dokumenten, oder aber auch Aufruf-Nachrichten, die Eingabeparameter beinhalten können, wie beispielsweise Kennungen oder auch sensitive Nutzerdaten.

Die Kommunikation zwischen Dienstanbieter und Dienstnutzer erfolgt zudem ohne Authentifizierung und es sind auch keine Maßnahmen vorgesehen, um die Integrität und Verfügbarkeit der angebotenen Web-Services zu gewährleisten. Da Web-Services auf den gleichen Technologien beruhen, die auch bei der Programmierung von Web-Applikationen verwendet werden, sind Web-Services allen aus diesem Umfeld bekannten Bedrohungen aus-

⁸ vgl. https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=uddi-spec

gesetzt. Beispiele sind Buffer-Overflow-Angriffe, SQL-Injection, Replay- oder auch DoS-Angriffe. Daneben gibt es jedoch einige Angriffe, die Web-Service-spezifisch sind.

Web-Service-spezifische Angriffe

Ausgangspunkt für WSDL und Access Scanning-Angriffe sind WSDL-Dokumente, die u.a. beschreiben, welche Operationen von einem Web Service bereitgestellt werden. Diese Schnittstellenspezifikation wird in der Regel automatisch generiert, wodurch automatisch Spezifikationen für alle Operationen, auch für die nicht-öffentliche zugreifbaren Operationen generiert werden. Dadurch, dass auch für diese eine Spezifikation erzeugt wird, sind sie im Prinzip ebenfalls über den Web-Service zugreifbar. Dies gilt auch für den Fall, dass die URIs dieser Operationen aus der WSDL-Datei entfernt wurde. Ein Angreifer kann auf der Basis von öffentlich verfügbaren Informationen gezielte Rate-Versuche hinsichtlich nicht-öffentlicher Operationen vornehmen und versuchen, auf diese zuzugreifen. Zum Beispiel ist zu vermuten, dass es zu einer öffentlich verfügbaren Operation namens *getItem* eine nicht-öffentliche zugängliche Operation namens *setItem* gibt. Ein Angreifer kann sein Glück versuchen und im Erfolgsfall auf diesem Wege unter Umgehung von Zugriffskontrollen auf nicht-öffentliche Operationen zugreifen. Da es sich hierbei in der Regel um sensitive Operationen handelt, ist mit solchen Angriffen potentiell ein großes Schadenspotential verbunden.

WSDL

Die so genannten *External Entity Attacks* richten sich gegen die Integrität der URIs, die in XML-Dokumenten enthalten sind. Sowohl SOAP Nachrichten, WSDL Beschreibungen, als auch UDDI Einträge sind als XML-Dokumente spezifiziert. Ein möglicher Angriff besteht darin, eine nicht schreibgeschützte URI in einem XML-Dokument so zu manipulieren, dass sie auf ein manipuliertes oder vertrauliches Dokument verweist, das auf diese Weise anderen Nutzern zugänglich gemacht wird.

XML-Attacke

Eine weitere kritische Schwachstelle ist die *XPath Injection*. Ausgangspunkt für diese Angriffe ist die Sprache XML Path Language (XPath). Diese wird verwendet, um schnelle Abfragen über XML-Dokumente zu ermöglichen, ohne dass das XML-Dokument vollständig geparsert werden muss. XPath-Abfragen werden häufig aus Eingabeparametern generiert, wobei mal wieder das Problem besteht, dass die Parameterwerte ungeprüft in die XPath-Abfragen übernommen und zur Ausführung gebracht werden. Ein Angreifer kann somit über manipulierte Parameter präparierten XPath-Code in die Abfragen an das XML-Dokument einschleusen.

XPath-Attacke

12.6.3 Web-Service Sicherheitsstandards

Die Web-Service Sicherheitsstandards (WSS) zählen zu den wichtigsten Grundlagen zur Durchsetzung von Sicherheitsanforderungen in Web-

Standards

Service-basierten Architekturen. Ergänzend dazu stehen mit WS-Policy, WS-Trust, WS-Privacy, WS-SecureConversation, WS-Federation und WS-Authorization eine Reihe von Standards und Drafts zur Verfügung, um u.a. Vertrauensbeziehung zu etablieren oder eine Authentifizierung der Parteien zu gewährleisten. Abbildung 12.11 gibt einen Überblick über die relevanten Web-Service-Sicherheitsstandards.

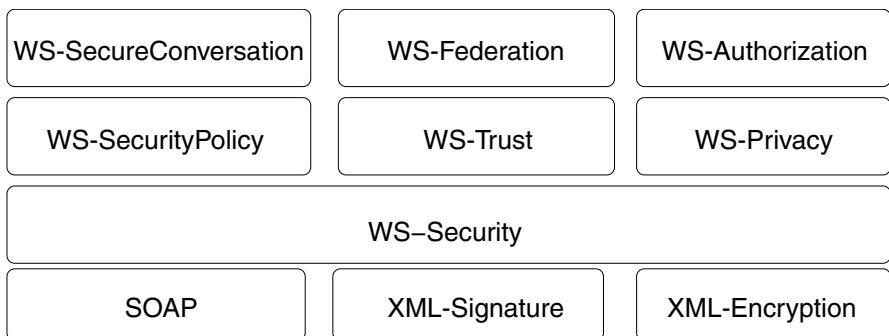


Abbildung 12.11: Web-Service Sicherheitsstandards

Zur Gewährleistung der Kommunikationssicherheit der mittels SOAP übertragenen XML-Nachrichten verwendet man üblicherweise SSL/TLS; hierdurch wird jedoch nur ein Teil der Sicherheitsprobleme von SOAP behoben. SSL/TLS authentifiziert die Endpunkte, jedoch nicht die einzelnen Dienste. Darüber hinaus kann mittels SSL/TLS bei orchestrierten Services keine Ende-zu-Ende Sicherheit gewährleisten werden. Durch die Orchestrierung von Diensten kann ein Dienstaufruf viele Zwischenstationen durchlaufen bis er beim tatsächlichen Diensterbringer ankommt, so dass an den jeweiligen Zwischenstationen die Daten ungeschützt vorliegen. Vor diesem Hintergrund wurden eine Reihe von Industriestandards entwickelt, um die Sicherheit von Web-Services zu erhöhen.

SAML

Die Web-Service- Sicherheitsstandards werden ergänzt durch den SAML-Standard (Security Assertion and Markup Language), der ebenfalls von der OASIS entwickelt wurde. SAML spezifiziert ein allgemeines XML-basiertes Framework zum Austausch von Sicherheitsinformationen zwischen Parteien von Online-Geschäftstransaktionen. Die Kernkonzepte von SAML sind die so genannten SAML-Assertions, also eine Art Bescheinigung, die zwischen den Parteien ausgetauscht werden. Derartige Assertions sind XML-Datenstrukturen, die nach einem festgelegten Schema aufgebaut sind und Aussagen über den (geprüften) Sicherheitsstatus eines Online-Partners beinhalten. Assertions sind damit in gewisser Weise vergleichbar mit Authentisierungs-Credentials wie Kerberos Tickets. Auch Assertions

werden von speziellen, vertrauenswürdigen dritten Instanzen erstellt und können dezentral von Service-Anbietern geprüft werden. Analog zu Kerberos verfolgt auch SAML das Ziel, ein Single-Sign-On in einer verteilten, hier Service-basierten, Umgebung aufzusetzen. Der SAML-Standard spezifiziert die Syntax derartiger Assertions sowie Regeln, auf welche Weise Assertions angefragt, erstellt, verteilt und genutzt werden können. Auf SAML gehen wir in Abschnitt 12.6.4 noch etwas genauer ein. Für das Zusammenwirken mit den Web-Service Standards wurde von der OASIS ein spezielles Profil entwickelt⁹, das festlegt, wie die SAML-Konzepte mit den Web-Service Konzepten zu nutzen sind.

WS-Sicherheit

Der Web Service Security (WSS) Standard der OASIS erweitert die SOAP Syntax, um einen sicheren Austausch von SOAP Nachrichten zu ermöglichen. Dazu definiert dieser Standard SOAP Elemente, Security Token und Mechanismen zu deren Verarbeitung, die Vertraulichkeit, Authentizität und Integrität auf Nachrichtenebene gewährleisten. WSS verwendet dazu die Verfahren zum Signieren und Verschlüsseln aus den beiden W3C Standards XML Digital Signature und XML Encryption¹⁰. WSS spezifiziert, wie das Signieren und Verschlüsseln von XML-Dokumenten auf die XML Struktur von SOAP Nachrichten angewendet werden müssen. Signier- und Verschlüsselungsoperationen können auf ganze XML-Dokumente oder Teile angewandt werden.

XML-Sicherheit

Beispiel 12.7 (Signierte und verschlüsselte SOAP-Nachricht)

Das nachfolgende XML Listing zeigt eine SOAP Nachricht mit verschlüsseltem und signiertem Inhalt, die Vertraulichkeit und Authentizität zusichert. Das Beispiel ist stark kondensiert, enthält aber die wesentlichen Elemente einer signierten und verschlüsselten SOAP Nachricht.

Beispiel

```
<?xml version=1.0 ?>
<S11:Envelope ...>
  <S11:Header>
    <wsse:Security>
      <wsu:Timestamp wsu:Id=T0 .../>
      <wsse:BinarySecurityToken
        ValueType=...X509v3
        wsu:Id=MyX509Token ...>
    </wsse:BinarySecurityToken>
```

⁹ <http://docs.oasis-open.org/wss/v1.1/wss-v1.1-spec-os-SAMLTokenProfile.pdf>

¹⁰ Vgl. <http://www.w3.org/TR/xmldsig-core/> und <http://www.w3.org/TR/xmlenc-core/>

```

<xenc:EncryptedKey>
  <ds:KeyInfo>
    <wsse:SecurityTokenReference>
      <wsse:KeyIdentifier ... />
    </wsse:SecurityTokenReference>
  </ds:KeyInfo>
  <xenc:CipherData ... />
  <xenc:ReferenceList>
    <xenc:DataReference
      URI=#enc1/>
  </xenc:ReferenceList>
</xenc:EncryptedKey>
<ds:Signature>
  <ds:SignedInfo>
    <ds:Reference URI=#T0 .../>
    <ds:Reference URI=#body ... />
  </ds:SignedInfo>
  <ds:SignatureValue ... />
  <ds:KeyInfo>
    <wsse:SecurityTokenReference>
      <wsse:Reference
        URI=#MyX509Token/>
    </wsse:SecurityTokenReference>
  </ds:KeyInfo>
</ds:Signature>
</wsse:Security>
</S11:Header>
<S11:Body wsu:Id=body >
  <xenc:EncryptedData
    wsu:Id=enc1>
    <xenc:CipherData> ...
    </xenc:CipherData>
  </xenc:EncryptedData>
</S11:Body>
</S11:Envelope>

```

Die XML Namespaces der SOAP Elemente sind der Einfachheit halber weggelassen. Die Namespace Identifier *wsse* und *wsu* beziehen sich auf den WSS Standard, *xenc* und *ds* beziehen sich auf XML Encryption und XML Signature und *S11* referenziert SOAP in der Version 1.1.

Das *wsse:Security* Element im Header beinhaltet die relevanten Information, damit der Empfänger die Nachricht entschlüsseln und die Signatur verifizieren kann. Das *BinarySecurityToken* enthält das Zertifikat des Senders, mit dessen privatem Schlüssel die Nachricht signiert ist. Das *EncryptedKey* Element speichert den symmetrischen Schlüssel, mit dem der Body der

SOAP Nachricht verschlüsselt ist. Dieser Schlüssel ist mit einem öffentlichen Schlüssel aus einem X.509 Zertifikat des Empfängers verschlüsselt, welcher über eine *SecurityTokenReference* angesprochen wird. Der eigentliche Nachrichtenschlüssel wird verschlüsselt in *CipherData* übermittelt. Das *EncryptedKey* Element enthält zudem Referenzen auf die mit diesem Schlüssel verschlüsselten Nachrichtenelemente.

Das letzte Header Element ist die Signatur der Nachricht. Das gleichnamige Element enthält eine Referenz auf das für die Signatur verwendete Security Token und eine Liste der signierten SOAP Felder. *SignedInfo* referenziert in diesem Beispiel den Body der Nachricht und einen Zeitstempel der Erstellung. Body und Zeitstempel werden gemeinsam signiert, um einen Replay Angriff zu verhindern. Würde man sie einzeln signieren, könnte ein Angreifer sowohl den Zeitstempel als auch die Signatur aus der Nachricht entfernen und die Nachricht beliebig oft wiederholen. Das Body Element der Nachricht enthält schließlich den verschlüsselten SOAP Request.

Die Web Service Nachricht ist für den Empfänger verifizierbar und entschlüsselbar, da sie alle benötigten Schlüssel und Zertifikate enthält.



WS-Secure Exchange

Das SOAP Protokoll ist zustandslos und implementiert einen unidirektionalen Nachrichtenaustausch. Ein unidirektionales Nachrichtenaustausch Schema ist nur sinnvoll einsetzbar, wenn zwischen Service Provider und Service Client nur wenige Nachrichten ausgetauscht werden müssen. Andernfalls wird der Overhead für das Erzeugen, das Übertragen und das asymmetrische Ver- und Entschlüsseln der Nachrichtenschlüssel unverhältnismäßig hoch.

Das Ziel von Web Services Secure Exchange (WS-SX) ist es, Standards für den Aufbau von Sicherheits-Kontexten, der WS-Secure-Conversation, für die Etablierung und Vermittlung von Vertrauensbeziehungen und für die Beschreibung von Security Policies für Web Service Endpunkte zu spezifizieren.

WS Trust

Im allgemeinen Fall besteht kein Vertrauensverhältnis zwischen Dienstnutzer und Diensteinbringer. Für diesen Fall wurde das Web Service Trust Modell definiert, das Protokolle zum Vertrauensaufbau zwischen den Service Partnern bereitstellt. Ein oder mehrere Security Token Services (STS) fungieren dabei als Trusted Third Party und stellen die benötigten Security Token für die Dienstnutzer aus, sofern diese die Policy des STS und des

WS Trust

Diensterbringern erfüllen. Sowohl der potentielle Dienstnutzer eines Web Services als auch der Web Service selbst können die Dienste eines Security Token Service beanspruchen. Beide Parteien nehmen dann gegenüber dem STS die Rolle des Requestors ein, wie diese in WS-Trust genannt wird. Die wichtigsten Typen von Anfragen an den STS sind die Ausgabe (Issuance), die Erneuerung (Renewal), der Widerruf (Cancel) und die Validierung (Validation) eines Security Token. Der STS selbst ist in WS-Trust auch als Web Service definiert.

Szenario

Ein mögliches Nutzungsszenario des WS Trust Modells könnte folgendermaßen aussehen. Ein Dienstnutzer besitzt für den Aufruf des gewünschten Services nicht die notwendigen Security Token. Welche Token er benötigt und woher er sie bekommen kann, erfährt der Nutzer aus der Policy des Web Services. Der Dienstnutzer sendet an den STS einen Security Token Request und erhält nach Vorlage der entsprechenden Credentials ein Security Token für den fraglichen Web Service. Mit diesem Security Token authentifiziert und autorisiert sich der Service Client gegenüber dem Web Service. Sollte der Web Service die Gültigkeit des vorgelegten Security Tokens nicht überprüfen können, so kann er diese Aufgabe an den ausstellenden oder einen anderen Security Token Service übertragen.

WS Secure Conversation

Secure Conversation

Während der WSS Standard auf die Vertraulichkeit und Authentizität einzelner SOAP Nachrichten abzielt, ermöglicht der WS Secure Conversation Standard (WS-SC) Sicherheitskontakte für den Austausch vieler Nachrichten zu etablieren. Dies Konzept ist vergleichbar mit den Security Associations, die man aus IPSec kennt. WS-SC führt dazu das Konzept des Security Context ein, das sich auf den Zustand eines Protokolls bezieht, in dem die Kommunikationspartner authentifiziert sind und über gemeinsame Session Schlüssel und eventuell weitere Security Attribute verfügen. Der Sicherheitskontext wird durch ein Security Context Token (SCT) repräsentiert.

Sicherheitskontext

WS-SC kennt drei Varianten, um einen Sicherheitskontext zu etablieren. So kann ein Security Token Service (STS) als Trusted Third Party fungieren und auf Anforderung ein Security Context Token für die Partner Web Services erstellen. Weiterhin kann einer der Partner selber ein solches SCT generieren, wenn der andere Partner ihm vertraut; in der dritten Variante handeln beide Partner das Token gemeinsam aus. Für die Anforderung eines SCT und dessen Weitergabe durch den Requestor setzt WS-SC auf den Protokollen und XML-Strukturen von WS Trust auf. Der Secure Conversation Standard erweitert die SecurityToken Struktur aus dem Web Services Security Standard und die WS-Trust Mechanismen durch zusätzliche Datenfelder wie einen eindeutigen Identifier für den Kontext und Mechanismen, um Schlüsselmaterial aus dem Security Context ableiten zu können.

WS Security Policy

Die WS-Policy¹¹ Spezifikation definiert ein allgemeines Framework, das es ermöglicht, die Eigenschaften, Anforderungen und Einschränkungen von Web Services zu spezifizieren. Diese werden als Policy Assertions entweder in die WSDL Beschreibung oder in den UDDI Eintrag des Web Services eingefügt. Der Standard für die Web Services Security Policy Languages (WS-SecurityPolicy) ergänzt das WS-Policy Framework um Assertion Typen, die vornehmlich Sicherheitseigenschaften aus den Standards WS Security, WS Secure Conversation und WS Trust referenzieren.

WS Policy

WS-Security Policy definiert fünf Klassen von Security Assertions, die verschiedene Aspekte der Kommunikation sicherer Web Services adressieren.

Assertions

1. Protection Assertions definieren, welche Teile einer SOAP Nachricht gesichert werden sollen.
2. Conditional Assertions bestimmen allgemeine Sicherheitsvoraussetzungen.
3. Security Binding Assertions beschreiben, welche Security Mechanismen verlangt werden.
4. Supporting Token Assertions geben an, welche Token Typen wie eingesetzt werden, um zusätzliche Sicherheitsbehauptungen, z.B. über Identität oder Rechte, zu spezifizieren.
5. Trust Assertions definieren, welche Formen von Vertrauenszusicherung verlangt werden.

Ein kleines textuelles Beispiel soll das Ineinandergreifen der verschiedenen Assertion Typen einerseits und das Zusammenspiel von WS-Policy mit WS-Trust und WS-SC andererseits darlegen.

Beispiel

Für einen Web Service S enthalte die WSDL Beschreibung die Protection Assertion ContentEncryptedElements mit einem XPath Attribut auf das SOAP Element, das in einem Request an S verschlüsselt übertragen werden muss. Diese Protection Assertion ist mit einer Binding Assertion vom Typ AlgorythmSuite und einer Token Assertion vom Typ SecurityContextToken verbunden. Das SecurityContextToken wiederum enthält Attribute, die dem Dienstnutzer mitteilen von welchem Security Token Service (IssuerName) er ein SCT mit welchen Inhalten (Claims) beziehen muss, um den Web Service S aufrufen zu können.

WS Security Policy ist damit eine der Kernspezifikationen, um die sichere Kooperation von Web Services dezentral zu organisieren.

¹¹ vgl. <http://www.w3.org/2002/ws/policy/>

12.6.4 SAML

SAML

SAML ist ein XML-basierter Standard zum Austausch von Authentifikations- und Autorisierungsinformationen. SAML (Security Assertion and Markup Language) wurde im Mai 2002 von der Standardisierungsorganisation OASIS verabschiedet¹². Sicherheitsinformationen werden in SAML in Form von Bescheinigungen, den so genannten *Assertions* ausgetauscht. SAML definiert Protokolle bestehend aus XML-basierten Anfrage- und Antwortnachrichtenformaten, so dass SAML-Bescheinigungen von entsprechenden SAML-Authorities anfordern können und als Antwort eine Bescheinigung erhalten. Die SAML-Protokolle sind generisch und können an unterschiedliche Kommunikationsprotokolle gebunden werden.

SAML unterscheidet drei Klassen von Bescheinigungen, nämlich die Authentisierungs-, die Autorisierungs- und die Attribut-Bescheinigung. Auf die drei Klassen gehen wir weiter unten noch etwas genauer ein. Eine SAML-Bescheinigung beinhaltet Aussagen über ein Subjekt, für das die Bescheinigung ausgestellt wurde, wie beispielsweise den Namen, die Art und Weise, wie es sich authentisiert hat, zu welcher Organisation es gehört, oder aber auch auf welche Ressourcen es zugreifen darf. Die ausstellende Instanz bescheinigt die Gültigkeit dieser Aussagen. eine Authentisierungs-Bescheinigung ist damit konzeptuell vergleichbar mit einem Kerberos-Ticket. Die Struktur einer Bescheinigung wird durch ein XML-Schema definiert.

An einer SAML-Interaktion sind mindestens zwei Parteien beteiligt. Das ist zum einen die so genannte Asserting-Party bzw. SAML Authority, die die Bescheinigungen erstellt. Sie ist vergleichbar mit einer CA in einer PKI. Zum anderen ist eine Relying Party beteiligt, die den ausgestellten Bescheinigungen vertraut. Zwischen den beiden Parteien müssen also Vertrauensbeziehungen existieren. SAML-Bescheinigungen werden für Subjekte ausgestellt, wobei ein Subjekt entweder ein Mensch oder aber auch ein Objekt, wie beispielsweise ein Service, ein Server oder eine Institution sein kann.

Beispieldaten

Das häufigste Einsatzszenario für SAML ist eine Multi-Domain Web-Umgebung¹³. Betrachten wir beispielsweise ein Szenario, in dem ein Flug inklusive Mietwagenanmietung online gebucht werden soll. In einem solchen Szenario unterhält der Benutzer eine Web-Session zu einer Web-Seite, wie beispielsweise *airline.com* und greift auf die Ressourcen dieser Seite zu. Mit einem Flug soll auch ein Mietwagen angemietet werden, so dass der Benutzer im Verlauf dieser Sitzung zur Web-Seite der Autovermietung *www.car.com* weiter geleitet wird. Für das Beispiel gehen wir davon aus,

¹² <http://www.oasis-open.org/committees/security/docs>

¹³ vgl. <http://docs.oasis-open.org/security/saml/Post2.0/ssotc-saml-tech-overview-2.0.pdf>

dass zwischen den beiden Web-Anbietern *airline.com* und *car.com* eine Geschäftsbeziehung besteht. Der Anbieter *www.car.com* stellt lediglich seine Autovermietungsdienste zur Verfügung, jedoch führt er keine Authentisierung der Nutzer durch. Hierfür verlässt er sich auf den Dienst seines Partner *airline.com*. Dieser agiert als Identitäts-Dienstleister, die die entsprechenden Identitätsprüfungen der Nutzer durchführt und korrekte Identitätsnachweise mit SAML-Assertions bescheinigt. Das heißt, der Dienst-Anbieter *car.com* vertraut den Assertions, die der Identitäts-Anbieter *airline.com* über Nutzer ausstellt. So könnte in diesem Szenario der Anbieter *airline.com* als SAML-Authority auftreten und eine Bescheinigung ausstellen, dass der aktive Nutzer sich ihm gegenüber identifiziert und authentisiert hat. Darüber hinaus könnte er noch bestätigen, dass der Nutzer über spezielle Identitätsattribute verfügt. Ein solches Attribut könnte den Nutzer beispielsweise als einen Premium-Kunden auszeichnen. Der Dienst-Anbieter *car.com* vertraut den Aussagen der Autorität *airline.com*, erzeugt eine lokale Sitzung für den Nutzer, ohne ihn seinerseits erneut zu authentisieren. Der Nutzer muss, entsprechend dem Singl-Sign-On Prinzip, also keine Authentisierungsinformationen mit dem Dienst-Anbieter *car.com* abstimmen. In dem skizzierten Szenario authentisiert sich der Nutzer zunächst bei seinem Identitäts-Provider bevor er versucht, die Ressourcen des Dienst-Anbieters zu nutzen.

Ein etwas anderes Szenario ergibt sich, wenn der Nutzer zunächst auf die Seiten eines Service-Anbieters navigiert und dort zunächst unkritische Ressourcen nutzt. Wenn der Nutzer im Laufe der Sitzung auf Ressourcen zugreifen möchte, die eine Authentisierung bzw. Autorisierung erfordern, kann der Dienst-Anbieter dem zuständigen Identitätsanbieter eine SAML-Authentisierungsanfrage senden, die dieser dann mit einer Authentisierungs-Bescheinigung beantwortet.

Assertions

Mit einer Authentifikations-Bestätigung wird festgehalten, dass das spezifizierte Subjekt mit dem angegebenen Authentifikationsverfahren und zu der angegebenen Zeit authentifiziert worden ist. Nachfolgend ist ein Beispiel einer Authentifizierungs-Assertion angegeben.

Authentifikation

```
<saml:Assertion  
MajorVersion= 1 MinorVersion= 0  
AssertionID=128.9.167.32.12345678  
Issuer=Smith Corporation  
IssueInstant=2001-12-03T10:02:00Z />  
<saml:Conditions  
NotBefore=2001-12-03T10:00:00Z  
NotAfter=2001-12-03T10:05:00Z />
```

```

<saml:AuthenticationStatement
    AuthenticationMethod=password
    AuthenticationInstant=2001-12-03T10:02:00Z>
    <saml:Subject>
        <saml:NameIdentifier
            SecurityDomain=smithco.com
            Name=joeuser />
    </saml:Subject>
</saml:AuthenticationStatement>
</saml:Assertion>

```

Das Beispiel ist eine Bestätigung, dass der Benutzer namens *joeuser* sich mittels eines Passwortverfahrens in dem Zeitraum zwischen 2001 – 12 – 03T10 : 00 : 00Z und 2001 – 12 – 03T10 : 05 : 00Z ausgewiesen hat.

Autorisierung

Eine Autorisierungsbestätigung spezifiziert, ob das Subjekt auf die angegebene Ressource zugreifen darf oder nicht.

```

<saml:Assertion ... />
<saml:Conditions ... />
<saml:AuthorizationStatement
    Decision=Permit
    Resource=http://jonesco.com/rpt_12345.htm>
    <saml:Subject>
        <saml:NameIdentifier
            SecurityDomain=smithco.com
            Name=joeuser />
    </saml:Subject>
</saml:AuthorizationStatement>
</saml:Assertion>

```

Die Autorisierungs-Bestätigung besagt, dass das Subjekt *joeuser* auf das Objekt *http://jonesco.com/rpt_12345.htm* unter der oben nicht näher spezifizierten Bedingung (Condition ...) zugreifen darf.

Attribut

Mit Attribut-Anweisungen können weitere Informationen über das spezialisierte Subjekt festgelegt werden. So kann beispielsweise mit einer Attribut-Assertion eine Aussage über die Zuordnung eines Benutzers zu einer bestimmten Abteilung des Unternehmens bestätigt werden. Das nachfolgende Beispiel gibt mit den entsprechenden Attributen Auskunft über Bezahlattribute eines bestimmten Subjekts, im Beispiel ist dies *http://smithco.com*.

```

<saml:AttributeStatement ... SAML:2.0:assertion">
    <saml:Attribute
        NameFormat="http://smithco.com"
        Name="PaidStatus">

```

```
<saml:AttributeValue>
    PaidUp
</saml:AttributeValue>
</saml:Attribute>
<saml:Attribute
    NameFormat="http://smithco.com"
    Name="CreditLimit">
    <saml:AttributeValue xsi:type="smithco:type">
        <smithco:amount currency="USD">
            500.00
        </smithco:amount>
    </saml:AttributeValue>
</saml:Attribute>
</saml:AttributeStatement>
```

Sicherheit

Das SAML-Framework spezifiziert Nachrichtenformate, Protokollschritte und Formate von Bescheinigungen, um Sicherheitsinformationen in einem Web-Umfeld zwischen unterschiedlichen Partnern auszutauschen. Zur Absicherung der Kommunikation und zum Aufbau von Vertrauensbeziehungen zwischen den beteiligten Parteien empfiehlt das Framework¹⁴ die Nutzung einer PKI sowie die Web-Service Sicherheitsstandards wie XML-Signature oder XML-Encryption, um digital signierte und bzw. oder verschlüsselte Bescheinigungen auszustellen. Der Transport der Bescheinigungen sollte mittels Standard-Verfahren wie SSL/TLS oder IPSec abgesichert.

Sicherheit

Problematisch ist, dass es keine enge Bindung zwischen dem Subjekt, für das die Bescheinigung ausgestellt wird, und der Bescheinigung selber besteht. Werden keine zusätzlichen Maßnahmen ergriffen, wie beispielsweise die bereits angesprochenen Sicherheitsprotokolle mit wechselseitiger Authentisierung der Kommunikationspartner, so besteht zusätzlich auch die Gefahr von Man-in-the-Middle Angriffen. Da Bescheinigungen nicht durch spezielle kryptografische Maßnahmen an Subjekte gebunden sind, kann ein Subjekt nicht nachweisen, dass die Bescheinigung aktuell ist und es der berechtigte Besitzer der Bescheinigung ist. Ein solches Bindungs- und Nachweiskonzept bietet demgegenüber Kerberos mit seinem Authenticator-Konzept. SAML-Bescheinigungen könnten somit abgefangen und mit Spoofing-Angriffen wieder eingespielt werden.

Das SAML-Framework unterstützt eine Attribut-basierte Zugriffskontrolle anstelle herkömmlicher Identitäts- basierter Kontrollen. Dadurch können detailliertere Zugriffskontrollregeln definiert werden, die einen Zugriff auf eine Ressource von spezifischen Attributen abhängig machen können; dies ist in

XACML

¹⁴ vgl. <http://docs.oasis-open.org/security/saml/v2.0/saml-sec-consider-2.0-os.pdf>

einem Web-Service Umfeld ein sehr geeignetes Konzept. SAML sieht aber keine Formate und Konzepte vor, um Sicherheits-Policies zu definieren. Für den Bereich der Zugriffskontrolle wird im Web-Service-Kontext sehr häufig auf die Sprachkonzepte von XACML (eXtensible Access Control Markup Language) zurückgegriffen. XACML ist eine XML-basierte Sprache¹⁵. Die Version 2.0 wurde 2005 und die Version 3.0, die u.a. eine Rechte delegation ermöglicht, wurde 2010 von der OASIS verabschiedet. Nachfolgend ist ein Beispiel für eine XACML-Policy angegeben. Das definierte Regelwerk erlaubt dem Subjekt *John* die Ausführung der Operation *open* (action) auf der Ressource *door*.

```

<Rule RuleId=? Effect="Permit">
<Description>Permit John to Open the Door</Description>
<Target>
  <Subjects><Subject>
    <SubjectMatch MatchId= ....
    <AttributeValue DataType= ... John </AttributeValue>
    <SubjectAttributeDesignator
      AttributeId= ...
      DataType= ...
      ...
    <ResourceMatch MatchId= ...
      <AttributeValue DataType= ... door </AttributeValue>
      <ResourceAttributeDesignator ...
        DataType="http://www.w3.org/2001/XMLSchema#anyURI"/>
      ...
    <ActionMatch MatchId= ...
      <AttributeValue DataType= ... open </AttributeValue>
      <ActionAttributeDesignator
        AttributeId= ...
        DataType="http://www.w3.org/2001/XMLSchema#string"/>
    </ActionMatch>
  </Action></Actions> </Target> </Rule>

```

Dies ist lediglich ein einfaches Beispiel. Zugriffsregeln können beispielsweise durch die Festlegung von Bedingungen noch weiter differenziert werden.

Fazit

Heute werden SOA-Anwendungen noch vordringlich in geschlossenen Umgebungen mit zentralen Kontrollinstanzen eingesetzt. Die Sicherheitsanforderungen von solchen geschlossenen Szenarien unterscheiden sich deutlich von offenen Szenarien, wo im Sinne eines Internets der Dienste, Dienste von beliebigen Anbietern bereitgestellt, zu neuen Diensten zusammengesetzt und genutzt werden..

¹⁵ vgl. <http://docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-cs-01-en.pdf>

Die bestehenden Standards sind stark Technologie-bezogen und lassen aufgrund unzureichender semantischer Spezifikation viel Implementierungsfreiraum, was wiederum zu Interoperabilitätsproblemen in der Praxis führt. Somit besteht ein Bedarf, Richtlinien für die sichere und effektive Nutzung bestehender Standards im Sinne von Best Practices zu entwickeln und Interoperabilitäts-Testbeds zu erarbeiten.

Interoperabilität

Das erforderliche domänenübergreifende Identitäts- und Rechtemanagement wird noch nicht zufriedenstellend unterstützt, auch wenn mit WS-Federation, SAML, WS-Trust, dem Security Token Service (STS) oder auch XACML bereits einige technische Grundlagen zur Verfügung stehen. Benötigt werden Lösungen, die analog zu den föderierten Identitäten im Kontext der Arbeiten der Liberty Allianz ein Single Sign On in dezentralen Web-Service Umgebungen mit wechselnden Partnern und wechselnden Diensteangeboten ermöglichen.

Identitätsmanagement

Auch bereits in geschlossenen Umgebungen muss SOA-Sicherheit die Bedürfnisse unterschiedlicher Beteiligter berücksichtigen, so dass der Frage der Policies und deren dezentralen Umsetzung in Zukunft eine entscheidende Rolle zukommen wird. Die WS-Policy-Standards und Beschreibungssprachen sind zu erweitern, damit auch nicht-technische Anforderungen, die sich durch Geschäftsprozesse ergeben, wie z.B. Zertifizierung für SOX-Compliance, oder aber SLAs und PLAs (Protection Level Agreements) berücksichtigt werden können. Um eine sichere Service-Komposition zu unterstützen, müssen Policy-Sprachen zukünftig in der Lage sein zu beschreiben, unter welchen Voraussetzungen ein Service mit anderen Services komponiert werden darf. Erforderlich sind u.a. Regelwerke, mit denen Kontexte und Vorbedingungen für eine zulässige Komposition oder aber auch Verpflichtungen spezifiziert werden können, die ein Service einzuhalten und deren Einhaltung über festzulegende Maßnahmen nachgeprüft werden kann.

Policy

Die sichere Komposition und Orchestrierung von Services gehört zu den zentralen, noch immer offenen Forschungsfragen. Für eine sichere Komposition von Services ist die Festlegung einer formalen Semantik der Sicherheitsspezifikationssprache erforderlich, so dass Kompositionskonzepte entwickelt werden können, die die Einhaltung von Sicherheitseigenschaften auch nach einer Service-Komposition garantieren. Für eine sichere Komposition sind zudem Konzepte erforderlich, damit ein Service-Anbieter den Nachweis erbringen kann, dass sein Service einen gewissen Sicherheitslevel einhält. Dies ist eine wichtige Voraussetzung im Zusammenhang mit dem Nachweis der Einhaltung von Compliance-Anforderungen.

Komposition

Die SOA-Aktivitäten finden eine konsequente Fortsetzung im Cloud-Computing. Cloud-Computing definiert ein flexibles IT-Bereitstellungsmo dell für Dienste und Ressourcen durch ein oder mehrere Anbieter. Es werden

Cloud

Daten und Anwendungen ausgelagert in die Cloud, die on-demand die benötigten Ressourcen, wie Speicher und Rechenfähigkeit zur Verfügung stellt. Das Cloud-Computing greift sehr stark auf Virtualisierungstechniken zurück, um eine isolierte Ausführung verschiedenere Anwendungen auf der gleichen Hardware-Plattform des Cloud-Anbieters zu gewährleisten. Die Sicherheit derartiger Virtualisierungsumgebungen (Hypervisor), die Überwachung der legitimen Zugriffe auf die Daten der Cloud-Anwender, das Verhindern unzulässiger Datenlecks, aber auch die Nachverfolgbarkeit, wo welche Daten verarbeitet werden, sind zentrale Fragestellung für die Cloud-Sicherheit.

13 Fallstudien: iOS-Ecosystem und Windows10

In diesem Kapitel werden exemplarisch in Abschnitt 13.1 das iOS-Ecosystems von Apple sowie in Abschnitt 13.2 die Sicherheitsarchitektur des Windows-Betriebssystems vorgestellt. Das iOS-Betriebssystem bildet den Kern des Apple-Ecosystems. Anders als beispielsweise Linux oder auch Windows verfolgt Apple konsequent ein Closed-World-Prinzip, das aus Sicherheitssicht einige Vorteile bietet, aber natürlich auch den Nachteil der großen Abhängigkeit vom Hersteller Apple nach sich zieht. Das Closed-World-Prinzip bedeutet, dass Apple vollständig kontrolliert, welche Software-Updates für den Betriebssystemkern oder auch welche Apps auf iOS-Geräte geladen werden dürfen. Auch ein sicheres Booten darf nur auf der Basis von Code erfolgen, der von Apple signiert wurde. Apps müssen eine Apple-interne Freigabeprüfung durchlaufen haben, bevor sie über den App-Store verfügbar gemacht werden können. Über das HomeKit-Framework unterstützt das Ecosystem auch die Vernetzung eines mobilen iOS-Geräts mit IoT-Geräte, wie einen Lichtsensor. Dafür muss das IoT-Gerät über ein spezielles MFI-Zertifikat von Apple verfügen und der Hersteller muss von Apple anerkannt sein, so dass eine Basis für Identitätskontrollen geschaffen ist.

13.1 iOS-Ecosystem

Apple hat für seine iOS-Plattform eine Sicherheitsarchitektur konzipiert und umgesetzt, die sowohl Hardware- als auch Softwaresicherheitskonzepte umfasst. Zu den technischen Grundlagen der iOS-Sicherheitsarchitektur gibt es nur wenige Veröffentlichungen. Die nachfolgenden Darstellungen basieren im Wesentlichen auf der Dokumentation von Apple in dem Whitepaper zur iOS-Sicherheit vom März 2017¹, sowie dem Vortrag von Ivan Krstic auf der Black Hat 2016 (Behind the Scenes of iOS Security).²

¹ Vgl. https://www.apple.com/business/docs/iOS_Security_Guide.pdf

² Anm. der Autorin: Da die öffentlich verfügbare Dokumentation an vielen Stellen einen Interpretationsspielraum lässt, ist es möglich, dass die nachfolgende Darstellung in imple-

13.1.1 iOS-Sicherheitsarchitektur im Überblick

Abbildung 13.1 gibt einen Überblick über die Sicherheitsarchitektur von iOS.

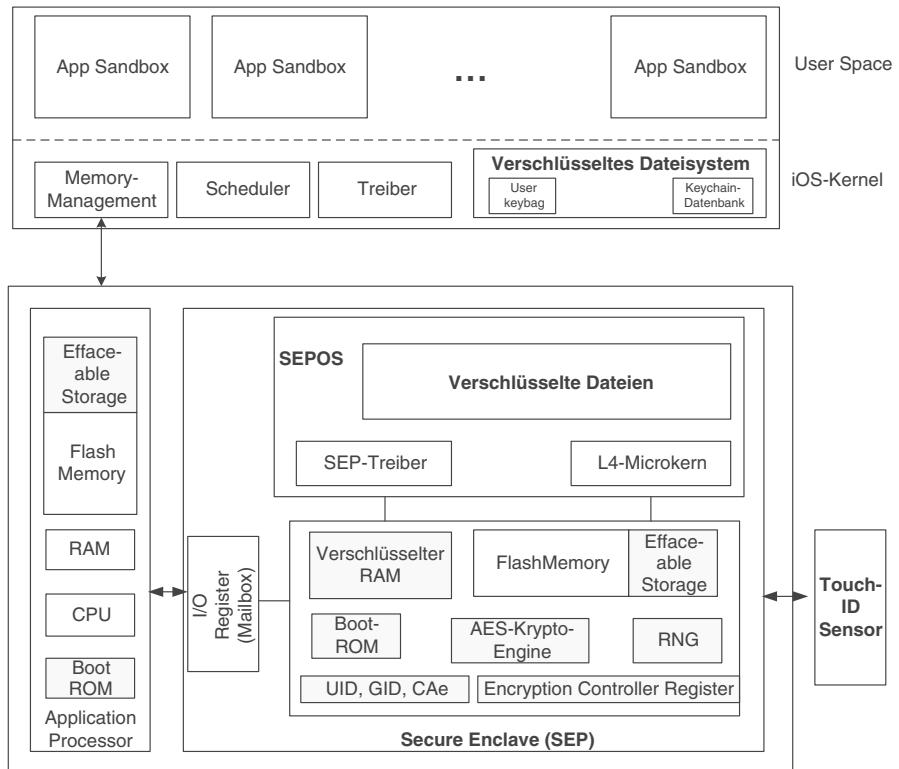


Abbildung 13.1: iOS-Architektur

iOS Architektur

Das Betriebssystem iOS läuft auf Apple A-Prozessoren, die auf 64-Bit dual-Core ARM-Prozessoren basieren. Eine wichtige Komponente der Apple-Hardware ist der Secure Enclave Prozessor (SEP). Der SEP ist ein Ko-Prozessor des A-Prozessors. Er bietet eine sichere Umgebung unter anderem zur Durchführung von Verschlüsselungen und besitzt ein eigenes Betriebssystem, das SEPOS, das unter anderem einen verschlüsselten Speicher zur Verfügung stellt. iOS unterstützt AES-256 Verschlüsselung und verfügt dafür über eine dedizierte Krypto-Engine, die Teil der Sicheren Enklave ist.

Der iOS-Kernel verwaltet unter Rückgriff auf die Sichere Enklave seine Dateien in einem verschlüsselten Dateisystem und stellt Sandboxes zur

mentierungstechnischen Details von der technischen Realität abweicht. Für Hinweise auf mögliche unsaubere Beschreibungen wäre ich deshalb sehr dankbar.

isolierten Ausführung von Apps im User-Space zur Verfügung. Seit dem iPhone5 in 2013 unterstützt iOS auch die Authentisierung mittels Fingerabdruck, wofür ein entsprechender Sensor über die Komponente Touch-ID in die Architektur integriert ist. Der Sensor interagiert mit der Enklave, um Fingerabdrücke zu prüfen. Sowohl die Enklave als auch der eigentliche Prozessor besitzen ein Boot-ROM, das beim Secure Boot ausgelesen wird. Der SEP führt ein eigenes Secure Boot basierend auf seinem Boot-ROM durch.

Jeder A-Prozessor besitzt eine eindeutige Geräte-Identität, die UID, die von der Sicheren Enklave erzeugt wird und nicht ausserhalb des SEP sichtbar ist. Weiterhin werden auch die Gruppen-ID (GID) sowie der öffentliche Schlüssel der Apple-CA bei der Geräte-Herstellung in der Enklave über das Boot-ROM bereit gestellt. Der GID ist fest in einer Generation von Apple A-Prozessoren eingebracht und ändert sich bei einem Generationswechsel.

Zum sicheren Löschen von Daten, wie insbesondere Schlüsseln, wird ein Low-Level-Konzept namens Effaceable Storage genutzt. Dies ist ein dedizierter NAND-Speicherbereich, der direkt adressiert werden kann, um auf Gatter-Ebene einzelne Datenblöcke direkt zu löschen. In diesem Speicher werden spezielle Schlüssel abgelegt, wie zum Beispiel der Dateisystem-Schlüssel, um ein schnelles Wipe (siehe Seite 673) zu ermöglichen.

Effaceable Storage

Der Grund für die Einführung von Effaceable Storage-Bereichen ist, dass NAND-Flash-Speicherzellen kein sicheres Löschen von Speicherinhalten garantieren. Da bei Flash-Speichern nach einer gewissen Zahl von Schreib- bzw. Löschvorgängen eine Art Verfall eintritt und einzelne Zellen ausfallen können, setzen Flash-Speicher den Wear-Leveling-Algorithmus ein, der Schreibzugriffe gleichmäßig auf das Medium verteilt. Auf Flash-Speicher wird über den Flash Translation Layer (FTL) zugegriffen, der das Wear-Leveling (dt. Verschleissnivellierung) unterstützt. Unter Nutzung von Wear-Leveling markiert der SSD-Controller beim Löschen eine bereits beschriebene Flash-Page möglicherweise lediglich als ungültig, anstatt sie direkt zu überschreiben. Auch nach mehrmaligem Löschen der NAND-Zellen können noch Reste vorhanden sein. Dieses Verhalten des Speichers, das kein sicheres Löschen der Speicherinhalte gewährleistet, ist natürlich für sensitive Daten, wie kryptografische Schlüssel, ungeeignet. Auch wenn man auf diese Daten typischerweise über die normale Schnittstelle des Speichermediums nicht zugreifen kann, ist ein Auslesen mit spezieller Elektronik möglich. Der Effaceable-Speicher greift deshalb direkt, ohne den FT Layer, auf die Zellen zu, wodurch kein Wear-Leveling durchgeführt wird und ein sicheres Löschen möglich ist.

Wear-Leveling

Im Folgenden gehen wir auf die Komponenten der Sicherheitsarchitektur etwas genauer ein.

13.1.2 Sichere Enklave

SEP

Die Sichere Enklave (SEP) ist ein Ko-Prozessor in den Apple-Prozessoren ab der A7 Prozessorfamilie. Das Ziel des SEP ist der Schutz der Nutzerdaten selbst bei Kompromittierung des Application-Prozessors (AP). Der SoC (System-on-Chip) besitzt ein eigenes Betriebssystem, SEPOS, das auf einer Variante des L4-Mikrokerns läuft. Die Enklave verwaltet einen verschlüsselten RAM-Speicher, verschlüsselte Dateien im Flash-Speicher sowie einen Hardware-basierten Zufallszahlengenerator (RNG) und ein eigenes Boot-ROM. Zur RAM-Verschlüsselung wird bei jedem Restart ein neuer, flüchtiger (ephemeral) Schlüssel erzeugt und mit der Geräte-UID verbunden. Ab der Apple A8 Familie wird der RAM Speicher auch über AES-CCM authentisiert. SEP Dateien werden mit Schlüsseln, die aus der UID abgeleitet werden und in die ein Anti-Replay-Zähler einfließt, verschlüsselt.

Kryptografische Operationen werden in der Enklave durch die dedizierte AES-Krypto-Engine durchgeführt, und die Enklave ist auch für das Schlüsselmanagement, wie Schlüsselerzeugung, zuständig. Dazu verwaltet die Enklave eine Schlüsselhierarchie, auf die wir im Zusammenhang mit dem Datei-Schutz näher eingehen. Die Enklave kommuniziert mit dem Applikations-Prozessor (AP) über eine kontrollierte Schnittstelle mittels eines sicheren Mailbox-Konzepts. Dieses ist als Interrupt-gesteuertes Message-Passing mit den I/O-Registern des SEP implementiert. Die Enklave führt ihren eigenen gesicherten Boot und ebenfalls einen eigenen sicheren Update-Prozess durch. Auf das sichere Boot des AP und das sichere Update von Firmware und Betriebssystem für den AP gehen wir weiter unten ein.

Krypto-Engine

Krypto-Engine

Jedes iOS-Gerät besitzt einen in Hardware implementierten AES-256 Krypto-Engine. Die Ver- und Entschlüsselung findet auf dem DMA-Datenpfad zwischen dem Flash-Speicher und dem RAM statt. Ab der A9-Prozessor-Generation besitzt der Krypto-Engine einen isolierten Bus, so dass Nutzerdaten nur über den DMA-Datenpfad in den RAM übertragen werden.

Hardware-Schlüssel und flüchtige Schlüssel

UID

Bei der Herstellung des Gerätes wird im SEP die eindeutige Geräte-Identifikation UID generiert, der nach dem Zerstören von Verbindungen nur noch vom SEP nutzbar ist. Zudem wird bei der Geräteherstellung die Gruppen-ID in die Sichere Enklave eingebracht. Die IDs sind 256-Bit AES Schlüssel. Die Geräte-UIDs werden gerätespezifisch erzeugt und sind nicht extern sichtbar und damit auch nicht gespeichert (kein Key-Escrow). Die Gruppen-ID ist ein Schlüssel, der für alle Geräte einer Produktfamilie, zum

Beispiel für alle Geräte mit A8-Prozessoren, gleich ist und nur für unkritische Operationen, wie beispielsweise ein Betriebssystem-Update, genutzt wird. Durch die Nutzung des GID zur Software-Verschlüsselung soll unter anderem ein Reverse-Engineering erschwert werden. Ver- und Entschlüsselungen erfolgen ausschließlich in Hardware in dem Krypto-Engine. Die in Silizium implementierten IDs sind weder über eine JTAG noch über andere Schnittstellen von außen zugreifbar, und damit besonders geschützt. Selbst die SEP-Software kann den UID und den GID nicht lesen, sondern lediglich nutzen. Der eindeutige Geräteschlüssel UID wird genutzt, um Datenschlüssel an das Gerät zu binden. Mit dem UID werden zudem Offline-Angriffe, bei denen der Flash-Speicher aus dem Gerät extrahiert wird, um beispielsweise den Passcode mittels Brute Force zu knacken, verhindert.

Außer den beiden genannten, in Hardware bereitgestellten, statischen Schlüsseln, werden weitere Schlüssel auf dem Gerät dynamisch unter Nutzung eines integrierten Hardware-Zufallszahlengenerators RNG generiert.

Flüchtige Schlüssel

Bei jedem (Re-)Starten des Geräts erzeugt die Enklave einen flüchtigen (ephemeral) Schlüssel K_{eph} , der mit der UID des Geräts verknüpft wird. Die Enklave nutzt diese Schlüssel zur Verschlüsselung und Authentisierung der Daten in ihrem RAM-Speicher. Der Schlüssel wird in einem spezifischen Register, dem Encryption Control Register abgelegt. Der flüchtige Schlüssel K_{eph} ist für eine Boot-Session gültig und wird gelöscht, wenn das Gerät herunter gefahren wird.

ephemeral Key

Daten, die im iOS-Dateisystem gespeichert werden, werden durch die Enklave verschlüsselt, wofür ein dateispezifischer Schlüssel der per-file-key (s.u.) genutzt wird. Die Metadaten einer jeden Datei werden separat mit einem anderen Schlüssel verschlüsselt und sind nur auf dem Gerät selbst wieder entschlüsselbar. Über die Metadaten ist auch die Datei mit der Geräte-UID verknüpft. Ein weiterer flüchtiger Schlüssel wird beim Booten zwischen dem AES-Krypto-Engine im Storage Controller und dem SEP etabliert. Dieser Schlüssel wird zum Schutz der Datei-Schlüssel verwendet, damit der Application-Prozessor niemals Kenntnis über den Datei-Schlüssel erlangt.

13.1.3 Touch ID

Die Touch ID ist ein Fingerabdruck-Sensor zur einfachen Authentisierung von Nutzern. Fingerabdrücke werden in der Sicheren Enklave überprüft (match), wofür im Speicher der Enklave Referenzwerte in Form von Node Maps abgelegt sind. Die Referenzwerte sind hash-artig gespeichert, so dass eine Wiederherstellung des Originalabdrucks aus dem gespeicherten Wert nicht möglich ist. Der SEP und der Touch ID-Sensor sind über ein Serial Peripheral Interface (SPI) verbunden. In dem Set-up für die Fingerabdrucker-

Touch ID

kennung werden bis zu fünf verschiedene Fingerabdrücke erfasst, und deren jeweiliger Hashwert wird in der Sicheren Enklave gespeichert. Beim Entblockieren des Geräts (Unlock) durch den Benutzer wird der aktuelle Scan des Fingerabdrucks an den SEP übertragen, in Vektoren umgerechnet und mit den gespeicherten Abdrücken verglichen. Dazu werden die erhobenen Daten sicher vom Sensor zum SEP übertragen. Die Basis dafür ist ein gemeinsamer AES-Schlüssel $K_{S,E}$, der zwischen dem Sensor und der Enklave bei der Geräteherstellung etabliert wird. Zum verschlüsselten Transfer der Sensordaten erzeugt die Enklave einen Session Key K_c und überträgt diesen verschlüsselt an den Sensor: AES-CCM($K_c, K_{S,E}$).

Der Sensor nutzt den Session Key K_c , um die aktuellen Fingerprint-Daten gesichert zur Prüfung an die Enklave zu senden. Der Scan des Fingerabdrucks wird temporär im verschlüsselten RAM-Speicher der Enklave für die Analyse abgelegt und nach erfolgter Analyse wieder gelöscht. Die erhobenen Fingerabdruckdaten sind nur im Touch ID-Sensor im Klartext vorhanden.

Seit der Version iOS 8 kann der Touch ID-Sensor auch von Apple-Payment-Apps genutzt werden, beispielsweise für Einkäufen im iTunes- und App-Store, und auch Apps von Drittanbietern können den Touch ID-Sensor nutzen. Dafür wird dann auch bei einem bereits entsperrten Gerät erneut der Fingerabdruck erforderlich, der dann als Zugangsschlüssel zur App genutzt wird. Des Weiteren können Apps Schlüssel durch den SEP generieren lassen und den Zugriff darauf mittels einer Touch ID Authentisierung schützen.

Secure Element

Falls auf dem Gerät ein Secure Element, beispielsweise für das sichere Bezahlen, integriert ist, so wird auch bei der Geräteherstellung ein gemeinsamer Schlüssel zwischen den beiden Komponenten etabliert.

Blockieren (lock) des Geräts

lock

Wie weiter unten noch detaillierter ausgeführt wird, werden Dateien mittels eines Datei-individuellen Datei-Schlüssels DK verschlüsselt und im iOS-Dateisystem verschlüsselt gespeichert. Der Datei-Schlüssel DK wird seinerseits von einem so genannten Klassenschlüssel CK der Schutzklasse der Datei verschlüsselt. Der Klassenschlüssel der Dateien der Schutzklasse *Complete Protection* wird mittels eines Master Key geschützt, der aus dem Passcode des Nutzers sowie der UID des Geräts abgeleitet wird. Um auf die Daten der Datei zugreifen zu können, müssen sie also entsperrt werden, wofür der CK erforderlich ist. Solange das Gerät nicht blockiert ist, ist der CK im verschlüsselten RAM der Enklave gespeichert und zum Entsperren für den SEP direkt verfügbar, so dass nicht bei jeder Datei-Zugriffsoperation der Passcode eingegeben werden muss. Der Klassenschlüssel CK wird direkt innerhalb von (10ms) aus dem RAM der Enklave gelöscht, wenn das Gerät gelockt wird und keine Authentisierung durch Touch ID aktiviert ist.

Der CK bleibt dann aber immer noch als verschlüsselter Blob im Flash des SEP präsent. Bei erneuter Eingabe des Passcodes kann der Master-Key berechnet, der CK damit wieder entschlüsselt und der Datei-Schlüssel wieder hergestellt werden.

Ist die Touch ID-Authentisierung aktiviert, so wird beim Locken der Klassenschlüssel CK nicht aus der Enklave gelöscht, sondern mit einem zufälligen (random) Schlüssel K_{rs} , der vom SEP generiert wird, verschlüsselt. Dieser Schlüssel K_{rs} wird an das SBIOS-Subsystem, das Bestandteil des SEP und für das Überprüfen der Fingerabdrücke verantwortlich ist, übertragen. Der SEP verschlüsselt den Klassenschlüssel mit dem Schlüssel K_{rs} , $\text{Blob} = \text{AES-CCM}(CK, K_{rs})$ und löscht danach den Schlüssel K_{rs} .

Entblockieren (unlock) des Geräts

Falls eine Authentisierung des Nutzers nur durch die Eingabe des Passcodes möglich ist, muss der Nutzer den Passcode eingeben und der Master-Key wird daraus generiert, so dass der verschlüsselte Klassenschlüssel aus dem Flash-Speicher wieder hergestellt werden kann. Die Datei-Schlüssel, die mit CK geschützt sind, können nun wieder entsperrt werden.

Bei einer aktivierte Touch ID-Authentifikation muss der Nutzer einen aktuellen Fingerprint vorweisen. Bei einem erfolgreichen Login mittels Fingerprint liefert das SBIOS-Subsystem den Schlüssel K_{rs} , um den Blob wieder zu entschlüsseln und den Klassenschlüssel CK wieder direkt zugänglich zu machen. Der Wrapper-Key K_{rs} wird nach 5 Fehlversuchen beim Touch ID Login sowie nach 48 Stunden gelöscht, so dass sich danach der Nutzer wieder mittels seines Passcodes authentisieren muss.

13.1.4 Systemsicherheit

Konzepte der Systemsicherheit sind in allen iOS-Geräten umgesetzt. Sie betreffen das sichere Booten und Konzepte für ein sicheres Software-Upload.

Sicheres Booten

Nach dem Einschalten eines Gerätes (Power-on) wird der Bootvorgang gestartet. Dazu initiiert der Applikations-Prozessor den Bootprozess und liest den Boot-Code aus seinem Boot-ROM. Dieser Code wird bei der Chip-Herstellung in den Speicher eingebracht und ist als read-only Speicher unveränderlich; der Code wird als Hardware Root of Trust betrachtet. Der Boot-Code enthält insbesondere den Apple Root CA Public Key (CA_e), der benutzt wird, um Code, der von Apple signiert wurde, zu validieren. Zum Signieren und Validieren wird ECC genutzt. Der private Apple Root CA Key (CA_d) ist nur Apple bekannt.

Secure Boot

Root CA
Public Key

Durch die Ausführung des Boot-Codes wird als erstes die Signatur des iBoot-Bootloaders mit dem öffentlichen Schlüssel CA_e aus dem Boot-ROM validiert. Ist die Signatur gültig, wird der Bootloader geladen, ausgeführt und als nächstes wird die Signatur des iOS-Kernels geprüft, bevor dieser geladen wird. Schlägt eine Signatur-Validierung fehl, so stoppt der Bootvorgang und das Gerät wechselt in den Recovery-Zustand und zeigt den *Connect to iTunes*-Bildschirm. In diesem Fall muss das Gerät via USB mit iTunes verbunden werden und wird auf die Basis-Einstellung des Herstellers zurückgesetzt.

Falls das iOS-Gerät über eine Mobilfunkschnittstelle verfügt, wird das Baseband-Subsystem auf analoge Weise durch den Baseband-Prozessor sicher gebootet.

Verfügt das iOS-Gerät über einen Secure Enclave-Ko-Prozessor, so führt der Secure Enclave Ko-Prozessor ebenfalls ein sicheres Booten in Bezug auf die vom übrigen Kernel separierte Software der Enklave durch. Der Bootvorgang startet mit dem Boot-ROM der Enklave.

System Software Authorization

Sicheres Update

Mit dem sicheren Booten wird sichergestellt, dass nur Code auf das Gerät geladen wird, der von Apple signiert wurde. Um zu verhindern, dass beim Booten eine veraltete, aber dennoch korrekt signierte Komponente geladen wird (downgrade), verwendet iOS das Konzept der *System Software Authorization*. Lädt ein Gerät ein iOS- oder Firmware-Update nach, so kann dies entweder über iTunes erfolgen oder Over-the-Air. Bevor die Updates installiert und ausgeführt werden dürfen, muss verifiziert werden, dass es sich um aktuelle Updates handelt. Dazu berechnet das Gerät die Hashwerte der Update-Routinen (z. B. iBoot, Kernel) und verbindet sich mit dem *Apple Installation Authorization Server*. Das Gerät übergibt die Liste seiner gemessenen Hashwerte, eineNonce N und die Identität des Geräts ECID³. Der Server prüft, ob die Hashwerte korrekt sind, und signiert die Liste der korrekten Hashwerte zusammen mit der ECID des Geräts und der Nonce N. Das Gerät verifiziert die Signatur und prüft, welche der Hashwerte durch die Signatur als korrekt bestätigt werden und installiert die entsprechenden Komponenten. Durch die Einbindung der Geräte-Identität ECID wird die Antwort des Servers für das Gerät personalisiert und die Frische der Antwort wird durch die Nonce N gewährleistet.

closed world

Die Boot- und Update-Prozesse stellen sicher, dass nur solche Software-Komponenten auf ein Gerät geladen werden, die von Apple signiert sind, es ist damit eine abgeschlossene Welt (closed world). Dies wurde in der breite-

³ Die UID verlässt den SEP nicht. Die ECID dagegen ist nach außen sichtbar, es handelt sich um einen 64-Bit eindeutigen Identifikator des Prozessors.

ren Öffentlichkeit stärker diskutiert, als 2015 das FBI im Zuge von Verbrechungsbekämpfungsmaßnahmen die Firma Apple aufforderte, eine spezielle Firmware-Update-Version des FBI zu signieren. Dieses Update sollte nach seiner Installation auf dem Gerät des zu observierenden Tatverdächtigen das FBI in die Lage versetzen, die auf dem Gerät gespeicherten, verschlüsselten Daten zu entschlüsseln. Ein solches Update kann gemäß des oben beschriebenen Prozesses nur unter Nutzung des geheimen Apple-Signier-Schlüssels in das Gerät eingebracht werden. Apple hat sich geweigert, diese Updates zu signieren.

13.1.5 Passcode

Durch das Setzen eines Passcodes wird automatisch der Dateischutz (siehe Abschnitt 13.1.6) ermöglicht. Ein Nutzer-Passcode kann eine 6- bzw. 4-stellige Zahl oder eine beliebig lange alphanumerische Zeichenkette sein. Der Passcode wird zur Nutzer-Authentisierung und zum Entblockieren des Gerätes verwendet, aber er wird auch in manchen Fällen zur Entschlüsselung von kryptografischen Schlüsseln benötigt. Dies wird im Abschnitt zur Dateiverschlüsselung erläutert.

Passcode

Aus dem Passcode des Nutzers wird zusammen mit dem eindeutigen Geräte-Schlüssel UID unter Nutzung einer festgelegten Key-Derivation-Funktion, in die u. a. auch ein Salt-Wert einfließt, der Master-Key berechnet, der benötigt wird, um in weiteren Schritten auf die Daten im verschlüsselten Speicher zuzugreifen. Durch die Verknüpfung mit dem Geräte-Schlüssel wird erreicht, dass ein Brute-Force-Angriff zum Knacken des Passcodes auf dem Gerät selbst erfolgen muss. Ein Zugriffsversuch mit der Prüfung des Passcodes dauert künstlich lange (80ms). Ein Brute-Force-Angriff auf einen 6-stelligen Passcode kann bis zu 5,5 Jahren dauern, während ein Brute-Force Angriff auf einen 4-stelligen Passcode nur ca. 15 Minuten benötigt. Zusätzlich kann man konfigurieren, dass nach 10 Fehlversuchen (die Anzahl ist über Policy einstellbar) ein automatisches Wipe stattfindet. Ein Wipe hat zur Folge, dass keine Datei auf dem Gerät mehr zugreifbar ist. Zusätzlich kann konfiguriert werden, dass das SEP eine bestimmte Wartezeit zwischen Fehlversuchen durchsetzt, wie beispielsweise eine Stunde warten nach 9 Fehlversuchen. Die aktuelle Wartezeit wird vom SEP bei einem Neustart erneut gestartet, so dass man die Wartezeit nicht durch ein Reboot umgehen kann.

13.1.6 Dateischutz

Daten im Flash-Speicher werden durch das Konzept der *Data Protection* geschützt. Das Konzept unterscheidet verschiedene Schutzklassen für Dateien. So ist es beispielsweise möglich, dass das Gerät im blockierten Zustand eingehende Ereignisse, wie Anrufe, bearbeiten und gleichzeitig den Schutz der

Datei-Schutz

Daten aufrecht erhalten kann. Eine solche Schutzklasse wird per Default für Mail, Kalender, Kontakte, Fotos und Gesundheitsdaten sowie Messages angewandt. Die Umsetzung des Schutzkonzepts basiert auf den zur Verfügung stehenden Hardware-Schutz-Mechanismen und der Definition einer Schlüsselhierarchie.

Mit der Erzeugung einer neuen Datei wird dieser eine Schutzklasse zugeordnet und auch ein Datei-Schlüssel DK erzeugt. Die Datei wird verschlüsselt im Flash-Speicher gespeichert. Der Datei-Schlüssel DK wird seinerseits geschützt, und es ist abhängig von der gewählten Schutzklasse der Datei, welche Maßnahme anzuwenden ist, um auf den Datei-Schlüssel zuzugreifen.

class key

Jede Schutzklasse definiert einen Klassen-Schlüssel (class-key). Eine Datei wird bei ihrer Erzeugung einer Klasse zugeordnet. Auf die Eigenschaften der Klassen gehen wir weiter unten noch ein. Um auf die Datei wieder zugreifen zu können, ist der Klassen-Schlüssel für die Klasse, der die Datei zugeordnet wurde, erforderlich. Es werden vier verschiedene Klassen-Schlüssel unterschieden:

- Klasse *Complete-Protection*: Dies ist ein 256-Bit AES Schlüssel, der nur verfügbar ist, wenn das Gerät entsperrt ist. Es ist somit die Eingabe des Passcodes erforderlich, um Klassen-Schlüssel dieser Klasse wieder zugreifbar zu machen. Zusätzlich wird auch der UID-Schlüssel des Geräts benötigt.
- Klasse *Protected-Unless-Open*: Dies ist ein ECC-Schlüsselpaar auf einer festgelegten Kurve, wobei der öffentliche Schlüssel stets zugreifbar ist, während der private Schlüssel nur verfügbar ist, wenn das Gerät entsperrt ist, wofür die Eingabe des Passcodes erforderlich ist. Für den Zugriff auf den privaten Schlüssel ist somit auch die Eingabe des Passcodes sowie der UID-Schlüssel erforderlich.
- Klasse *Until-First-User-Authentication*: Dies ist ein 256-Bit AES Schlüssel, der verfügbar ist, nachdem der Nutzer zumindest einmal das Gerät nach dem Booten entsperrt, also den Passcode eingegeben hat.
- Klasse *No-Protection*: Dies ist ein 256-Bit AES Schlüssel, der immer verfügbar ist.

Datei-Schlüssel

Jede Datei f wird blockweise mit einem eigenen 256-Bit AES Schlüssel (DK) mittels AES-CBC (bzw. AES-XTS bei A8-Prozessoren) verschlüsselt, wobei der erforderliche Initialisierungsvektor IV aus dem Blockoffset der Datei abgeleitet wird. Der Datei-Schlüssel DK wird mit dem Klassen-Schlüssel CK der Klasse von f verschlüsselt (wrapped) und als Key-Blob in den Metadaten der zu schützenden Datei f gespeichert.

Die Metadaten wiederum werden mit dem Dateisystem-Schlüssel (file system key) K verschlüsselt. Dieser Schlüssel wird, verschlüsselt mit dem Geräte-Schlüssel UID, im EffaceableSpeicher auf dem Gerät abgelegt und ist damit nicht geeignet, Vertraulichkeit von Daten herzustellen. Er dient vielmehr dazu, mit einem einzigen Kommando, einem Wipe, zu erreichen, dass keine Datei mehr auf dem Gerät zugreifbar ist, wenn der Schlüssel K gelöscht wird. Abbildung 13.2 veranschaulicht das Zusammenspiel der verschiedenen Schlüssel. Die Datenblöcke der Datei f sind mit dem Datei-Schlüssel DK verschlüsselt.

File system key

wipe

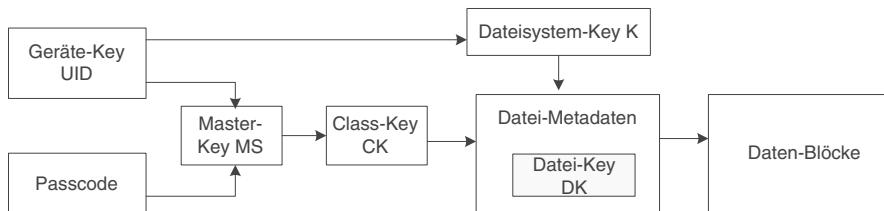


Abbildung 13.2: Schlüsselhierarchie für den Dateischutz

Um die Datei f zu öffnen, muss der Datei-Schlüssel DK aus den Metadaten der Datei f extrahiert werden. Dazu ist der Klassenschlüssel CK erforderlich, mit dem DK verschlüsselt wurde, sowie auch der Dateisystem-Schlüssel K, mit dem die Metadaten verschlüsselt sind. Zur Entschlüsselung des Dateisystem-Schlüssels K muss der in Hardware verankerte, nicht migrierbare Geräte-Schlüssel UID verfügbar sein. Um den Klassenschlüssel CK zu entschlüsseln, wird der Master-Key benötigt, der wiederum die Kenntnis des Passcodes und die UID erfordert. Abhängig von der Schutzklasse ist nur die Kenntnis des UID-Geräteschlüssels erforderlich, so dass bei der Erstellung des Master-Keys der Passcodeteil entfällt. Ist der Datei-Schlüssel DK durch den SEP erfolgreich entschlüsselt worden, wird er mit einem flüchtigen Schlüssel K_{wrap} verschlüsselt, der vom SEP beim Booten mit dem Storage-Controller ausgetauscht wird. Der so gewrappte Datei-Schlüssel DK wird an den Application-Prozessor weitergegeben, der ihn für Dateizugriffe an den Storage-Controller übergibt. Auf diese Weise ist sicher gestellt, dass der AP niemals einen Datei-Schlüssel DK im Klartext vorliegen hat. Durch die Schlüsselhierarchie ist es möglich, einen Passcode einfach zu wechseln, da nur die betroffenen Schlüssel neu verschlüsselt werden müssen, nicht jedoch die eigentlichen Datenblöcke.

Nachfolgend werden die verschiedenen Schlüsseltypen zusammen mit ihrer Funktion noch einmal im Überblick zusammengefasst.

- **UID:** Der UID ist ein eindeutiger Geräteschlüssel, der fest in Hardware implementiert ist und bei der Geräteherstellung generiert wird. Er ist Bestandteil der Sicheren Enklave. Er dient zur Bindung von Daten und Schlüssel an das Gerät.
- **Dateisystem-Schlüssel:** Der Schlüssel wird nach Installation und nach einem Wipe erzeugt, indem er mit einem von der UID abgeleiteten Schlüssel verschlüsselt und im Effaceable-Speicher abgelegt wird.
- **Ephemeral Key:** Das sind flüchtige Schlüssel, die für die Dauer der Boot-Session erzeugt werden. Einer dieser flüchtigen Schlüssel wird in ein dediziertes Register im SEP abgelegt und zur Verschlüsselung der Daten im RAM-Speicher des SEP verwendet (verschlüsselter RAM). Ein anderer flüchtiger Schlüssel wird für den sicheren Transfer von Dateischlüsseln vom SEP zum Application-Prozessor verwendet.
- **Master-Key:** Der Schlüssel wird aus dem Passcode, einem Salt-Wert und der UID abgeleitet und dient zum Wrappen von Klassenschlüsseln der Schutzklassen. Der SEP berechnet den Master-Key aus dem Passcode und der UID und stellt ihn der Krypto-Engine zum Ver- bzw. Entschlüsseln zur Verfügung. Falls der Master-Key nur von der Geräte-UID abhängig sein soll, entfällt die Teilberechnung, die den Passcode einbezieht:

$$\text{Master-Key} = KDF_2(KDF_1(\text{Passcode} \mid \text{Salt}), \text{UID})$$

Der Master-Key wird nicht gespeichert, sondern bei Bedarf aus dem aktuell einzugebenden Passcode wieder hergestellt.

- **Class Key** der Klasse *Complete-Protection*: Der Schlüssel dient zur Verschlüsselung von Datei-Schlüsseln der Dateien der Klasse Complete-Protection. Der Klassen-Schlüssel wird seinerseits unter Nutzung des Master-Keys verschlüsselt gespeichert und bleibt nutzbar, bis das Gerät blockiert wird. Zur Entschlüsselung des Klassen-Schlüssels ist die Eingabe des Passcodes sowie die UID des Geräts erforderlich.
- **Class Key** der Klasse *Protected-Until-First-User-Authentication*: Die Funktion des Klassen-Schlüssels ist analog zum Klassen-Schlüssel für Dateien mit dem Attribut Complete-Protection, jedoch steht der Schlüssel auch bei blockierten Geräten zur Verfügung und muss erst nach einem Re-Boot wieder durch die Eingabe des Passcodes entschlüsselt werden.
- **Class Key** der Klasse *No-Protection*: Der Klassen-Schlüssel wird mit dem Geräte-Schlüssel UID verschlüsselt.
- **Class Key** der Klasse *Protected-Unless-Open*: Das Klassen-Schlüsselpaar wird zusammen mit einem flüchtigen Datei-Schlüsselpaar zur Berechnung eines Hilfsschlüssels verwendet, mit dem der Datei-Schlüssel verschlüsselt wird. Der private Klassen-Schlüssel wird mit dem Master-

Key geschützt gespeichert und erfordert die Eingabe des Passcodes zur Wiederherstellung.

- **Datei-Schlüssel:** Ein Datei-Schlüssel wird bei Datei-Erzeugung zur Datei-Verschlüsselung generiert, mit dem Class-Key der zugeordneten Schutzklasse verschlüsselt und verschlüsselt in den Metadaten der Datei in Dateisystem gespeichert.
- **Passcode:** Der Passcode ist ein 6- bzw. 4-stelliger Zugangscode, der mit dem Gerät-Schlüssel UID verschlüsselt auf dem Gerät gespeichert und damit an das Gerät gebunden wird.
- **Apple Root CA Public Key:** Der öffentliche Schlüssel wird bei der Herstellung des Geräts im Boot-ROM abgelegt. Er dient zur Validierung der Apple-Code-Signaturen beispielsweise bei Kernel-Updates.

Datenschutz-Klassen

Bei der Erzeugung einer Datei wird deren Klasse festgelegt, wobei die Klassenzuordnung spezifiziert, unter welchen Bedingungen der Klassen-Schlüssel wiederhergestellt wird, so dass die Daten der Datei zugreifbar sind. Es werden vier Basis-Klassen unterschieden, die aber durch benutzerspezifische Policies ergänzt werden können.

Klassen

Complete-Protection: Ein Klassen-Schlüssel der Klasse *Complete-Protection*, der mit dem Master-Key geschützt wird, wird genutzt. Sehr schnell nach dem Blockieren des Geräts, nach ca. 10 Sekunden, wird der entschlüsselte Klassenschlüssel aus dem SEP-RAM gelöscht und die damit geschützten Dateien sind erst wieder zugreifbar, wenn erneut der Passcode eingegeben wurde.

No-Protection: Ein Klassenschlüssel der Klasse *No-Protection* wird genutzt. Der Klassen-Schlüssel wird nur mit der UID des Gerätes verschlüsselt und im Effaceable Storage abgelegt. Falls für eine Datei keine explizite Klassenzuordnung durchgeführt wurde, wird diese Klasse als Default gewählt. Dadurch wird gewährleistet, dass alle Dateien verschlüsselt werden und der Wipe-Mechanismus dafür sorgt, dass nach dessen Ausführung keine Datei mehr zugreifbar ist.

Protected-Until-First-User-Authentication: Ein Klassen-Schlüssel der Klasse *Protected-Until-First-User-Authentication* wird genutzt. Im Unterschied zur Klasse *Complete-Protection* wird der entschlüsselte Klassen-Schlüssel beim Blockieren des Geräts nicht aus dem RAM des SEP gelöscht, sondern erst bei einem Re-Boot verworfen. Das heißt erst ein Re-Boot erfordert wieder eine Passcode-Eingabe, so dass dies nur einen Schutz vor Reboot-Angriffen darstellt. Die Dateien dieser Schutz-

Klasse sind auch bei gelocktem Gerät zugreifbar. Das heißt sie können geöffnet, beschrieben, gelesen werden, da der Klassen-Schlüssel zur Entschlüsselung des Datei-Schlüssels im SEP verfügbar ist. Diese Klasse ist vergleichbar mit einer Festplattenverschlüsselung bei Laptops oder PCs.

Protected-Unless-Open: Dies ist eine interessante Klasse, da sie den Schutz der Datei garantiert, solange sie geschlossen ist, aber dennoch beispielsweise das Herunterladen von Daten, wie zum Beispiel ein Mail-Attachement, ermöglicht, auch wenn das Gerät blockiert ist.

Im Folgenden erläutern wir das Konzept der Schutzklasse *Protected-Unless-Open* am Beispiel des Herunterladens einer Datei während das Gerät gelockt ist. Um die Schutzanforderungen der Klasse umzusetzen, aber gleichzeitig eine derartige Nutzung zu ermöglichen, wird ein Mechanismus benötigt, der

- für die neu erzeugte Datei einen Datei-Schlüssel erzeugt und diesen Datei-Schlüssel geschützt in den Metadaten der Datei ablegt, wie dies durch einen Klassen-Schlüssel herkömmlich geleistet wird. Da die Aktivitäten im Hintergrund erfolgen, soll jedoch hierfür keine Eingabe des Passcodes erforderlich sein (keine Nutzerinteraktion).
- Der Datei-Schlüssel wird nach dem Schließen der Datei aus dem SEP gelöscht.
- Da die Daten der Datei geschützt sein sollen, wenn die Datei geschlossen ist (analog zu Dateien mit dem Attribut Complete-Protection), muss nun aber zum Entschlüsseln des Datei-Schlüssels der Nutzer-Passcode erforderlich sein.

Um diese Anforderungen zu erfüllen, werden zwei asymmetrische Schlüsselpaare sowie ein EC-Diffie-Hellman-Verfahren verwendet. Das Vorgehen ist im Groben wie folgt:

1. Für die neue Datei wird ein Datei-Schlüssel DK erzeugt.
2. Für Dateien der Schutzklasse *Protected-Unless-Open* existiert ein Klassen-Schlüsselpaar ($class_d, class_e$).
 - Der private Klassen-Schlüssel $class_d$ wird mit dem Master-Key geschützt, und zu dessen Wiederherstellung ist die Eingabe des Passcodes und die UID erforderlich.
 - Der öffentliche Klassen-Schlüssel $class_e$ wird nicht speziell geschützt im Flash gespeichert.
3. Für die Datei wird ein dateispezifisches Schlüsselpaar ($file_d, file_e$) erzeugt.

- Der öffentliche Datei-Schlüssel $file_e$ wird in den Metadaten der Datei f abgelegt.
 - Der private Datei-Schlüssel $file_d$ wird nach der Berechnung des DH-Geheimnisses s gelöscht.
4. Mittels des Diffie-Hellmann-Verfahrens auf Elliptischen Kurven (ECDH) wird das DH-Geheimnis s bestimmt, mit
 $s = ECDH(file_d, class_e)$.
 5. Unter Nutzung des SHA256 wird der Datei-Schlüssel DK verschlüsselt:
 $File_{wrap} = AES(DK, SHA256(s))$.
Der $File_{wrap}$ wird ebenfalls in den Metadaten der Datei abgelegt und der private Schlüssel $file_d$ wird gelöscht.
 6. Sobald die Datei geschlossen wird, wird auch der Datei-Schlüssel DK aus dem SEP gelöscht.

Nach dem Schließen der Datei sind die Datenblöcke geschützt. Für ein erneutes Öffnen der Datei muss der Datei-Schlüssel wieder hergestellt werden. Anders als bei den anderen Schutzklassen ist hierfür nun aber kein Klassen-Schlüssel, sondern der individuell berechnete Schlüssel $File_{wrap}$ wiederherzustellen. Hierfür ist das DH-Geheimnis s erforderlich. Da der Schlüssel $file_d$ nicht mehr verfügbar ist, muss das DH-Geheimnis s mit der anderen möglichen Schlüsselkombination berechnet werden, nämlich mit

$$s = ECDH(file_e, class_d)$$

Das bedeutet, dass zur Berechnung von s jetzt der private Klassen-Schlüssel $class_d$ benötigt wird. Dieser ist durch den Master-Key geschützt im Flash abgespeichert, so dass die Eingabe des Nutzer-Passcodes erforderlich ist, um den Master-Schlüssel wieder herzustellen. Der öffentliche Schlüssel $file_e$ ist in den Metadaten der Datei gespeichert, die wiederum mit dem Dateisystem-Schlüssel K geschützt sind; es wird also lediglich der UID des Geräts benötigt, um den Schlüssel K wiederherzustellen und den Schlüssel $file_e$ zu extrahieren. Mit dem berechneten DH-Geheimnis s kann der Datei-Schlüssel DK wieder hergestellt, $DK = AES(File_{wrap}, SHA256(s))$ und die Dateiblöcke können wieder entschlüsselt werden.

Prozessschritte des AP und SEP beim Öffnen einer Datei

Im Folgenden wird der Ablauf bei einem *Open* auf eine Datei f der Schutzklasse *Complete-Protection* detaillierter beschrieben, um die Zusammenhänge zwischen den Schlüsseln der Schlüsselhierarchie noch einmal detaillierter darzustellen.

Gegeben sei ein Szenario, in dem eine Datei f bereits erzeugt, aber geschlossen ist und verschlüsselt im Flash-Speicher des Application-Prozessor liegt. Das bedeutet, dass die Daten der Datei mit einem dateispezifischen Schlüssel DK_f verschlüsselt sind, und dieser Schlüssel DK mit dem Klassen-Schlüssel der Klasse CK_{cp} *Complete-Protection* verschlüsselt und in den Metadaten der Datei f gespeichert ist. Weiterhin gilt, dass die Metadaten mit dem Dateisystem-Schlüssel K verschlüsselt sind.

Um die Datei f zu öffnen, muss der Datei-Schlüssel DK_f wiederhergestellt werden, wofür der Klassen-Schlüssel CK_{cp} wiederhergestellt werden muss. Dieser Schlüssel ist mit dem Master-Key verschlüsselt und in der Schlüssel-Datei namens *user-key-bag*-Datei (vergleiche Abschnitt 13.1.7), die alle Klassen-Schlüssel verschlüsselt beinhaltet, im Flash gespeichert. Da *user-key-bag* (ukb) eine Datei ist, ist sie gemäß der durchgehenden iOS-Konzeption mit einem eigenen Datei-Schlüssel DK_{ukb} verschlüsselt. Die Datei ist als eine *No-Protection*-Datei erzeugt, so dass der Datei-Schlüssel DK_{ukb} mit dem Klassen-Schlüssel CK_{np} der *No-Protection*-Klasse verschlüsselt ist und in den Metadaten der Datei ukb abgespeichert ist. Die Metadaten sind wie stets mit dem Dateisystem-Schlüssel K verschlüsselt.

Der vergrößerte Ablauf ist dann wie folgt, wobei für die Notation gilt: $C[\text{data}, \text{key}] = \text{AES}(\text{data}, \text{key})$, also die Verschlüsselung von *data* mit dem Schlüssel *key*.

1. Verschlüsselte Metadaten der Datei f an den SEP übergeben:

AP → SEP: $C[\text{meta-daten-f}, K], C[\text{daten-f}, DK_f]$.

Der Datenblock meta-daten-f enthält den verschlüsselten Datei-Schlüssel $C[DF_f, CK_{cp}]$.

2. Aktivitäten des SEP:

- Um den Blob $C[\text{meta-daten-f}, K]$ zu entschlüsseln, wird der Schlüssel K benötigt, der verschlüsselt als $C[K, \text{UID}]$ im Effaceable-Speicher des SEP vorliegt und vom SEP direkt unter Nutzung der UID wiederhergestellt werden kann. Der Schlüssel K wird im verschlüsselten RAM des SEP abgelegt.
- Mit dem Schlüssel K kann der Blob $C[\text{meta-daten-f}, K]$ entschlüsselt und auf den in den Metadaten gespeicherten Blob $C[DF_f, CK_{cp}]$ zugegriffen werden.
- Um den Blob $C[DF_f, CK_{cp}]$ zu entschlüsseln, ist der CK_{cp} erforderlich, der seinerseits mit dem Datei-Schlüssel des Keybags DK_{ukb} verschlüsselt ist und als Blob $C[CK_{cp}, DK_{ukb}]$ in der Datei ukb gespeichert ist.

3. Der Application-Prozessor überträgt die verschlüsselten Daten der *user-key-bag*-Datei an den SEP:

AP → SEP: C[meta-daten-ukb, K], C[ukb, DK_{ukb}]).

Der Datenblock meta-daten-ukb enthält den verschlüsselten Datei-Schlüssel C[DK_{ukb} , CK_{np}].

4. Aktivitäten des SEP:

- Mit dem Schlüssel K aus dem RAM des SEP kann der Blob C[meta-daten-ukb, K] entschlüsselt und es kann auf den in den Metadaten gespeicherten Blob C[DK_{ukb} , CK_{np}] zugegriffen werden.
- Um den Blob C[DK_{ukb} , CK_{np}] zu entschlüsseln, ist der CK_{np} erforderlich, der seinerseits mit dem Gerät-Schlüssel UID (No-Protection-Klasse) verschlüsselt ist und vom SEP ohne weitere Nutzereingabe wieder hergestellt werden kann.
- Mit dem CK_{np} wird der Blob C[DK_{ukb} , CK_{np}] entschlüsselt und der Datei-Schlüssel DK_{ukb} extrahiert.
- Mit dem DK_{ukb} kann der Blob C[ukb, DK_{ukb}] entschlüsselt werden, wodurch der in der Datei ukb gespeicherte Blob C[CK_{cp} , master – key] zugreifbar wird. Zu dessen Entschlüsselung muss der Master-Key regeneriert werden, was die Eingabe des Passcodes durch den Nutzer erfordert.

5. User → AP: Passcode

6. AP → SEP: Passcode

7. Aktivitäten des SEP:

- Der SEP berechnet den master-key=KDR(passcode, UID) und entschlüsselt den Klassen-Schlüssel CK_{cp} damit. Der Master-Key wird wieder gelöscht.
- Der CK_{cp} wird im SEP-RAM und der Blob C[CK_{cp} , master – key] im SEP-Flash gespeichert.
- Mit dem Klassen-Schlüssel CK_{cp} kann der Blob C[DK_f , CK_{cp}] entschlüsselt und der Datei-Schlüssel DK_f zur Entschlüsselung des Datenblocks C[daten-block-f, DK_f] wiederhergestellt werden. Der Application-Prozessor benötigt den Schlüssel zur Weitergabe an den Storage Controller. Da jedoch Datei-Schlüssel nie im AP in Klartext verfügbar sein sollen, um mögliche Informations-Leakages zu reduzieren, wird der Datei-Schlüssel DK_f mit einem flüchtigen Schlüssel des SEP gewrapped und entsprechend verschlüsselt an den AP weitergereicht. Ein solcher Schlüssel wird bei jedem Booten des Geräts im SEP erzeugt und ist für die Dauer der Bootsession gültig.

- Nach der Entschlüsselung des Datenblocks wird der Datei-Schlüssel DK_f gelöscht und kann für den nächsten zu entschlüsselnden Datenblock der Datei f direkt wieder hergestellt werden, da der CK_{cp} im RAM der Enklave abgelegt ist.

Wird das Gerät gelockt, so bleiben die Daten geöffnet, um ein Instant-on beim Unlock des Geräts zu ermöglichen.

- Beim Sperren des Geräts löscht der SEP den Klassen-Schlüssel CK_{cp} aus dem RAM⁴, der Zugriff auf die Datenblöcke von f ist nicht mehr möglich (protected).
- Für einen Zugriff muss der Blob $C[CK_{cp}, \text{master-key}]$, der im Flash des SEP gespeichert ist, entschlüsselt werden. Dies erfordert jedoch ein unlock und die Eingabe des Nutzer-Passcodes, um den Master-Key wieder herzustellen.

13.1.7 Keybags

Keybag

Klassen-Schlüssel CK werden in speziellen Datenstrukturen, den Keybags, verwaltet. iOS unterscheidet fünf verschiedene Keybag-Typen: *user*, *device*, *backup*, *escrow*, *iCloud Backup*. Nachfolgend werden die charakteristischen Eigenschaften der fünf Keybag-Typen erläutert.

User keybag: Enthält die verschlüsselten Klassen-Schlüssel, die bei normalen Dateioperationen benutzt werden, wie beispielsweise einen Klassen-Schlüssel der Klasse *Complete-Protection*. Bei Eingabe des Passcodes wird dann ein solcher Klassen-Schlüssel aus dem *user keybag* geladen und in der Enklave entschlüsselt. Die Datenstruktur wird ihrerseits in einer Datei im verschlüsselten Dateisystem gespeichert und besitzt das Klassenattribut *No-Protection*.

Device keybag: Enthält die verschlüsselten Klassen-Schlüssel, die bei Dateioperationen benutzt werden, die gerätespezifische Daten betreffen. Dieser Typ wird bei Geräten benötigt, die von mehreren Nutzern gemeinsam genutzt werden. Im Einzelnutzerbetrieb, das ist der Normalfall, sind *user keybag* und *device keybag* gleich.

Backup keybag: eine solche Datenstruktur wird erzeugt, wenn iTunes ein verschlüsseltes Backup erstellt. Diese Datenstruktur wird auf dem System, auf dem das Backup gehalten wird, gespeichert. Für das Backup

⁴ Falls die biometrische Authentisierung mittels Touch ID aktiviert ist, wird mit dem Klassenschlüssel wie weiter oben beschrieben verfahren.

werden neue Schlüssel generiert, die Klassen-Schlüssel der zu sichern-Daten werden mit den neuen Schlüsseln verschlüsselt und diese werden in dem *backup keybag* gespeichert. Der Zugriff auf die Datenstruktur *backup keybag* erfordert die Eingabe eines Passwortes, das der Nutzer in iTunes festlegen kann. Auf diese Weise ist es möglich, ein verschlüsseltes Backup auch auf anderen iOS-Geräten wiederherzustellen, da für die Entschlüsselung nicht der UID-Geräte-Schlüssel erforderlich ist. Dies gilt jedoch nicht für solche Einträge im Backup, die mit dem Attribut *This Device Only* verschlüsselt wurden (siehe unten); diese Einträge können nur wieder auf dem ursprünglichen Gerät wiederhergestellt werden.

Falls der Nutzer ein unverschlüsseltes Backup initiiert, werden die Daten unverschlüsselt, unabhängig von ihrer spezifizierten Schutzklasse, gespeichert. Es wird lediglich sichergestellt, dass die Daten nur auf dem Gerät wiederhergestellt werden können (durch Nutzen der Geräte UID), von dem sie gesichert wurden.

Escrow keybag: Dieses Konzept erlaubt es iTunes, ein Backup zu erstellen oder es erlaubt, ein Device-Management durchzuführen, ohne dass der Nutzer seinen Passcode eingeben muss. Der *escrow keybag* ist auf dem Rechner gespeichert, der für eine Synchronisation mit iTunes genutzt wird. Bei der erstmaligen Verbindung eines mit dem Passcode-Konzept geschützten Geräts zu iTunes wird der Passcode erfragt und es wird ein *escrow keybag* generiert. Dieser enthält alle Klassen-Schlüssel des Geräts, und der keybag wird mit einem neu generierten Schlüssel verschlüsselt. Dieser Schlüssel wird in einer Datei auf dem Gerät gespeichert, die die Klasse *Protected-Until-First-User-Authentication* besitzt, also den Passcode erfordert, um zugegriffen zu werden. Soll ein Backup mit iTunes nach einem Re-Boot erfolgen, ist der Passcode erforderlich, um die Daten im *escrow keybag* zugreifbar zu machen.

iCloud Backup keybag: Diese Datenstruktur ist vergleichbar mit der *backup keybag* Datenstruktur, jedoch enthält sie nur asymmetrische Klassen-schlüssel der Klasse *Protected-Unless-Open*, so dass iCloud-Backups im Hintergrund durchgeführt werden können. Beim Backup werden die ver-schlüsselten Daten, außer den Daten der Klasse *No Protect*, vom Gerät in der iCloud gespeichert, und die Klassen-Schlüssel werden durch iCloud-Schlüssel geschützt. Wie bei einem unverschlüsselten iTunes-Backup wird eine Bindung der gesicherten Daten an die UID des Geräts vor-genommen.

13.1.8 Keychain

keychain

Die Keychain ist eine Datenbank unter anderem für Nutzerpassworte, Schlüssel, Zugangstoken. Sie ist implementiert als SQLite-Datenbank und im verschlüsselten Dateisystem gespeichert. Der Zugriff auf die Einträge in der Datenbank wird durch den Security-Dämon *securityd* kontrolliert. Die Zugriffsberechtigung hängt unter anderem von der *Keychain-access-group* ab, so dass Daten aus der Datenbank über diese Gruppenzugehörigkeit auch zwischen Apps gemeinsam genutzt werden können, wobei aber die gemeinsame Nutzung nur zwischen Apps des gleichen Entwicklers zulässig ist.

Analog zum Dateischutzkonzept basierend auf den Klassenattributen, werden auch die Einträge der Keychain durch ein eigenes Klassenkonzept mit eigenen Klassen-Schlüsseln geschützt. So können Einträge beispielsweise mit Attributen versehen werden, die festlegen, dass auf den Eintrag nur zugegriffen werden kann, wenn das Gerät entblockiert ist. Einträge in der Keychain-Datenbank können mittels ACLs geschützt werden, so dass beispielsweise gefordert werden kann, dass der Nutzer den Passcode eingegeben muss, oder dass der Nutzer seine Präsenz über den Touch ID-Mechanismus nachweisen muss. ACLs werden in der Secure Enclave ausgewertet.

Nachfolgend werden einige Beispiele für Einträge, die von iOS erzeugt werden, angegeben. Für jeden Eintrag ist diejenige Zugriffsklasse aufgeführt, deren Zugriffsanforderungen beim Zugriff auf die Einträge zu erfüllen sind. Das Attribut *non-migratory* besagt, dass der Eintrag nicht auf ein anderes Gerät migriert werden darf.

Eintrag	Zugriffsschutz
WiFi Passworte	After First Unlock
Mail-Konten	After First Unlock
Exchange Konten	After First Unlock
VPN Passworte	After First Unlock
Zugangstoken für soziale Netzwerke	After First Unlock
iCloud Token	After First Unlock
Find my Phone Token	Always
Sprachnachricht	Always
iTunes Backup	When unlocked, non migratory
Safari Passworte	When unlocked
VPN Zertifikate	Always, non migratory
SIM PIN	Always, non migratory

13.1.9 App-Sicherheit

iOS führt auf verschiedenen Ebenen Maßnahmen durch, um Risiken durch mögliche Angriffe, die sich durch manipulierte und infizierte Apps, die auf das Gerät geladen und ausgeführt werden, ergeben könnten, zu reduzieren. Dazu gehören Maßnahmen vor der Ausführung der App, sowie solche, die zur Ausführungszeit greifen.

Code-Review und Code-Signaturen:

Bevor eine App über den App Store zum Download verfügbar gemacht wird, wird ein App-Code-Review durch Apple durchgeführt. Mit dem Review wird geprüft, ob sich die App entsprechend ihrer Funktionalität verhält. Unter <https://developer.apple.com/app-store/review/guidelines/2016-06-13/> findet man Aussagen dazu, welches App-Verhalten zu einer Ablehnung in Bezug auf die Aufnahme in den App Store führen kann. Dazu zählt unter anderem das Vorhandensein versteckter Features in der App, Apps, die ausführbaren Code installieren oder starten, aber (natürlich) auch Apps, die persönliche Angriffe beinhalten, wie Verleumdungen, oder auch Apps, die eine mögliche Verletzung der Privatsphäre bewirken können.

Code-Review

Ausführbarer App-Code muss mit einem privaten Schlüssel signiert sein (Code-Signatur), zu dessen öffentlichem Schlüssel es ein von Apple ausgestelltes, gültiges Zertifikat gibt. Entwickler, die signierte Apps erstellen wollen, müssen sich im iOS-Developer-Programm registrieren. Apple prüft dann ihre reale Identität und stellt ein Zertifikat aus.

Code-Signatur

Bevor der Code auf dem Gerät zur Ausführung gebracht wird, erfolgt eine Signaturprüfung. Seit der Version iOS 8 ermöglicht iOS, dass Apps Code von Dritten in ihren jeweiligen Adressraum nachladen können. iOS validiert auch alle Code-Signaturen von nachgeladenen, dynamisch gebundenen Bibliotheken, um die Risiken für das Einschleusen von Malware zu reduzieren.

Schutzmaßnahmen zur Laufzeit:

Alle Apps, die von Dritten erstellt und auf dem Gerät ausgeführt werden, laufen in einer Sandbox ab. Das verhindert den unkontrollierten Zugriff auf Dateien anderer Apps und auch Manipulationszugriffe auf Systemressourcen. Jede App erhält ein eigenes Home-Verzeichnis für ihre Dateien. iOS definiert APIs und Dienste, um einen Datenaustausch zwischen Apps zu ermöglichen. So kann eine App über das *Extension*-Konzept anderen Apps Funktionen zur Verfügung stellen, die aber nicht in einem gemeinsamen Adressraum ausgeführt werden. Ein anderes Konzept ist das der *Entitlements*. Diese sind vergleichbar mit Android-Permissions. Bei der Entwicklung einer App gibt der Entwickler an, welche Zugriffe die App benötigt, wozu sie also erteilt werden muss; das sind die Entitlements. Das iOS generiert aus diesen Entwicklerangaben eine Datenstruktur und signiert sie, so dass sie nicht unbemerkt geändert werden kann.

Sandbox

Systemdateien und Ressourcen werden in der Betriebssystempartition verwaltet und sind ebenfalls von 3rd Party-Apps isoliert. Die Betriebssystempartition ist als read-only-Partition gemountet und lässt damit keinen Schreibzugriff auf die dort gespeicherten Systemdateien und Ressourcen zu. Der überwiegende Teil des Betriebssystems wird jedoch in nicht-privilegierten Modus ausgeführt und ist, wie alle Apps von Drittanbietern auch, dem Nutzerkonto *mobile* zugeordnet.

ASLR

Vorkonfigurierte Apps (built-in) nutzen den Speicherschutz der Address Space Layout Randomisation (ASLR), um sich gegen Buffer-Overflow- oder auch ROP-Angriffe⁵ zu schützen. Auch ausführbarer Code, System-Bibliotheken, dynamischer Loader, der Stack und der Heap sind mittels ASLR geschützt.

non-executable

iOS nutzt auch das XN-Feature des ARM-Prozessors (eXecute Never), mit dem Speicherseiten als nicht ausführbar (non-executable) markiert werden. Seiten, die sowohl als beschreibbar als auch als ausführbar markiert sind, können in Ausnahmefällen und unter Kontrolle des Betriebssystems diese beiden Zugriffsrechte ausüben. So benötigt beispielsweise Safari diese Ausnahmeregelung für seinen JavaScript JIT⁶-Compiler

Durch Einschränkungen in der Funktionalität werden weitere mögliche Angriffsvektoren ausgeschlossen. So ist es nicht möglich, eine Shell zu starten, oder Kommandos wie ls, rm, ps auszuführen.

Fazit

Die Schichtenweise aufgebaute iOS-Sicherheitsarchitektur (vgl. Abbildung 13.3) basiert auf gut durchdachten und gut auf einander abgestimmten Konzepten.

Mit der Secure Enclave wird eine hardwarebasierte, vertrauenswürdige Umgebung für sicherheitsrelevante Operationen geschaffen, so dass alle kryptografischen Operationen in der Secure Enclave stattfinden. Schlüsselmaterial gelangt nie ungeschützt zum Applikations-Prozessor oder zu Apple. Mit dem Effaceable Storage-Bereich wird zudem dafür Sorge getragen, dass zentrale Schlüssel, wie der Dateisystem-Schlüssel, bei Bedarf, beispielsweise beim Wipe, sicher gelöscht werden und keine Daten mehr im Klartext auf dem Gerät zugreifbar sind. Prozesse und Apps werden in einer Sandbox ausgeführt und erhalten initial lediglich Zugriff auf ihr eigenes Heimverzeichnis.

Alle Dateien im iOS-Dateisystem werden verschlüsselt, wobei Datei-individuelle Schlüssel zum Einsatz kommen, und die Daten in der verschlüsselten Form über die Verknüpfung mit der Geräte-UID an das Gerät

⁵ Return Oriented Programming

⁶ Just in time

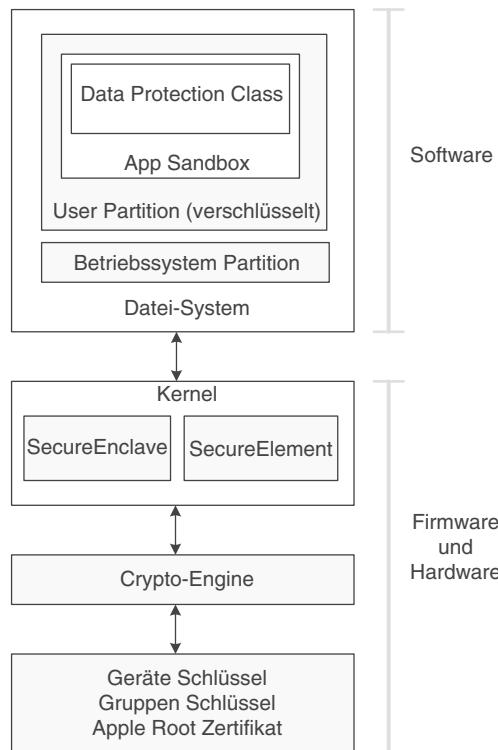


Abbildung 13.3: Sicherheitsarchitektur von iOS

gebunden sind, auf dem sie gespeichert werden. Dateien können unterschiedliche Schutzklassen besitzen, so dass der Zugriff auf die Daten neben der Kenntnis des korrekten Geräteschlüssels auch die Kenntnis bzw. Eingabe des Nutzer-Passcodes erfordern kann. Mit dem Konzept des verschlüsselten Backups ist es möglich, dass Daten des iOS-Dateisystems verschlüsselt auf einem Backup System gespeichert und durch die Umverschlüsselung beim Backup auch auf einem anderen Gerät wiederhergestellt werden können.

Der Ansatz der Geschlossenen Welt (closed world) ermöglicht es, über eine sichere Bootkette nur signierten Code zu laden bzw. auch nur signierte und aktuelle Kernel-Updates aufzuspielen. Dies schützt vor dem Einschleusen von manipuliertem Kernel-Code und vor dem Wiedereinspielen von veralteten Updates. Weiterhin dürfen nur Apps geladen werden, die eine Apple-Signatur besitzen bzw. von einem registrierten App-Entwickler stammen und über den App-Store freigegeben sind. Um aus dieser Closed-World-Umgebung auszubrechen, wird bereits seit den ersten iOS-Versionen von Nutzergruppen kontinuierlich versucht, mittels Jailbreak⁷ Schwachstellen in der Hardware, im iOS-Betriebssystem, unter anderem in der Bootkette,

Jailbreak

⁷ vgl. https://www.theiphonewiki.com/wiki/Jailbreak_Exploits

oder auch im User-Space, unter anderem im Safari-Webbrowser, zu finden und auszunutzen. Das Ziel ist in der Regel, Root-Berechtigungen zu erlangen und Kontrollen, wie das Code-Signing, ASLR oder auch das Sandboxing des iOS auszuschalten, so dass auch eigene Apps geladen werden können. Apple versucht im Gegenzug bei jedem Release die ausgenutzten Schwachstellen zu patchen, so dass ein klassisches Hase-Igel-Rennen stattfindet.

13.1.10 Apple Pay

Apple Pay

Apple Pay ist ein Dienst für mobile, elektronische Zahlungen (mobile Payment) an herkömmlichen Kassenterminals, den Point of Sale (PoS) Terminals oder aber auch über Apps. Die mobile Bezahlung wird so abgewickelt, dass Daten, die die Bezahl-Transaktion betreffen, abgesichert ausgetauscht werden, so dass die beteiligten Parteien jeweils nur die Information erhalten, die sie zur Bearbeitung der Transaktion benötigen (need-to-know). Der Dienst wurde von Apple im September 2014 angekündigt und ab 2015 zunächst in den USA für iPhones, iPads, AppleWatch und Mac eingeführt. Der mobile Bezahl-dienst hat mittlerweile eine weite Verbreitung und Akzeptanz in vielen Ländern gefunden.

Abbildung 13.4 gibt einen groben Überblick über die Komponenten, die auf einem iOS-Gerät bzw. an der Abwicklung von Bezahltransaktionen bei Apple Pay beteiligt sind.

Apple Pay orientiert sich mit seinen Konzepten an der EMV⁸ Payment Tokenisation Specification (vgl. <http://www.emvco.com/specifications>), um möglichst viele der bestehenden Infrastrukturkomponenten und Prozesse bei Händlern nutzen zu können, ohne dass die Händler Modifikationen oder Ergänzungen ihrer Kassenterminals vornehmen müssen. So erfolgt die Kommunikation zwischen einem Apple Pay-Gerät und einem PoS-Terminal in einem Geschäft über NFC (Near Field Communication), und die dabei übermittelten Informationen können ohne weitere Bearbeitung durch den Händler direkt an die Banken zur Prüfung weitergeleitet werden.

Während bei herkömmlichen elektronischen Zahlungsvorgängen Informationen über die genutzten Kreditkarten an den Händler weitergegeben werden und von dort ggf. in falsche Hände gelangen können, wird durch das Konzept der Tokenisation anstelle der Kreditkartendaten nur noch ein Token bei einem Zahlungsvorgang genutzt. Nur die Bank ist in der Lage, das Token der Kreditkarte zuzuordnen, um die eigentliche Transaktion auszuführen. Ein analoges Token-Konzept setzt Apple Pay mit der Nutzung von sogenannten Geräteaccountnummern, den Device Account Numbers (DAN), um. Die DAN wird von der Bank erzeugt, die die Kreditkarte ausgegeben hat, die der Nutzer für die Nutzung mit App Pay im Wallet registriert hat.

DAN

⁸ Europay International, MasterCard und VISA

Die DAN dient als Pseudonym für die reale Kartennummer. Dieses Pseudonym wird in iOS-Geräten abgelegt und bei Zahlungstransaktionen an die Händler weitergegeben. Auf diese Weise liegen beim Händler keine Kreditkartennummern aus App Pay-Transaktionen vor. Für jede Transaktion wird zudem ein individueller Sicherheitscode generiert, der ein Wiedereinspielen von Transaktionen verhindern soll. Die EMV-Spezifikation erfordert zusätzlich eine Maßnahme zur Überprüfung der Authentizität des Kredit- oder Geldkartenbesitzers (Consumer Device Cardholder Verification Method (CDCVM)). Beim Apple Pay-Dienst wird dies durch eine Passcode- oder Touch ID-basierte Authentisierung umgesetzt.

Komponenten des Apple Pay-Dienstes

Der Apple Pay-Dienst umfasst ein Secure Element, die Secure Enclave, einen NFC-Controller, Apple Pay-Server und Bank-Server (vgl. Abbildung 13.4), auf deren Funktion und Zusammenspiel wir nachfolgend eingehen. Wir beschränken uns dabei auf die Erläuterung der Bezahlabläufe über

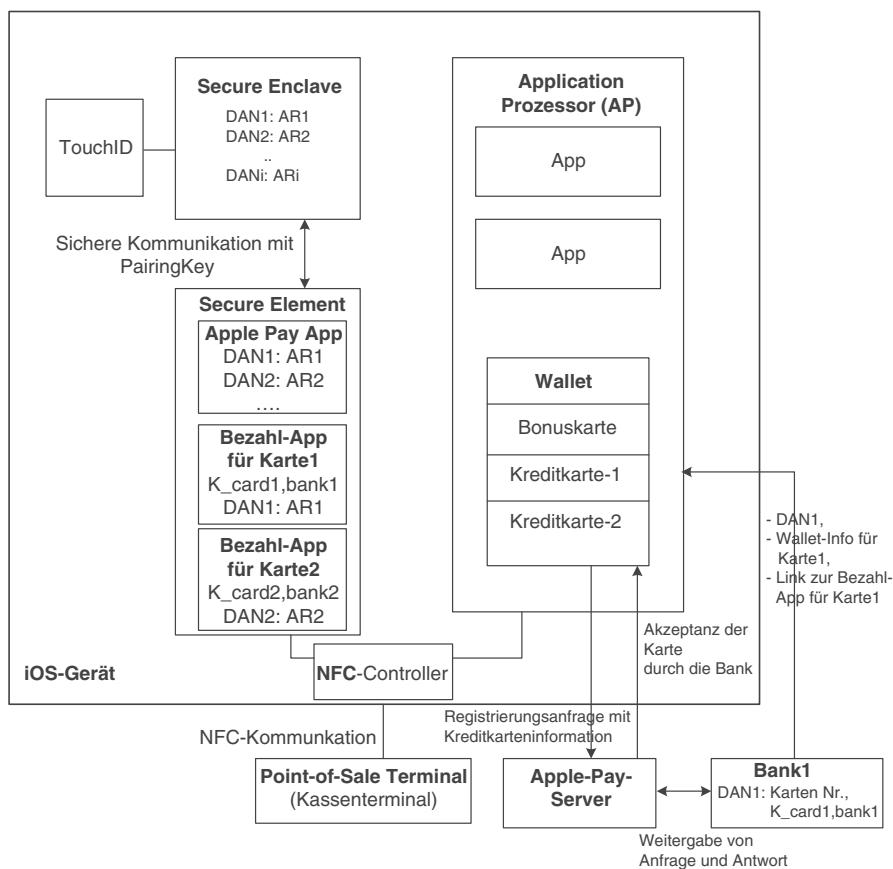


Abbildung 13.4: Überblick über die Komponenten bei Apple Pay

NFC an einem Kassenterminal in einem Geschäft. Für die Darstellung der Bezahlabläufe direkt aus einer App (in-App) oder über den Webbrower sei auf das iOS-Whitepaper⁹ verwiesen.

Ein vergrößerter Ablauf, der darstellt, wie der Nutzer eine Kreditkarte im Wallet seines iOS-Gerätes registriert ist in den Abbildungen 13.5 und 13.6 skizziert. Abbildung 13.5 stellt dabei den Ablauf bei der Registrierung einer Kreditkarte im Wallet des Nutzers auf dem iOS-Gerät dar.

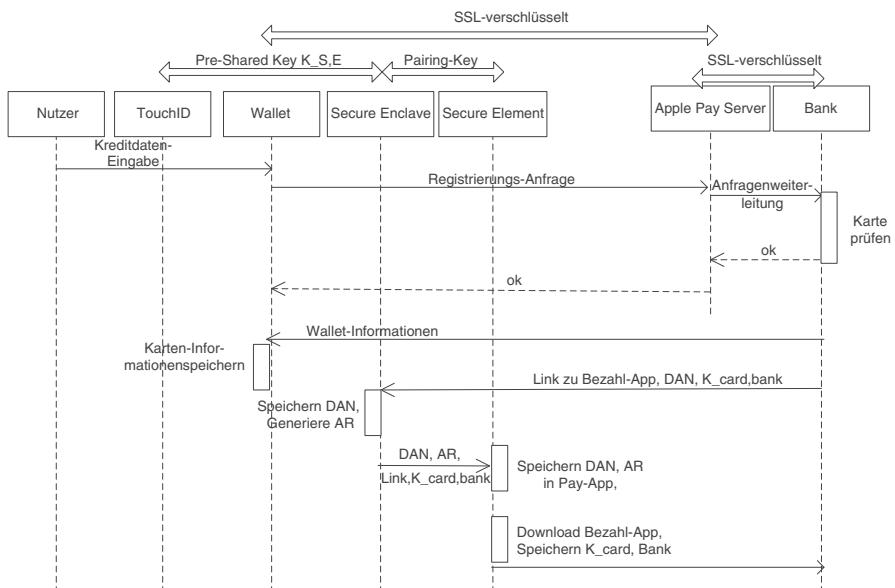


Abbildung 13.5: Ablauf einer Kartenregistrierung bei Apple Pay

Eine NFC-basierte mobile Bezahlung an einem PoS ist in der Abbildung 13.6 skizziert. In der folgenden Beschreibung wird der Ablauf erläutert.

Wallet

Die elektronische Geldbörse, Wallet (früher Passbook genannt), ist eine Applikation unter iOS, die verwendet wird, um Kredit-, Debit-, Kunden- und Bonuskarten hinzuzufügen und zu verwalten und um Zahlungen mit Apple Pay abzuwickeln. Der Benutzer kann sich im Wallet seine Karten und Informationen zu den Karten, wie die letzten Zahlungen, anzeigen lassen.

Secure Element

Das Secure Element ist ein zertifizierter Chip, auf dem die Java Card Middleware ausgeführt wird, die die Anforderungen an elektronische Zahlungssysteme der Finanzindustrie wie den EMV-Standard erfüllt. Im Secure Element wird eine spezifische App, die *Apple Pay App*, ausgeführt, um Apple Pay durchzuführen. Daneben enthält das Secure Element zertifizierte

⁹ vgl. https://www.apple.com/business/docs/iOS_Security_Guide.pdf

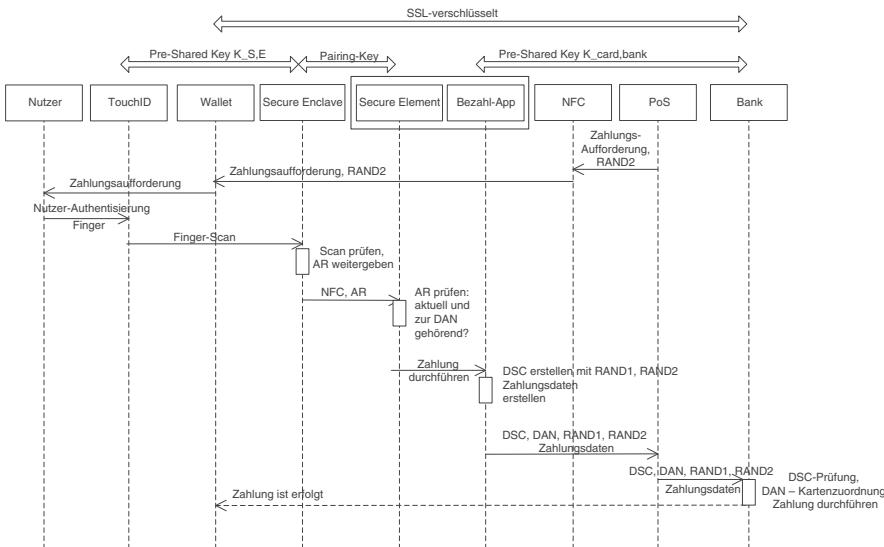


Abbildung 13.6: Ablauf einer Bezahlung mit Apple Pay am PoS

Bezahl-Apps (Payment Apps) von den Banken, die die Kreditkarten ausgestellt haben, die der Nutzer für Apple Pay auf dem Gerät registriert hat.

Wenn der Nutzer eine Kredit-, Debit- oder Prepaidkarte seiner elektronischen Geldbörse, dem Wallet, auf dem Gerät hinzufügt und für die Nutzung in Apple Pay registrieren möchte, wird eine Anfrage mit den Informationen über die zu registrierende Karte SSL-verschlüsselt an einen Apple Pay-Server gesendet. Der Server entschlüsselt die Daten und verschlüsselt sie neu mit einem Schlüssel, der nur der Bank des Kunden oder einem autorisierten Dienstleister bekannt ist. Zusammen mit Informationen über das Gerät, wie die Telefonnummer, und über die Accountaktivität des Nutzers, wie die Anzahl der iTunes-Transaktionen, werden die verschlüsselten Daten an die Bank des Nutzers gesandt, um ein Token als Karten-Pseudonym, nämlich eine Geräteaccountnummer, ausstellen zu lassen. Die Kommunikation erfolgt SSL-verschlüsselt.

Apple Pay-Server

Apple Pay-Server sind auch zuständig für die Abwicklung von Bezahlungen innerhalb von Apps, jedoch werden dabei keine auf den Nutzer zurückführbare Transaktionsinformationen auf dem Server gespeichert und auch keine Informationen darüber, was der Benutzer gekauft hat.

Wenn die Bank des Nutzers von einem Apple Pay-Server um die Freigabe der Kreditkarte zur Nutzung für Apple Pay gebeten wird, führt die Bank eine Prüfung durch. Falls die Kreditkarte in Ordnung ist, generiert die Bank ein Token für das iOS-Gerät, also die bereits erwähnte Geräteaccountnummer, und erzeugt einen für das iOS-Gerät personalisierten, gemeinsamen Schlüssel $K_{card,bank}$.

Bank

Registrierung

Über das *Link and Provisioning*-Kommando von Apple Pay initiiert die Bank das Herunterladen der bereitgestellten Information zur Kreditkarte in das Wallet, die Installation der Bezahl-App mit assoziiertem personalisiertem Schlüssel $K_{card,bank}$ im Secure Element und die Speicherung der Geräteaccountnummer als Pseudonym für die eigentliche Kreditkartennummer, sowohl in der Secure Enclave als auch im Secure Element. Der Schlüssel $K_{card,bank}$ dient bei der Abwicklung von Zahlungstransaktionen zum sicheren Austausch von Zahlungsdaten zwischen dem Secure Element des Geräts und der Bank. Auf diese Weise wird sichergestellt, dass auch der Apple-Pay-Server keine Informationen über durchgeführte Transaktionen erhält.

Die Secure Enclave generiert bei der Kartenregistrierung zusätzlich eine Zufallszahl, den Authorization Random-Wert (AR-Wert), der mit dem Pairing-Schlüssel (siehe unten) verschlüsselt und im Secure Element gespeichert wird.

Secure Enclave

Soll eine Zahlung durchgeführt werden, muss diese zunächst über die Eingabe des Passcodes oder mittels Touch ID autorisiert werden. Die Überprüfung der Authentizität erfolgt über die Secure Enclave, die ihrerseits über die serielle Schnittstelle mit dem Secure Element kommuniziert. Für den sicheren Datenaustausch wird bei der Geräteherstellung ein Pairing-Schlüssel in der Enclave generiert, der aus der Geräte-UID sowie der Identität des Secure Elements abgeleitet ist, und bei der Geräteherstellung sicher im Secure Element abgelegt wird. Hat der Nutzer die Bezahltransaktion autorisiert, generiert die Secure Enclave eine mit dem Pairing-Schlüssel verschlüsselte Nachricht für das Secure Element, in der die Art der Transaktion (kontaktlos oder über eine App) und der Authorization Random-Wert (AR-Wert) der genutzten Kreditkarte enthalten sind.

AR

Das Secure Element entschlüsselt die Autorisierungsnachricht der Secure Enclave und prüft den AR-Wert, bevor es den Bezahlvorgang über die zugeordnete Bezahl-App der Kreditkarte anstößt. Der AR-Wert, der ja auch in der Secure Enclave gespeichert ist, erlaubt eine Kreditkarte im Wallet zu sperren, indem der zugeordnete AR-Wert in der Enclave als ungültig markiert wird, beispielsweise wenn der Passcode deaktiviert wird, oder wenn das Gerät vom Recovery Modus wiederhergestellt wird. Die Secure Enclave kann dann keine gültigen Zahlungsautorisierungen mehr ausstellen.

Bezahl-App

Die zuständige Bezahl-App im Secure Element wickelt die Transaktion ab, indem jeder Bezahltransaktion ein transaktionsspezifischer Sicherheitscode und die Geräteaccountnummer zugeordnet wird. Der einmal nutzbare, transaktionsspezifische Sicherheitscode (dynamic security code (DSC)) wird aus dem Wert des aktuellen Transaktionszählers (ta-counter), einer Zufallszahl RAND1, die von der Bezahl-App generiert wird, einer Zufallszahl RAND2, die vom PoS-Terminal (bei einer NFC-Transaktion) oder dem Apple Pay

Server (bei einer In-App-Transaktion) generiert wird, aus dem personalisierten, gemeinsamen Schlüssel $K_{card,bank}$, sowie aus ggf. noch weiteren Daten, abhängig von der involvierten Bank, abgeleitet:

$$\text{DSC} = \text{KDF}(\text{ta-counter}, \text{RAND1}, \text{RAND2}, K_{card,bank}, \text{data}).$$

Die Geräteaccountnummer, der Sicherheitscode DSC und Informationen, die für die Transaktion erforderlich sind, werden via NFC an das PoS-Terminal des Geschäfts gesendet. Vor der Genehmigung der Zahlung kann die Bank die Zahlungsdaten überprüfen, indem der Sicherheitscode DSC geprüft wird, das heißt indem überprüft wird, ob er frisch ist, also ob die aktuellen Zufallszahlen und der aktuelle Zählerstand verwendet wurden, und ob der Code mit dem korrekten Gerät verknüpft ist, was durch die korrekte Nutzung des personalisierten Schlüssels $K_{card,bank}$ nachgewiesen wird.

Über den NFC-Controller erfolgt die drahtlose Kommunikation zwischen dem Application-Prozessor (AP) und dem Secure Element, bzw. zwischen dem Secure Element und dem PoS-Terminal. Nur Zahlungsanfragen von Terminals in unmittelbarer Nähe des Apple Pay-Geräts werden vom NFC-Controller weiter bearbeitet.

NFC

Fazit

Apple Pay stellt eine mobile, elektronische Bezahlösung zur Verfügung, die mittlerweile weltweit von vielen Banken und Geschäften akzeptiert und genutzt wird. Das umgesetzte Sicherheitskonzept versucht konsequent das Paradigma des need-to-know zu beherzigen. Da bei der Apple Pay-Bezahlung mittels NFC an PoS-Terminals keine sensiven Kreditkarteninformationen an das Terminal transferiert werden, Pseudonyme statt der eigentlichen Kreditkartennummer genutzt werden, sensitive Daten nur verschlüsselt übertragen werden, und die Bezahlung explizit durch den Nutzer autorisiert werden muss, was durch die Secure Enclave als vertrauenswürdigen Ko-Prozessor geprüft wird, bietet Apple Pay eine solide, sichere und zudem einfach nutzbare mobile Bezahlösung. Apple Pay bietet mit seinen Sicherheitskonzepten ein deutlich höheres Sicherheitsniveau als dies bei der Bezahlung mittels herkömmlicher Kreditkarten erreicht wird.

13.1.11 HomeKit-Framework

Das HomeKit-Framework definiert eine Heiamtomatics-Infrastruktur, die es ermöglicht, Zubehörkomponenten (genannt accessories) herstellerübergreifend sicher zu vernetzen. Die entsprechenden Komponenten, wie Lichtschalter, Kameras, Türschlösser oder auch Thermostate lassen sich mit der Home-App auf einem iOS-Gerät steuern und überwachen. Komponenten, die über HomeKit vernetzt werden sollen, müssen von Herstellern

HomeKit

stammen, die ein MFI¹⁰-Zertifikat von Apple und einen speziellen Apple-Microchip besitzen. Das MFI-Zertifikat bestätigt, dass die Komponente die Anforderungen von Apple erfüllt. Um Komponenten über die Home-App zu steuern, müssen in einem ersten Schritt die Komponente und das iOS-Gerät gekoppelt (pairing) werden. Die dafür erforderliche Kommunikation findet über WLAN oder Bluetooth LE statt.

Pairing:

HomeKit-Identität

Sowohl das iOS-Gerät, im Folgenden A genannt, als auch jede zu koppelnde Komponente generieren eine so genannte HomeKit-Identität. Dabei handelt es sich um ein Ed25519 EC-Schlüsselpaar (e, d) bestehend aus einem öffentlichen Schlüssel e und dem zugehörigen privaten Schlüssel d . Auf dem iOS-Gerät A wird dessen Schlüsselpaar (e^A, d^A) in der Keychain geschützt abgelegt.

SRP

Gegeben sei eine Zubehörkomponente, beispielsweise ein Lichtsensor L, der mit dem iOS-Gerät A des Nutzers gekoppelt werden soll. Dafür authentisieren sich die beteiligten Geräte A und L. Zur Geräteauthentisierung wird das *Secure Remote Pairing Protokoll* (SRP) (RFC2945) verwendet, mit dem unter Nutzung des DH-Verfahrens ein gemeinsames Geheimnis s berechnet wird. Die Basis dafür bilden die öffentlichen Schlüssel e^A bzw. e^L der HomeKit-Identitäten der zu koppelnden Geräte. Diese öffentlichen Schlüssel werden ausgetauscht. Um Man-in-the Middle-Angriffe auszuschließen, muss die Authentizität der ausgetauschten Schlüssel geprüft werden. Da eine PKI mit X.509-Zertifikaten für viele IoT-Szenarien zu aufwändig wäre, wird beim SRP-Protokoll ein Passwort-basierter Authentisierungsansatz genutzt. Das erforderliche Passwort ist beim HomeKit-Framework ein 8-stelliger Code. Dieser Code wird vom Hersteller der Zubehörkomponente erzeugt, in der Komponente gespeichert und zusätzlich auf der Komponente angebracht oder zum Beispiel im Handbuch der Komponente hinterlegt. Der Nutzer muss diesen Code an seinem iOS-Gerät A eingeben bzw. über die Kamera einscannen. Der Pairing-Ablauf ist in Abbildung 13.7 skizziert und wird nachfolgend erläutert.

Schritt 1: Beide Geräte tauschen ihre öffentlichen Schlüssel aus.

Schritt 2: Beide Geräte berechnen das gemeinsame DH-Geheimnis s .

Schritt 3: Der Nutzer gibt an dem iOS-Gerät A den 8-stelligen Code des Herstellers für die zu koppelnde Komponente L ein.

Schritt 4: Das iOS-Gerät A verschlüsselt diesen Code mit einem aus s abgeleiteten Schlüssel K , $K=\text{HKDF-SHA-512}(s)$. Zur Verschlüsselung wird das Verfahren ChaCha20-Poly1305 (vgl. RFC 7905) verwendet.

¹⁰ Made for iPhone, iPad

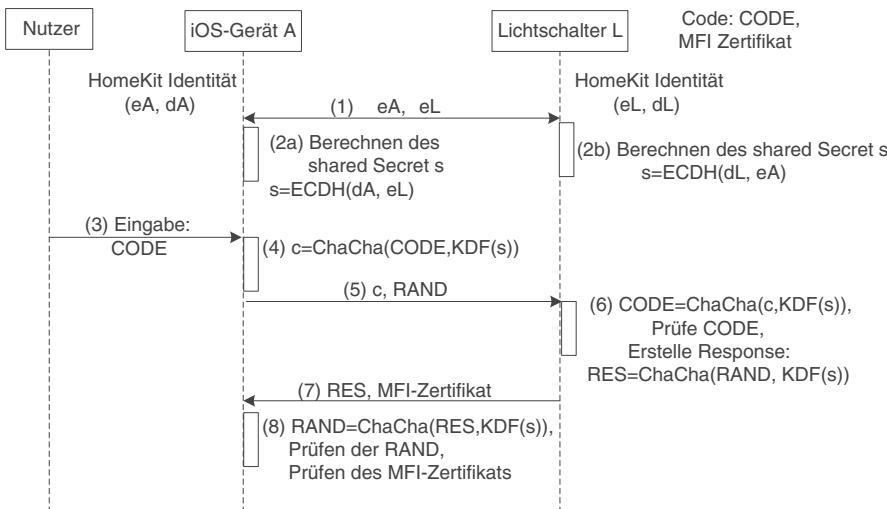


Abbildung 13.7: Pairing einer Zubehörkomponente L mit einem iOS-Gerät A

Schritt 5: Gerät A sendet den Chiffretext zusammen mit einer Challenge RAND an die Komponente L.

Schritt 6: Die Komponente berechnet ihrerseits den Schlüssel K aus dem gemeinsamen DH-Geheimnis s , entschlüsselt den Code, vergleicht ihn mit dem eigenen, gespeicherten Code und akzeptiert das Gerät A als authentisch, wenn der Code übereinstimmt. Durch die Eingabe des Codes am iOS-Gerät ist sichergestellt, dass ein Man-in-the-Middle, der seinen öffentlichen Schlüssel eingeschleust hat, aufgedeckt wird.

Die Komponente L berechnet dann unter Nachweis der Kenntnis des gemeinsamen Geheimnisses s die Antwort RES.

Schritt 7: L sendet die Antwort RES zusammen mit dem MFI-Zertifikat an A zur Prüfung zurück.

Schritt 8: Das Gerät A prüft die Aktualität der Antwort sowie das Zertifikat. Nach Abschluss des Pairings kann eine gesicherte Kommunikation aufgebaut werden.

Kommunikation zwischen iOS-Gerät A und Komponente L

Für die Kommunikation zwischen dem iOS-Gerät A und der gekoppelten Komponente L wird eine sichere Session unter Nutzung des *Station-to-Station-Protokolls* (STS) [57] aufgebaut. Dazu erzeugen sich die Kommunikationspartner A und L für jede Session ein temporäres EC-Schlüsselpaar (e_s^A, d_s^A) bzw. (e_s^L, d_s^L).

Sichere Session

Die jeweiligen öffentlichen Schlüssel (e_s^A) und (e_s^L) werden ausgetauscht und mit den privaten Schlüsseln (d_s^A) bzw. (d_s^L) der HomeKit-Identität signiert:

$Sig_A = EC((e_s^A, d^A)$, bzw. $Sig_L = EC((e_s^L, d^L)$. Beide Parteien berechnen mittels ECDH ein gemeinsames Geheimnis sc , aus dem der gemeinsame Sitzungsschlüssel $K_{A,L}$ abgeleitet wird:
 $K_{A,L} = \text{HKDF-SHA-512}(sc)$.

Nutzen der HomeKit-Infrastruktur

HomeKit ermöglicht die direkte Interaktion zwischen einem iOS-Gerät und einer gekoppelten Zubehörkomponente. Es können darüber hinaus *Räume* und *Szenen* definiert werden, womit sich mehrere Sensoren über ein Kommando ansprechen lassen. So kann man einen Raum *Schlafzimmer* definieren und mit einem Kommando, wie beispielsweise dem Kommando *Abendstimmung*, alle Leuchten in dem Raum gleichzeitig dimmen. Aktivitäten von gekoppelten Komponenten kann man zudem auch in sogenannten *Szenen* konfigurieren. Beispielsweise kann eine Szene *Verlasse-Haus* konfiguriert sein, so dass beim Absetzen des Kommandos *Verlasse-Haus* automatisch die konfigurierten Ereignisse bei den eingebundenen Zubehörkomponenten ausgelöst werden. Das Aktivieren der Szene könnte beispielsweise das Ausschalten des Lichts in allen Räumen, das Herunterregeln der Heizung und das automatisierte Schließen der Fenster und Türen anstoßen, das heißt die Home-App des iOS-Geräts versendet entsprechende Kommandos an die verbundenen Zubehörkomponenten. Durch die Einbindung der Sprachsteuerung Siri können Kommandos auch als Sprachbefehle abgesetzt werden. Die Aktivierung sicherheitskritischer Aktivitäten, wie das Öffnen eines Türschlosses, kann zusätzlich die Eingabe des Passcodes oder des Fingerabdrucks am Touch ID des iOS-Geräts erfordern.

HomeKit-Daten

Daten aus dem HomeKit-Framework eines Nutzers, dazu gehören Daten zu gekoppelten Zubehörkomponenten, Home-Kit-Identitäten, aber auch Raum- bzw. Szenen-Beschreibungen, werden verschlüsselt auf dem iOS-Gerät des Nutzers abgelegt, wofür ein Schlüssel aus der HomeKit-Identität des Nutzers und aus einer Nonce abgeleitet wird. Die Daten werden zudem in einer Datei mit dem Attribut *Protected-Until-First-User-Authentication* verwaltet, so dass zusätzlich auch noch der Passcode des Nutzers erforderlich ist, um auf die Daten zuzugreifen. HomeKit-Daten dürfen nur in verschlüsselten Backups extrahiert werden.

mehrere Nutzer

Die HomeKit-Infrastruktur kann auch von verschiedenen Nutzern gemeinsam genutzt werden. Dazu verteilt der Master-Nutzer, das ist der Nutzer mit der Berechtigung, weitere Nutzer hinzuzufügen, die öffentlichen Schlüssel der HomeKit-Identitäten der anderen Nutzer an die Zubehörkomponenten. Diese sind dann auch berechtigt, Kommandos an die Zubehörkomponenten zu senden. Die dafür erforderliche sichere Verbindung wird in der üblichen Weise mit dem STS-Protokoll aufgebaut.

Zugriff über das Internet

Die HomeKit-Infrastruktur ermöglicht auch die Steuerung von Komponenten über das Internet. Dafür wird im heimischen Netz eine Hub-Komponente benötigt, die als Proxy für das iOS-Gerät des Nutzers fungiert und die Kommandos an die gekoppelten Komponenten absetzt. Ein solcher Hub kann beispielsweise ein AppleTV oder ein iPad sein. Damit das Hub-Gerät im Auftrag des Nutzers aktiv werden kann, erhält es das Schlüsselpaar der HomeKit-Identität des Nutzers. Diese Schlüssel werden über einen sicheren Kanal vom iOS-Gerät zum Hub-Gerät übertragen. Der Kanal wird als eine sichere Session nach dem gleichen Vorgehen wie oben beschrieben etabliert. Das heißt, dass auch ein AppleTV oder iPad eine HomeKit-Identität erzeugt und damit ein Pairing mit dem Gerät des Nutzers durchführt. Anschließend wird eine sichere Session mittels des STS-Protokolls aufgebaut.

Hub-Gerät

Ein Fernzugriff auf die Zubehörkomponenten kann auch über die iCloud erfolgen. Entweder haben die Zubehörkomponenten selbst einen Zugriff auf die iCloud, oder der Nutzer installiert in der Heimautomatisierungsumgebung einen iPad, der mit der iCloud interagiert und als Hub dient.

iCloud

13.2 Windows 10

Windows 10 [183] ist ein 32-Bit und 64-Bit Multi-Programming, Multi-Threading Betriebssystem. Windows 10 führt die verschiedenen Windows-Varianten für PCs, Spielkonsolen, Smartphones etc. in einem OneCore Betriebssystem-Ansatz zusammen. Das bedeutet, dass die Basisversion des Betriebssystems auf alle oben genannten Plattformen portierbar ist, aber auch auf IoT-Plattformen, wie beispielsweise Raspberry Pi 2.

Windows 10

13.2.1 Architektur-Überblick

Wie Unix/Linux, so unterscheidet auch Windows nur zwei Arbeitsmodi, auch wenn die zugrundeliegende Hardware, wie der x86, vier unterschiedliche Privilegien-Modi unterstützen würde. Windows ist ein Betriebssystem mit hybrider Kernel, in dem die meisten Treiber und Betriebssystem-Dienste im Kernel-Modus im Adressbereich des Betriebssystem-Kerns ausgeführt werden und sich somit den gemeinsamen virtuellen Speicher teilen (vgl. Abbildung 13.8). Dies eröffnet die Gefahr, dass die Komponenten sich wechselseitig kompromittieren, wobei insbesondere die Treiber-Software, die im Kernel-Modus mit hohen Privilegien ausgeführt wird, ein mögliches Sicherheitsrisiko darstellt. Aus diesem Grund muss unter Windows 10 ein Treiber zunächst das Windows Hardware Quality Lab (WHQL) durchlaufen, in dem der Treiber-Code formal evaluiert und signiert wird. Nur signierte und frei gegebene Treiber dürfen auf ein Windows 10 System geladen werden.

Modi

Anwendungsprogramme werden im Benutzermodus (user mode) ausgeführt und besitzen keine Berechtigungen, direkt auf Hardware-Komponenten oder Betriebssystembereiche zuzugreifen.

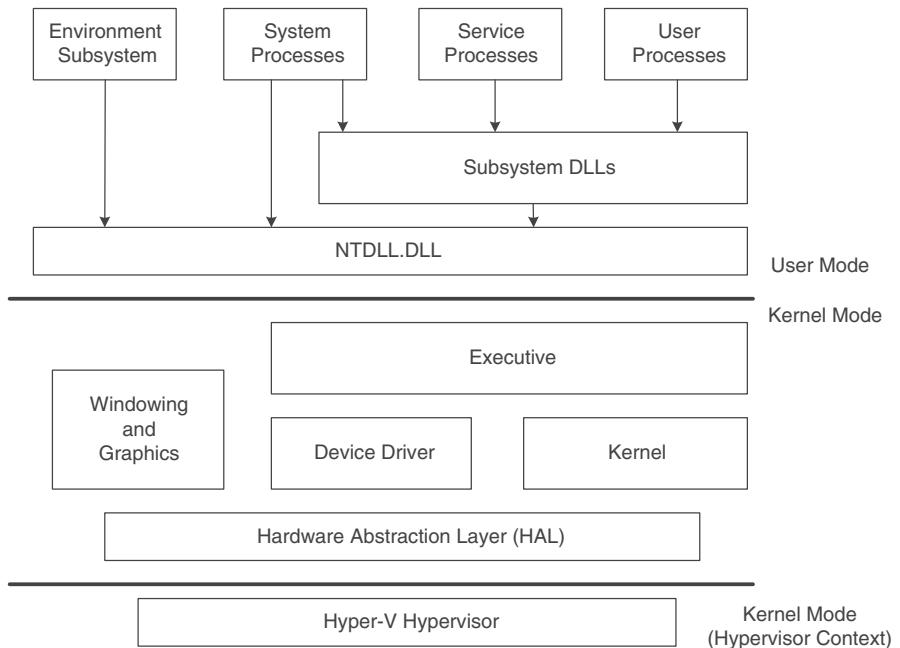


Abbildung 13.8: Überblick über die Windows 10 Betriebssystem-Architektur

Wie dem Architektur-Bild 13.8 zu entnehmen, ist auf der untersten Ebene der Hypervisor angesiedelt, der unter Windows 10 eine Weiterentwicklung von Hyper-V ist und spezielle Sicherheitsdienste, bekannt als *Virtualization-Based Security* (VBS), anbietet.

Virtualization-based Security

VBS

Windows 10 realisiert eine Virtualisierungs-basierte Sicherheitsarchitektur, die eine Isolierung sicherheitskritischer Systemteile ermöglicht. Diese Isolierung wird z.B. für den Local Security Authentication Service (Lsass) (siehe unten) genutzt. Zudem werden orthogonal zu den zwei Arbeitsmodi noch zwei Vertrauenslevel, die virtuellen Trust-Level (VTL) VTL0 und VTL1, umgesetzt. Anwendungs- und Kernel-Code werden im Vertrauenslevel VTL0 (normale Welt) ausgeführt und haben keine Kenntnis von einem weiteren Vertrauenslevel. Komponenten im Vertrauenslevel VTL1 (sichere Welt) besitzen höhere Privilegien und sind vor den Komponenten des Levels VTL0 verborgen und für diese nicht zugreifbar. Damit wird sichergestellt, dass auch Komponenten des Betriebssystemkerns, die im Level VTL0 ab-

laufen, den Hypervisor nicht kompromittieren können, da dieser in dem höheren Vertrauenslevel VTL1 abläuft. Obwohl auch der Hypervisor im Kernel-Modus ausgeführt wird, ist es nur ihm gestattet, spezielle CPU Instruktionen zu nutzen, wie VT-x auf Intel-Prozessoren oder SVM auf AMD. Damit kann der Hypervisor sowohl eine Isolation gegenüber den Kernel-objekten implementieren, als auch seine Monitoringaufgabe wahrnehmen. Schadcode, der im Level VTL0 aktiv ist, kann somit nicht auf Komponenten des Levels VTL1 übergreifen. Level VTL1 setzt auf einem Trusted Boot auf und nutzt Hardware-Unterstützung, um eine vertrauenswürdige Ausführungsumgebung für die Level VTL1-Komponenten anzubieten. Das VBS Subsystem umfasst folgende Komponenten.

VBS-Subsystem

- Der *Device Guard* ermöglicht es, Code-Signing Regelwerke festzulegen, die differenziertere Regeln beschreiben, als das Kernel-Mode-Code-Signing (KMCS) Regelwerk, das mit Windows 8.1 eingeführt wurde.
- Der *Hyper Guard* schützt ausgewählte Kernel- und Hypervisor-Datenstrukturen.
- Der *Credential Guard* schützt vor unautorisiertem Zugriff auf Zugangs-Credentials, wie Passworte oder Kerberos-Tickets.
- Der *Application Guard* realisiert eine gehärtete Sandbox für den Edge-Browser von Microsoft¹¹.
- Der *Host Guardian and Shielded Fabric* nutzt einen virtuellen TPM (vTPM), um eine virtuelle Maschine vor der Plattform, auf der sie ausgeführt wird, zu schützen.

Architektur-Komponenten

Der *Hardware Abstraction Layer* (HAL) enthält die hardwareabhängigen Systemteile und isoliert diese vom Kernel, den Treibern und der Executive. Die Windows *Executive* umfasst die Betriebssystem-Basisfunktionen wie Speichermanagement, Prozess- und Thread-Management, Ein-/Ausgabe, Networking, Interprozesskommunikation und Sicherheit. Der *Windows-Kernel* enthält die low-level Betriebssysteme-Dienste wie das Thread-Scheduling, das Interrupt-Handling oder auch das Synchronisieren von Multiprozessoren. Die *Geräte-Treiber* umfassen sowohl die Hardware-Treiber als auch Treiber für das Dateisystem und das Netzwerk. Das *Windowing und Graphics System* implementiert die grafische Nutzungs-schnittstelle (GUI).

HAL

Im User-Mode sind die Nutzer-, Service, System und Environment Prozesse angesiedelt. *Nutzer-Prozesse* entsprechen Anwendungsprogrammen.

Prozesse

¹¹ Bem.: Der Edge Browser hat den Internet Explorer als Standard Browser abgelöst.

Die *Service-Prozesse* führen, wie der Name schon sagt, Systemdienste aus, unabhängig davon, ob ein Nutzer eingeloggt ist. Drucker-Spooler oder Komponenten des Microsoft SQL-Servers sind Beispiele für solche Prozesse. Die *System-Prozesse* sind eine Anzahl festgelegter Prozesse, wie der Login-Prozess oder der Session-Manager, die nicht vom Service-Control Manager gestartet werden. Die *Environment Subsystem Server* Prozesse implementieren Dienste für den Nutzer aus der Betriebssystem-Umgebung, wie eine erweiterte POSIX-Umgebung namens SUA (Subsystem for Unix-based Applications). Die Prozesse im User Mode dürfen die Windows-Betriebssysteme-Dienste nicht direkt aufrufen, sondern bedienen sich dafür der dynamischen Link Libraries (DLL), die die Aufrufe aus den Anwendungsprogrammen auf die nativen Betriebssystem-Aufrufe abbilden, die wiederum in der *Ntdll.dll* implementiert sind.

Der überwiegende Teil von Windows ist in C, nur wenige Teile sind in C++ programmiert. Assembler wird nur für Systemteile genutzt, die direkt mit der Hardware interagieren, wie das Interrupt-System.

Registry

Registry

Informationen, die zum Booten und Konfigurieren des Systems benötigt werden, wie Informationen über die zugrunde liegende Hardware, Software und die Benutzer, werden in einer zentralen Datenbank, der Registry, abgelegt. Die Registry kann man als ein spezielles Dateisystem auffassen, das Verzeichnisse¹² und sehr kleine Dateien enthält. Das Verzeichnis *HKEY_LOCAL_MACHINE*, im Folgenden mit *HKLM* abgekürzt, ist eines der Wichtigsten. Es enthält fünf Unterverzeichnisse. Das Hardwareverzeichnis beschreibt, welche Treiber welche Hardwarebereiche kontrollieren. Das

SAM

(Security Account Manager) Verzeichnis enthält sicherheitsrelevante Informationen wie Benutzernamen, Gruppen, Passworte, die für einen Systemzugang benötigt werden. Der Zugriff auf diese Daten ist sehr eingeschränkt, so dass nur das Systemkonto, das eine eigene Benutzerkennung darstellt, die entsprechenden Rechte besitzt. Dennoch ist es möglich, über einen Umweg, direkten Zugriff auf die SAM-Daten zu erhalten. Über den Befehl *at.exe* lassen sich zeitgesteuerte Programme starten, was normalerweise für Backup-Zwecke verwendet wird. Über diesen Befehl lässt sich aber auch der Registry-Editor starten, der dann, wie alle Programme, die mit *at.exe* gestartet werden, mit den Rechten des Systemkontos ausgeführt wird. Auf diese Weise erlangt man über den Registry-Editor direkten Zugriff auf die Daten.

Das Security-Verzeichnis enthält allgemeine Informationen über die gelgenden Sicherheitsregelwerke wie die Festlegung der Minimallänge für Passwör-

¹² Verwirrenderweise werden diese Verzeichnisse unter Windows Key genannt.

te oder die Anzahl der möglichen Fehlversuche. Weitere Verzeichnisse sind das Softwareverzeichnis, in dem Softwarehersteller Informationen zum Umgang mit ihren Produkten ablegen können (z.B. welche Treiber zu nutzen sind, oder wie ein Paket wieder zu deinstallieren ist) und das Systemverzeichnis, das alle Informationen zum Booten des Systems (z.B. welche Treiber geladen werden müssen) enthält. Wird das System heruntergefahren, so werden die meisten Einträge in der Registry über atomare Transaktionen auf der Festplatte gespeichert.

Edge Browser

Der neue Standard-Browser Edge bietet einige zusätzliche Sicherheitsmerkmale. So bieten die beiden Technologien (1) Content Security Policy und (2) HTTP Strict Transport Security einen stärkeren Schutz gegen Cross-Site-Scripting-Attacken und informieren die attackierten Webseiten über Angriffsversuche. Durch die Nutzung von isolierenden Containern wird versucht, die Ausbreitung von Schäden zu begrenzen. Zudem wird die Angriffsfläche reduziert, indem verwundbare Technologien wie ActiveX nicht mehr unterstützt werden.

Edge

13.2.2 Sicherheits-Subsystem

Nachfolgend werden die wichtigsten Komponenten und Datenbanken des Windows-Sicherheits-Subsystem (vgl. Abbildung 13.9) kurz vorgestellt.

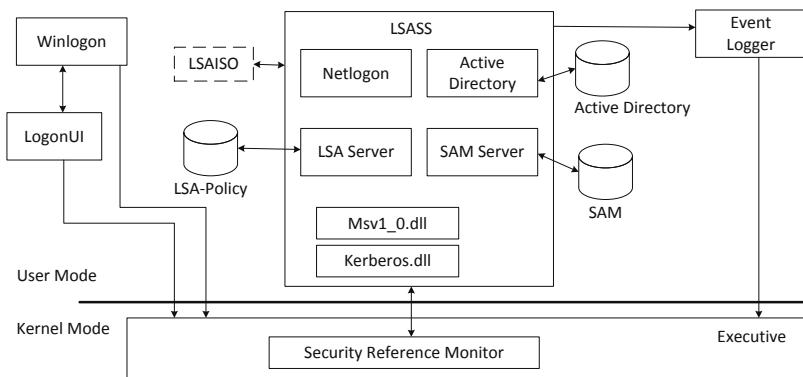


Abbildung 13.9: Windows 10 Sicherheitssubsystem

- Der Local Security Authority Subsystem Service (Lsass) ist ein Prozess im User-Modus, der das Programm *Lsass.exe* ausführt, das die systemlokale Sicherheitsrichtlinie durchsetzt. Diese Richtlinie (policy) umfasst beispielsweise Regeln darüber, welche Nutzer zum Login berechtigt

Lsass

sind, welche Berechtigungen an Nutzer vergeben sind, welche Passwort-Policy gilt, oder auch welche Vorgaben für ein Audit zu erfüllen sind. Der Lsass-Prozess führt die Nutzer-Authentisierung durch und sendet Audit-Nachrichten an den Event Log. Der größte Teil der Aufgaben wird durch eine Bibliotheksfunktion, *Lsassrv.dll*, die der Prozess lädt, implementiert.

- Der Prozess LSAIso.exe, auch unter dem Namen *Credential Guard* bekannt, ist ein isolierter User-Mode Prozess, der als Trustlet im Trust-level VTL1 ausgeführt wird. Der Prozess speichert die Hashwerte von Zugangstoken für Benutzer in seinem Prozessadressraum. Dieser Adressraum ist sogar vor einem Zugriff durch Kernel-Prozesse geschützt. Der Lsass selbst speichert nur einen verschlüsselten Blob der Passwort-Hashes.
- Die *Lsass Policy Datenbank* verwaltet die lokalen Sicherheitsregelwerke und -Konfigurationen. Die Datenbank ist in einem mittels Zugriffskontrolllisten (ACLs) geschützten Bereich der Registry, in **HKLM_Security**, gespeichert.
- Der *Security Account Manager* (SAM) besteht aus einer Menge von Prozeduren, mit denen auf die Datenbankeinträge, die die Benutzerkennungen und Gruppen enthalten, zugegriffen werden kann. Der SAM-Dienst ist in der *Samsrv.dll* implementiert und wird in den Lsass Prozess geladen.
- Die lokalen Benutzerkennungen und Gruppen sowie die Passworte und weitere Attribute werden in der SAM Datenbank in der Registry unter **HKLM_SAM** verwaltet.
- Das Active Directory ist ein Verzeichnisdienst mit einer Datenbank, die Informationen über Objekte einer Netzdomäne enthält. Das Active Directory verwaltet Informationen über die Benutzer und Gruppen in einer Domäne, über deren Passworte sowie über deren Berechtigungen. Hiermit sind allgemeine Rechte wie ein Shut-Down-Recht gemeint, nicht aber Zugriffsrechte zur Nutzung einzelner Dateien. Diese werden über ACLs festgelegt. Jede Domäne kann eine eigene Sicherheitsrichtlinie festlegen. Eine Domäne fasst eine Menge von Rechnern zu einer Verwaltungseinheit zusammen. Ein Active Directory Server wird über die *Ntdsa.dll* implementiert und im Lsass Prozess ausgeführt. Zwischen verschiedenen Windows Domänen wird unter Nutzung des Kerberos-Protokolls (vgl. Abschnitt 10.4.2) eine transitive Vertrauensbeziehung etabliert. Wie alle Objekte von Windows, so sind auch die Objekte im Active Directory über Zugriffskontrollisten vor unautorisierten Zugriffen geschützt.

Active Directory

- *Authentifizierungspakete* enthalten Dynamic Link Libraries (dll), die im Lsass Prozess und im Client Prozess ausgeführt werden und die Authentifizierungs-Policy des Windows-Systems implementieren. Windows verwendet zwei Standard Pakete, nämlich Kerberos und Msv1_0. Msv1_0 ist das Default-Authentifizierungspaket auf einem Stand-alone-System. Zur Authentifikation wird der Benutzername und das Benutzerpasswort benötigt. Msv1_0 berechnet den Hashwert des Passwortes und überprüft den gehaschten Wert mit dem in der SAM-Datenbank gespeicherten. Nach einer erfolgreichen Authentifikation erzeugt das Msv1_0 ein Access Token (s.u.) für den Benutzerprozess.

Msv1_0

Das Kerberos-Protokoll wird von Rechnern in der Windows Domäne zur Authentifikation verwendet. Der Kerberos-Dienst vergleicht den Benutzernamen und das gehashte Passwort mit den Angaben, die über den Benutzer im Active Directory abgelegt sind. Bei einem erfolgreichen Login transferiert der Kerberos-Dienst das Zugangsticket über das Netz zu demjenigen Rechner, an dem der authentifizierte Benutzer das Login durchgeführt hat. Der Kerberos Key Distribution Center (KDC) ist ein Bestandteil des Active Directories. Die unter Windows implementierte Kerberos-Version ist eine Erweiterung der Standardversion 5 und damit proprietär. Sie ermöglicht es, dass in den Tickets auch öffentliche Schlüssel eines asymmetrischen Verfahrens verwendet werden, um auf diese Weise auch eine smartcardbasierte Authentifikation zu unterstützen. Ferner wird im Ticket die Information über die Benutzeridentität und Gruppenzugehörigkeit abgelegt; dies wird von der Zugriffskontrolle benötigt.

Kerberos

- *Winlogon* ist der interaktive Logon-Prozess, der im User-Modus abläuft. Dieser Prozess wird aktiv, wenn ein Benutzer die spezielle Tastenkombination CTRL-ALT-DEL¹³ zum sicheren Einloggen gedrückt hat. Beim sicheren Login ruft der Tastaturtreiber den Winlogon-Systemprozess aufruft, der das korrekte Login auf dem Bildschirm darstellt und die vollständige Kontrolle über den Bildschirm übernimmt. Das sichere Login verhindert, dass ein gefälschtes Login-Programm (z.B. über ein Trojanisches Pferd) die Authentifikation unterläuft.

Winlogon wickelt den interaktiven Dialog zum Login ab und erzeugt eine Shell, wenn sich der Benutzer erfolgreich eingeloggt hat.

- Das Logon User Interface *LogonUI* ist ein Prozess, der im User-Modus abläuft und dem Nutzer eine Schnittstelle zur Eingabe der beim Login geforderten Authentisierungs-Credentials, wie Passworte oder Smartcards anbietet.

¹³ Dies wird auch als SAS (Secure Attention Sequence) bezeichnet.

- *Netlogon* ist ein Dienst, der eine sichere Verbindung zu einem Domänen-Controller aufbaut. Über diese sichere Verbindung werden Anfragen und Antworten eines interaktiven Logins versandt. Der Netlogon-Dienst wird auch für Active Directory Logins verwendet.
- Der Kernel Security Device Driver (KSecDD) ist eine Bibliothek im Kernel-Modus, die eine spezielle Schnittstelle bereitstellt, die von anderen Security-Subsystem-Komponenten des Kernels genutzt wird, um mit dem Lsass Prozess zu kommunizieren, der im User-Modus ausgeführt wird.
- *AppLocker* ist ein Mechanismus, mit dem Systemadministratoren spezifizieren können, welche ausführbaren Dateien, welche DLLs und Skripte von welchen Nutzern und Gruppen ausgeführt werden dürfen.
- Der *Security Reference Monitor* (SRM) ist eine Komponente in der Windows Executive (Ntoskrnl.exe), die im Kernel-Modus ausgeführt wird. Der SRM definiert die Datenstruktur der Access-Token (s.u.), mit der ein Sicherheitskontext festgelegt wird. Der Manager kontrolliert die Zugriffe auf Objekte, verwaltet die Zugriffsrechte von Nutzern und generiert die für ein Security Audit benötigten Nachrichten.

Während der Systeminitialisierung erzeugt der SRM den *SeRmCommand Port*, zu dem der Lsass Prozess eine Verbindung aufbaut. Sobald der Lsass Prozess gestartet wird, erzeugt er seinerseits den *SeLsaCommand Port* mit dem sich der SRM verbindet. Nachdem diese sichere Verbindung etabliert ist, lauscht keiner der beiden Kommunikationsendpunkte mehr an den Ports, so dass ein Angreifer, der versucht eine Verbindung zu einem der beiden Ports aufzubauen, keine Antwort erhält; eine Verbindung wird nicht etabliert.

13.2.3 Datenstrukturen zur Zugriffskontrolle

Identifikation

Jeder Benutzer und jede Gruppe unter Windows wird systemintern eindeutig¹⁴ durch eine SID (Security ID) identifiziert. Die SID ist eine Datenstruktur im Binärformat, die eine variable Menge von Werten umfassen kann. Beim Starten eines Prozesses wird die Benutzer SID an den Prozess vererbt, d.h. jeder Prozess und alle seine Threads laufen unter der ID des Benutzers.

Access Token

Access Token

Nach der erfolgreichen Authentifikation eines Benutzers erzeugt das Sicherheitssubsystem für ihn eine fälschungssichere Datenstruktur, das Access Token, das den Sicherheitskontext eines Prozesses definiert. Das Access Token dient zur Identifikation des zugreifenden Subjekts und zur Bereitstellung

¹⁴ SIDs sollen weltweit eindeutig sein.

von Informationen, die die jeweiligen Objektmanager (z.B. der Prozessmanager für Prozess- und Threadobjekte, das Kommunikationssystem für Ports und der Ein/Ausgabe-Manager für Dateiobjekte) für die Zugriffskontrolle benötigen. Das Token wird automatisch mit jedem Prozess und allen Threads des Benutzers assoziiert und muss bei jedem Zugriff präsentiert werden. Ein Thread kann aber auch ein eigenes Token anfordern, wodurch das Token des Prozesses für den Thread überschrieben wird.

Abbildung 13.10 zeigt ein Beispiel eines Access Tokens. Der Identifikationsteil enthält die Benutzer-SID sowie die SIDs der Gruppen, in denen der Benutzer und damit der ausführende Prozess bzw. Thread Mitglied ist. Im Privilegianteil können spezielle Berechtigungen zum Zugriff auf sicherheitskritische Systemdienste wie zum Beispiel auf den Dienst zur Erzeugung von Zugriffstoken oder zum Shutdown des Rechners angegeben werden. Über den Privilegianteil ist es möglich, die Allmacht eines Superusers zu beschränken, indem individuelle Rechte einzelnen Prozessen zugebilligt werden. Die meisten Benutzer besitzen aber keinerlei solche Privilegien. Die Default-ACL definiert Standardberechtigungen, die automatisch für jedes vom Benutzer erzeugte Objekt vergeben, also in die dem Objekt assoziierte Zugriffskontrollliste (s.u.) eingetragen werden, falls keine anderweitige Zugriffskontrollliste für das Objekt erzeugt wird.

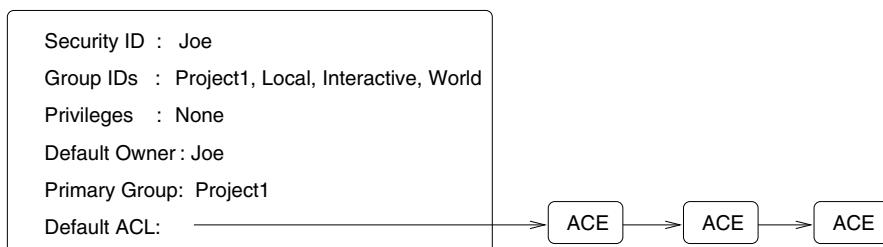


Abbildung 13.10: Windows Access Token

Unter Windows ist es möglich, dass ein Thread sein Access Token an einen Server weiterreicht, so dass dieser auf die geschützten Dateien und Objekte des Threads zugreifen darf. Diesen Vorgang nennt man Impersonation. Über das Impersonation-Konzept kann man auch eingeschränkte Tokens, das so genannte Restricted Tokens, weiterreichen, so dass der Server, der im Auftrag und mit den Rechten eines Clients tätig wird, nur die zur Erledigung der Aufgabe notwendigen Rechte (need-to-know) erhält. Eine Einschränkung des Rechtekontextes kann man auf dreierlei Weisen erreichen. Zum einen durch das Entfernen von Privilegien, zum anderen durch das Deaktivieren von Gruppenmitgliedschaften und schließlich durch das Hinzufügen einer Liste von beschränkten SIDs, die die Identität und Rechte des in Ausführung

Impersonation

befindlichen Programms beschreiben. Bei der Zugriffskontrolle werden sowohl die Benutzer-SID und zugehörigen Rechte als auch die restricted SID geprüft. Nur wenn beide den gewünschten Zugriff erlauben wird der Zugriff auch gewährt. Will man beispielsweise festlegen, dass ein bestimmtes Programm nur auf genau eine Datei zugreifen darf, so kann man eine restricted SID für das Programm erzeugen und einen Eintrag in der ACL der Datei konstruieren, die den Zugriff auf die Datei für diese SID erlaubt. Wenn dies die einzige ACL eines Objektes ist, die Erlaubnis-Rechte für die SID enthält, ist die gewünschte Policy realisiert.

Sicherheitslevel

Jeder Prozess hat ein primäres Access Token und kann, wie oben beschrieben, noch weitere Tokens erhalten, um im Auftrag eines Clients tätig zu werden. Dies sind dann die bereits angesprochenen Impersonation und Restricted Tokens. Um einen Missbrauch des Impersonation-Konzepts zu verhindern, kann eine Impersonation nur mit Einwilligung des Clients stattfinden. Ein Client kann den Sicherheitslevel bei der Rechte delegation explizit festlegen. Windows unterscheidet hierbei vier Stufen, die von keinerlei Rechten für den Server, über die Weitergabe der Client-Identität (SID) und der Privilegien ohne jedoch der vollständigen Impersonation, der Impersonation auf dem lokalen System bis hin zu einer Impersonation, die auch für entfernte Rechner gilt, reicht. Die Impersonation auf dem lokalen System ist die Default-Einstellung, falls keine explizite Angabe durch den Client gemacht wird.

Security Descriptor

Security Descriptor

Die bereits angesprochenen Zugriffskontrolllisten sind Bestandteil des Security Descriptors eines Objekts. Ein solcher Deskriptor besteht aus einem Header gefolgt von einer Discretionary ACL mit ACEs (Access Control Element). Der Header enthält die SID des Objekt-Owners sowie eine Gruppen SID. Dies ist die Primärgruppe des Objekts¹⁵. Die wichtigsten ACE-Elemente sind Erlaubnis- und Verbots-ACEs.

ACE

Ein Erlaubnis-ACE (allow) spezifiziert einen Identifikator SID sowie eine Bitmap, die festlegt, welche Operationen die Subjekte (Prozesse) mit der Identifikation SID auf dem Objekt ausführen dürfen. In analoger Weise wird ein Verbots-ACE (deny) verwendet. Das heißt, ein Deny-ACE spezifiziert einen Identifikator SID sowie eine Bitmap, die festlegt, welche Operationen die Subjekte (Prozesse) mit der Identifikation SID auf dem Objekt nicht ausführen dürfen. Jedes Objekt kann bis zu 16 verschiedene Zugriffsrechts typen festlegen. Standardtypen, die auf jedes Objekt anwendbar sind, sind zum Beispiel `synchronize`, das es einem Prozess erlaubt, auf einen spezifischen Zustand zu warten, `write_owner`, das das Schreibrecht an den Objekt-Owner vergibt, `write_DAC`, das den Schreibzugriff auf die DACL

¹⁵ Diese Information wird nur unter POSIX verwendet.

erlaubt, oder auch `read_Control`, das den Zugriff auf den Security Descriptor gestattet. Zu beachten ist, dass eine ACL, die keine ACE-Einträge enthält, festlegt, dass kein Subjekt ein Zugriffsrecht an dem Objekt besitzt. Demgegenüber bedeutet das Fehlen einer ACL, dass alle Subjekte alle Rechte an dem Objekt besitzen. Die Reihenfolge der eingetragenen ACEs in einer ACL ist wesentlich, da die Zugriffskontrolle die ACEs sequentiell durchläuft. Defaultmäßig werden die Deny-ACEs vor die Allow-ACEs platziert, jedoch erlauben es die Operationen zur Manipulation der ACL, die die Win32-Schnittstelle zur Verfügung stellt, eine ACL auch mit einer anderen Reihenfolge aufzubauen.

Neben der Discretionary ACL besitzt jeder Security Descriptor auch noch einen Eintrag für eine SACL, das ist die System Access Control List. Diese ist in ihrer Form analog zur DACL, besitzt aber eine andere Aufgabe. Ein Eintrag in dieser System ACL spezifiziert, welche Operationen, die auf dem zugehörigen Objekt ausgeführt werden, in der systemweiten Ereignis-Logdatei zu protokollieren sind. Damit ist es also möglich, objektspezifische Audit-Informationen zu sammeln.

Beispiel 13.1 (Security Descriptor und DACL)

Abbildung 13.11 zeigt ein Beispiel für einen Security Descriptor.

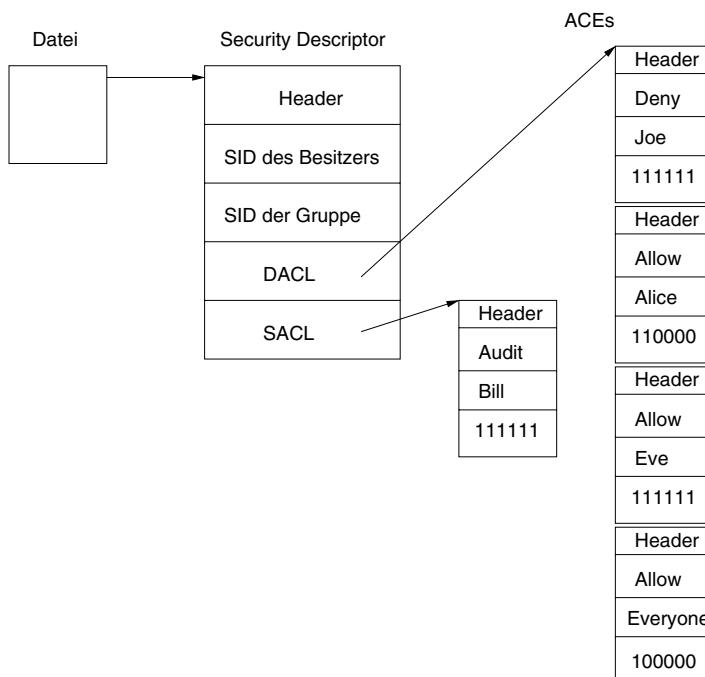


Abbildung 13.11: Beispiel eines Windows Security Descriptors

Die DACL des Deskriptors enthält vier Einträge. Der erste Eintrag besagt, dass der Benutzer Joe keinerlei Rechte an dem Objekt erhält, während gemäß dem zweiten und dritten ACE die Benutzerin Alice Lese- und Schreibrechte und die Benutzerin Eve sogar alle Rechte erlangt. Die SID *everyone* bezieht sich auf alle Benutzer und vergibt Rechte an diese. Diese Rechtevergabe kann aber durch eine weitere explizite Rechtevergabe, die dem *everyone*-Eintrag in der ACL folgt, überschrieben werden. Der SACL-Eintrag in dem Beispieldeskriptor besagt, dass jede Operation, die Bill auf der Datei ausführt, in der Log-Datei protokolliert wird.

MFT

Die Security Descriptoren werden in der Master File Table (MFT) (vgl. Abbildung 13.12) verwaltet.

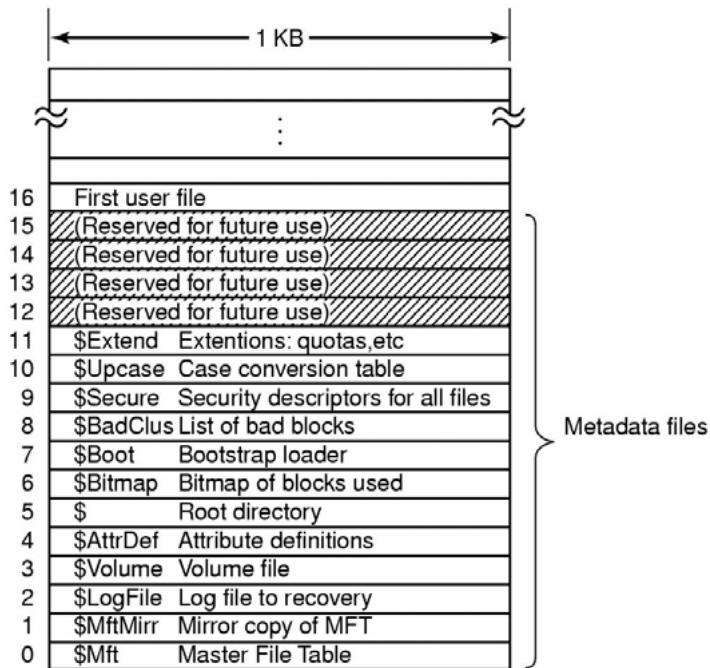


Abbildung 13.12: Struktur der Master File Table

Die MFT besteht aus einer linearen Folge von Records und jeder dieser Records beschreibt eine Datei bzw. ein Verzeichnis. Anders als unter Unix werden die Berechtigungsinformationen aber nicht in der Dateibeschreibung abgelegt (vgl. Abbildung 12.8), sondern als eigener Eintrag in der MFT, die beim Bootvorgang von der Festplatte in den Hauptspeicher geladen wird.

Die Aufgaben des Windows MFT sind vergleichbar mit dem Superblock und den inodes unter Unix. Bereits an dieser Stelle sei darauf hingewiesen, dass die Berechtigungsinformationen zwar beim Booten in einen Bereich geladen werden, der nur mit privilegierten Rechten gelesen bzw. modifiziert werden kann, aber die Master File Table natürlich auch auf der Festplatte persistent gespeichert ist. Weitere Schutzmaßnahmen, wie eine Verschlüsselung der sensiblen MFT-Einträge auf der Festplatte, um einen unautorisierten direkten Zugriff darauf zu unterbinden, sind standardmäßig nicht vorgesehen. Dies gilt übrigens in gleicher Weise für die inodes unter Unix/Linux, die dort die Zugriffsberechtigungen verwalten.

Dynamische Rechtevererbung

Windows implementiert eine dynamische Rechtevererbung, indem Berechtigungen automatisch von einem Verzeichnis auf sämtliche seiner untergeordneten Verzeichnisse und Dateien übergehen. Entsprechende geerbte ACEs werden durch ein Flag gekennzeichnet. Wenn die DACL eines Verzeichnisses geändert wird, werden die geänderten DACL-Einträge automatisch in die mit dem Vererbungsflag versehenen ACEs der DACLs der untergeordneten Verzeichnisse und Dateien kopiert. Die Vererbung von Berechtigungen auf untergeordnete Ordner und Dateien kann man auch beschränken.

Rechtevererbung

13.2.4 Zugriffskontrolle

Jeder Zugriff auf ein Objekt durch einen Benutzerprozess erfordert die Vorlage eines so genannten Object Handles. Diese Handles werden von den Objektmanagern ausgestellt und sind Prozessen eindeutig zugeordnet; sie entsprechen Capabilities. Möchte ein Prozess erstmalig auf ein Objekt o zugreifen, so führt er einen Open-Aufruf aus und gibt dabei die gewünschten Zugriffsrechte (z.B. suspend oder terminate für ein Thread-Objekt) an. Der zuständige Objektmonitor leitet den Aufruf an den Security Reference Monitor weiter, welcher ein Bestandteil des Executives ist (vgl. Abbildung 13.9). Dieser im Systemmodus ausführbare Code des Betriebssystemkerns wird als vertrauenswürdig betrachtet, so dass er als einziger für Objektzugriffe keine Object Handles benötigt, sondern direkt auf Speicheradressen zugreifen darf.

Object Handle

Der Referenzmonitor überprüft anhand der Zugriffskontrollliste des Objekts o , ob der Prozess zur Durchführung der gewünschten Zugriffe tatsächlich berechtigt ist. Ist dies der Fall, so übergibt er dem zuständigen Objektmanager eine Liste der gewährten Berechtigungen für diesen Prozess. Der Objektmanager erzeugt nun seinerseits ein Object Handle, stattet dieses mit den gewährten Berechtigungen aus und übergibt das Handle an den Prozess.

Referenzmonitor

Berechtigungsprüfung

Windows verwendet zwei Algorithmen, um die Menge der Zugriffsrechte zu einem Objekt zu bestimmen. Der erste Algorithmus wird angewandt, um für einen Prozess dessen maximale Rechte an einem spezifischen Objekt zu bestimmen. Der zweite Algorithmus bestimmt für spezielle gewünschte Zugriffe (Desired-Access-Maske), ob diese erlaubt sind. Dazu wird die Funktion *AccessCheck* verwendet. Im Folgenden beschränken wir uns auf die Beschreibung des zweiten Algorithmus, wobei wir nicht alle Details der Prüfung wiedergeben, sondern uns auf die wesentlichen Aspekte beschränken.

Als Eingabe für die Berechtigungskontrolle dient das Access Token des zugreifenden Subjekts, das u.a. die Benutzer SID und die Gruppen SID enthält. Ferner wird in der Desired Access Maske die Menge der gewünschten Zugriffsoperationen übergeben. Zur Berechtigungsprüfung eines Subjekts vergleicht der Referenzmonitor die Daten im Access Token des Subjekts sowie dessen Desired-Access-Maske mit den Daten in der DACL im Security Descriptor. Falls für das Objekt keine DACL existiert, wird der gewünschte Zugriff gewährt. Falls der Zugreifer der Eigentümer des Objekts ist, gewährt der Referenzmonitor die *read-control*, *write-DACL*-Rechte und prüft die ACL nicht weiter, falls dies die einzigen Zugriffswünsche des Subjekts waren, ansonsten wird die DACL durchsucht. Wie bereits gesagt, wird die DACL in einer First-to-Last Reihenfolge abgearbeitet. Jeder ACE-Eintrag in der ACL, dessen SID mit der SID im Access Token des Subjekts (entweder Subjekt SID oder eine der aktiven Gruppen SIDs des Subjekts) übereinstimmt, wird geprüft. Handelt es sich um einen Erlaubnis-ACE (allow), so werden alle in der Allow-Maske des Eintrags gewährten Rechte¹⁶ dem Subjekt zugestanden. Falls damit alle angeforderten Rechte gewährt werden können, terminiert die Suche. Handelt es sich um einen Deny-Eintrag, der mindestens eines der angeforderten Zugriffsrechte verbietet, so wird der gesamte Zugriff verboten und die Überprüfung terminiert. Ist die gesamte DACL abgearbeitet und konnten nicht alle gewünschten Rechte gewährt werden, so wird der Zugriff ebenfalls verweigert, anderenfalls ist der Zugriff erlaubt und der Referenzmonitor stellt einen Objekt-Handle für nachfolgende Zugriffe auf das Objekt für das Subjekt aus. Der Referenzmonitor überprüft zuerst nicht vererbte Berechtigungen und anschließend die vererbten, wodurch mögliche Widersprüche zwischen vererbten und nicht vererbten ACL-Einträgen aufgelöst werden. Das heißt, die nicht geerbten Rechte haben Vorrang vor den geerbten Rechten.

Access Check

Reihenfolge der ACEs

Aus der Beschreibung des Algorithmus wird klar, dass die Reihenfolge der ACE-Einträge in der DACL wichtig ist und das Verhalten der Berechtigungs-

¹⁶ Rechte, für die die Bitmap in der ACE den Wert 1 besitzt.

kontrolle beeinflusst. Betrachten wir als Beispiel ein Objekt, dessen DACL einen Allow-ACE-Eintrag enthält, der Benutzern mit der Gruppen-SID *Team* das Schreibrecht gewährt, sowie einen Deny-Eintrag, der der Benutzer-SID *Joe* jeglichen Zugriff untersagt. Steht der Allow-Eintrag vor dem Deny-Eintrag, so erhält der Benutzer *Joe*, falls er der Gruppe *Team* angehört, Schreibzugriff, während er keinen Zugriff erhält, wenn der Deny-Eintrag vor dem Allow-Eintrag platziert ist. ACEs werden in der Deny-vor-Allow Reihenfolge angeordnet.

Hervorzuheben ist an dieser Stelle noch einmal, dass dem Eigentümer eines Objekts stets das Schreibrecht auf die DACL gewährt wird. Damit wird sichergestellt, dass auch dann ein Objekt noch geöffnet werden kann, wenn die DACL leer ist. Das heißt, dass in diesem Fall der Eigentümer neue ACEs in die DACL eintragen und so das Objekt zugreifbar machen kann.

Ein Object Handle wird wie ein Zugriffsausweis verwendet. Bei einem Objektzugriff weist nämlich der Prozess nur noch sein Handle vor und der Objektmanager prüft, ob der Dienst, den der Prozess auf dem Objekt aufruft, in der Berechtigungsliste des Handles eingetragen ist. Zu beachten ist aber, dass bei einer derartigen Kombination aus Zugriffsausweisen und -listen auf eine direkte Aktualisierung von Zugriffsrechten verzichtet wird, um den zu leistenden Verwaltungsaufwand zu reduzieren und um die Performanz des Systems zu steigern. Eine Rechterücknahme wird für einen Prozess, der ein Objekt geöffnet hat und ein Handle dafür besitzt, erst dann wirksam, wenn der Prozess das Objekt wieder schließt und danach erneut versucht, es zu öffnen. Da viele Objekte erst mit dem Terminieren von Prozessen wieder geschlossen werden, kann sich die Verzögerung einer Rechterücknahme über lange Zeiträume erstrecken.

Probleme

Beachtenswert ist ferner, dass Betriebssystemroutinen, die im privilegierten Modus ausgeführt werden, keine Handles kennen, sondern stets Zeiger verwenden, um auf Objekte zuzugreifen. Dies ist natürlich ein sehr effizientes Vorgehen, hat aber gleichzeitig zur Folge, dass keine Berechtigungsprüfung für derartige Betriebssystemroutinen durchgeführt und der Windows-Kern als vertrauenswürdig betrachtet wird.

13.2.5 Encrypting File System (EFS)

Durch die Entwicklungen in den Bereichen mobiler Endgeräte sowie der Vernetzung sind die Zugriffskontrollen herkömmlicher Betriebssysteme häufig nicht mehr ausreichend, um den Objektschutz wirksam sicherzustellen. Gleichzeitig mit dem Wert der Daten steigt auch der Anreiz für einen direkten Diebstahl der Datenträger, man denke hier nur an den Diebstahl von Notebooks mit sensiblen Daten auf der Festplatte. Erhält ein Angreifer phy-

sischen Zugriff auf einen Rechner und kann ein Betriebssystem booten, das ein Umgehen der Zugriffskontrolle ermöglicht, so liegen alle Daten auf der Festplatte des Rechners offen. So existieren beispielsweise frei verfügbare Programme, die einen Zugriff auf NTFS-Dateien der Microsoft Betriebssysteme unter Umgehung der Zugriffskontrollen ermöglichen. Verschlüsselnde Dateisysteme bieten einen geeigneten Schutz.

EFS

Das unter Windows eingesetzte *Encrypting File System* (EFS) ist in den Betriebssystemkern integriert, so dass eine transparente, automatische Dateiver- und -entschlüsselung durchgeführt wird. Mit EFS können NTFS-Dateien, jedoch keine FAT16/32-Dateien verschlüsselt werden.

Dateiverschlüsselung

Mehrbenutzer

Zum Einsatz kommt bei der Verschlüsselung ein hybrides Verfahren. Eine Mehrbenutzerfähigkeit wird von EFS durch eine Kombination aus asymmetrischen und symmetrischen Kryptoverfahren ermöglicht. Zur Verschlüsselung einer Datei f wird ein symmetrischer Schlüssel FEK (File Encryption Key) verwendet, der automatisch vom EFS generiert wird. Der Dateischlüssel FEK wird geschützt auf der Festplatte abgelegt, indem er mit allen öffentlichen Schlüsseln der berechtigten Benutzer der Datei f mittels RSA verschlüsselt wird. Zur Speicherung des verschlüsselten Dateischlüssels wird ein spezielles EFS-Dateiattribut, das so genannte DDF (Data Decryption Field) verwendet. Der verschlüsselte Dateischlüssel und die Liste aller öffentlichen Schlüssel der berechtigten Benutzer werden als zusätzliche Dateiattribute in der EFS-Dateibeschreibung der verschlüsselten Datei f verwaltet.

Entschlüsselung

Um die Datei f vor einem Zugriff zu entschlüsseln, muss das Dateisystem der verwendeten Schlüssel FEK wiederherstellen können. Da der Schlüssel mit den öffentlichen Schlüsseln aller berechtigten Benutzer verschlüsselt wurde, benötigt das System den Zugriff auf den zugehörigen privaten Schlüssel des zugreifenden Benutzers. Idealerweise sollte dieser auf einem geschützten Speichermedium wie einer Smartcard abgelegt sein. Obwohl Windows Smartcards unterstützt, werden die privaten Schlüssel, die EFS benötigt, nicht auf solchen Smartcards gespeichert. Vielmehr wird bei der erstmaligen Verwendung von EFS durch den Benutzer automatisch ein Schlüsselpaar bestehend aus einem Private und einem Public-Key erzeugt.

Privater Schlüssel

Der private EFS-Schlüssel des Benutzers wird mittels eines symmetrischen Verfahrens, das von der CryptoAPI, die von dem Microsoft Betriebssystem zur Verfügung gestellt wird, verschlüsselt und auf der Festplatte gespeichert. Der hierfür benötigte Schlüssel wird entweder vom Passwort des Benutzers abgeleitet oder von einem Schlüssel, der auf einer Smartcard gespeichert ist, falls ein Smartcard-basiertes Login aktiviert ist. Auf diese Weise kann das EFS den privaten EFS-Schlüssel des Benutzers bereits zum Zeitpunkt des

Logins entschlüsseln. Der Schlüssel wird dann für den Rest der Sitzung im virtuellen Adressraum des EFS-Prozesses abgelegt und erst beim Logout des Benutzers wieder gelöscht. Um zu verhindern, dass diese sensitive Schlüsselinformation durch ein Paging (virtuelle Speicherverwaltung) ausgelagert und damit offen auf der Festplatte liegt, verwendet das EFS zur Speicherung von Verschlüsselungsschlüssel nur nicht-auslagerbare Seiten. Für den öffentlichen Schlüssel des Benutzers stellt EFS ein Zertifikat aus, das entweder von einer konfigurierten CA zertifiziert oder, falls keine solche CA existiert, von EFS selbst zertifiziert wird.

Abbildung 13.13 gibt einen groben Überblick über den Ablauf beim EFS.

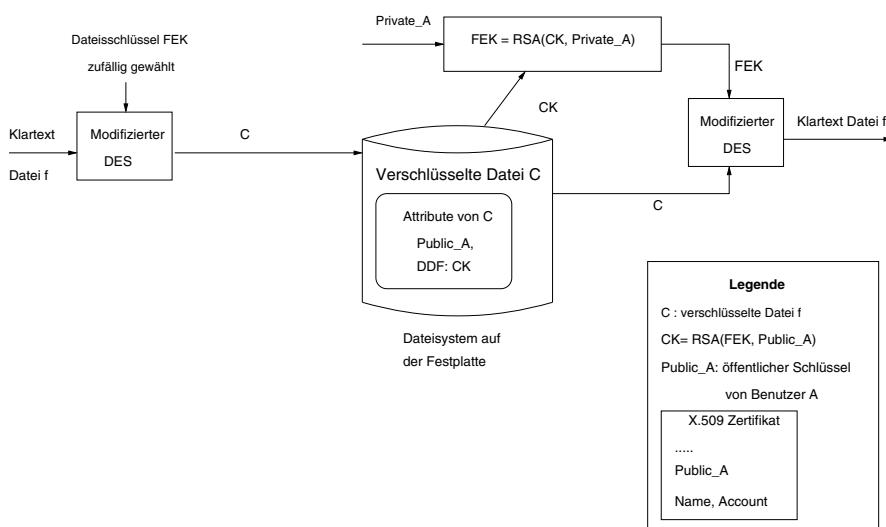


Abbildung 13.13: Verschlüsselung von Dateien unter EFS

Das Bild veranschaulicht, dass die mit *FEK* verschlüsselte Datei *f* als Kryptotext *C* auf der Festplatte abgelegt ist und der Deskriptor von *f* als zusätzliches Attribut den Kryptotext *CK* enthält. *CK* ist der mit dem öffentlichen Schlüssel *Public_A* des Benutzers A verschlüsselte Dateischlüssel. **Hinweis:** In der Abbildung wird als Verschlüsselungsverfahren der modifizierte DES (DESX) genutzt, jedoch werden in den aktuelleren Betriebssystem-Versionen der AES eingesetzt.

Ein Benutzer kann eine Dateiverschlüsselung auf zweierlei Weisen initiieren. Zum einen ist es möglich, über die Kommandozeile mit einem speziellen Befehl die Verschlüsselung eines bestimmten Verzeichnisses zu veranlassen. Zum anderen kann dies auch über den Browser initiiert werden, indem im Browser ein zu verschlüsselndes Verzeichnis (ggf. mit allen Unterverzeichnissen und Dateien) markiert wird. Durch die Integration des EFS in das

Betriebssystem ist es in der Lage, automatisch auch alle temporären Dateien von markierten Dateien, also von Dateien, die verschlüsselt werden sollen, zu ver- und entschlüsseln. Wird eine Datei in ein EFS-Verzeichnis kopiert, in dem alle Unterverzeichnisse bzw. Dateien ebenfalls verschlüsselt werden sollen, so führt EFS dies automatisch für die kopierte Datei durch.

Schlüssel-Recovery

Recovery

Die EFS-Funktionalität ist nur dann nutzbar, wenn mindestens ein Recovery-Schlüssel konfiguriert ist. Dies erfordert mindestens einen so genannten Recovery Agenten. Bei Stand-alone Rechnern ist defaultmäßig der Administrator der Recovery Agent, während bei Domänen dies per Default der Domänen-Administrator ist. Es ist jedoch auch möglich, beliebige andere Benutzer als Recover-Agenten festzulegen; dies wird in der Sicherheitspolicy des Systems als Bestandteil des *Group Policy Objects* konfiguriert. Ein Recovery-Agent besitzt die Berechtigung, Recovery-Schlüssel zu erzeugen und verschlüsselt abgelegte Dateien wieder herzustellen. Zu diesem Zweck wird stets eine Kopie des verwendeten Dateischlüssels mit einem Recovery-Schlüssel eines Recovery-Agenten verschlüsselt und in einem weiteren EFS-Dateiattribut, dem Data Recovery Field (DRF) abgespeichert (vgl. Abbildung 13.14).

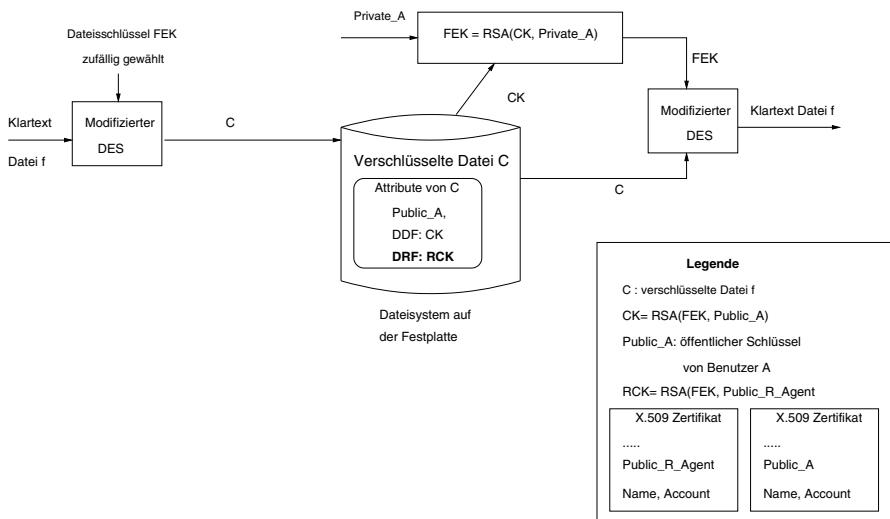


Abbildung 13.14: Schlüsselrecovery unter EFS

Der zur Verschlüsselung verwendete öffentliche Schlüssel des Recovery-Agenten wird aus dem X.509-Zertifikat des Agenten extrahiert, das in der *Encrypted Data Recovery Agent Policy* (EDRA) des Rechners abgelegt ist. In diesem Zertifikat muss in dem Feld *enhanced key usage* der Wert *File*

Recovery eingetragen sein. Der zugehörige private Recovery-Schlüssel kann sicher außerhalb des Systems aufbewahrt werden. Die EDRA besteht aus einer Menge von X.509-Zertifikaten und beim Öffnen einer verschlüsselten Datei wird stets geprüft, ob der Recovery-Schlüssel noch gültig ist. D.h. es wird geprüft, ob das zum Recovery-Schlüssel gehörige Zertifikat bereits abgelaufen ist. In diesem Fall muss die Recovery-Information erneuert werden, bevor die Datei wieder verschlüsselt abgelegt werden kann. Das bedeutet natürlich, dass die Zertifikate von Recovery-Schlüsseln und die zugehörigen privaten Recovery-Schlüssel über lange Zeiträume aufbewahrt werden müssen, damit auch auf solche verschlüsselte Dateien noch zugegriffen werden kann, die über lange Zeiträume nicht geöffnet worden sind. Insbesondere kann es natürlich sein, dass ein Recovery-Agent zwischenzeitlich bereits seine Schlüssel gewechselt hat, so dass für einen Agenten unter Umständen mehrere Zertifikate und private Schlüssel verwaltet werden müssen, damit die Daten auch nach langen Ruhepausen noch zugreifbar sind.

Sicherheitsprobleme bei EFS

Zu beachten ist, dass verschlüsselte EFS-Dateien, die in ein FAT-Dateisystem kopiert werden, vorher entschlüsselt werden, da das FAT die Verschlüsselung nicht unterstützt. Die EFS-Verschlüsselung erstreckt sich auch nicht auf den Transfer von Dateien über eine Netzverbindung. Hierzu sind weitere Maßnahmen erforderlich, wie beispielsweise das Etablieren eines VPN mittels IPSec (vgl. Abschnitt 14.3), oder man verwendet Backup Tools. Diese ermöglichen sowohl die verschlüsselte Speicherung als auch eine verschlüsselte Übertragung.

Der nachlässige Umgang mit den privaten Schlüsseln (von Benutzer oder Recovery-Agents) führt häufig zu Problemen. Da unter Windows eine Änderung des Benutzer-Passwortes durch den Administrator dazu führt, dass der rechtmäßige Benutzer seine eigenen Dateien nicht mehr entschlüsseln kann, sollte man seinen privaten Schlüssel, den EFS ja automatisch zusammen mit dem öffentlichen Schlüssel erzeugt und im System speichert, zusätzlich auf einem externen Medium ablegen. EFS bietet dafür eine Export-Funktion an, über die der private Schlüssel und das Zertifikat in einem PKCS#12-Format abgelegt und mit einem vom Benutzer festzulegenden Passwort geschützt werden. Nach dem erneuten Import des Zertifikats sind die Daten auch nach einer solchen Passwortänderung oder dem Verlust des auf dem Rechner gespeicherten Zertifikats wieder zugreifbar. Natürlich kann man auch die Recovery-Funktionalität verwenden, um an die verschlüsselten Dateien zu gelangen. Dazu benötigt man jedoch die Unterstützung des zuständigen Recovery-Agenten, bzw. desjenigen Benutzers, der das Passwort für diese Kennung besitzt. Ein offensichtliches Problem von EFS besteht darin, dass die privaten Benutzerschlüssel verschlüsselt im Speicher abgelegt und damit

privater Schlüssel

Angriffen ausgesetzt sind. Wird ein solcher Schlüssel mit einem Schlüssel verschlüsselt, der aus dem Passwort des Benutzers abgeleitet wurde, so bietet es sich förmlich an, dieses meist schwächste Glied der Sicherheitskette zu brechen und die vielfältigen Möglichkeiten eines Passwort-Cracking-Angriffs zu nutzen, um an den privaten Schlüssel des Benutzers zu gelangen.

Recovery

Jeder Benutzer mit physischem Zugriff auf einen Stand-alone betriebenen Rechner kann das Passwort der Administrator-Kennung auf einen nur ihm bekannten Wert ändern. Nach dem Zurücksetzen des Administrator-Passwortes ist ein Einloggen als Administrator möglich und falls der lokale Administrator der Recovery-Agent des Systems ist (was er ja bei Stand-alone Rechnern defaultmäßig ist), kann er mit dem Administrator-Recovery-Schlüssel jede verschlüsselt abgelegte Datei entschlüsseln. Die Voraussetzung hierfür ist natürlich, dass der private Recovery-Schlüssel des Recovery-Agenten auf der Festplatte des lokalen Rechners abgelegt wurde. Dies ist zwar eine grobe Nachlässigkeit des verantwortlichen Systemadministrators, jedoch leider keineswegs unüblich. Hat ein Angreifer Administrator-Zugriff erlangt, so kann er alternativ natürlich auch das Passwort eines spezifischen Benutzers ändern, sich als dieser Benutzer einloggen und erhält damit ebenfalls Zugriff auf dessen verschlüsselt abgelegte Daten, da im aktuellen EFS keine weiteren Authentifikationsmechanismen erforderlich sind, um den privaten Benutzerschlüssel auszulesen.

Das Einloggen als lokaler Administrator auf die gerade beschriebene Weise führt bei Domänen-Rechnern jedoch i.d.R. nicht zur Offenlegung aller verschlüsselten Dateien, da hier per Default die Kennung des Domänen-Administrators der Recovery-Agent ist und nicht die lokale Administrator-Kennung. Gelingt es einem Angreifer physischen Zugriff auf den Domänen-Controller zu erhalten, so ist unter den gleichen Voraussetzungen, wie sie gerade für den Stand-alone Fall beschrieben wurden, ein Angriff auf die Domänen-Rechner möglich.

Physischer Zugriff

Wie im vorherigen Abschnitt bereits beschrieben, existieren verfügbare Tools, so dass ein Rechner, auf den ein Angreifer physischen Zugriff besitzt, mit einem anderen Betriebssystem gebootet werden kann. Mit diesen Tools können die Einträge in der SAM-Datenbank überschrieben werden. Nach dem Re-Boot der Maschine gelten dann diese Neueinträge.

14 Sicherheit in Netzen

In Kapitel 3 wurden bereits wichtige Problembereiche heutiger Kommunikationsprotokolle und häufig verwendeter Dienste der Anwendungsebene angesprochen. Mit Verschlüsselungsverfahren, Hashfunktionen, digitalen Signaturen sowie den diversen Konzepten und Protokollen zur Sicherstellung der Authentizität von Kommunikationspartnern haben wir wirksame Schutzmechanismen kennen gelernt. Weitestgehend offen blieb bislang jedoch noch die Frage nach deren gezieltem Einsatz zur Steigerung der Sicherheit heutiger bzw. zukünftiger Kommunikationsnetze.

Firewalls liefern eine erste Abwehrlinie (first line of defense) und sind wichtige Sicherheitskonzepte in heutigen vernetzten Systemen. Abschnitt 14.1 erklärt die Konzepte und Architekturen, die im Zusammenhang mit der Firewall-Technologie von Bedeutung sind, und geht auf deren sicherheitssteigernde Möglichkeiten, aber auch auf deren Grenzen und Risiken ein.

Die Möglichkeiten von Sicherheitsprotokollen hängen sehr stark von der Schicht ab, auf der die Sicherheitsdienste zum Einsatz kommen. Abschnitt 14.2 beschäftigt sich mit allgemeinen Eigenschaften der Kommunikations- und Transportprotokollen der Schichten 2 - 4 und dem Aufbau von Virtuellen privaten Netzen, den VPNs. Mit der IPsec-Protokollfamilie werden in Abschnitt 14.3 spezielle Protokolle der Schicht 3 vorgestellt. Als de facto Standard für den Bereich der TCP-basierten sicheren Kommunikation gilt das Transport Layer Security (TLS) Protokoll. TLS wird in Abschnitt 14.4 behandelt. In Abschnitt 3.4.1 wurden die Sicherheitsprobleme des Domain Name Service (DNS) diskutiert. Abschnitt 14.5 stellt mit DNSSEC die Sicherheitserweiterung von DNS vor, mit der wesentliche Probleme beseitigt werden können. Auf die Sicherheit spezielle Anwendungsdienste, wie sichere e-Mail gehen wir dann in Abschnitt 14.6 ein. Die dort behandelten Sicherheitsprotokolle stellen nur eine Auswahl dar und dienen dazu, die generellen Vorgehensweisen exemplarisch zu verdeutlichen. Weitere Sicherheitsprotokolle auf der Anwendungsebene werden mit dem Signal-Protokoll in Abschnitt 14.7 für einen Ende-zu-Ende verschlüsselten Messengerdienst, und mit der Blockchain-Technologie in Abschnitt 14.8 diskutiert.

Kapitelüberblick

14.1 Firewall-Technologie

Firewalls haben die Aufgabe, durch Kontrollen und Filterungen von Datenpaketen die Weiterleitung solcher Pakete zu verhindern, die eine mögliche Bedrohung für die Daten und Komponenten eines Netzsegments bedeuten könnten.

14.1.1 Einführung

Brandmauer

Bereits in Kapitel 3 haben wir vielfältige Bedrohungen kennen gelernt, die sich aus einer Anbindung eines Rechners oder eines Netzsegments an ein öffentliches Netz, wie das Internet, ergeben können. Zu deren Abwehr und zur Kontrolle von Zugriffen in offenen Systemen werden heute nahezu standardmäßig Firewall-Systeme, kurz Firewalls, eingesetzt. Der Begriff der Firewall entstammt dem Bereich des Feuerschutzes bei Gebäudekomplexen. Dort haben Firewalls als Brandschutzmauern zwischen Gebäuden die Aufgabe, die Ausbreitung von Bränden von einem Gebäudekomplex auf den nächsten einzudämmen.

kontrollierte
Netzübergänge

Übertragen auf offene Rechnernetze bedeutet das, dass ein Firewall-System unterschiedliche Netze gegeneinander abzuschotten hat. In IT-Systemen übernimmt die Firewall jedoch normalerweise nicht eine vollständige Isolierung und Abschottung, wie dies die Brandschutzmauer leistet, sondern es werden gezielt und kontrolliert Übergänge zwischen den Netzen zugelassen. Dazu werden die Netze so konfiguriert, dass jedes Teilnetz nur über eine einzige Stelle, nämlich den Firewall, betreten und verlassen werden kann (vgl. Abbildung 14.1). Laut dem RFC 2828 wird eine Firewall wie folgt definiert

„A firewall is an internetwork gateway that restricts data communication traffic to and from one of the connected networks (the one said to be *inside* the firewall) and thus protects that network's system resources against threats from the other network (the one that is said to be *outside* the firewall).“

Firewall-Systeme, die häufig auch als Security Gateways bezeichnet werden, findet man in der Praxis meist bei der Kopplung eines als vertrauenswürdig und sicher betrachteten privaten Netzes bzw. Rechners an ein als unsicher eingestuftes öffentliches Netz. Sie können aber auch verallgemeinert verwendet werden, um lokale Teilnetze voreinander zu schützen und individuell kontrollierbare Verwaltungs- und Vertrauensbereiche zu errichten. Solche Firewalls werden häufig als Intranet-Firewalls bezeichnet. Ein Beispiel für den internen Einsatz von Firewalls ist die Trennung von Teilnetzen der Buchhaltung und der Forschungs- und Entwicklungslabors. Da Intranet-Firewalls sich aber konzeptionell nicht von einem Internet-Firewall unterscheiden, werden sie im Folgenden nicht separat betrachtet.

Das Firewall-Prinzip legt fest, dass jeglicher Datenverkehr von innen nach außen und umgekehrt durch die Firewall zu leiten ist. Dadurch ist es möglich, Zugriffe zu kontrollieren sowie Informationen über erfolgreich bzw. nicht erfolgreich durchgeföhrte Zugriffe zu protokollieren. Diese Protokollinformation kann anschließend sowohl zu Abrechnungszwecken (engl. *accounting*) als auch zur Aufdeckung von Angriffsversuchen analysiert werden. Die Errichtung von Schutzwällen führt zu einer Partitionierung

zentraler
Kontrollpunkt

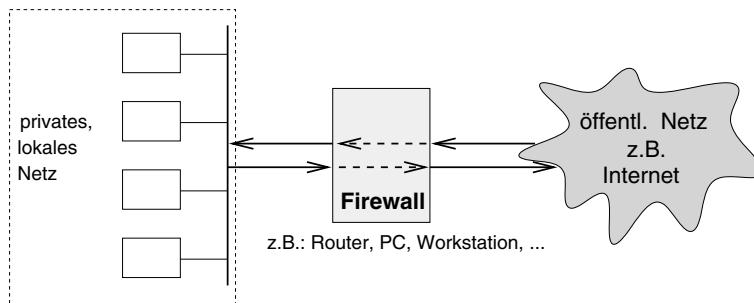


Abbildung 14.1: Kontrollierte Netzübergänge mit einem Firewall

des Netzes, was die Auswirkungen von Angriffen beschränkt. Diese gegeneinander kontrollierbar abgegrenzten Netze sind vergleichbar mit den Schutzdomänen, die wir in Kapitel 12 auf der Abstraktionsebene der maschinennahen Programmierung mit den virtuellen Adressräumen und auf der programmiersprachlichen Ebene mit den Schutzumgebungen für Prozeduren bzw. Klassen besprochen haben. Nach diesen Vorbemerkungen ergibt sich folgende informelle Definition eines Firewall-Systems.

Definition 14.1 (Firewall)

Ein Firewall besteht aus einer oder mehreren Hard- und Softwarekomponenten, die zwei Netzwerke koppeln und sicherstellen sollen, dass jeglicher Verkehr zwischen den beiden Netzen durch die Firewall geleitet wird. Sie realisiert eine Sicherheitsstrategie, die Zugriffsrestriktionen¹ und ggf. Protokollierungs- sowie Authentifikationsanforderungen umfasst. Die Firewall leitet nur diejenigen Datenpakete weiter, die diese Strategie erfüllen, und führt Authentifikationen sowie ein Auditing gemäß der festgelegten Firewall-Policy durch.

Firewall



Mit getroffenen Präzisierungen ist klar, dass man die Aufgaben einer Firewall eher mit denen einer Zugbrücke als mit einer undurchlässigen

Analogie zur
Zugbrücke

¹ Hier im Sinne von zulässiger oder unzulässiger Weiterleitung von Datenpaketen.

Schutzmauer vergleichen kann, die den Zugang zu einer Burg kontrolliert. Die Burg kann nur über den kontrollierten Zugang über die Zugbrücke betreten und wieder verlassen werden.

Sicherheitsstrategie

Gemäß Definition 14.1 ist eine Firewall eine zentrale Kontrollinstanz, in der Sicherheitsdienste der Zugriffskontrolle und optional noch Authentifikations- und Auditingdienste zusammengeführt sind. Diese Sicherheitsdienste setzen eine festgelegte Sicherheitsstrategie um. Eine einfache solche Zugriffskontrollstrategie haben wir bereits in Beispiel 6.2 kennen gelernt.

Definition 14.1 verdeutlicht, dass auf eine Firewall natürlich auch die Aussage zutrifft, die wir schon mehrfach für sichere IT-Systeme gemacht haben, dass nämlich die Sicherheit keine absolute Eigenschaft ist, sondern stets relativ zu den spezifizierten Anforderungen bewertet werden muss. Ist die zugrunde liegende Strategie unvollständig, grobgranular oder womöglich inkonsistent, so überträgt sich das auf die Firewall. Damit ist klar, dass die Qualität der durchgeföhrten Kontrollen und der Grad der erreichbaren Sicherheit unmittelbar von den Anforderungen seiner Strategie abhängig sind. Der Einsatz einer Firewall garantiert selbstverständlich nicht per se eine höhere Sicherheit, sondern er kann bei einer fehlerhaften und unsachgemäßen Konfigurierung sogar erhebliche Sicherheitsprobleme nach sich ziehen. Wir werden darauf an gegebener Stelle noch zurückkommen.

Kenntnisse

Eine präzise und möglichst vollständige Erfassung (Modellierung) der Sicherheitsanforderungen ist wichtig, um durch den Einsatz einer Firewall auch den gewünschten zusätzlichen Grad an Sicherheit zu erzielen. Zur Aufstellung einer solchen Strategie sind sowohl Kenntnisse über die Sicherheitsschwachstellen im Einsatz befindlicher Kommunikationsprotokolle und von Netzwerkdiensten, so wie wir sie in Kapitel 3 besprochen haben, als auch Kenntnisse über allgemeine Sicherheitsgrundfunktionen und allgemeine Prinzipien erforderlich (vgl. Kapitel 4.4). Dies verdeutlicht, dass das Konfigurieren und Betreiben einer Firewall keineswegs eine triviale Aufgabe ist und nur speziell ausgebildeten Personen überlassen sein sollte. Heutige Firewalls müssen so konfiguriert werden, dass sie mit komplexen Inhalten und Protokollen umgehen können. Dazu gehört die Filterung und Kontrolle von Multimediadaten, von ausführbarem, mobilem Code als Anhang von Mails oder als Download von einer Web-Seite, ebenso wie die Behandlung von Protokollen, die dynamisch in Web-Browser per Plug-in integriert werden.

Firewall-Klassen

Zur Realisierung der allgemeinen Aufgaben einer Firewall, die mit Definition 14.1 umrissen sind, werden in der Praxis unterschiedliche Firewall-Klassen sowie Architekturen verwendet. An Klassen unterscheidet man Paketfilter, Proxies und Applikationsfilter (engl. *application-level gateway*). Diese Typen werden häufig kombiniert eingesetzt, wodurch sich

unterschiedliche Firewall-Architekturen ergeben. Die häufigsten Architekturausprägungen enthalten Paketfilter, die in Netzwerkroutern integriert sind. Paketfilter werden meist zusätzlich mit verschiedenen Applikationsfiltern, die auf Servern realisiert werden, oder mit Proxy-Servern, die als Erweiterungen bestehender Kommunikationsprotokolle angesehen werden können, kombiniert. Da Firewalls sicherheitskritische Komponenten sind, müssen diese vor Angriffen besonders geschützt werden. Zu diesem Zweck stattet man Firewall-Rechner meist mit erheblich weniger Software als herkömmliche PCs oder Workstations aus, um dadurch *a priori* die Menge der möglichen Angriffspunkte zu minimieren.

Zur Firewall-Thematik existieren bereits viele Bücher (u.a. [40]), die häufig sehr technisch oder produktorientiert sind und dadurch sehr konkrete Hinweise zum praktischen Einsatz von Firewalls geben. In Abgrenzung hierzu beschränken wir uns darauf, die grundlegenden Konzepte wesentlicher Firewall-Klassen und Architekturen zu erklären, aber auch deren jeweilige Grenzen aufzuzeigen.

14.1.2 Paketfilter

Paketfilter Firewalls sind Systeme, die, wie Abbildung 14.2 verdeutlicht, auf den ISO/OSI-Schichten 3 und 4, also der Netzwerk- und der Transportschicht (vgl. Seite 99), angesiedelt sind.

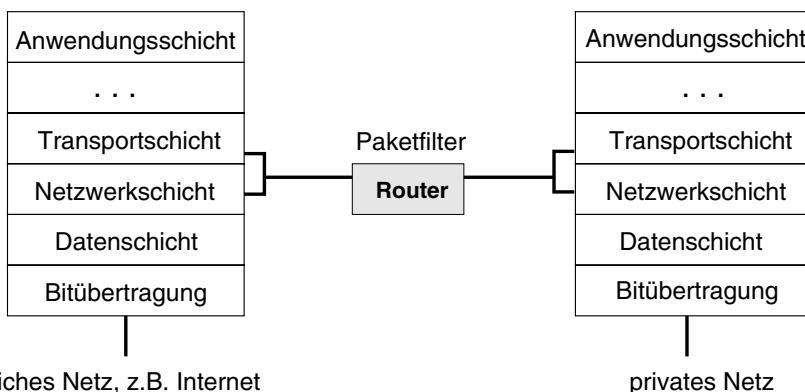


Abbildung 14.2: Einordnung eines Paketfilters

Sie haben die Aufgabe, Datenpakete, die mittels der Schicht 4 bzw. Schicht 3 Protokolle übertragen werden, nach den von der Sicherheitsstrategie vorgegebenen Kriterien zu filtern. Da Paketfilter in der Regel zwischen einem internen Netz und dem Internet eingesetzt werden, handelt es sich bei den betreffenden Protokollen in der Regel um TCP/IP bzw. UDP/IP.

Durch die Einordnung der Kontrollkomponenten in die Protokollsichten ist gleichzeitig klar, welche Informationen zur Durchführung der Kontrollen verfügbar sind, nämlich die Informationen aus den IP- bzw. TCP- oder UDP-Headern (siehe Abbildung 3.4).

Filterinformation

Mit Paketfiltern können somit Datenpakete auf der Basis von Sender- und Empfängeradressen (die IP-Adresse), Ports, Optionen, Acknowledgement-Bit oder aber des Protokolltyps gefiltert werden. Paketfilter-Firewalls können aber auch die Daten untersuchen, die nicht im Header-Teil sondern im Payload des Pakets stehen. Ein Beispiel für eine solche Information könnte eine Adresse einer Web-Seite sein, die von einem Benutzer angefordert wird. Paketfilter können auch bereits einfache Plausibilitätsüberprüfungen durchführen. Ein Beispiel hierfür ist die Überprüfung, dass die tatsächliche Paketgröße der behaupteten Größe entspricht. Auf diese Weise lassen sich bereits einige Denial-of-Service-Angriffe, die auf absichtlich falsch konstruierten Paketen basieren, erkennen und abwehren. Eine Filterregel ist vergleichbar mit einem Eintrag in einer Zugriffskontrollmatrix. Abweichend von der üblichen Matrixnotation werden hierbei Berechtigungen (blockieren, erlauben), Subjekte (Sendeadresse und Sendeport), Objekte (Zieladresse und Zielport) und weitere Entscheidungsinformationen (u.a. Optionen) in einem Listeneintrag zusammengefasst.

weitere Funktionen

Neben dem Durchlassen und Blockieren von Paketen aufgrund von Filterregeln kann ein Paketfilter-Firewall auch Fehlermeldungen an den Absender eines nicht weitergeleiteten Pakets senden, einen Eintrag in eine Logdatei schreiben, wenn ein Paket abgelehnt wird, oder bei bestimmten Paketen einen Alarm setzen, um den Administrator unmittelbar zu informieren. Fortgeschrittenere Paketfilter können aber auch die Pakete gezielt manipulieren, zum Beispiel um eine neue Adresse mit dem Network Address Translation (NAT) Protokoll oder neue Portnummern mittels PAT (Port Address Translation) einzutragen, oder um das Paket an einen anderen Empfänger, z.B. an einen Proxy-Server, zu senden. Die Adressumwandlung mittels NAT hat den Vorteil, dass die Rechner im geschützten Netz Adressen haben, die nicht nach draußen bekannt sind. Jeglicher Verkehr zu oder von diesen Rechnern muss „genattet“ werden. Dies ermöglicht zum einen eine sehr gute Kontrolle des Netzverkehrs und zum anderen werden zusätzlich die internen Netzstrukturen verschleiert. Andererseits ergeben sich durch das „Natten“ aber auch Probleme für eine Firewall, die dafür ja Pakete modifizieren muss. Für Pakete mit dynamischen Adressen ist das Logging sehr viel schwieriger durchzuführen. Über ein Logging möchte man nachhalten, welche Pakete tatsächlich zu bzw. von bestimmten IP-Adressen gesandt bzw. empfangen wurden.

Um mögliche Angriffe auf fehlerhafte TCP/IP-Stack-Implementierungen abzuwehren, kann eine Firewall auch in die Lage versetzt werden, illegale Optionen in Datenpaketen zu erkennen und zu entfernen oder ungewöhnliche Kombinationen aus gesetzten Flags bzw. gewählten Optionen zu verändern. Paketfilter können sogar auch ihre Filterregeln ändern, zum Beispiel, um eine Antwort auf ein UDP-Paket zu akzeptieren, oder um den gesamten Verkehr von einem Rechner zu blockieren, nachdem von diesem problematische Pakete empfangen wurden.

Filterung nach Portnummern

Werden Paketfilter nur auf der Netzwerkschicht eingesetzt, so muss man sich beim Treffen von Filterungsentscheidungen mit derjenigen Information begnügen, die auf dieser Schicht durch die Kommunikationsprotokolle in den Datenpaketen bereitgestellt wird. Da die Schicht 3 eine Ende-zu-Ende-Verbindung zwischen Endsystemen etabliert, nicht aber zwischen Prozessen auf diesen Systemen, stehen lediglich die IP-Adressen des Senders und Empfängers zur Identifizierung zur Verfügung. Eine größere Differenzierung erlauben die Portadressen, die mit der Schicht 4 eingeführt werden. Ein Paketfilter auf der Transportschicht ermöglicht die Filterung nach Portnummern, so dass man Datenpakete nach den Diensten, die mit ihnen in Anspruch genommen werden sollen, unterscheiden kann. Dies stellt eine deutliche Verbesserung gegenüber einer adressenbasierten Kontrolle dar, da eine dienstbezogene Zugriffskontrolle ermöglicht wird. Das nachfolgende Beispiel verdeutlicht die Vorteile einer portbasierten Filterung.

*Granularität
der Filterung*

Beispiel 14.1 (Filterung von SMTP-Paketen)

Das SMTP (Simple Mail Transport Protocol) stellt einen Mail-Dienst zur Verfügung, der vom anbietenden Server üblicherweise an die Portadresse 25 gebunden ist. Ein Clientrechner, der diesen Dienst nutzen will, baut eine Verbindung auf und erhält von seinem Betriebssystem dafür dynamisch eine Portadresse zugewiesen, an die die Antworten zu richten sind. Dynamisch zugeteilte Ports besitzen einen Wert größer als 1023, weil die Portadressen 0 ... 1023 für Systemzwecke reserviert sind. Ein Datenpaket, das den Mail-Dienst nutzen soll, enthält damit als Absendeport eine Portadresse > 1023 sowie den Empfängerport 25, während das Antwortpaket des Servers gerade umgekehrt als Absendeport den Wert 25 und als Empfängerport eine Adresse > 1023 angibt.

Mail-Dienst

Ein Paketfilter, der Zugriffe auf den SMTP-Dienst nur auf Basis der Adressen filtert, muss also in beiden Richtungen solche Pakete passieren lassen, die zulässige IP-Sende- bzw. -Empfängeradressen besitzen.

Filterregeln

verfeinerte Regel

Falls der Filter aber auch die Portadressen berücksichtigen kann, so lassen sich folgende differenziertere Filterregeln aufstellen:

Aktion	Sendeadr.	Sendeport	Zieladr.	Zielport
erlauben	extern	> 1023	intern	25
blockieren	extern	> 1023	intern	$\neq 25$
erlauben	intern	25	extern	> 1023
blockieren	intern	$\neq 25$	intern	> 1023

Das heißt, dass alle von einem Client (extern) eingehende Pakete, die an einen anderen als an den Port 25 gerichtet sind, ebenso blockiert werden, wie die Pakete des Servers (intern), die von einer anderen Portnummer als der mit der Portadresse 25 abgesendet werden.



Screening Router

Paketfilter können relativ einfach unter Nutzung vorhandener Router realisiert werden. Ein Router ist meist auf der Netzwerkschicht, der Schicht 3, des OSI-Modells eingeordnet und hat im Wesentlichen die Aufgabe, IP-Pakete über eine oder mehrere Netzwerkschnittstellen zu empfangen und sie gemäß seiner Routinginformationen sowie der im Paket enthaltenen Empfängeradresse über eine seiner Netzwerkschnittstellen weiterzuleiten. Viele der kommerziellen Routerprodukte besitzen die Fähigkeit, Pakete nach vorgegebenen Regeln zu analysieren, so dass sie unmittelbar als Paketfilter eingesetzt werden können. Man bezeichnet diesen Routertyp auch häufig als Screening Router.

zustandslos

Die Sicherheitsstrategie eines Paketfilters wird durch Filterregeln in einer Filtertabelle oder über Filterausdrücke (siehe Beispiel 14.2) spezifiziert. Einfache Paketfilter-Firewalls sind zustandslos. Das hat zur Konsequenz, dass eine Entscheidung, einen Zugriff zu blockieren oder weiterzuleiten, allein von den aktuell in dem zu analysierenden Datenpaket vorliegenden Informationen abhängig gemacht werden kann.

Beispiel 14.2 (Filterregeln)

Betrachten wir als Beispiel die einfachen Sprachkonstrukte, mit denen Filterregeln in Form von Zugriffskontrollisten (ACL) formuliert werden können. Wir beschränken uns auf zustandslose Regeln, obgleich auch zustandsbehaftete Regeln festlegbar sind. Eine Regel hat grob folgendes Format:

```
access-list {acl number} {permit|deny} [protocol]
[source address and mask]
[source port number or range]
[destination address and mask] ...
```

Damit lässt sich z.B. folgende Filterregel spezifizieren.

Filtern primitiver Spoofing-Versuche:

Ein einfacher Spoofing-Angriff besteht darin, dass der Angreifer eine lokale IP-Adresse als Absendeadresse in sein Paket einträgt, das er von außen versucht ins Netz zu schleusen (vgl. Abbildung 14.3). Das bedeutet, dass

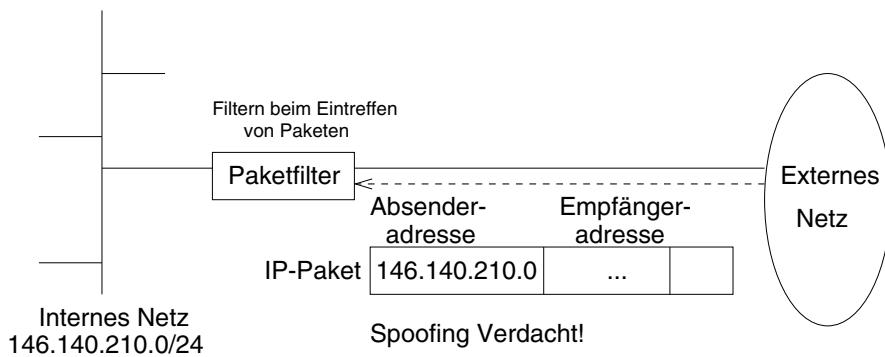


Abbildung 14.3: Erkennen einfacher Spoofing-Angriffe

die Filterregel festlegen muss, dass ein von außen kommendes Paket mit einer internen Absendeadresse zu blockieren ist. Sei der Adressbereich des internen Netzes durch 146.140.210.0/24 gegeben, dann leistet folgende Filterregel das Gewünschte:

```
acc 101 deny ip 146.140.210.0.0.0.0 2555 any
```

Mit dem Schlüsselwort *any* wird eine Art Wildcard festgelegt, die hier besagt, dass die Zieladresse jede beliebige sein kann.



Fortgeschrittenere Paketfilter sind *zustandsbehaftet* (engl. *stateful*), so dass sie Informationen über beobachtete Pakete speichern und dies bei der Filterentscheidung berücksichtigen können. Derartige Paketfilter nennt man auch häufig *dynamische Filter*², da sie ihr Filterverhalten von den Paketen abhängig machen, die sie beobachten. Der Rückgriff auf Zustandsinformationen beim dynamischen Etablieren von Zugriffsregeln ist beispielsweise beim Filtern von UDP oder DNS-Paketen sehr hilfreich. Anhand von Zu-

*zustandsbehaftet,
dynamisch*

² oder auch stateful inspection, content based access control

standsinformationen über bereits ausgetauschte Datenpakete ist die Firewall in der Lage zu erkennen, ob es sich bei den ankommenden Paketen um Antworten auf zuvor gesendete Anfrage-Pakete handelt, die die Firewall passieren dürfen. Dazu muss der Filter sich die ausgesendeten UDP- bzw. DNS-Anfrage-Pakete für einige wenige Sekunden (setzen von Timeouts) in einer Tabelle abspeichern (vgl. Abbildung 14.4).

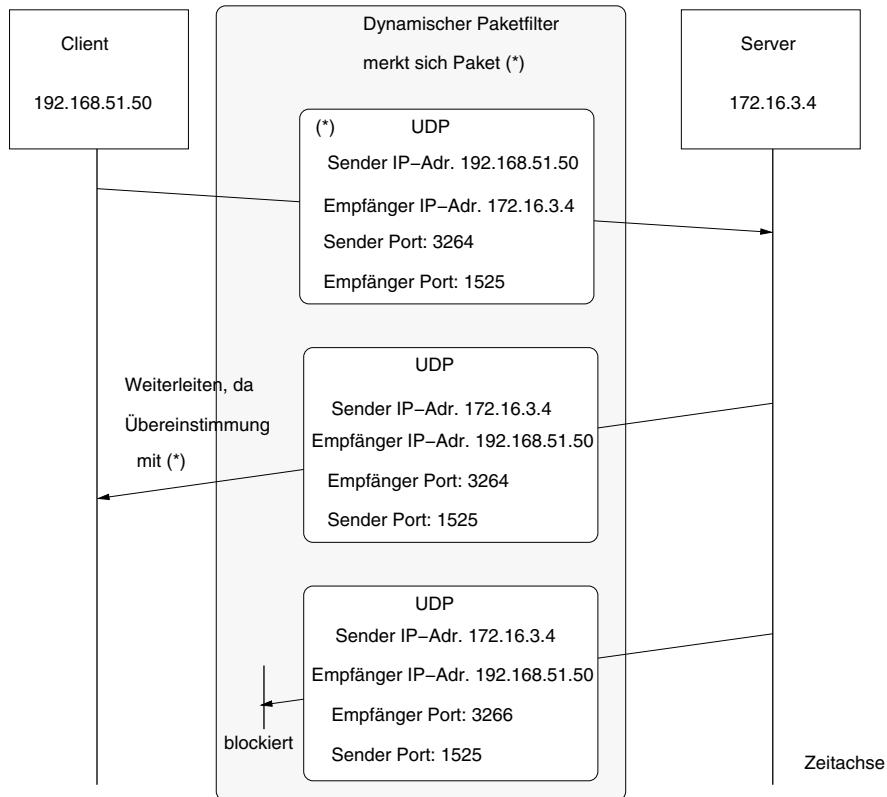


Abbildung 14.4: Dynamische Filterung von UDP-Paketen

Callback-Problem

Auf diese Weise lässt sich auch das Callback-Problem von aktiven FTP-Verbindungen lösen. Ein dynamischer Firewall kann die ausgehende FTP-Verbindung überwachen und die darin spezifizierte Datenverbindung, den Port, extrahieren und protokollieren. Sendet der angefragte Server über diesen Port seine Antwort zurück (callback), so ist die Firewall in der Lage, dieses Paket als eine legitime Antwort zu erkennen und passieren zu lassen. Filter, die zusätzlich zum Header auch den Content, also die Nutzdaten des Pakets untersuchen, nennt man häufig intelligente Paketfilter. In heutigen Systemen werden in der Regel dynamische und intelligente Paketfilter eingesetzt.

Filtern von Multi-Media-Protokollen

Multimedia-Protokolle, wie Conferencing Protokolle, sind aus der Sicht von Paketfilter-Firewalls sehr problematisch und kaum geeignet filterbar. Bekannte Beispiele für Conferencing-Protokolle sind das T.120 und H.323³. Das T.120 wird für den Dateitransfer, Chat, Whiteboard und das Application Sharing verwendet. T.120 arbeitet TCP/IP-basiert und verwendet den festgelegten Port 1503. Problematisch an T.120 ist, dass es über diesen einen Port gleichzeitig die sicherheitstechnisch relativ unkritischen Dienste Chat sowie Whiteboard und die problematischen Dienste des Dateitransfers und der gemeinsamen Nutzung von Anwendungen anbietet. Durch diese Bündelung von Diensten kann das undifferenzierte Öffnen des Zugangs zum internen Port 1503 durch eine Paketfilter-Firewall ganz erhebliche Sicherheitsprobleme nach sich ziehen. Benötigt werden hier Proxy-Lösungen, wie wir sie nachfolgend kennenlernen, die eine Filterung zugeschnitten auf die Anwendung ermöglichen.

H.323 deckt Audio- und Video-Conferencing ab, wobei eine große Anzahl von sowohl UDP als auch TCP-basierten Verbindungen zum internen Netz aufgebaut werden. Jeder Datenstrom erfordert einen dynamisch allokierten TCP-Port für den Kontrollkanal sowie einen dynamisch allokierten Port für den Datenkanal. Audio- und Video-Daten werden separat übertragen und für jede Richtung wird ein eigener Kanal aufgebaut. Eine normale Video-Konferenzschaltung erfordert mindestens acht dynamisch gebundene Ports, so dass dieses Protokoll von einer zustandslosen Paketfilter-Firewall nicht zu schützen ist. Eine dynamische Firewall kann zumindest das Aushandeln der zu verwendenden Ports protokollieren. Da jedoch die ankommenden H.323-Daten von einem entfernten Host nicht den Charakteristika einer UDP-Antwortnachricht entsprechen, kann ein Paketfilter H.323-Pakete nicht geeignet filtern. Kenntnisse über die spezifischen Eigenschaften von H.323 wären hierfür erforderlich. Da aber H.323 schwierig zu parsen sind (z.B. werden IP-Adressen an nicht festgelegten Stellen in H.323-Pakete eingebettet), gibt es auch kaum H.323-spezifische Anwendungsfilter. Das weit verbreitete Conferencing Programm NetMeeting von Microsoft basiert sowohl auf T.120 als auch auf H.323 und ist entsprechend schwierig zu filtern.

Viele der heutigen Standardbetriebssysteme wie Unix/Linux-Varianten oder die Systeme der Windows-Familie bieten integrierte Paketfilterdienste an, die unabhängig von Freeware oder kommerziellen Firewall Produkten sind. Beispielsweise ist das System *ipchains* bzw. *netfilter* in den Linux-Kern integriert.

T.120

H.323

NetMeeting

³ Es handelt sich hierbei nicht um ein Akronym.

Vorteile

Vorteile und Grenzen von Paketfiltern

Wesentliche Vorteile von Paketfiltern sind deren preiswerte und mit vorhandener Technologie relativ einfache Realisierbarkeit, deren transparente Nutzbarkeit, da weder Anwendungsprogramme modifiziert werden noch Benutzer mit dem Filter interagieren müssen, sowie deren Fähigkeiten, einige Klassen von IP-Spoofing Angriffen und Routingangriffen abwehren bzw. eindämmen zu können. Mit einer korrekt platzierten Paketfilter-Firewall, kann über eine einzelne zentrale Komponente ein gesamtes Subnetz abgesichert werden. Im Gegensatz zu Proxys, die eine aufwändige Umleitung von Paketen erfordern, arbeiten Paketfilter sehr effizient, da sie direkt in den Verbindungsweg von Paketen eingebunden sind. Diesen Vorteilen stehen jedoch eine ganze Reihe von Nachteilen und Problembereichen gegenüber.

Grenzen

Authentizität und Integrität

Die Filterungsentscheidung basiert auf Angaben zur Subjektidentifikation, nämlich auf den IP-Adressen, die nicht authentifiziert sind. Anhand dieser Informationen kann man zwar zwischen internen und externen Adressen unterscheiden, aber diese geben keine Auskunft darüber, ob die Adressen ihrerseits authentisch zum Absender des Pakets gehören. Maskierungsangriffe bzw. Man-in-the Middle-Angriffe sind möglich. Das Analoge gilt auch für die Filterung von Diensten anhand von Portadressen. Der Eintrag eines Absender-Ports in einem Paket ist nicht per se vertrauenswürdig, da jeder, der den Rechner, von dem das Paket stammt, kontrolliert (z.B. über Root-Zugriffsrechte bei Unix-Systemen oder jeder Benutzer eines vernetzten privaten PCs), diesen Eintrag fälschen und einen beliebigen Dienst an den Port binden kann. Beim Aufstellen von Filterregeln sollte man deshalb darauf achten, Pakete, die von außen nach innen geleitet werden sollen, nur an wenige Ports, an die lokal ein vertrauenswürdiger Dienst gebunden ist, zu leiten. Wenn dieser Dienst nur genau seine spezifizierte Funktionalität durchführt, kann dadurch die Möglichkeit eines nicht vertrauenswürdigen externen Servers für Angriffe eingeschränkt werden. Auch die übrigen Informationen des IP-Headers, die zur Kontrolle herangezogen werden, sind unbemerkt manipulierbar, da sie in den herkömmlichen Kommunikationsprotokollen weder verschlüsselt noch mittels eines Message-Authentication Codes geschützt werden.

schwache Prüfsummen

Die zur Fehlerkontrolle verwendeten Prüfsummen können von einem Angreifer nach einer Manipulation neu berechnet und dem Paket-Header hinzugefügt werden, wodurch sich die unautorisiert durchgeföhrten Modifikationen auf einfache Weise verschleiern lassen. So stützt man sich zum Beispiel bei der Filterung von TCP-Verbindungen auf das ACK-Flag ab, um zu erkennen, ob eine neue Anfrage von außerhalb vorliegt oder ob es sich um Antwortpakete auf eine Anfrage handelt. Bei neuen Anfragen

ist das ACK-Flag nicht gesetzt, so dass so zwischen Anfrage- und Antwortpaketen unterschieden werden kann. Die Sicherheit des Filtervorganges basiert hier darauf, dass dieses ACK-Flag nicht fälschbar ist, was in einer feindlichen Netzumgebung selbstverständlich nicht gewährleistet ist. Für UDP-Verbindungen ist eine solche Art der Filterung nicht möglich, da keine Verbindung aufgebaut wird, sondern Pakete unabhängig voneinander versendet werden. Der Filter muss hier allein darauf vertrauen, dass die Angabe des Sendeports richtig ist, was aber natürlich nicht der Fall sein muss.

Von den in Definition 14.1 angesprochenen Eigenschaften kann ein Paketfilter nur solche abdecken, die mit einer einfachen Zugriffskontrollstrategie formuliert sind. Authentifikationsanforderungen gehen über die Möglichkeiten dieser Filter hinaus. Eine Realisierung der Filterung auf der Netzwerk- bzw. Transportebene hat den Nachteil, grobgranular und zu wenig differenziert zu sein. Die Subjekte, für die die Sicherheitsstrategie Berechtigungen vergibt, werden über IP-Adressen sowie Ports identifiziert. Es handelt sich somit um Endsysteme bzw. Dienste, wobei aber eine Differenzierung nach Benutzern nicht möglich ist. Die Filterung nach Portadressen ist ebenfalls nur sehr grob, da man damit nicht eindeutig die eigentlichen Anwendungsdienste, die man filtern möchte, erfasst. Beim Aufstellen der Filterregeln kann man also nur hoffen, dass nur wirklich der vermutete Dienst oder das vermutete Protokoll an dem in der Filterregel spezifizierten Port ausgeführt wird. Dieser lose Zusammenhang zwischen Portadresse und Dienst kann natürlich von einem Angreifer manipuliert werden, so dass über beispielsweise den Portmapperdienst ein problematischer Dienst an eine Portadresse gebunden wird, die von der Firewall als vertrauenswürdig eingestuft wurde.

Zustandslose Paketfilter können Zugriffe nur erlauben oder blockieren. Abstufungen aufgrund von Kontextinformationen oder der Zugriffsvergangenheit sind nicht möglich. Die fehlende Zustandsinformation verhindert beispielsweise eine gezielte Filterung von UDP-basierten Diensten, so dass man sich mit der Filterung verbindungsorientierter, TCP-basierter Dienste begnügen muss. Zustandslose Paketfilter realisieren darüber hinaus eine statische Sicherheitsstrategie, die die Möglichkeiten zur Festlegung von Zugriffsverboten auf solche Dienste einschränkt, denen eine feste, a priori bekannte Portnummer zugeordnet ist. Dies führt zu Problemen bei der Filterung von bekannten und sehr häufig verwendeten Diensten wie aktivem FTP oder Multi-Media-Protokollen, da diese Portadressen verwenden, die dynamisch festgelegt werden (Callback-Problem).

Dynamische und zustandsbasierte Paketfilter lösen einige dieser Probleme, jedoch können sie zu erheblichen Performanceinbußen führen, da die Filterung nach contentbezogenen Daten sowie nach Zustandskontexten sehr viel aufwändiger ist als die reine Adressen- bzw. Portfilterung. Mit dem Auf-

grobgranulare
Kontrolle

zustandslos

dynamische Filter

rechterhalten von Zustandsinformationen werden derartige Firewalls darüber hinaus anfällig gegen Denial-of-Service-Angriffe. Ferner gehen nach einem Reboot des Filters alle Zustandsinformationen verloren und es werden Pakete abgewiesen, die eigentlich zulässig sind. Problematisch ist auch, dass Informationen über Pakete wie beispielsweise UDP-Pakete gespeichert werden, zu denen es unter Umständen nie ein Antwortpaket geben wird. Da auch dynamische Firewalls ihre Entscheidungen auf der Basis der Sender IP-Adresse und Ports treffen, kann ein Angreifer durch einen Maskierungsangriff ein UDP-Paket als Reply auf eine vorab gesandte UDP-Nachricht einschleusen. Dazu muss er nur dieses gesendete Paket abfangen und die Sende-IP-Adresse durch seine eigene IP-Adresse ersetzen, so dass sein Reply-Paket akzeptiert wird.

Schließlich ist auch noch anzumerken, dass auch eine zustandsbasierte Firewall nur einen sehr eingeschränkten Zustandsüberblick verwaltet, da alles andere viel zu aufwändig wäre. Damit ist klar, dass eine solche Firewall natürlich auch nicht einen ganzen Datenstrom als eine Einheit betrachten und kontrollieren kann, sondern trotz einiger Informationen aus dem Ablaufkontext letztendlich doch nur einzelne Pakete überprüft. Maßnahmen zur Überprüfung eines ganzen Datenstroms sind somit auf höhere Protokollebenen ggf. bis in die Anwendungen hinein zu verlagern. Protokolle zur e-Mail-Verschlüsselung, wie wir sie in Abschnitt 14.6 behandeln, sind Beispiele dafür, dass ein gesamter Datenstrom (hier eine Mail) abgesichert wird, indem u.a. ein Hashwert über den Mail-Körper berechnet und dieser signiert wird.

14.1.3 Proxy-Firewall

Eine einfache, generische Proxy-Firewall ist auf der Transportschicht gemäß ISO/OSI angesiedelt. Sie wird aus diesem Grund auch häufig als Verbindungs-Gateway (engl. *Circuit-level Gateway*) bezeichnet. Das Proxy-Konzept als Stellvertreterkonzept wird in der Informatik an unterschiedlichen Stellen verwendet (z.B. beim entfernten Methodenaufruf (RMI) oder beim Caching). Im Zusammenhang mit Firewalls verbirgt sich hinter den Proxies eine sehr einfache Idee. Anstatt dass ein Benutzer sich auf einem Vermittlungsrechner einloggen muss, um von hier aus Zugriffe auf das Internet durchzuführen, werden ihm spezialisierte Dienste auf einem Gatewayrechner angeboten, die er aufrufen kann und die als seine Stellvertreter (Proxy) agieren. Auf diese Weise kann man sicherstellen, dass ein Benutzer nur eine wohl definierte und beschränkte Menge von Programmen auf dem Gateway nutzen kann.

Eine Proxy-Firewall wird in der Literatur häufig als spezielle Ausprägung eines Applikationsfilters betrachtet. Da aber Applikationsfilter auf einer

höheren OSI-Schicht aufsetzen und damit konzeptionell andere Fähigkeiten besitzen können, folgen wir dieser Sicht hier nur insofern, als dass wir an gegebener Stelle auf die Gemeinsamkeiten beider Ansätze hinweisen.

Sowohl Proxy-Filter als auch Applikationsfilter werden üblicherweise auf einem Multi-Homed-Rechner realisiert. Dabei handelt es sich um einen Rechner, der mehrere Netzwerkkarten besitzt und als Bastionsrechner (engl. *bastion-host*) bezeichnet wird. Der Name röhrt daher, dass alle Verbindungen zwischen einem internen und einem externen Netz ausschließlich über diesen Rechner geleitet werden, so dass er sozusagen an vorderster Front als eine Bastion zur Angriffsabwehr dient.

Die Aufgabe einer Proxy-Firewall besteht darin, gegenüber dem Client als Server aufzutreten sowie für den Server den Part des Clients zu übernehmen und dadurch als Vermittler zwischen beiden zu agieren. Die Firewall übernimmt eine Art Stellvertreterfunktion. Die in diesem Abschnitt behandelten Firewalls stellen generische Proxy-Dienste zur Verfügung, da sie nicht auf einzelne Dienste der Anwendungsebene zugeschnitten sind, sondern mehrere Dienste unterstützen.

Bei einem Verbindungsaufbau wendet sich der Client an die Proxy-Firewall, die die Zulässigkeit des beantragten Verbindungsaufbaus überprüft. Da es sich bei der Proxy-Firewall um eine eigenständige Softwarekomponente handelt, ist sie fähig, Zustandsinformationen zu verwalten und Sicherheitsdienste durchzuführen, die über einfache Kontrollen hinausgehen. Dazu gehören insbesondere die Authentifikation des aufrufenden Clients und die Protokollierung durchgeföhrter Aktivitäten. Die Authentifikation kann zum Beispiel unter Nutzung des Kerberos-Systems (vgl. Abschnitt 10.4.2) realisiert werden.

Eine Proxy-Firewall übernimmt die Funktion eines Vermittlers von TCP- und UDP-Verbindungen zwischen Client und Server. Die Fähigkeit, Zustandsinformationen zu speichern, bietet analog zu den dynamischen Paketfiltern die Möglichkeit, auch für UDP-Pakete Anfragen, die von einem internen Client an einen externen Server gerichtet sind, von dessen Antworten zu unterscheiden. Dazu müssen von allen ausgesendeten Paketen deren Adressen und Ports gespeichert werden. Nur Pakete, deren Empfängeradresse sowie -port mit der gespeicherten Sendeadresse sowie dem gespeicherten Sendeport übereinstimmen und für die zusätzlich gilt, dass deren Sendeadresse und -port mit der gespeicherten Empfängeradresse und dem Empfängerport übereinstimmen, werden als externe Antworten auf eine Anfrage zugelassen.

Zugriffsrestriktionen können ferner von Kontextinformationen abhängig gemacht werden, wie zum Beispiel die Anzahl der bereits bestehenden Ver-

Bastion-Rechner

Proxy

Verbindungsaufbau

Vermittler

differenzierte Kontrollen

bindungen beim FTP-Dienst oder der aktuellen Uhrzeit. Auf diese Weise lässt sich eine Überlastung des Dienstes zu Stoßzeiten verhindern. Diese bedingten Kontrollen ermöglichen eine wesentlich differenziertere Filterung von Zugriffen als mit einfachen, zustandslosen Paketfiltern.

Vorteile und Grenzen von Proxy-Firewalls

Vorteile

Der generische Proxy-Dienst arbeitet anwendungs- und herstellerunabhängig, so dass unterschiedliche Anwendungen auf einem einheitlichen Dienst aufsetzen können. Da viele Anwendungsdienste (u.a. SMTP, HTTP, NNTP) auf einem Store-and-forward-Prinzip basieren, eignen sie sich sehr gut für eine vermittelnde Filterung. Eine Proxy-Firewall verhindert die direkte Etablierung einer TCP-Verbindung zwischen Client und Server durch die Firewall hindurch, die ja bei Paketfiltern möglich ist. Die vermittelnde Weiterleitung von Datenpaketen ermöglicht es, auch für bestehende Verbindungen noch Kontrollen durchzuführen. Die Erweiterung der Sicherheitsdienste der Proxies um Authentifikations- und Protokollierungsfähigkeiten ist ein weiterer wichtiger Vorteil dieser Firewall-Klasse gegenüber den einfachen Paketfiltern. Die zur Filterung verwendeten generischen Proxies sind allgemein nutzbar, so dass insbesondere auch neue Anwendungsdienste auf deren Basis gefiltert werden können. Da ein Proxy keine direkte Verbindung zwischen dem geschützten Netz und der Außenwelt erlaubt, führen derartige Firewalls implizit eine Adressumsetzung durch. D.h. durch den Einsatz von Proxys werden die realen IP-Adressen und Netzstrukturen hinter der Firewall nach außen verborgen.

Stateful vs. Proxy

Durch die qualitativen Verbesserungen von Paketfilter-Firewalls zu dynamischen, adaptiven und intelligenten Paketfiltern ist aber der Unterschied zwischen Paketfiltern und generischen Proxy-Firewalls nur noch sehr gering. Beide Firewall-Klassen sind funktional nahezu äquivalent.

Grenzen

nicht anwendungs-spezifisch

Da Proxy-Firewalls auf der Transportebene und damit unterhalb der Anwendungen angesiedelt sind, können sie nur allgemeine Vermittlungsdienste zur Verfügung stellen, ohne die Spezifika einzelner Anwendungsprotokolle zu berücksichtigen. Das bedeutet, dass aufgrund der fehlenden Kenntnisse über die Anwendung keine gezielte Beschränkung der Funktionalität der einzelnen Dienste möglich ist, sondern diese nur als Ganzes zugelassen oder verboten werden können. So sind beispielsweise HTML-Verbindungen zwar als Ganzes kontrollierbar, aber eine unterschiedliche Behandlung der übermittelten Datenpakete, abhängig davon, ob es sich um aktive Inhalte oder um unkritische Datenpakete handelt, ist nicht möglich.

Client-Code Modifikation

Ein Proxy-Firewall auf der Transportebene erfordert in der Regel die Modifikation der Software auf der Clientseite, da Aufrufe, die an einen bestimmten

Server gerichtet sind, explizit auf den Proxy-Server umzuleiten sind. Das bedeutet, dass auf der Clientseite der Source-Code zugreifbar sein muss, damit die Modifikationen durchgeführt werden können. Dies ist aber insbesondere bei proprietären Softwarepaketen selten der Fall.

14.1.4 Applikationsfilter

Wie der Name bereits sagt, sind Applikationsfilter (engl. *application-level gateway*) auf der obersten ISO/OSI-Schicht, also der Anwendungsschicht, angesiedelt. Sie erweitern die Dienste eines Verbindungs-Gateways, indem sie anstelle eines generischen einen dedizierten Proxy-Dienst zur Verfügung stellen. Im Gegensatz zu den Proxies der Transportebene besitzen Applikationsfilter Kenntnisse über die jeweilige Anwendung, die sie zu filtern haben, so dass sie dienstspezifische Kontrollen durchführen können. Ein anwendungsspezifischer FTP-Proxy kann beispielsweise die Nutzdaten analysieren und unterscheiden, ob es sich um Befehle wie PUT oder GET, um Dateiangaben wie HTTP-URLs oder um Programme handelt, und diese dann gemäß der festgelegten Sicherheitsstrategie differenziert behandeln. Verbietet beispielsweise die Strategie schreibende Zugriffe auf einen anonymen FTP-Server, so muss der Applikationsfilter die FTP-PUT-Kommandos filtern. Kommerziell verfügbare Firewalls bieten meist standardmäßig Proxy-Server dediziert für wichtige TCP-basierte Dienste an. Dazu gehören Proxies für Telnet, FTP, SMTP und HTTP. Zu beachten ist, dass ein Applikationsfilter nur solche Dienste filtern kann, für die ein Proxy-Server existiert. Alle anderen Verbindungswünsche werden automatisch blockiert.

dedizierter Proxy

Beispiel 14.3 (Telnet-, SMTP-, NNTP- und FTP-Proxies)

In Abbildung 14.5 ist ein Beispiel für einen Applikationsfilter skizziert, der dedizierte Proxies für den Telnet-, SMTP-, NNTP- und FTP-Dienst enthält. Der Applikationsfilter schottet das zu schützende interne Netz, das LAN, gegen das unsichere Internet ab. Alle Nachrichten vom oder in das LAN müssen den Filter über die beiden Schnittstellen passieren. Der FTP-Proxy kontrolliert alle FTP-Zugriffe zwischen LAN und Internet. Hier können die bereits angesprochenen protokollspezifischen Analysen durchgeführt und zum Beispiel auch Zugriffsprofile von einzelnen Benutzern erstellt werden. Der Telnet-Proxy (remote login) kann so konfiguriert sein, dass Zugriffe externer Benutzer zunächst authentifiziert werden. Der SMTP-Proxy⁴ hat zu verhindern, dass ein Internetbenutzer interaktiv mit einem potentiell fehlerhaften sendmail-Programm kommuniziert. Dieses Programm ist

Applikationsfilter

⁴ Filtern von E-Mail-Verkehr.

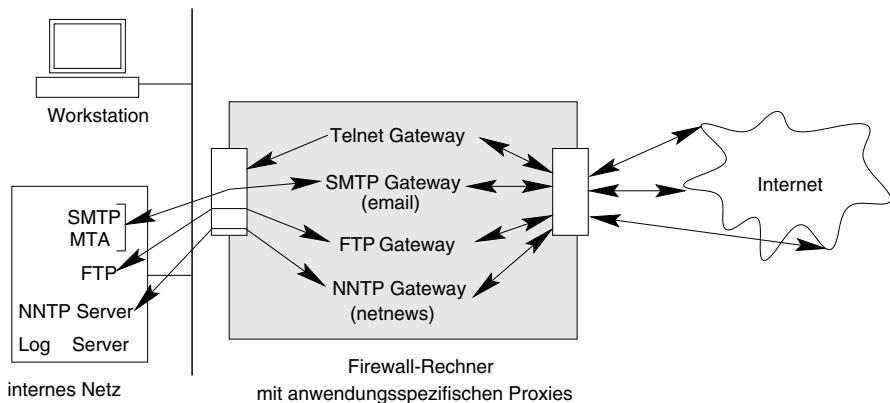


Abbildung 14.5: Applikationsfilter

weit verbreitet, da es sehr leistungsfähig ist. Gleichzeitig ist es jedoch sehr umfangreich und schwer zu konfigurieren, so dass es in der Vergangenheit bereits häufig die Ursache von Sicherheitsproblemen war und auch nach wie vor ist. Zur Abwehr von Angriffen könnte der Proxy ankommende Nachrichten zunächst auf die Festplatte schreiben und im Hinblick auf bekannte Angriffsversuche untersuchen, bevor er sie an den Nachrichten-Agenten (MTA) des Mail-Dienstes weiterreicht.



Schutz des Firewalls

Um die Firewall-Rechner, die die Proxy-Server beherbergen, vor Angriffen zu schützen, sollten diese nur eine rudimentäre Funktionalität zur Verfügung stellen. Das bedeutet, dass Binaries von Kommandos und von Bibliotheken, die nicht zur Aufrechterhaltung der Firewall-Funktionalität benötigt werden wie beispielsweise Editoren oder grafische Oberflächen, entfernt werden sollten. Weiterhin ist die gleichzeitige Verwendung von Firewall-Rechnern als dedizierte Server, wie NFS- oder WWW-Server, zu vermeiden und auch normale Benutzer-Accounts sollten auf diesen Rechnern nicht zugelassen werden.

Des Weiteren sollten keine Standardübersetzer, wie der C-Compiler, installiert sein, so dass ein Angreifer keine Programme auf der Firewall übersetzen und binden lassen kann.

Vorteile und Grenzen von Applikationsfiltern

Mit Applikationsfiltern können differenzierte Authentifikationen und Überprüfungen durchgeführt werden. Da die Filter zustandsbehaftet sind, lassen sich Nutzungsprofile erstellen und anhand auffälliger Zugriffsmuster Angriffsversuche auf das zu schützende Netz frühzeitig erkennen. Derartige Analysen möglicher Einbruchsversuche werden häufig durch spezifische Softwarekomponenten durchgeführt und sind unter dem Begriff

Intrusion Detection

Netz-basierte Intrusion Detection (IDS) bekannt. Erkannte Angriffsversuche können von den Analyseprogrammen protokolliert und/oder sofort dem zuständigen Administrator über Alarmsmeldungen z.B. via E-Mail angezeigt werden. Als Reaktion auf einen vermuteten Einbruchsversuch können alle nachfolgenden Zugriffe des Angreifers automatisch blockiert werden oder aber dem Angreifer wird eine präparierte Ausführungsumgebung zur Verfügung gestellt, in der er, ohne Schaden anzurichten, seine Angriffe durchführen kann. Derartige präparierte Umgebungen nennt man Honeypots. Sie werden eingesetzt, um Angreifer von dem lokalen Netz abzulenken und ihn in die präparierte Umgebung zu locken. Es ist aber sicherlich nur eine Frage der Zeit, bis Angreifer diese Ablenkung erkennen und ihre Angriffsversuche auf anderem Weg fortsetzen.

Eine wesentliche Aufgabe von Applikationsfiltern ist neben der Überwachung und Filterung auch die anwendungsspezifische Protokollierung von Zugriffen, so dass auf dieser Basis eine Abrechnung der in Anspruch genommenen Dienstleistungen durchgeführt werden kann. Die Kenntnisse über die zu filternde Anwendung ermöglichen es beispielsweise, dass ein FTP-Proxy-Server nicht alle übertragenen Datenpakete protokollieren muss, sondern sich darauf beschränken kann, nur die übertragenen Befehle sowie die Antworten darauf zu loggen. Auf diese Weise erhält man sehr viel kleinere und übersichtlichere Log-Dateien. Sowohl das Accounting als auch die Zugriffskontrolle kann benutzerspezifisch erfolgen, da im Gegensatz zur Netzwerkebene, die nur IP-Adressen kennt, auf der Anwendungsebene detailliertere Informationen zur Verfügung stehen.

Da alle Datenpakete durch die Firewall geschleust werden, kann sie lokale Kopien der angeforderten Daten in einem lokalen Cache verwalten. Bei einer großen Anzahl von Zugriffen kann dies zu einer signifikanten Verbesserung der Performanz führen. Andererseits kann es aber durch die Anwendungsproxies auch zu erheblichen Performanzeinbußen kommen, da in einer Anwendungsfirewall für jeden Proxy-Dienst in der Regel ein eigener Prozess gestartet wird.

Applikationsfilter sind zugeschnitten auf Dienste der Anwendungsebene, so dass sie eine dienstspezifische Filterung ermöglichen und damit eine weit stärkere Differenzierung erlauben, als dies mit den generischen Proxies der Transportebene möglich ist. So lassen sich insbesondere auch Dienste mit eingeschränkter Funktionalität betreiben. Ein Beispiel hierfür ist ein eingeschränkter FTP-Dienst, der die Ausführung von Put-Kommandos nicht unterstützt. Demgegenüber war bisher nur das Alles-oder-Nichts-Prinzip realisierbar. Da Applikationsfilter die Nutzdaten der übertragenen Datenpakete analysieren, lassen sich auch aktive Inhalte oder MIME-Dokumente, filtern, indem die Nutzdaten nach spezifischen Schlüsselworten durchsucht

Accounting

Performanz

dedizierte Kontrollen

werden. So kann ein HTTP-Proxy in einer HTML-Seite nach dem Schlüsselwort <applet> suchen, diesen Eintrag ersetzen und an seiner Stelle dem Benutzer eine Mitteilung in das Browserfenster schreiben, mit der Information, dass das an dieser Stelle normalerweise erscheinende Applet blockiert wurde. Das Blockieren von Java-Class Dateien scheint zunächst relativ einfach, da jede solche Klasse mit dem Hexadezimalen Bytecode CAFEBABE beginnen muss. Werden solche Klassen jedoch z.B. mittels ZIP komprimiert und in dieser gepackten Form durch die Firewall geschleust, so kann diese Überprüfung bereits nicht mehr erfolgreich durchgeführt werden.

Standardproxies

Der Zuschnitt auf einzelne Dienste erfordert jedoch, dass für jeden zu filternden Dienst ein eigener Proxy-Server existiert. Um flexibel auf Änderungen und Ergänzungen des Dienstangebots reagieren zu können, ist es also wünschenswert, dass entsprechende Proxies weitestgehend automatisch aus einer Dienstspezifikation generiert und einem bestehenden Filtersystem hinzugefügt werden können. Dies wird jedoch von existierenden Produkten noch nicht unterstützt, so dass man sich in der Praxis meist mit einigen wenigen Standarddiensten zufrieden geben muss, für die die benötigten Proxies bereits existieren. Dazu gehören HTTP, Telnet, SMTP, NNTP und FTP.

Grenzen

Für Applikationsfilter, wie für alle anderen Filterungskomponenten, gilt, dass sie die Protokollschwächen und Probleme der gefilterten Dienste und Protokolle höchstens geringfügig beheben können. Mögliche Diensterweiterungen sind neben der bereits angesprochenen Authentifikation auch die Verschlüsselung des Datenverkehrs zwischen dem Filter und seiner Außenwelt. Die Hauptaufgabe eines Applikationsfilters besteht jedoch darin, sicherheitskritische Aktionen von Benutzern im Verlauf einer Dienstnutzung bzw. beim Dienstauftruf zu erkennen und zu blockieren. Das Spektrum an differenzierten Filterungsmöglichkeiten eines dedizierten Proxies wird dabei im Wesentlichen durch den zu filternden Dienst bestimmt. Ermöglichen die gefilterten Dienste keine Unterteilung in sicherheitskritische und zu blockierende sowie in unkritische Aktionen, sondern erbringen ihre Funktionalität nur dann vollständig, wenn alle Aktionen zugelassen sind, so ist auch durch den Einsatz von Applikationsfiltern keine nennenswerte Steigerung der Sicherheit im Vergleich zu generischen Proxies zu erzielen.

14.1.5 Architekturen

Üblicherweise werden Kombinationen von Paket- und Applikationsfiltern in einer Firewall-Architektur eingesetzt. Typische Konfigurationen sind Dual-Homed Firewalls, überwachte Rechner (engl. *screened-host*) und überwachte Teilnetze (engl. *screened-subnet*). Diese Architekturen werden wir im Folgenden kurz vorstellen.

Dual-Homed Firewall

Dual-Homed Rechner besitzen zwei Netzwerkschnittstellen, die Verbindungen zu zwei Netzsegmenten herstellen. Der Rechner ist damit sozusagen in zwei Teilnetzen „zu Hause“ (dual-homed). Verbindungsrechner, wie Bridges auf der OSI-Schicht 2 und Router auf der Schicht 3, sind typische Komponenten, die mehrere Netzsegmente koppeln und für jedes Segment eine eigene Netzwerkkarte besitzen. Die wesentliche Eigenschaft des Dual-Homed Rechners besteht darin, dass er keine IP-Routing und -Forwarding Dienste anbietet. Durch die Dienstdeaktivierung werden die beiden Netzsegmente voneinander isoliert, so dass Pakete nicht direkt weitergeleitet werden, sondern analysiert und gegebenenfalls umgelenkt werden können.

dual-homed

Eine Dual-Home Firewall-Architektur⁵ basiert auf einem solchen Host, der ein Bastionsrechner mit einem Applikationsfilter ist. Ergänzend dazu werden dem Bastionsrechner noch zwei Paketfilter vor und nachgeschaltet. Abbildung 14.6 veranschaulicht diese Architektur.

Isolierung

Konfiguration

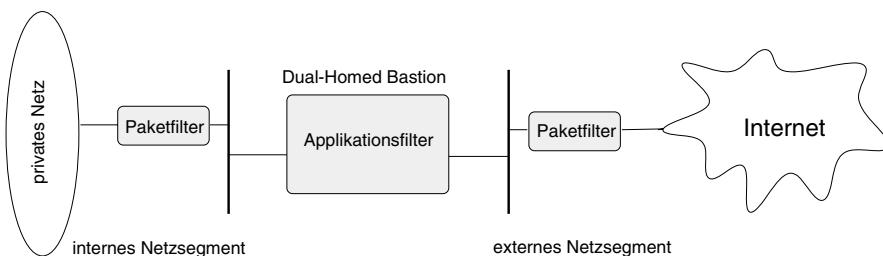


Abbildung 14.6: Dual-Homed Firewall

Die Paketfilter, die den Bastionsrechner mit dem externen bzw. mit dem internen Netz verbinden, haben die Aufgabe, nur korrekt adressierte IP-Pakete an den Bastionsrechner weiterzuleiten. Sie bilden einen ersten einfachen Schutzwall, der durch die dedizierten Proxy-Dienste des Applikationsfilters verstärkt wird. Will ein Benutzer einen Dienst im unsicheren Netz nutzen, so muss er sich dazu zunächst auf dem Dual-homed Host einloggen, die dortigen Dienste nutzen und ggf. erarbeitete Ergebnis-Daten manuell vom Host exportieren. Dies ist natürlich wenig benutzungsfreundlich und ein dazu benötigtes Remote Login wird auch nicht von allen Betriebssystemen uneingeschränkt unterstützt (z.B. erlaubt Windows ein Remote Login nur mit Einschränkungen). Es bedeutet natürlich auch, dass auf einem Dual-homed Host Benutzerkennungen eingerichtet und Anwendungsdienste konfiguriert werden müssen. Dadurch ergeben sich die üblichen Sicherheitsprobleme wie Buffer-Overflow Schwachstellen etc. Da zudem der Dual-homed Host auch

⁵ Sie ist auch als Split Screened Subnet-Architektur bekannt.

einen Single-Point of Failure darstellt, bietet diese Architektur für heutige Nutzungsszenarien nur eine unzureichende Lösung und ist kaum noch üblich.

Screened-Host Firewall

screened-host

Im Gegensatz zum Dual-Homed ist ein überwachter Firewall lediglich mit dem privaten Netz verbunden und benötigt damit nur eine Netzwerkschnittstelle. Um den Bastionsrechner abzuschirmen, wird ebenfalls ein Paketfilter und zwar in Form eines Screening-Routers eingesetzt. Dieser überwacht den Datenverkehr zwischen dem externen Netz und dem Bastionsrechner im internen Netz. Abbildung 14.7 veranschaulicht diese Architektur. Anders als

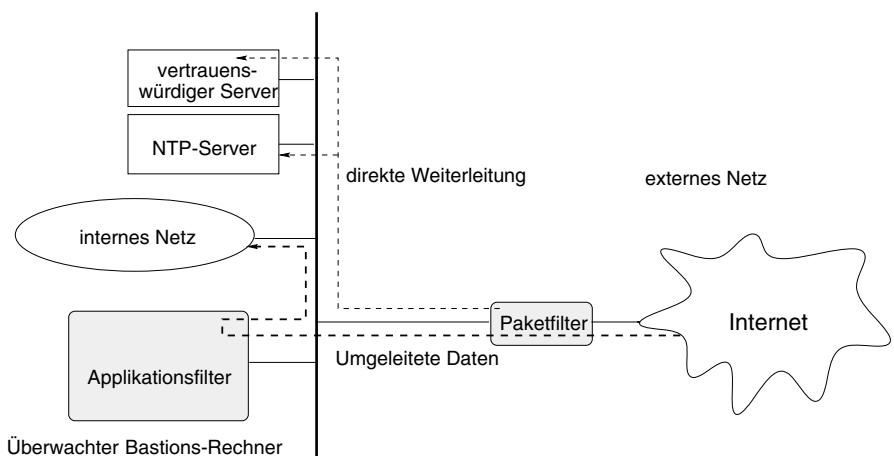


Abbildung 14.7: Screened-Host Firewall

bei der Dual-Homed Firewall liegen hier Bastionsrechner und Paketfilter im gleichen Netzsegment. Der Screening-Router ist deshalb so zu konfigurieren, dass jeglicher Datenverkehr aus dem externen in das interne Netz zunächst diesen Filter durchläuft und dann an den Bastionsrechner zur weiteren Bearbeitung weitergeleitet wird. Der Router ist somit eine sicherheitskritische Komponente, deren Routingtabelle vor unautorisierten Zugriffen besonders zu schützen ist.

*direkte
Paketvermittlung*

Im Gegensatz zur Dual-Homed Konfiguration ist es hier jedoch möglich, dass der Paketfilter spezielle Datenpakete direkt an den Empfänger weiterleitet, also den Applikationsfilter umgeht. Dies kann für unkritische Dienste wie dem Zeitdienst oder aber auch für solche vertrauenswürdigen Dienste sinnvoll sein, für die keine Proxies im Applikationsfilter existieren. Natürlich birgt diese größere Flexibilität bei unsachgemäßem Einsatz auch erhebliche Sicherheitsrisiken. Screened-Host Konfigurationen sollten deshalb nur dann

eingesetzt werden, wenn diese Flexibilität tatsächlich benötigt wird und das interne Netz keine sehr hohen Sicherheitsanforderungen stellt.

Screened-Subnet Firewall

Durch eine Ergänzung der Screened-Host Konfiguration um ein zusätzliches Netzsegment wird das interne Netz vom externen isoliert, wodurch sich ein höherer Grad an Sicherheit erzielen lässt. Im Gegensatz zur Dual-Homed Konfiguration ist der Bastionsrechner aber nicht mit beiden Netzsegmenten verbunden, sondern nur mit dem externen. Zur Errichtung der isolierten Netzsegmente werden dem Bastionsrechner je ein Screening-Router vor- und nachgeschaltet, wodurch ein nach innen und außen isoliertes Subnetz entsteht, das häufig als Perimeter Netzwerk oder demilitarisierte Zone⁶ (DMZ) bezeichnet wird. Das Subnetz kann neben dem Applikationsfilter noch weitere dedizierte Server enthalten wie zum Beispiel einen WWW-, DNS-, Mail- oder FTP-Server, für die ein kontrollierter Zugang vom bzw. zum externen Netz benötigt wird. Durch die Einbettung dieser Server in das geschützte Subnetz wird erreicht, dass Strukturen des internen Netzes (u.a. Benutzernamen, Rechnernamen, Mail-Alias-Namen) versteckt werden und Angreifer, die es zwar geschafft haben, einen dieser Server unter ihre Kontrolle zu bringen, immer noch keinen direkten Zugriff auf das interne Netz besitzen. Sie müssen dazu noch den zweiten Paketfilter erfolgreich passieren.

screened-Subnet

Die Architektur eines Screened-Subnet Firewalls ist in Abbildung 14.8 zu sehen.

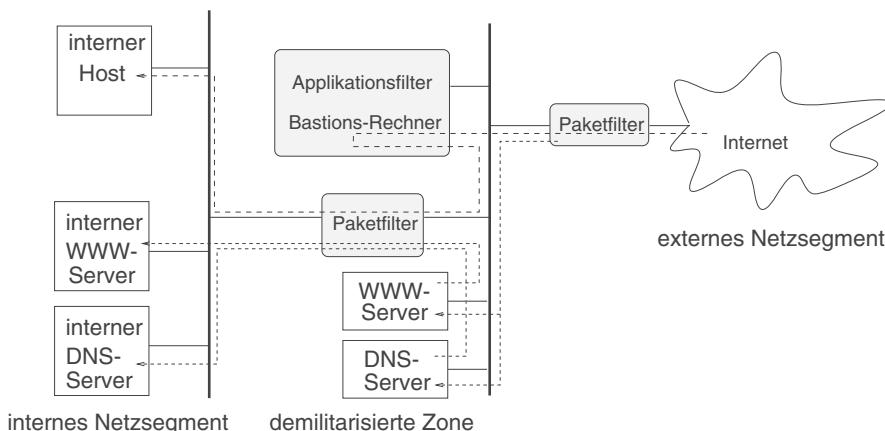


Abbildung 14.8: Screened-Subnet Firewall

⁶ Nach dem Streifen Niemandsland zwischen Nord- und Südkorea.

eingeschränkte
Flexibilität

Analog zur Screened-Host Konfiguration erlaubt die Konfiguration mit überwachten Teilnetzen, dass der Applikationsfilter gezielt umgangen werden kann und dass Datenpakete direkt Servern in der demilitarisierten Zone zugestellt werden können. Durch das zusätzliche Netzsegment, das die demilitarisierte Zone etabliert, ist es hier jedoch nicht mehr möglich, IP-Pakete direkt zwischen externem und internem Netz auszutauschen. Der Grad an Sicherheit ist somit gegenüber dem Screened-Host Ansatz erhöht.

Fazit

Firewalls zählen zu den am häufigsten eingesetzten Schutzkonzepten im Bereich der Rechnernetze. Die vorangegangene Diskussion der verschiedenen Firewall-Typen und -Konfigurationen sollte deutlich gemacht haben, dass ein gut konfiguriertes Firewall-System erheblich zur Steigerung der Sicherheit beitragen kann. Die Qualität einer Firewall wird maßgeblich von der zugrunde liegenden Sicherheitsstrategie bestimmt. Kenntnisse über die lokale Netzinfrastruktur sind unabdingbar, um sicherzustellen, dass eine eingerichtete Firewall nicht gezielt umgangen werden kann.

Web-Services

Problematisch ist die Vielzahl von Web-Services und Web-basierter Anwendungen. Damit möglichst viele Anwender diese Dienste nutzen können, werden sie über das HTTP-Protokoll und damit (standardmäßig) über den Port 80 angeboten. Dieser Port ist jedoch in der Regel von den Paketfilter-Firewalls freigeschaltet und bietet somit ein großes Einfallstor für Angriffe. Applikationsfilter-Firewalls könnten konzeptuell natürlich Maßnahmen zur Abwehr von Angriffen über den Port 80 bereit stellen, da sie das anwendungsbezogene Filtern von Daten ermöglichen. Dies erfordert jedoch, dass die Firewall für jeden dieser Web-Dienste einen dedizierten Proxy enthält. Das Konfigurieren und Verwalten derartiger Proxy-Parks wird sehr schnell sehr komplex und schwierig, da für diese Nicht-Standarddienste noch keine allgemein verfügbaren Proxies existieren. Es ist somit wichtig, Sicherheitskonzepte und Maßnahmen in die Anwendungen selber zu integrieren anstatt auf den Schutz von Firewalls und Intrusion Detection Systemen zu vertrauen.

keine Virenabwehr

Grenzen der Möglichkeiten von Firewalls liegen in der Abwehr von Angriffen, die sich aus den übertragenen Daten ergeben. Da Firewalls keine semantische Überprüfung der Daten durchführen, können keine Aussagen über deren Funktionalität gemacht werden. Durch eine Stichwortsuche lassen sich nur sehr offensichtliche Angriffe aufdecken, wie zum Beispiel Code, der eine bekannte Virenkennung enthält. Durch allgemein übliche Datenkomprimierungs- und Codierungstechniken lassen sich derartige Informationen leicht verschleiern.

Tunneling-Technik

Firewalls stoßen aber auch bei der Verwendung der Tunneltechnik (engl. *tunneling*) an ihre Grenzen. Tunneling (vgl. Abschnitt 14.2.2) ist eine in der

Netzwelt sehr weit verbreitete Vorgehensweise, um Datenpakete gekapselt in einem „Umschlag“ durch ein Netzsegment, das nur als Transportsystem genutzt wird, zu schleusen. Der Umschlag besteht aus den Informationen (u.a. Adressen, Prüfsummen), die das Transportsystem zur Beförderung des Datenpaketes benötigt. Die eingekapselten Daten werden dann erst am Zielort weiter interpretiert. Tunneling wird auch eingesetzt, um Daten in einem verteilten Unternehmensnetz unter Zuhilfenahme eines dazwischen liegenden Transportsystems, z.B. des Internets, gekapselt zu transportieren.

Problematisch ist, dass das Tunneling dazu missbraucht werden kann, die Filterregeln einer Firewall zu umgehen. Voraussetzung dafür ist aber, dass ein interner und ein externer Rechner zusammenarbeiten. Wenn diese die zu tunnelnden Pakete in ein Paket oder in eine Nachricht so einkapseln, dass das gekapselte Paket von der Firewall als zulässig betrachtet und weitergeleitet wird, so können beliebige Daten durch die Firewall hindurch geschleust werden. Die gekapselten Daten werden dann vom Empfänger ausgepackt und weiterverarbeitet. Streng genommen lässt sich das mit dieser Art von Tunneling verbundene Problem auf das allgemeine Problem reduzieren, das entsteht, wenn ein autorisierter Benutzer seine Berechtigungen missbraucht. Von Firewalls kann man selbstverständlich hierfür keine Lösung erwarten. Dies ist vielmehr die Aufgabe der verwendeten Software, die durch ein gezieltes Rechtemanagement dafür zu sorgen hat, dass zum einen die Möglichkeiten zum Missbrauch durch eine minimale Rechtevergabe eingeschränkt und zum anderen die Auswirkungen eines möglichen Missbrauchs durch vollständige und differenzierte Kontrollen eingegrenzt werden.

Tunneln des
Firewalls

Firewalls sind eine pragmatische Antwort auf einige bestehende Sicherheitsprobleme. Sie versuchen mit mehr oder minder gezielten Maßnahmen, bekannte Sicherheitslöcher a posteriori zu stopfen. Sie werden zwar evolutionär weiterentwickelt, indem kontinuierlich auf aufgetretene Sicherheitsprobleme reagiert und die Firewall-Fähigkeiten entsprechend angepasst werden. Die Kernprobleme, nämlich Designfehler der verwendeten Software und das Fehlen einer systematischen Integration von Sicherheitsfunktionen in den Software-Engineering-Prozess, werden aber mit Firewalls nicht beseitigt. Das Paradigma der Perimeter-Sicherheit, das durch die Firewall-Technologie umgesetzt wird, ist unzureichend, um vernetzte Systeme geeignet zu schützen. Firewalls sind als Maßnahmen einzurordnen, die die existierenden Mängel heutiger Kommunikationsprotokolle nicht beheben, sondern durch zusätzliche Kontrollen versuchen, die Auswirkungen dieser Mängel zu begrenzen. Das nachträgliche Stopfen von Sicherheitslöchern ist jedoch, wie wir schon mehrfach hervorgehoben haben, unbefriedigend, da zum einen versucht werden kann, diese zusätzlichen Kontrollen zu umgehen, um die Lücken, die ja nach wie vor vorhanden sind, weiter auszunutzen. Zum

Stopfen von
Sicherheitslücken

anderen besteht die Gefahr, dass durch solche nachträglichen Bereinigungsmaßnahmen die Sicherheitsprobleme nur unvollständig behoben werden oder durch mangelhaft implementierte Kontrollen sogar noch zusätzliche Probleme entstehen. In den folgenden Abschnitten widmen wir uns Ansätzen, die versuchen, durch integrative Sicherheitsmaßnahmen zur Steigerung der Sicherheit beizutragen.

14.2 Sichere Kommunikation

Die entwickelten und in der Entwicklung befindlichen Protokolle zur Errichtung sicherer Kommunikationsverbindungen verwenden unterschiedliche Kombinationen von Mechanismen zur Verschlüsselung, Authentifikation und zur Berechnung eindeutiger Hashwerte. Hierbei werden in der Regel diejenigen Verfahren eingesetzt, die wir bereits im Kapitel 7 behandelt haben. Dennoch unterscheiden sich die Protokolle wesentlich durch ihre angebotenen Fähigkeiten, die sich aus der Kombination der Mechanismen und aus der Einordnung in den OSI-Stack ergeben.

Kommunikations- versus Anwendungssicherheit

Im Groben kann man zwischen solchen Protokollen unterscheiden, die als reine Kommunikationsprotokolle dienen und einen Nachrichtenstrom zwischen Kommunikationspartnern absichern, und solchen, die auf der Anwendungsebene dediziert Anwendungsdaten absichern. Klassische Vertreter der ersten Klasse von Protokollen sind das IPSec, TLS oder auch SSH, während XMLSec, S/MIME oder PGP zu den bekanntesten Vertretern der zweiten Protokollklasse zählen. Auch das Signal-Protokoll fällt eher in die Klasse der Anwendungsprotokolle, da es auf Instant-Messaging-Dienste zugeschnitten ist.

*Kommunikations-
protokolle*

Sichere Kommunikationsprotokolle sind dadurch charakterisiert, dass eine Interaktion zwischen den beteiligten Kommunikationspartnern stattfindet, während der sich die Partner authentifizieren und sich auf die zu verwendenden Verfahren einigen, sowie gemeinsame Kommunikations- und falls erforderlich auch MAC-Schlüssel austauschen. Gängige Protokolle zur Etablierung eines sicheren Kommunikationskanals zwischen Partnern gewährleisten in der Regel die Schutzziele der Authentizität, Integrität und Vertraulichkeit, wobei sich die Authentizität auf den Datenursprung und die Integrität sich auf einzelne Datenpakete und nicht auf eine ganze Transaktion bezieht. Nachdem diese vorbereitende Phase, die in der Regel eine Anzahl von auszutauschenden Nachrichten erfordert und entsprechend aufwändig ist, erfolgreich abgeschlossen ist, können die beteiligten Kommunikationspartner über den etablierten sicheren Kanal effizient und sicher ihre Daten

austauschen. Der Overhead der vorbereiteten Schritte ist immer dann sinnvoll, wenn der Kanal für einen ganzen Datenstrom etabliert wird (z.B. Download von WWW-Seiten), so dass dieser mit möglichst geringen Verzögerungen ver- und entschlüsselt werden kann. Der Schutz der Daten beginnt direkt vor deren Übertragung über den Kanal und endet auf der Empfängerseite, wo eine Entschlüsselung stattfindet und die Daten im Klartext zur Weiterverarbeitung vorliegen.

Demgegenüber sind Anwendungsprotokolle dadurch charakterisiert, dass einer der Partner alle zur sicheren Kommunikation benötigten Informationen (u.a. verwendete Verfahren, verwendete Schlüssel, Zertifikate) in geeigneter Weise mit der zu übermittelnden Nachricht kombiniert und diese Informationen in einem gemeinsamen Schritt übermittelt (z.B. für eine E-Mail oder für ein XML-Dokument). Der Empfänger kann zu einem späteren Zeitpunkt diese Anwendungsdaten prüfen und verarbeiten (u.a. authentifizieren des Senders, Integritätsprüfung, Entschlüsselung). Sender und Empfänger von Nachrichten sind hierbei also in der Lage, ihre jeweiligen Bearbeitungsschritte zu bündeln und, vergröbert betrachtet, in einem Schritt durchzuführen. Man spricht deshalb auch von einem One-Pass-Processing⁷ der Daten. Im Gegensatz zu den Kommunikationsprotokollen ermöglichen die sicheren Anwendungsprotokolle, dass der Schutz der Daten auch noch nach der erfolgten Übertragung erhalten bleibt. So können zum Beispiel eMails verschlüsselt und signiert gespeichert werden.

Anwendungs-
protokolle

Zum anderen wird durch die Einordnung der Protokolle in die Schichten des ISO/OSI-Modells ein allgemeiner Rahmen festgelegt, der die jeweiligen Fähigkeiten der Protokolle begrenzt.

14.2.1 Verschlüsselungs-Layer

Wir unterscheiden grob zwischen zwei Klassen von Ansätzen, nämlich der Klasse der Verbindungsverschlüsselung und der Klasse der Ende-zu-Ende-Verschlüsselung. Sicherheitsprotokolle der Klasse der Verbindungsverschlüsselung finden sich auf den unteren ISO/OSI-Schichten 1 und 2, während Ende-zu-Ende-Sicherheitsprotokolle auf den höheren Schichten 3 bis 7 angesiedelt sind.

Ansätze

Je tiefer ein Protokoll in der OSI-Schicht eingeordnet wird, desto allgemeiner und weniger differenziert sind seine Sicherheitsdienste, die dafür aber für die Protokolle der höheren Schichten transparent nutzbar sind. Die Einordnung eines Sicherheitsprotokolls auf einer höheren OSI-Schicht ermöglicht Protokolle, die zwar dediziert für jede Anwendung entwickelt werden müssen, dafür aber auch einen auf spezifische Anforderungen der

generelle
Charakteristika

⁷ Manchmal spricht man hier auch von One-Step Processing.

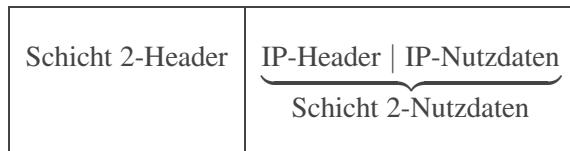
Anwendung zugeschnittenen Sicherheitsdienst bieten können. Zum Beispiel können eMails einzeln und unabhängig verschlüsselt und bei Bedarf signiert werden. Im Folgenden beschäftigen wir uns zunächst mit allgemeinen Charakteristika der zwei Klassen von Ansätzen bevor wir uns konkreten Protokollen des Kommunikations- und Transportsystems zuwenden. Auf sichere Anwendungsprotokolle der Schicht 7 gehen wir abschließend ein.

Verbindungsverschlüsselung

Schicht 2

Bei der Verbindungsverschlüsselung (engl. *link encryption*) werden die Daten unmittelbar vor ihrer Übertragung, also auf den Schichten 1 oder 2, verschlüsselt. Link-Verschlüsselung wird heute vordringlich in drahtlosen Netzen, wie WLAN und Bluetooth (vgl. Kapitel 15) verwendet. Bei WLAN erfolgt beispielsweise eine direkte, verschlüsselte Kommunikation zwischen dem Endgerät und dem Access-Point.

In Netzen, die nicht nur aus einem lokalen Netz (LAN), sondern aus mehreren gekoppelten Teilnetzen bestehen, werden Datenpakete über Kopplungs- bzw. Vermittlungskomponenten, wie Bridges, Switches oder Router, weitergeleitet. Die Ende-zu-Ende-Zustellung der Pakete erfolgt durch Router auf der Basis der IP-Empfängeradresse in dem jeweiligen Datenpaket. Die IP-Adresse wird bereits mit dem Header der Schicht 3-Protokolle den Nutzdaten des Datenpaketes hinzugefügt und mit dem Übergang auf die Verbindungs Ebene durch den Header der Schicht 2-Protokolle ergänzt, wodurch der IP-Header ein Teil der Nutzdaten der Schicht 2 wird:



IP-Adressen

Da bei der Verbindungsverschlüsselung die Nutzdaten der Schicht 2 verschlüsselt werden, sind auch die IP-Adressen ein Bestandteil des Kryptotextes. Somit müssen die Router jedes Paket zunächst entschlüsseln, um an die für das Routing benötigten Informationen zu gelangen. Das hat zur Konsequenz, dass die Nutzdaten nur während der direkten Übermittlung zwischen Routern verschlüsselt sind, in den Vermittlungsrechnern (Routern) jedoch im Klartext vorliegen. Abbildung 14.9 veranschaulicht dies.

sicherheitskritische Router

Aus dem Gesagten ergibt sich unmittelbar, dass beim Einsatz kryptografischer Verfahren auf dieser niedrigen Schicht die Router in hohem Maß für die Kommunikationssicherheit verantwortlich sind. Zum einen bilden sie ein lohnendes Angriffsziel, da sie die Datenpakete im Klartext zwischenspeichern, und zum anderen müssen die Endsysteme und damit deren Nutzer der

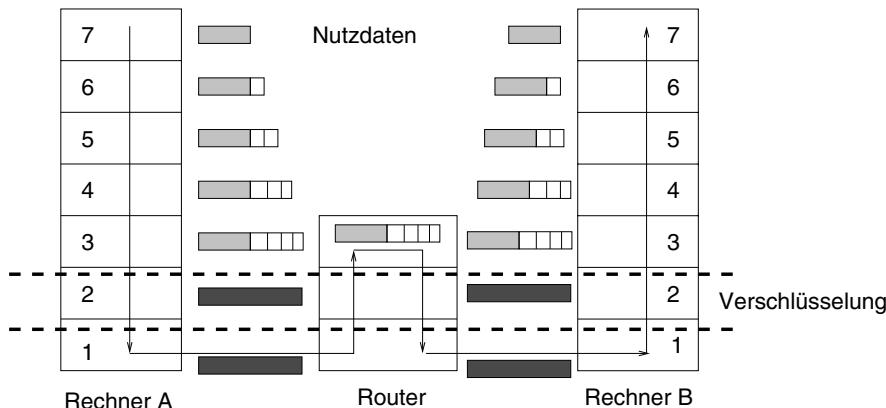


Abbildung 14.9: Verbindungsverschlüsselung

korrekten Arbeitsweise dieser Vermittlungskomponenten vertrauen. Dies ist aber gerade in großen Netzen wie dem Internet sehr problematisch, weil der Endbenutzer Komponenten vertrauen muss, über die er keine Kenntnis und erst recht keine Kontrolle besitzt.

Da die Datensicherungsebene (Schicht 2) nicht fähig ist, zwischen Endsystemen oder gar Prozessen bzw. Benutzern zu differenzieren, existiert für jede physische Verbindung zwischen unmittelbar benachbarten Systemen ein Schlüssel, der dann für jeden Nachrichtenaustausch zwischen diesen Endsystemen verwendet wird. Das bedeutet, dass für alle Prozesse dieser beiden Systeme, die miteinander kommunizieren, bzw. für alle Daten, die über diese physische Verbindung geleitet werden, derselbe Schlüssel benutzt wird und zwar völlig losgelöst von den unterschiedlichen Sicherheitsbedürfnissen dieser Prozesse bzw. Anwendungsdaten. Die Schicht 2 besitzt keine Kenntnisse über die spezifischen Anforderungen höherer Schichten, so dass die Sicherheitsdienste auf alle Datenpakete in uniformer Weise angewandt werden. Eine gezielte Auswahl oder auch gezielte Verstärkung bzw. Abschwächung der Sicherheitsmaßnahmen (Letzteres kann zum Beispiel aus Effizienzgründen attraktiv sein) ist aufgrund der mangelnden Kenntnisse jedoch nicht möglich. Die Authentifikation der Kommunikationspartner beschränkt sich auf eine grobgranulare Authentifikation von Endsystemen, da nur diese als Kommunikationspartner auf der Ebene 2 in Frage kommen. Eine dienst- oder benutzerspezifische Authentifikation ist wiederum aus Mangel an differenzierenden Informationen nicht möglich. Um dieses Defizit zu beheben, bedient sich beispielsweise das 802.11i-Protokoll (WLAN) des 802.1X Frameworks, das auf der Anwendungsebene angesiedelt ist und eine differenzierte Authentifikation ermöglicht (vgl. Abschnitt 15.4.6). Die klare OSI-Schichtung wird dadurch aufgeweicht.

Problembereiche

Die unspezifischen Kenntnisse haben schließlich auch zur Folge, dass auf der Ebene 2 nur eine Alles-oder-Nichts-Verschlüsselungsstrategie verfolgt werden kann, so dass unter Umständen auch zu viel verschlüsselt wird. Darunter kann die Effizienz und damit die Akzeptanz derartiger Protokolle erheblich leiden.

Vorteile

Diesen ganzen Einschränkungen steht als Hauptvorteil die einfache und vollständig transparente Handhabung der Sicherheitsprotokolle durch die Protokolle höherer Schichten gegenüber. Diese können unmodifiziert auf den Schicht 2-Protokollen aufsetzen, wodurch sie zumindest deren einfache sicherheitssteigernde Eigenschaften übernehmen. Weiterhin bietet die Verbindungsverschlüsselung den Vorteil, dass Angriffe, die auf eine Verkehrsflussanalyse und/oder auf die Erstellung von Kommunikationsprofilen abzielen, wesentlich erschwert werden, da ja die IP-Adressen in verschlüsselter Form übertragen werden. Noch ein Vorteil ergibt sich beim gleichzeitigen Einsatz von Firewalls und der Verbindungsverschlüsselung. Da die Nachrichten ab der Schicht 3 aufwärts vollständig im Klartext vorliegen, kann eine Firewall, die auf einer dieser höheren Ebenen angesiedelt ist, diese gezielt nach Schlüsselworten, Strukturen etc. filtern.

Ende-zu-Ende-Verschlüsselung

Schicht 3 und höher

Bei der Ende-zu-Ende-Verschlüsselung (engl. *end-to-end encryption*) werden die Nutzdaten eines Datenpakets vom Absender verschlüsselt und erst vom Empfänger wieder entschlüsselt. Das bedeutet, dass sie während des gesamten Transportes als Kryptotexte übertragen und auch verschlüsselt in den Vermittlungskomponenten zwischengespeichert werden. Abbildung 14.10 veranschaulicht dies und macht gleichzeitig deutlich, dass zwar die Nutzdaten verschlüsselt bleiben, dafür aber die Verkehrsdaten offen liegen, so dass Verkehrsflussanalysen möglich sind. Da das Risiko dieser Angriffe jedoch für viele Anwendungsbereiche als relativ gering eingestuft wird, bedeutet dies nur einen kleinen Nachteil, der darüber hinaus durch die Durchführung weiterer Verschlüsselungsschritte auf niedrigeren Ebenen ja auch auf einfache Weise beseitigt werden könnte. IPSec und TLS sind bekannte Vertreter der Ende-zu-Ende (E2E) Verschlüsselung.

Abhängig davon, auf welcher OSI-Schicht die Ende-zu-Ende-Sicherheitsprotokolle angesiedelt sind, besitzen sie unterschiedliche Fähigkeiten.

E2E-Verschlüsselung auf der Netzwerkebene

Schicht 3

Sicherheitsprotokolle auf der Netzwerkebene (Schicht 3) können zwar eine sichere Ende-zu-Ende-Verbindung zwischen verschiedenen, nicht notwendig physisch benachbarten Endsystemen realisieren, aber eine weitergehende Differenzierung ist nicht möglich. Dazu sind Informationen über Prozesse und/oder über die einzelnen Benutzer erforderlich. Solche Informationen

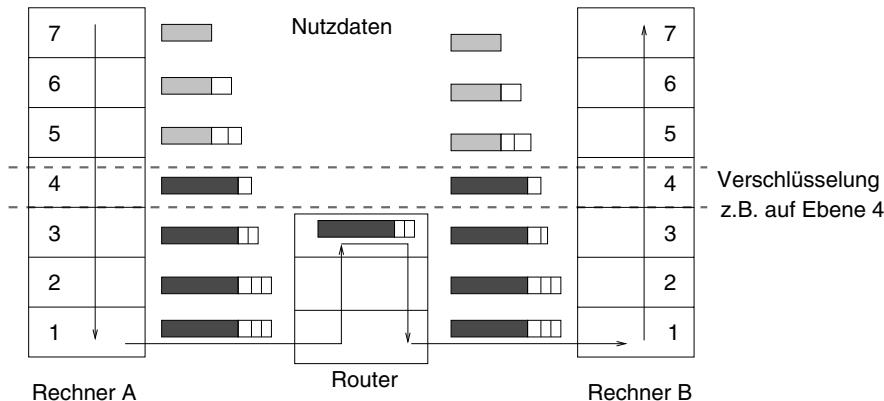


Abbildung 14.10: Ende-zu-Ende-Verschlüsselung auf Schicht 4

stehen aber erst mit Protokollen ab der Schicht 4, der Transportebene, bereit. Das hat zur Konsequenz, dass auf der Netzwerkebene Datenpakete verschiedener Prozesse nicht voneinander unterschieden werden können, so dass zur Sicherung der Kommunikationsverbindungen von Anwendungsprozessen, die unter Umständen völlig unterschiedliche Sicherheitsanforderungen besitzen, die gleichen Schlüssel verwendet werden. Auch die Authentifikation kann nur auf der Basis von Endsystemen stattfinden. Ansätze, die versuchen, durch benutzer- oder sitzungsbasierte Schlüssel eine Abhilfe zu schaffen, erfordern den Zugriff auf Informationen, die von Protokollen höherer Schichten zur Verfügung zu stellen sind und somit zusätzliche, nicht in die Protokolle der Schicht 3 integrierte Maßnahmen darstellen, wie die Verwaltung der Security Associations (SA) in einer SA-Datenbank beim IPSec-Protokoll (siehe Seite 751 ff).

Host-Schlüssel

Ein Vorteil der Schicht 3-Protokolle ist, dass sie Standardnetzdienste anbieten, die von beliebigen Anwendungen in transparenter und damit einfacher Weise genutzt werden können. Ein Beispiel für ein Sicherheitsprotokoll auf der Schicht 3 ist IPv6 als Nachfolger des bestehenden IPv4. IPv6 implementiert die mit der Internet-Sicherheitsarchitektur IPSec festgelegten Sicherheitsdienste. Auf IPSec gehen wir in Abschnitt 14.3 noch genauer ein.

E2E-Verschlüsselung auf der Transportebene

Sicherheitsprotokolle auf der Transportebene (Schicht 4) und auf Ebenen zwischen der Transport- und der Anwendungsebene (Schicht 5 und 6) bieten eine Ende-zu-Ende-Verschlüsselung zwischen Prozessen als Kommunikationspartnern. Sie stellen, analog zu den Verbindungsgateways bei den Firewalls, einen anwendungsunabhängigen Sicherheitsdienst zur Verfügung, so dass sie beliebige Anwendungsprotokolle wie Telnet, HTTP oder FTP mit Diensten zur Gewährleistung der Kommunikationsvertraulichkeit, der Datenintegrität und der Authentizität von Kommunikationspartnern anreichern.

Schichten 4-6

nur verbindungsorientiert

Alle entwickelten und in Entwicklung befindlichen Sicherheitsprotokolle dieser OSI-Schicht setzen auf einem zuverlässigen und verbindungsorientierten Übertragungsprotokoll, meist ist dies TCP, auf. Die Sicherheitsanforderungen von verbindungslosen Protokollen wie UDP und der darauf aufsetzenden Anwendungen werden somit von den existierenden Protokollen nicht abgedeckt. Diese müssen entweder auf Protokolle niedrigerer Ebenen zurückgreifen oder erfordern anwendungsspezifische, also individuelle Sicherheitserweiterungen.

Firewall-Problem

Da die Protokolle der Transportebene darauf abzielen, eine sichere Verbindung zwischen einem Client- und einem Serverprozess aufzubauen, können sie nicht ohne weiteres zusammen mit Applikationsfilter-Firewalls eingesetzt werden. Das Problem ergibt sich daraus, dass die dedizierten Proxies der Applikationsfilter von den Protokollen der Schicht 4 als mögliche Mittelsmänner eines Man-in-the-Middle-Angriffs betrachtet werden. Zur Problemlösung sind entweder Tunnelungs-Techniken einzusetzen oder man muss sich mit Firewall-Konfigurationen mit geringeren Filterungsfähigkeiten, wie Paketfiltern, zufrieden geben. Schließlich ist natürlich auch noch anzumerken, dass eine Ende-zu-Ende-Verschlüsselung, die auf einer Schicht n eingesetzt wird, verhindert, dass Firewall-Komponenten auf einer der Schichten $i < n$ die Nutzdaten und ggf. auch Verkehrsdaten der Datenpakete gezielt nach Indikatoren für Bedrohungen inspizieren können.

Fazit

OSI-Einordnung

Abbildung 14.11 ordnet die unterschiedlichen Ansätze zusammen mit einigen Protokollrepräsentanten in das ISO/OSI-Modell ein. Anwendungsprotokolle kann man, wie die Abbildung verdeutlicht, unterscheiden nach solchen, die wie S/HTTP oder S/MIME direkt auf der Transportebene aufsetzen, und solchen wie PGP, die ihrerseits Protokolle der Anwendungsebene nutzen, um ihre Funktionalität zu erbringen.

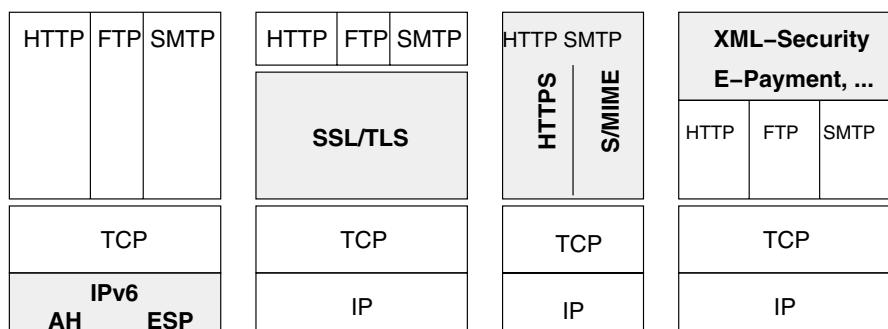


Abbildung 14.11: ISO/OSI-Einordnung der Sicherheitsprotokolle

Grenzen

Unabhängig von ihrer OSI-Einordnung gilt für alle Sicherheitsprotokolle, dass die Sicherungsmaßnahmen ausschließlich die Kommunikationsverbindungen betreffen. Sind die Daten empfangen, so liegen sie unverschlüsselt in den Speichern (u.a. auf der Festplatte, im Arbeitsspeicher, in Caches) der beteiligten Endsysteme. Auch für die Verwaltung und Speicherung der geheimen Informationen wie der Schlüssel der MACs und der geheimen Schlüssel der kryptografischen Verfahren muss auf die Betriebsssoftware der genutzten Endsysteme oder auf externe geschützte Komponenten wie Smart Cards zurückgegriffen werden. Sind diese Informationen nicht vor unautorisierten Zugriffen geschützt, so werden auch die Sicherheitsdienste der Protokolle ausgehöhlt. Es gilt hier wieder einmal das Bild vom schwächsten Glied in der Kette.

14.2.2 Virtual Private Network (VPN)

Verbindungs- und Ende-zu-Ende-Verschlüsselungen werden auch bei der sicheren Kommunikation mittels virtueller privater Netze (VPN) verwendet.

Definition 14.2 (Virtuelles privates Netz)

Unter einem virtuellen privaten Netz verstehen wir eine Netzinfrastruktur, bei der Komponenten eines privaten Netzes über ein öffentliches Netz wie dem Internet miteinander kommunizieren, wobei sie die Illusion besitzen, das Netz zu ihrer alleinigen Verfügung zu haben.

VPN

Diese informelle Definition ist die Grundlage für zwei verschiedene Funktionsbereiche, die herkömmlich mit dem Begriff VPN verknüpft werden. Zum einen wird die Virtualisierung verwendet, um ein Bandbreiten- und QoS⁸-Management über unterschiedliche Netze zu etablieren. Zum anderen dient es als Grundlage für eine sichere Kommunikation. Auf diesen Aspekt konzentrieren wir uns in diesem Unterabschnitt.

Durch die Verwendung einer Tunneltechnik werden die Sicherheitseigenschaften, die im privaten Netz gelten, auch im öffentlichen Netz aufrechterhalten. Die Nutzung des öffentlichen Netzes bleibt für die Benutzer eines VPNs völlig transparent; ihnen erscheint die Kommunikationsverbindung als eine dedizierte private Verbindung. Ein VPN hat somit die Aufgabe, die Benutzer des Netzes zu authentifizieren, die Vertraulichkeit der übertragenen Daten zu garantieren, die erforderlichen Schlüssel zu erzeugen und diese regelmäßig zu erneuern. Das VPN muss auch Audit- sowie Accountinginformationen über seine Nutzer bereitstellen. Darüber hinaus hat es dafür zu

Aufgaben

⁸ Quality of Service

sorgen, dass die privaten Adressen auch bei der Kopplung an ein öffentliches Netz noch privat bleiben, so dass deren Struktur und Bedeutung für Außenstehende nicht ersichtlich ist.

Anwendungsbereiche

VPNs sind für ganz unterschiedliche Anwendungsbereiche sinnvoll einsetzbar. So lassen sich mit VPNs geografisch verteilte Organisationen (z.B. global operierende Unternehmen) verbinden, die mehrere verschiedene Subnetze nutzen, wobei die Standardtechniken und Komponenten (u.a. Router) eines Wide Area Netzes (WAN) verwendet werden können. VPNs ermöglichen das verteilte und kooperative Arbeiten über große Entfernung hinweg unter Nutzung der vorhandenen Telekommunikations- und Datenkommunikationstechnologie. Mittels eines VPNs lassen sich auch geschützte entfernte Zugriffe von autorisierten Klienten auf ein Unternehmensnetz realisieren. Dazu zählen beispielsweise entfernte Zugriffe von Mitarbeitern, die zur Erledigung ihrer Arbeit auf die Server ihres Unternehmens von zu Hause oder von unterwegs zugreifen müssen. Dazu muss sowohl auf dem Client- als auch auf dem Serverrechner ein VPN-Modul installiert werden, das sich in den Protokollstack des jeweiligen Betriebssystems einklinkt und die notwendigen Verschlüsselungs- und Authentifikationsaufgaben durchführt. In der Praxis wird beim Aufsetzen von VPNs für externe Zugriffe (z.B. für Telearbeitsplätze) häufig ein zwar pragmatischer, aber aus Sicherheitssicht sehr bedenklicher Weg gewählt, indem für alle Verbindungen der gleiche gemeinsame Schlüssel verwendet wird, der im Client- und Servermodul eingetragen ist. Da ein Schlüsselwechsel den manuellen Eingriff auf allen externen Clients erfordert würde, werden diese Schlüssel auch in der Regel gar nicht oder sehr selten gewechselt, so dass sich hieraus die bekannten Angriffs möglichkeiten ergeben. Neben der Absicherung externer Zugänge kann ein VPN ebenso sinnvoll in einem Intranet eingesetzt werden.

Beispiel 14.4 (Intranet-VPN)

Intranet-VPN

Betrachten wir als Szenario ein solches Netz, das neben den Rechnern für die Mitarbeiter auch einen Server enthält, der als vertrauenswürdiger Rechner sensible Unternehmensdaten verwaltet und aus Sicherheitssicht heraus eigentlich isoliert betrieben werden müsste. Eine solche Isolierung würde natürlich Probleme in Bezug auf die Verfügbarkeit der Daten nach sich ziehen, da ein direkter Zugriff über die Mitarbeiterrechner unterbunden wird. Ein VPN ermöglicht es, diesen Server kontrollierbar an das Unternehmensnetz anzukoppeln und die Verfügbarkeit der Daten zu gewährleisten. Mittels des VPN-Servers können die Authentizität und die Berechtigungen der Zugriffswilligen kontrolliert und der Schutz der sensiblen Daten kann sichergestellt werden.



In heutigen Netzen werden VPN-Verbindungen häufig auf der Basis von IPSec oder von TLS etabliert. Auf beide Protokolle gehen wir weiter unten noch ein.

Tunneling

Die logische Verbindung zwischen privaten Komponenten über ein anderes Netz hinweg wird durch die Tunnelungstechnik erreicht. Mit Tunnelung bezeichnet man die Nutzung einer Netzinfrastruktur zum Transfer von Daten von einem Netzwerk zu einem anderen (vgl. Abbildung 14.12). Die

Tunneling

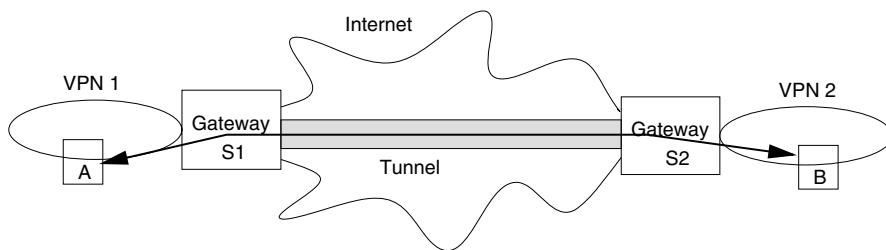


Abbildung 14.12: Tunnelung in VPNs

transportierten Daten sind Frames bzw. Pakete, die durch einen zusätzlichen Header gekapselt werden. Dieser Header enthält Routinginformationen, so dass die gekapselten Daten dem Empfänger am Tunnelende, dies ist in der Abbildung eines der Gateways S1 oder S2, zugestellt werden können. Dort werden die Daten entpackt und, falls notwendig, über das private Netz an den eigentlichen Empfänger weitergeleitet. Als Transitnetz kann ein beliebiges anderes Netz dienen; in der Praxis wird in der Regel das Internet verwendet.

Bekannte Tunnelungsprotokolle der Schicht 2 sind das Point-to-Point-Tunneling Protocol (PPTP) und das Layer 2 Tunneling Protocol (L2TP). Das von Microsoft entwickelte PPTP ist veraltet, weshalb wir hier nicht weiter darauf eingehen. So setzt PPTP die Authentifikations- und Verschlüsselungsdienste von PPP ein. Zur Authentifikation kann entweder PAP (Password-Authentication Protocol) oder CHAP (Challenge-Handshake Authentication Protocol) genutzt werden. Beide Protokolle entsprechen nicht mehr dem Stand der Technik. Zur Verschlüsselung setzt PPP DES-CBC ein, das beim heutigen Stand der Technik auch nicht mehr ausreichend ist.

PPTP

Im Gegensatz zum PPTP ist L2TP nicht auf das IP-Protokoll zugeschnitten, sondern kapselt PPP-Frames durch einen zusätzlichen Header so, dass das gekapselte Paket von einem beliebigen paketvermittelnden Protokoll wie IP weitergeleitet werden kann. L2TP ermöglicht auch die Einrichtung mehrerer Tunnels zwischen zwei Endpunkten, um Tunnelverbindungen von

L2TP

unterschiedlicher Qualität aufzubauen. Beim Aufbau eines VPN mittels PPP-Verbindungen müssen die Endsysteme stets eine direkte Verbindung zu demjenigen PPP-Server aufbauen, der als Netzserver für das VPN agiert. Hierbei können sehr hohe Leitungskosten entstehen, wenn es sich bei dem Endsystem um ein mobiles Gerät handelt, das sich in großer physischer Entfernung zum Server befindet. Durch das L2TP ist es nun möglich, statt über Telefonleitungen oder Wählverbindungen, ein VPN auch über eine Infrastruktur von IP-Routern aufzubauen. Ein Benutzer hat damit die Möglichkeit, auch über das Internet oder andere öffentliche Netze auf ein Intranet zuzugreifen.

EAP

EAP

Auch das L2TP verwendet die Authentifikationstechniken des PPP. Da jedoch das herkömmliche PPP nur eine Passwort-Authentifikation durchführt, bei der der Benutzername sowie das Passwort im Klartext übertragen werden, erfolgt nur eine sehr schwache Authentifikation der Kommunikationspartner. Zur Lösung dieses Problems existieren bereits Erweiterungen für PPP unter dem Namen Extensible Authentication Protocol (EAP). Das EAP sieht die dynamische Integration von Authentifikationsmodulen vor, die es ermöglichen, nach Bedarf unterschiedliche Authentifikationsverfahren zu verwenden. Grundlage von EAP ist das Challenge-Response-Verfahren zur Authentifikation. Abbildung 14.13 zeigt den EAP-Protokollstack.

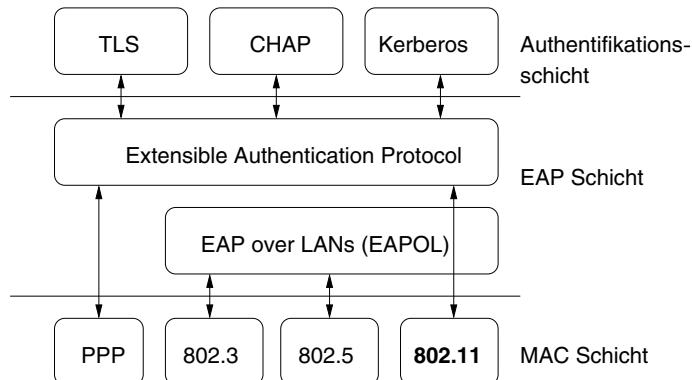


Abbildung 14.13: Der EAP Protokollstack

VPN-Gateway und Firewalls

In der Praxis findet man drei gängige Architekturvarianten, um VPN-Gateways und Firewalls sinnvoll zu koppeln.

Kopplung mit Firewalls

- Wird der VPN-Gateway **vor** der Firewall platziert, so kann der Datenstrom von der Firewall analysiert werden. Die Dienste einer Firewall und eines VPN-Gateways lassen sich hierdurch separieren, wodurch Abhängigkeiten verringert werden. Ein gravierender Nachteil dieser Architektur liegt natürlich darin, dass alle Daten offen in der Firewall vorliegen.
- Wird das VPN-Gateway **hinter** die Firewall platziert, so erreicht man damit eine bessere Unterstützung von Peer-to-Peer Sicherheit sowie einen Schutz des VPN-Gateways selber. Jedoch kann auf diese Weise die Firewall natürlich die verschlüsselten Daten nicht filtern. Zudem kann es zu Problemen kommen, falls die Firewall gleichzeitig eine Netzwerkadress-Umrechnung, also ein NAT, durchführt.
- Schließlich lassen sich VPN-Gateway und Firewall auch **parallel** und unabhängig konfigurieren. Diese Konfigurationsvariante wird selten verwendet, da sie problematisch zu konfigurieren ist. Außerdem ist es schwierig festzulegen, welche Daten noch über die Firewall geleitet werden müssen und welche direkt über das Gateway weitergereicht werden.

14.3 IPSec

IPSec ist ein IETF Sicherheitsstandard und definiert eine Sicherheitsarchitektur für das Internetprotokoll IPv4 und dessen Nachfolger IPv6. IPSec spezifiziert Sicherheitsdienste auf der Schicht 3, um einen vertraulichen, integeren und authentifizierten Transport von IP-Paketen zu ermöglichen. IPSec definiert eine Familie von Protokollen, die aus den Protokollen für den Schlüsselaustausch (IKE-Protokoll), für den verschlüsselten und integeren Datentransfer (ESP-Protokoll) und dem Protokoll für einen authentifizierten Datentransfer (AH-Protokoll) besteht. Die IPSec-Spezifikationen legen einige kryptografische Verfahren fest, die jedes IPSec-System anbieten muss, damit eine Interoperation zwischen unterschiedlichen Protokollimplementierungen sichergestellt ist. Entsprechend seiner Einordnung als Schicht 3-Protokoll werden sichere Kommunikationskanäle zwischen Endsystemen, zwischen Endsystemen und Gateways (z.B. Router oder Firewalls) sowie zwischen zwei Gateways aufgebaut. Die Architektur des IPSec ist im RFC4301 (Security Architecture for the Internet Protocol) beschrieben; wir erklären im Folgenden die wichtigsten Eigenschaften.

Einordnung

Es existiert eine Vielzahl von IPSec-Implementierungen. Man unterscheidet, ob die Implementierung in den Endgeräten oder in den Gateways bzw. Routern erfolgt. Bei den Endgeräten unterscheidet man wiederum zwei Varianten, nämlich die direkte Integration der IPSec-Dienste in das Betrieb-

Implementierungen

system oder die Veränderung des IP-Stacks um eine zusätzliche Schicht. Die direkte Implementierung von IPSec in die Netzwerkschicht des jeweiligen Betriebssystems hat den Vorteil, dass die vorhandenen Dienste der Schicht 3 direkt genutzt werden können (z.B. Fragmentierung, Kontext-Informationen, Sockets), so dass eine effiziente Implementierung möglich ist und die gesicherte Datenübertragung für die Anwendungsebene transparent unterstützt wird. IPSec ist in viele Betriebssysteme integriert. Zu nennen ist hier unter anderem das Open-Source Projekt Openswan⁹, das eine Implementierung von IPSec für Linux bietet.

Nachteilig an der Betriebssystem-integrierten Lösung ist, dass Anwender, die spezifische, differenzierte Sicherheitslösungen für VPNs und Intranet-Bereiche anbieten möchten, auf die Dienste des jeweiligen Betriebssystems angewiesen sind. Um hier eine größere Flexibilität zu erlangen, werden in solchen Fällen vollständige eigene Lösungen implementiert, indem eine zusätzliche Schicht in dem Protokollstack zwischen die Netzwerkschicht und die Datensicherungsschicht integriert wird. Diese Bump in the Stack (BITS) genannte Lösung hat jedoch den Nachteil, dass einige Dienste der Schicht 3 doppelt implementiert werden müssen.

Einsatzgebiete für IPSec

Einsatzgebiete

Ein typisches Einsatzszenario für IPSec ist ein Unternehmensnetz, das aus mehreren einzelnen LANs besteht, die räumlich verteilt (z.B. Filialen, Tochterunternehmen) und über das Internet angebunden sind. In diesem Szenario würde das IPSec Protokoll in alle denjenigen Netzwerkkomponenten abgewickelt werden, die ein solches lokales Netz mit der Außenwelt verbinden. Die entsprechenden Komponenten sind typischerweise Router und Firewalls, so dass die von diesen Komponenten durchgeführten Sicherheitsdienste wie die Verschlüsselung transparent für die Server und sonstigen Rechner im LAN sind. Das IPSec-Protokoll wird auch häufig für die sichere Kommunikation im drahtlosen Nahverkehrsbereich eingesetzt. Ein Beispielszenario hierfür ist der über IPSec gesicherte mobile Zugriff auf einen zentralen Server (z.B. um Termine zu synchronisieren) mittels eines Smartphones unter Nutzung unsicherer WLAN-Verbindungen.

Ein typisches Einsatzgebiet der Implementierung von IPSec in Routern oder Gateways ist der Wunsch in einer Organisation sämtliche Internet-Verbindungen abzusichern, aber eine entsprechende Vorkehrung für das Intranet nicht vorzusehen, da diese Verbindungen als ausreichend sicher angesehen werden. Bei der Implementierung von IPSec in Router bzw. Gateways unterscheidet man ebenfalls zwei Varianten. Die direkte Implementierung in die Router, wie sie von allen gängigen Firewall-Lösungen

⁹ Siehe <http://www.openswan.org/>.

angeboten wird, ist vergleichbar mit der Betriebssystem-integrierten Lösung. Bei der Bump in the Wire (BITW) Implementierung, die als Analogon zur BITS-Lösung gesehen werden kann, wird ein eigenes Gerät benötigt, das die Aufgabe der Verarbeitung der IP-Pakete gemäß der IPSec-Protokolle übernimmt. Derartige Geräte sind über physikalische Leitungen direkt mit der physikalischen Schnittstelle des Routers verbunden.

14.3.1 Überblick

Die eigentlichen Sicherheitsdienste werden von zwei IPSec Protokollen, dem Authentication Header Protokoll und dem Encapsulating Security Payload Protokoll erbracht.

Das Authentication Header (AH) Protokoll (vgl. RFC4302) hat die Aufgabe, die Authentizität des Paketursprungs, die Datenintegrität eines verbindungslos übermittelten Pakets sowie optional einen Schutz vor Wiedereinspielungen zu gewährleisten. Das Encapsulating Security Payload (ESP) Protokoll (vgl. RFC4303) ermöglicht darüber hinaus eine vertrauliche verbindungslose Datenkommunikation und bietet einfache Maßnahmen zur Abwehr von Verkehrsflussanalysen. Beide Protokolle können sowohl einzeln als auch in Kombination eingesetzt und in zwei Modi, nämlich dem Tunnel- und dem Transportmodus betrieben werden.

AH und ESP

Jedem IP-Paket werden ein oder mehrere IPSec-Header hinzugefügt. Ein solcher IPSec-Header ist ein AH- oder ein ESP-Header und enthält Informationen, die der Empfänger benötigt, um das jeweilige Sicherheitsprotokoll abzuwickeln. Im Wesentlichen handelt es sich hierbei um Hashwerte, falls die Daten-Authentizität festgestellt werden soll sowie eine Art Zeiger, der SPI (s.u.), der auf einen Datenbankeintrag beim Empfänger verweist, in dem die sicherheitsbezogenen Informationen abgelegt sind, die der Empfänger benötigt, um das IPSec-gesicherte Paket weiter verarbeiten zu können. Das Format eines IPSec-Datenpakets ergibt sich damit im Groben wie folgt:

IPSec-Header



Transport- und Tunnelmodus

Im Transportmodus werden die Sicherheitsdienste der Protokolle nur auf die Pakete höherer OSI-Schichten angewandt, während sie sich im Tunnelmodus auf das ganze IP-Paket erstrecken. Der Transportmodus wurde ursprünglich für einen Einsatz zur Peer-to-Peer Kommunikation verwendet. Die IPSec-Information wird über den IPSec-Header zwischen dem IP-Header und dem Payload (Daten) eingefügt. Die praktische Bedeutung dieses Modus ist heutzutage jedoch gering.

Tunnelmodus

Der Tunnelmodus kann insbesondere von Firewalls und Sicherheits-Gateways verwendet werden, um einen Tunnel zwischen zu schützenden Endsystemen oder zwischen Teilnetzen, wie virtuellen privaten Netzen, die durch ein unsicheres Netz verbunden sind, aufzubauen (vgl. Abbildung 14.12). Zu diesem Zweck werden die Empfängeradressen des ursprünglichen Pakets (IP-Header und ggf. TCP-Header) eingekapselt, indem diesen ein neuer IP-Header vorangestellt wird, der die IP-Adressen der vermittelnden Gateways (in Abbildung 14.12 sind das die Gateways S1 und S2) als Absender- und Empfängeradresse enthält. Dadurch erreicht man, dass die Originaladressen, hier des Absenders A und des Empfängers B, während des gesamten Transportes des Pakets von S1 zu S2 über ein unsicheres Übertragungsnetz versteckt werden und damit ein einfacher Schutz gegen Verkehrsflussanalysen erzielbar ist.

Kapselung

Klar ist natürlich, dass auch durch ein Voranstellen (Kapseln) von neuen IP-Headern die Daten des Originalpakets noch nicht wirklich versteckt sind; diese werden lediglich als Ganzes als Payload des neuen Pakets betrachtet. Router und Paketfilter-Firewalls inspizieren diesen Payload nicht, so dass die Information vordergründig versteckt ist. Um eine tatsächliche Kapselung und ein Verbergen der Daten dieses Payloads zu gewährleisten, müssen diese selbstverständlich verschlüsselt werden. Hierfür verwendet man das ESP. Abbildung 14.14 veranschaulicht diese Kapselung der Datenpakete.

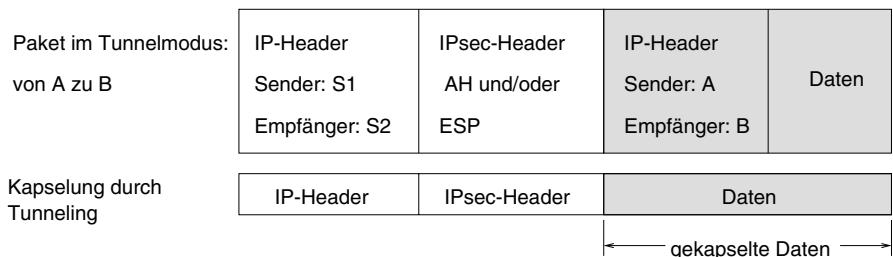


Abbildung 14.14: Kapselung im IPSec Tunnelmodus

Mittels des Tunnelmodus ist es somit möglich, IPSec-Paket auch von Firewalls bearbeiten zu lassen, da die von diesen zur Filterung benötigten Informationen in dem zusätzlichen Header unverschlüsselt zur Verfügung gestellt werden. Da der Tunnelmodus stets anwendbar ist, kann man den Transportmodus eigentlich als überflüssig betrachten. Seine Existenz hat eher historische Bedeutung. Die generelle Verwendung des Tunnelmodus erfordert lediglich etwas mehr Overhead durch den zusätzlichen Header. Dieser Overhead ist aber in der Regel vernachlässigbar.

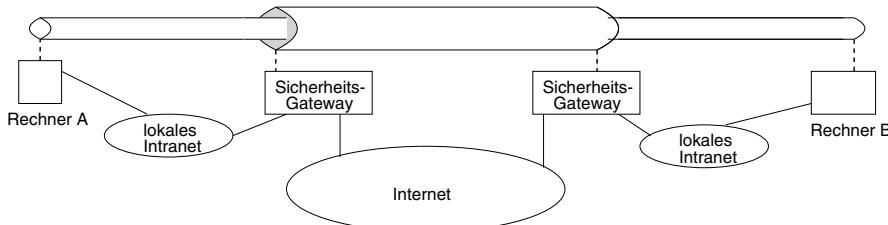


Abbildung 14.15: Geschachtelte Tunnels

Datenpakete können mittels des Konzepts des Tunnelmodus natürlich auch mehrfach gekapselt werden. Abbildung 14.15 veranschaulicht die Situation. Hier wird z.B. ein verschlüsselter Kanal zwischen A und B sowie zusätzlich ein anderer verschlüsselter Kanal zwischen Firewalls oder Gateways etabliert.

geschachtelter
Tunnel

14.3.2 Security Association und Policy-Datenbank

Die Informationen darüber, welche Verschlüsselungsverfahren sowie Schlüssel aktuell für eine IPSec-Verbindung zum Einsatz kommen, werden mit dem Konzept der Security Association (SA) zur Verfügung gestellt. Die Werte der Attribute einer SA-Datenstruktur basieren auf dem Sicherheitsregelwerk, das für das jeweilige IPSec-Endsystem bzw. den IPSec-Gateway gilt. Darauf gehen wir weiter unten noch etwas genauer ein. Eine IPSec Security Association ist unidirektional, so dass bei bidirektionalen Verbindungen für jede Verbindung sowohl jeweils verschiedene Krypto-Verfahren als auch unterschiedliche Schlüssel festlegbar sind. Abhängig von den Festlegungen der Sicherheitspolicy werden für eine Verbindung nur das AH, nur ESP oder beide Protokolle verwendet. Entsprechend wird für jedes angewandte Protokoll eine eigene SA etabliert. Dies wird als SA-Bundle bezeichnet. Eine SA stellt im Prinzip ein Abkommen zwischen zwei oder mehr Kommunikationspartnern dar und enthält u.a. folgende Informationen:

SA

SA-Information

- die Authentifikationsverfahren, Modi und Schlüssel für das AH-Protokoll, falls AH verwendet wird, oder
- die Verschlüsselungsverfahren, Authentifikationsverfahren, Modi und Schlüssel für das ESP-Protokoll, falls ESP verwendet wird,
- die Angaben über den potentiell erforderlichen Initialisierungsvektor IV; diese Information wird nur für das ESP benötigt,
- die Lebensdauer der Schlüssel bzw. der ganzen SA sowie
- die IP-Adresse desjenigen Endsystems bzw. Subsystems, auf das sich die Vereinbarungen der SA beziehen. Gelten die SA-Festlegungen für mehrere Empfänger, so kann dies also auch die Adresse eines ganzen Netzes oder Teilnetzes sein.

- Falls die Kommunikationspartner eine Multi-Level-Sicherheitsstrategie implementieren, enthält die SA auch die Sicherheitsklassifikation (confidential, secret, unclassified) der zu schützenden Daten.

SAD

Die SAs eines Endsystems bzw. Sicherheits-Gateways sind in der Security Association Database (SAD) des jeweiligen Systems zusammengefasst. Da die Verbindungen unidirektional sind, unterscheidet man zwischen der SA-Datenbank für ausgehende und der für hereinkommende Datenpakete. Damit der Empfänger eines IP-Pakets die zugehörige SA ermitteln kann, sind SAs eindeutig identifizierbar. Dazu dient der 32-Bit Security Parameter Index (SPI) zusammen mit der IP-Adresse des Kommunikationsendpunktes. Diese Informationen sind dem IPSec-Header von IP-Paketen zu entnehmen, der als AH- und/oder ESP-Header den IPv4- bzw. IPv6-Header ergänzt (siehe z.B. Abbildung 14.16, die den Header für das AH-Protokoll beschreibt).

Will nun ein Absender A mit dem Empfänger B eine IPSec-Verbindung aufbauen, so extrahiert A aus seiner von ihm verwalteten Ausgangs-SA Datenbank die dort für die Verbindungen mit B hinterlegten Informationen. Dazu gehört neben den SA-Daten auch der SPI, über den der Empfänger B die gleichen Informationen in seiner Eingangs-SA Datenbank finden kann.

Sicherheitsregelwerk

SPD

Es ist klar, dass die SAs nur ein Realisierungskonzept darstellen, um Festlegungen eines zugrunde liegenden Sicherheitsregelwerks, häufig auch als Strategie bezeichnet, umzusetzen. Ein solches Regelwerk legt für ein IPSec-System fest, welche Sicherheitsdienste in welcher Ausprägung für die verschiedenen Verbindungen eingesetzt werden sollen. Die Strategiespezifikation wird in der Security Policy Database (SPD) verwaltet. Diese enthält sowohl für eingehende als auch für abgehende Pakete Einträge, die angeben, ob für ein IP-Paket, das das Profil des Eintrags erfüllt, IPSec angewandt (apply), das Paket vernichtet (discard) oder ob es, ohne sich Sicherheitsdiensten unterziehen zu müssen, weitergeleitet (bypass) werden soll.

Kontrolle

Für jedes ankommende oder abgehende IP-Paket wird auf die SPD zugegriffen, um daraus zu entnehmen, wie mit dem Paket zu verfahren ist. Sollen die Sicherheitsdienste von IPSec angewandt werden, so verweist, falls vorhanden, der Strategieeintrag auf die zugehörigen SA-Einträge in der SA-Datenbank für ausgehende Pakete des IPSec-Systems. Andernfalls wird für ausgehende Pakete eine solche SA dynamisch unter Verwendung des Internet Key Exchange Protokolls (IKE) (s.u.) erzeugt. Ein ankommendes Paket, für das keine zugehörige SA in der SAD existiert, wird automatisch vernichtet.

SPD-Eintrag

Um eine neue SA für eine Verbindung erzeugen zu können, muss die Datenbank einen Eintrag besitzen, der die erforderlichen Festlegungen enthält.

Jeder SPD-Eintrag spezifiziert eine Menge von Selektoren, die diejenigen Pakete beschreiben, auf die die Festlegungen des Eintrags anzuwenden sind. Ein solcher Selektor kann eine IP-Adresse eines einzelnen Senders bzw. Empfängers oder ein ganzer Adressbereich sein. Die Auswahl eines ganzen Adressbereichs beim Empfänger ist beispielsweise dann sinnvoll, wenn alle diese Endsysteme sich hinter einer Firewall befinden und dieselbe SA verwenden. Analogen gilt natürlich auch für den Adressbereich von Absender, die Rechner hinter einer Firewall beschreiben können.

Es können ferner Sende- und Empfangssports angegeben und sogar spezifische Benutzer-Namen, ein Rechner-Name oder ein Name eines Sicherheits-Gateways in Form von DNS oder X500-Namen spezifiziert werden. Um Multi-Level-Security-Strategien zu unterstützen, können auch Sicherheitsklassifikationen spezifiziert werden. Mit den Selektoren lassen sich überschneidende Adressbereiche angeben, wodurch auch mehrere SPD-Einträge für ein Datenpaket anwendbar sein können. Um Inkonsistenzen zu vermeiden, werden die Einträge in einer linearen Ordnung durchmustert und der erste Eintrag, dessen Selektor-Bedingungen von dem Datenpaket erfüllt werden, wird gewählt.

Neben dem Selektor enthält der Eintrag die Aktionen, die für die ausgewählten Datenpakete zu verwenden sind. Diese Aktionen umfassen die oben bereits erwähnten Discard-, Apply- und Bypass-Anweisungen. Soll IPSec angewendet werden, so spezifiziert der Eintrag, ob ESP und/oder AH einzusetzen ist und welche kryptografischen Verfahren sowie Hashverfahren mit welchen Schlüssellängen und welche Modi sowie welche Initialisierungsvektoren einzusetzen sind. Weiterhin wird die Lebenszeit der SAs, die auf der Grundlage dieser Strategieeinträge dynamisch erzeugt werden, festgelegt.

Aktionen

Beispiel 14.5 (SPD-Eintrag)

Ein Eintrag in einer SPD eines IPSec-Systems kann beispielsweise folgenden Auszug einer Spezifikation enthalten:

SPD-Eintrag

src: 192.168.2.1 – 192.168.2.10

dest: 192.168.3.1

IPSec-action: *esp req cipher AES*
 integrity SHA2 keylen 128
 expiry (seconds) 60
 transport
 ah req integrity hmacsha1 keylen 160
 expiry (seconds) 60
 tunnel

Die Spezifikationen beziehen sich (Selektor) auf die Absenderadressen des Bereichs 192.168.2.1 – 192.168.2.10 und die Empfängeradresse 192.168.3.1. Es ist das ESP-Protokoll im Transport-Modus mit dem AES im CBC Modus zur Verschlüsselung und mit dem SHA2 mit einem 128-Bit Hashwert zu verwenden. Jede davon abgeleitete ESP-SA hat eine Lebensdauer von 60 Sekunden. Die Datenpakete sind zusätzlich durch das AH-Protokoll im Tunnel-Modus zu kapseln. Als Hashfunktion ist das HMAC-SHA-1 Verfahren mit einem 160-Bit Hashwert anzuwenden.



IKE-Eintrag

Ein Strategieeintrag beschreibt die zu verwendenden Verfahren, wobei auch mehrere zur Auswahl angegeben sein können, aber er spezifiziert natürlich weder die zu nutzenden kryptografischen noch die MAC-Schlüssel. Diese Schlüssel und die tatsächlich eingesetzten Verfahren werden mit dem IKE-Protokoll zwischen den kommunizierenden IPSec-Systemen ausgehandelt. Die in diesem Protokoll übermittelten Datenpakete müssen authentifiziert und vor unautorisierten Manipulationen sowie Informationsgewinnungen geschützt sein. Somit ist auch zur Abwicklung des IKE-Protokolls zunächst eine sichere Verbindung aufzubauen und es sind die dazu zu benutzenden Verfahren und Sicherheitsattribute festzulegen. Das heißt, dass die Strategiedatenbank für die zu sichernden Verbindungen auch die notwendigen IKE-Aktionen spezifiziert, also die Authentifikations-, Verschlüsselungs- und Hashverfahren festlegt, die beim Aufbau der IKE-Verbindung zu verwenden sind.

Granularität der Schlüsselvergabe

Es bleibt zu klären, welche Differenzierungsmöglichkeiten IPSec bei der Vereinbarung von Schlüsseln vorsieht. Entsprechend der OSI-Einordnung des AH- und ESP-Protokolls können die Kommunikationsendpunkte eigentlich nur Endsysteme sein, die über ihre IP-Adressen identifiziert werden. Da aber über die Strategiedatenbank, die ja auf der Anwendungsebene angesiedelt ist, zusätzliche Informationen zur Verfügung stehen, ist es unter IPSec möglich, eine differenziertere Schlüsselvergabe zu realisieren. Es werden drei Granularitätsstufen unterschieden.

Host-Orientierung

Die gröbste Granularität ergibt sich durch die endsystemorientierte (engl. *host-oriented*) Schlüsselvergabe, die besagt, dass ein einziger Schlüssel für alle Verbindungen zu einem dedizierten Empfänger-Endsystem benutzt wird. Das heißt, dass alle Benutzer zweier so gekoppelter Endsysteme zur Übermittlung ihrer Daten zwischen diesen Systemen einen gemeinsamen Schlüssel verwenden müssen, unabhängig davon, ob es sich um vertrauenswürdige oder nicht vertrauenswürdige Benutzer handelt. Dies eröffnet Angriffspunkte, auf die wir weiter unten noch zurückkommen werden.

Eine feinere Granularität bietet die benutzerorientierte Schlüsselvergabe, bei der aber immer noch nicht zwischen einzelnen Anwendungen des Benutzers differenziert wird. Das hat zur Konsequenz, dass man nicht zwischen sicherheitskritischen Datenpaketen wie privaten oder geschäftlichen E-Mails, die besondere Schutzmaßnahmen erfordern, und unkritischen Paketen, wie einer HTTP-Verbindung, bei der nur auf allgemein zugängliche Informationen zugegriffen wird, unterscheiden kann. Die uniforme Verwendung des Benutzerschlüssels führt dazu, dass unter Umständen große Datenmengen mit demselben Schlüssel verschlüsselt werden und damit einem Angreifer Möglichkeiten zur Durchführung einer Chosen- oder Known-plaintext-Attacke zur Verfügung stehen.

Benutzer-
orientierung

Die feinste Granularität lässt sich mit der verbindungsorientierten Schlüsselvereinbarung erzielen. Hier wird für jede logische Verbindung eines Benutzers ein eigener Schlüssel verwendet. Da jedoch nach wie vor keine Informationen über die Anwendungen vorliegen, kann auch die verbindungsorientierte Verschlüsselung keine dedizierten Anwendungsanforderungen erfüllen.

Verbindungs-
orientierung

SA-Management

Das Aushandeln der SA und der benötigten Schlüssel ist kein Bestandteil der IPSec-Protokolle, sondern erfolgt entweder statisch und dann auch meist manuell oder durch Protokolle auf höheren OSI-Schichten. Die manuelle Konfigurierung von Schlüsseln und SAs ist jedoch wegen ihrer mangelhaften Skalierbarkeit nur für kleine Netze attraktiv. Für einen breiten Einsatz von IPSec im Internet sind dagegen automatisiert ablauffähige und skalierbare Managementprotokolle erforderlich. Solche Protokolle sind u.a. mit dem Internet Security Association Key Management Protocol (ISAKMP) (vergl. RFC 2408) entwickelt worden. Mit dem ISAKMP werden Protokolle und Formate spezifiziert, um SAs einzurichten, auszuhandeln, zu modifizieren und wieder zu löschen. Auf das IKE-Protokoll, das als Bestandteil von ISAKMP diese Aufgaben erfüllt, gehen wir weiter unten noch etwas näher ein. Da die auszuhandelnden Sicherheitsattribute, Typen und Formate abhängig von ihrem jeweiligen Anwendungskontext eine unterschiedliche Bedeutung haben, ist für einen spezifischen Anwendungskontext stets eine Interpretation anzugeben. Die zu verwendenden Attribute mit ihren Bedeutungen sowie die unterstützten kryptografischen Verfahren werden in einer Interpretationsdomäne (engl. *domain of interpretation (DOI)*) zusammengefasst und über einen eindeutigen Identifikator identifiziert. Die DOI-Spezifikation für den Anwendungskontext der IPSec-Protokolle ist in [143] beschrieben.

SA-Management

ISAKMP

DOI

14.3.3 AH-Protokoll

Sicherheitsdienste

Das Authentication Header Protokoll (AH) (vgl. RFC 4302) hat die Aufgabe, die Quelle eines Datenpakets, also den Datenursprung, zu authentifizieren, die Integrität des Pakets sicherzustellen sowie optional Wiedereinspielungen (Replays) zu erkennen und abzuwehren. Das AH-Protokoll bietet seine Integritäts- und Authentizitätsfunktionalitäten als verbindungslosen Dienst an. Das hat zur Folge, dass ohne die optionalen, zusätzlichen Maßnahmen, wie Sequenznummern zur Abwehr von Replays nur die Integrität einzelner Datenpakete gesichert wird, eine Integritätsaussage über die aus den Paketen zusammengesetzte gesamte Nachricht aber nicht möglich ist. Aus diesem Grund ist es dringend anzuraten, die Option zur Abwehr von Replays zu wählen, da das IP-Protokoll selbst ja verbindungslos arbeitet und keine Reihenfolgenüberprüfungen durchführt.

Die Verschlüsselung der Daten ist kein Bestandteil des Funktionsspektrums des AH-Protokolls. Dessen vordringliches Ziel ist vielmehr die Abwehr von Address-Spoofing-Angriffen und Maskierungsversuchen. Da die Verfahren zur Integritäts- und Authentifikationskontrolle keinen Exportbeschränkungen unterliegen, bietet das AH-Protokoll eine Sicherung in diesen Bereichen ohne staatliche Einschränkungen. Die Internet Assigned Numbers Authority (IANA) hat dem AH-Protokoll die Nummer 51 zugewiesen, so dass dieser Wert in den Headern der IPv4-Pakete, die das AH-Protokoll einsetzen, anzugeben ist. Das AH-Protokoll ist für darüber liegende Schichten transparent; das heißt, deren Funktionalität wird durch den Einsatz des Protokolls nicht beeinflusst.

Integrität

Im herkömmlichen Internetprotokoll der Netzwerkebene, dem IPv4, sind zwar bereits Verfahren zur Fehlererkennung vorhanden, nämlich die 16- oder 32-Bit Polynom-Prüfsummen eines Cyclic-Redundancy-Codes (CRC). Diese sind jedoch zu schwach, um gezielte Angriffe auf die Datenintegrität der übertragenen Datenpakete abzuwehren. Sie dienen lediglich dazu, Bitfehler zu erkennen, die durch unzuverlässige Übertragungsmedien auftreten können. Aus diesem Grund werden im AH-Protokoll Message Authentication Codes (MAC), also kryptografisch sichere Hashfunktionen mit Schlüssel, verwendet.

Headerinformation

Die zur Durchführung der Schutzaufgaben notwendigen Informationen werden mit jedem IP-Paket über den Authentication Header, der eine Ausprägung des IPSec-Headers ist (vgl. Abbildung 14.14), übermittelt. Der Authentication Header eines Datenpakets enthält einen SPI, um die assozierte SA zu identifizieren, optional eine Sequenznummer, um Wiedereinspielungen zu erkennen, sowie ein Authentifikationsfeld mit dem MAC-Wert der Daten des IP-Pakets, wobei die Länge dieses Feldes von der Länge des MAC-Wertes

des jeweils verwendeten MAC-Verfahrens abhängt. Alle diese Authentifikationsdaten stehen im Klartext im Header.

Abbildung 14.16 veranschaulicht den Aufbau eines AH-Headers. Der reservierte Bereich ist für eine zukünftige Nutzung vorgesehen und muss stets den Wert Null enthalten. Die SPI-Werte 1 – 255 sind von der IANA ebenfalls für zukünftige Aufgaben reserviert. Der SPI-Wert Null soll nur lokal genutzt werden und darf nicht in Datenpaketen, die über das Netz übertragen werden, verwendet werden. Da jedes IPSec-System seine SAs in eigenen SA-Datenbanken verwaltet, muss der Absender eines Pakets denjenigen SPI angeben, der die benötigte SA in der Datenbank des Empfängers identifiziert. Diese Informationen sind beim Einrichten von SAs zwischen Sender und Empfänger auszutauschen.

Header-Format

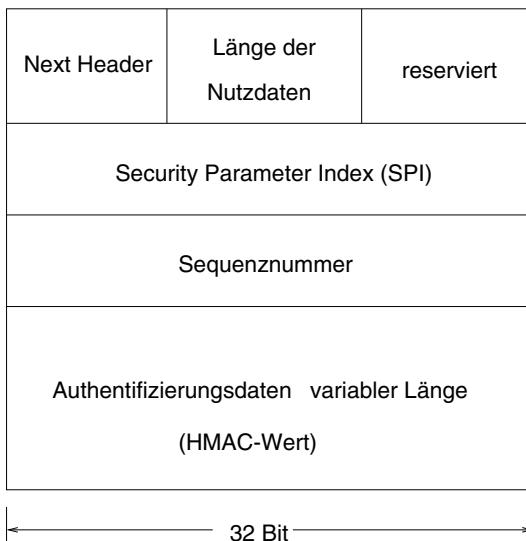


Abbildung 14.16: AH-Format

Jeder SA ist ein Sequenznummernzähler zugeordnet und jedesmal, wenn ein Paket über eine SA versendet wird, wird deren Sequenznummernzähler monoton um eins erhöht. Da die ursprüngliche 32-Bit Sequenznummer für High-Speed-Anwendungen nicht ausreichten, wurde in der neuen Spezifikation die Option der Extended (64-bit) Sequence Number (ESN) eingeführt. Diese Option ist bei dem aktuellen Schlüsselaustauschprotokoll IKEv2 standardmäßig aktiviert, d.h. 32-Bit Sequenznummern müssen explizit verhandelt werden.

Sequenznummer

Die aktuelle Sequenznummer wird in das Sequenznummernfeld des AH eingetragen. Um ein zirkuläres Wiederholen von Sequenznummern und damit das Vorhandensein gültiger Pakete mit gleichen Sequenznummern zu

verhindern, etabliert ein Sender stets eine neue SA mit einem neuen MAC-Schlüssel, wenn die Sequenznummer ihren maximalen Wert von $2^{64} - 1$ erreicht hat. Beim Einrichten einer SA zwischen Sender und Empfänger wird der Sequenznummernzähler mit Null initialisiert.

Replay-Erkennung

Sequenznummern dienen dazu, das Wiedereinspielen von Paketen zu erkennen. Da IP ein unzuverlässiges Protokoll ist, kann nicht davon ausgegangen werden, dass die gesendeten Pakete in der korrekten Reihenfolge zugestellt werden. Um Replay-Attacken zu erkennen, muss der Empfänger ein Paketfenster der Größe W implementieren, wobei defaultmäßig $W = 64$ ist. Die am weitesten rechts liegende Ecke des Fensters (vgl. Abbildung 14.17) repräsentiert die größte Sequenznummer N , die der Empfänger

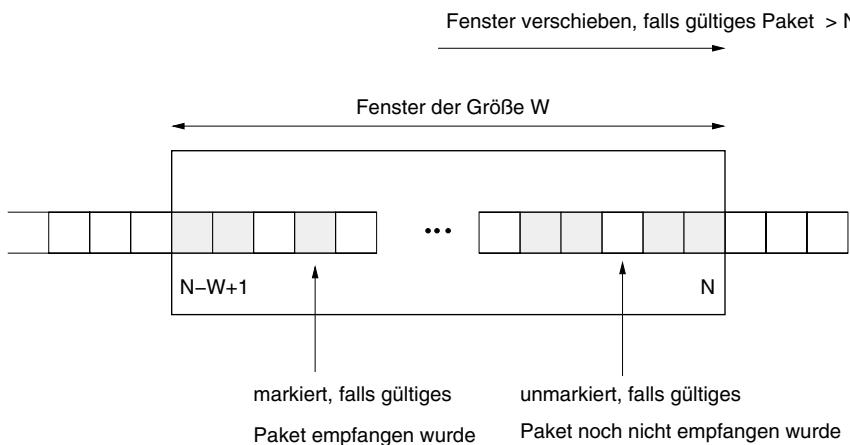


Abbildung 14.17: IPSec Replay-Erkennung

bislang empfangen hat. Für jedes Paket, das eine Sequenznummer $i \in \{N-W+1, \dots, N\}$ enthält und das korrekt authentifiziert wurde, wird der zugehörige Fenster-Slot markiert. Wird ein Paket empfangen, das eine Sequenznummer $i < (N - W + 1)$ enthält oder für das die Authentifizierung fehlgeschlagen ist, so wird das Paket verworfen und dieses Ereignis wird protokolliert (möglicher Replay-Angriff). Beim Empfang eines gültigen Pakets mit einer Sequenznummer größer als N wird das Fenster nach rechts verschoben.

MAC-Berechnung

Die RFC-Spezifikation des AH (siehe RFC4302) schreibt vor, dass das Verfahren HMAC-SHA-1 von jedem IPsec-System anzubieten ist. Ferner sollte AES-XCBC-MAC unterstützt werden und HMAC-MD5 kann verwendet werden. Da jedoch MD5 und SHA-1 nicht mehr sicher sind, wurde im RFC 4868 die Nutzung von HMAC-SHA-256, HMAC-SHA-384, und HMAC-SHA-512 in IPsec spezifiziert. Das BSI empfiehlt die Nutzung die-

ser Varianten¹⁰. Der Sender berechnet den MAC-Wert für die Daten des IP- und des AH-Headers sowie für die Nutzdaten des zu sendenden IP-Pakets. Sei K der geheime MAC-Schlüssel, der zwischen Sender und Empfänger vereinbart wurde, und seien die zu authentifizierenden Daten durch die Nachricht M gegeben. M besteht also aus dem IP-Header und dem AH-Header sowie den Nutzdaten. Das AH-Protokoll berechnet den Hashwert von M mittels

$$\text{HMAC}(M, K) = H(K \oplus \text{opad}, H(K \oplus \text{ipad}, M)).$$

Dies entspricht der auf Seite 378 bereits besprochenen HMAC-Berechnung mit den dort erklärten Werten der Bitmuster opad und ipad . In obiger Berechnung steht das Funktional H als Stellvertreter für den bei einem konkreten Einsatz des AH gewählten MAC-Algorithmus, also für MD5, SHA-1 etc. Da sich einige Daten des IP-Headers während des Datentransportes verändern können, man denke zum Beispiel an das Time-To-Live Feld oder an den Hop-Counter, dürfen diese Felder in die MAC-Berechnung nicht eingehen und werden dazu auf Null gesetzt. Abbildung 14.18 veranschaulicht das vergrößerte Format eines AH-geschützten IP-Pakets.

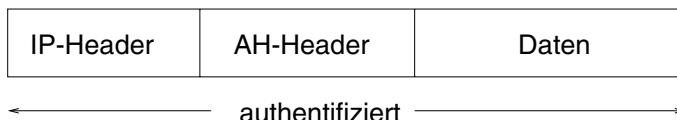


Abbildung 14.18: Ein mit dem AH-Protokoll geschütztes IP-Paket

Eine IP-Fragmentierung darf erst nach der Berechnung der Authentifikationsdaten erfolgen und die Fragmente müssen vor der Verifikation der Authentifikationsdaten wieder zum Originalpaket zusammengesetzt werden.

Ein offensichtliches Problem bei der Verwendung von AH ist das Zusammenspiel mit NAT (Network Address Translation (vgl. Seite 116)). Da durch das NAT Protokoll die interne IP-Adresse eines lokalen Rechners auf eine globale IP-Adresse umgesetzt, also verändert wird, verändert sich natürlich damit auch der MAC-Wert des AH-Protokolls und AH wird das Paket verwerfen. Wir gehen weiter unten auf Lösungsmöglichkeiten für dieses Problem noch ein.

NAT

14.3.4 ESP-Protokoll

Das Encapsulating Security Payload Protokoll ESP (siehe RFC4303) umfasst neben den Diensten, die bereits für das AH-Protokoll angegeben wurden, noch zusätzlich die Fähigkeit, Daten verschlüsselt zu transportieren. Entsprechend der beiden Modi, in denen das ESP-Protokoll verwendet

ESP-Sicherheitsdienste

¹⁰ vgl.<https://www.bsi.bund.de/SharedDocs/Downloads/DE/.../BSI-TR-02102-3.pdf>

werden kann, bezieht sich die Verschlüsselung entweder nur auf die Nutzdaten (engl. *payload*), dies ist im Transportmodus der Fall, oder aber auf das gesamte IP-Paket, wenn der Tunnelmodus verwendet wird.

CBC-Modus

Das ESP-Protokoll schreibt vor, dass jedes zur Verschlüsselung verwendete Verfahren eine symmetrische Chiffre sein muss. Da die ESP-Pakete auf der IP-Ebene in beliebiger Reihenfolge zugestellt werden, muss jedes Paket alle Informationen beinhalten, die für eine korrekte Entschlüsselung notwendig sind. Das bedeutet zum Beispiel bei der Wahl einer Blockchiffre, dass die Länge des zu verschlüsselnden Klartextes stets ein Vielfaches der Blockgröße der Chiffre beträgt. Um dies zu gewährleisten, sind die Klartexte mit Paddingmustern (siehe Seite 303 ff) aufzufüllen, die vom Empfänger nach der Entschlüsselung wieder zu entfernen sind. Das ESP-Datenpaket muss deshalb auch Informationen über das Paddingmuster und die Länge des durch Padding aufgefüllten Datenbereichs enthalten. Wird die Blockchiffre darüber hinaus im CBC-Modus verwendet (siehe Kapitel 7.5.3), so wird für jedes Paket ein Initialisierungsvektor IV benötigt, der somit auch mit dem Paket übertragen werden muss. In der Regel wird er mit den ersten Bytes der Nutzdaten festgelegt.

Kryptoverfahren

Jede ESP-Implementierung muss den NULL-Algorithmus und das Verfahren Tripel-DES im CBC-Modus und sollte die Verfahren AES-CBC mit 128-Bit Schlüssel bzw. AES-CTR enthalten.

Für den Bereich der Authentifikationsverfahren wird in Übereinstimmung mit der AH-Spezifikation gefordert, dass jede Implementierung den HMAC-SHA-1 mit einem Ausgabewert von 96-Bits unterstützen muss. Eine Implementierung sollte auch AES-XCBC-MAC und kann HMAC-MD5 anbieten. Auch der NULL-Authentifikationsalgorithmus muss von ESP-Implementierungen unterstützt werden.

NULL-Verfahren

Durch die Spezifikation wird gefordert, dass der NULL-Algorithmus (siehe RFC 2410) verpflichtend unterstützt werden muss. Das hat zur Konsequenz, dass aus einer technischen Sicht ESP zwar stets eine Verschlüsselung durchführt, jedoch durch die Wahl des NULL Verschlüsselungsalgorithmus de facto auf eine Verschlüsselung verzichtet werden kann. Die Sicherheitsexperten haben es sich nicht nehmen lassen, einen eigenen RFC zum Null-Verschlüsselungsalgorithmus zu spezifizieren. Die Lektüre dieses RFCs ist sehr amüsant und durchaus empfehlenswert. Nachfolgend ist ein Auszug aus dem RFC angegeben, der dies untermauert.

„NULL does nothing to alter plaintext data. In fact, NULL, by itself, does nothing. NULL is a block cipher the origins of which appear to be lost in antiquity. Despite rumors that the National Security Agency suppressed publication of this algorithm, there is no evidence of such action on their part. Rather, recent archaeological evidence sug-

gests that the NULL algorithm was developed in Roman times, as an exportable alternative to Ceaser ciphers. However, because Roman numerals lack a symbol for zero, written records of the algorithm's development were lost to historians for over two millennia.

2.1 Keying Material

Like other modern ciphers, e.g., RC5, the NULL encryption algorithm can make use of keys of varying lengths. However, no measurable increase in security is afforded by the use of longer key lengths.“

Analog zu dem oben bereits Besprochenen, werden die zur Abwicklung des ESP-Protokolls erforderlichen Informationen dem ESP-Header entnommen, der jedem ESP-geschützten IP-Paket anzufügen ist. Wie der AH-Header so enthält auch der ESP-Header einen SPI, um die Security Association, die für die Verbindung gelten soll, zu identifizieren. Abbildung 14.19 veranschaulicht das Format eines ESP-Pakets.

Da das ESP-Protokoll sowohl zum Schutz der Vertraulichkeit als auch zur Authentifikation dient, ist in der assoziierten SA ein Verschlüsselungs- und ein Authentifikationsverfahren zu spezifizieren.

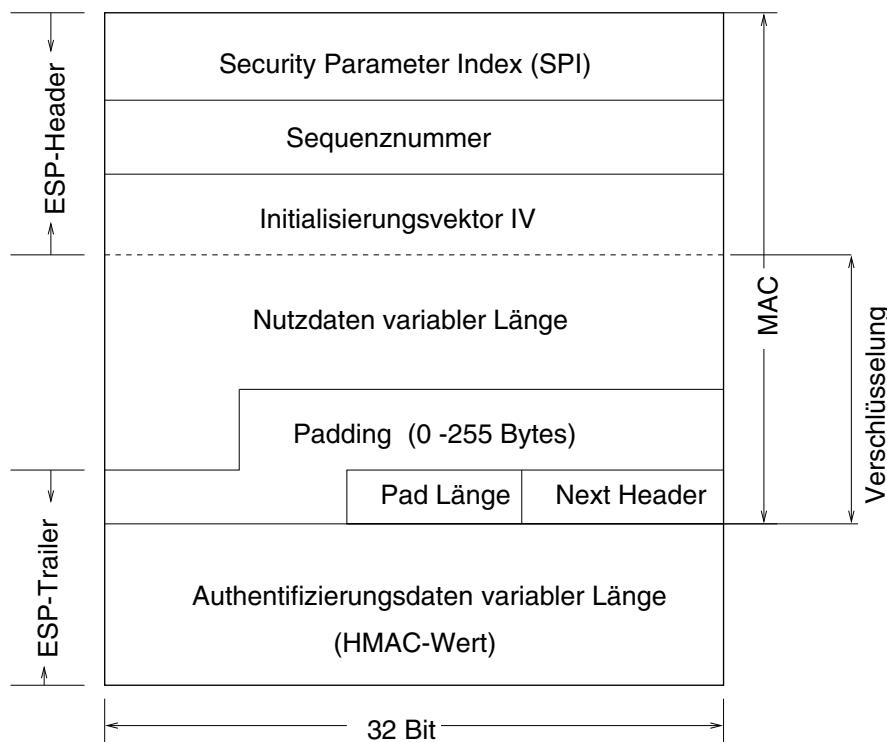


Abbildung 14.19: ESP-Format

vergrößertes
Format

Abbildung 14.20 zeigt das vergrößerte Format eines ESP-geschützten IP-Pakets, bestehend aus einem IP-Header, einem ESP-Header, den verschlüsselten Daten sowie einem ESP-Trailer.

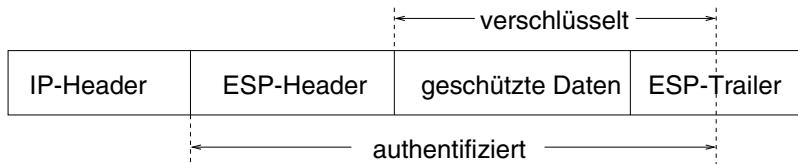


Abbildung 14.20: Ein mit dem ESP-Protokoll geschütztes IP-Paket

Klartexte

Das ESP-Protokoll verschlüsselt nur einen Teil des ESP-Trailers eines Datenpakets und übermittelt den ESP-Header vollständig im Klartext, um den Empfänger mit den übermittelten Informationen in die Lage zu versetzen, das Datenpaket korrekt zu verarbeiten. Zu den Klartextinformationen gehören in erster Linie der SPI-Wert zur Bestimmung der verwendeten kryptografischen Verfahren und die Sequenznummer des Pakets zur Erkennung von Wiedereinspielungen. Der unverschlüsselte Teil des ESP-Trailers enthält aber auch die Authentifizierungsdaten, also den MAC-Wert.

Authentifikation

Im Gegensatz zum AH-Protokoll erstreckt sich die MAC-Berechnung nur auf den ESP-Header, die geschützten Daten sowie den verschlüsselten Teil des Trailers. Das bedeutet, dass die Daten des IP-Headers weder authentifiziert sind, noch einer Integritätskontrolle unterliegen. Bei einem alleinigen Einsatz des ESP-Protokolls im Transportmodus (s.u.) wird der Datenursprung somit nicht authentifiziert, wodurch Maskierungsangriffe nach wie vor möglich sind. Da nur der IP-Header veränderliche Informationen enthält, legt die ESP-Spezifikation keine Beschränkungen für die MAC-Berechnung fest.

Zu den verschlüsselten Trailer-Daten gehören das Paddingmuster, die Angabe über die Länge der Paddingsequenz und das Next-Header-Feld, das den Typ der Nutzdaten spezifiziert. Das Feld enthält eine Protokollnummer, die einen Extension-Header des IPv6 oder ein Protokoll einer höheren Schicht identifiziert. Hierzu werden die von der Internet Assigned Numbers Authority vergebenen IP-Protokollnummern verwendet. Die IANA hat dem ESP offiziell die Protokollnummer 50 zugewiesen.

Einige ESP-Problembereiche

Verschlüsselung

Man beachte, dass ESP auch so eingesetzt werden darf, dass allein eine vertrauliche Kommunikation oder lediglich eine Authentizitätsprüfung erfolgt. In diesen Fällen enthält die SA nur jeweils das erforderliche Verfahren. Eine derartige Einschränkung der Sicherheitsdienste ist jedoch nicht unproblematisch, weil den Anwendern mit der Verwendung des ESP-

Sicherheitsprotokolls ein Grad an Sicherheit suggeriert wird, der aber aufgrund der Beschränkungen gar nicht erzielt werden kann. Wird als Verschlüsselungsverfahren beim ESP eine Blockchiffre im CBC-Modus verwendet, so ist das Protokoll anfällig für so genannte Cut-and-Paste-Angriffe (s.u.), die einen Maskierungsangriff zur Folge haben, da allein mit den Verschlüsselungsverfahren keine Authentifikation durchführbar ist.

Auch die Verwendung von ESP (auch bereits im Transport-Modus) führt zu Problemen im Zusammenspiel mit dem NAT-Protokoll. Da ESP ebenfalls die Header-Daten der Protokolle oberhalb von Ebene 3 als Payload betrachtet und verschlüsselt, werden unter anderem Portadressen und TCP/UDP-Prüfsummen in IPSec-Paketen verschlüsselt. Durch NAT verändern sich IP-Adressen und damit auch die Prüfsumme, die aber nicht von NAT korrigiert werden kann, da diese Information ja verschlüsselt abgelegt ist. Ebenso werden in manchen Fällen von NAT auch Portadressen geändert. Auch diese Änderungen lassen sich nicht in ein mit ESP-geschütztes Paket eintragen. Der RFC 3715 spezifiziert das Zusammenspiel zwischen NAT-Boxen und IPSec, das im Wesentlichen Bestandteil der Phasen des IKE-Protokolls (s.u.) ist.

Auch im Zusammenspiel von IPSec-ESP mit Firewalls (vgl. Kapitel 14.1) ergeben sich Probleme. Bereits einfache Paketfilterfirewalls versuchen über ihre Filterregeln ein- und ausgehende Pakete auch anhand der Portadressen zu filtern. Da aufgrund der Ende-zu-Ende Verschlüsselung von IPSec diese Informationen jedoch durch das ESP verschlüsselt im Paket abgelegt sind, greifen die entsprechenden Filterregeln nicht. Ein solches Paket muss entweder verworfen werden oder die Firewall unkontrolliert tunneln.

NAT-Problem

Firewalls

14.3.5 Schlüsselaustauschprotokoll IKE

Wie wir bereits erklärt haben, sind die Security Associations ein Realisierungskonzept, um die in der Security Policy Database (SPD) festgelegten Sicherheitseigenschaften zu implementieren. Ein IPSec-Eintrag in der SPD spezifiziert, welche Verfahren zur Gewährleistung einer vertraulichen und/oder authentifizierten Kommunikation zwischen den angegebenen Kommunikationspartnern zu verwenden sind. Dagegen sind aber keine Festlegungen darüber getroffen, wie die Verbindungen mit diesen Sicherheitseigenschaften tatsächlich aufgebaut werden sollen. Dies gehört zur Aufgabe des Internet Key Exchange Protokolls (IKE) (RFC4306).

Das IKE-Protokoll beschreibt die Schritte zum Austausch eines gemeinsamen authentifizierten Schlüssels zwischen Kommunikationspartnern. Zur Berechnung gemeinsamer geheimer Schlüsselinformationen wird stets das Diffie-Hellman-Verfahren verwendet, über das sich die Kommunikationspartner somit nicht abstimmen müssen.

IKE

IKE-SA**IKE-Aufgabe**

Das IKE-Protokoll wird immer dann benötigt, wenn für ein abzusendendes Datenpaket keine SA in der Security Association Database für ausgehende Pakete des Senders gefunden wird (NULL SADB-Zeiger) und gleichzeitig im SPD-Eintrag für die Verbindung über die Apply-Forderung die Anwendung von IPSec gefordert wird. Dann enthält, wie oben bereits beschrieben, der Policy-Eintrag auch die benötigten Festlegungen über die zum Aufbau einer sicheren Verbindung einzusetzenden Authentifikations- und Verschlüsselungsverfahren. Das Sendesystem initiiert das IKE-Protokoll, das durch den Austausch der erforderlichen Sicherheitsinformationen die Erzeugung der benötigten SAs sowohl auf der Initiatorseite als auch auf der Empfängerseite vorbereitet. Da die im Verlauf des IKE-Protokolls ausgetauschten Datenpakete ebenfalls zu schützen sind, müssen auch für die IKE-Kommunikation die zu verwendenden kryptografischen sowie Authentifikationsverfahren festgelegt sein. Dazu setzt man wiederum das Konzept der Security Associations ein, so dass, analog zu dem vorher Besprochenen, die Sicherheitseigenschaften von IKE-Verbindungen mittels IKE-SAs definiert werden. Das bedeutet natürlich, dass für das Schlüsselaustauschprotokoll ebenfalls eine Strategie festgelegt sein muss, da ja die SAs nur Realisierungskonzepte darstellen.

IKE-Policy

Die IKE Strategiedatenbank enthält eine nach Prioritäten geordnete Liste von Strategieeinträgen, genannt Protection Suites, mit Angaben über die für den Austausch anzuwendenden Verfahren. Da der Austausch geheimer Schlüsselinformationen stets mittels des Diffie-Hellman-Verfahrens (DH) erfolgt, muss dieses in der Suite nicht festgelegt werden. Freiheitsgrade bleiben hier nur für die Wahl der Parameter des DH-Verfahrens, so dass diese Bestandteil der Suites sind. Jede solche Suite definiert mindestens den zu verwendenden Verschlüsselungsalgorithmus, das Hashverfahren, die für das Diffie-Hellman-Verfahren benötigten Informationen (Modul, primitive Einheitswurzel, öffentliche Schlüssel) sowie das Authentifikationsverfahren (z.B. Pre-shared Key).

Protokollablauf

Das IKE-Protokoll zerfällt in zwei Phasen. In der ersten Phase werden die SAs, die zur Abwicklung des IKE-Protokolls selbst benötigt werden, ausgehandelt und in der zweiten Phase erfolgt dann der vertrauliche und authentifizierte Austausch von Informationen zur Erzeugung von IPSec-SAs. Im Gegensatz zu den IPSec-SAs ist eine IKE-SA bidirektional, d.h. sie wird sowohl auf ausgehende als auch auf einkommende Pakete angewandt. IKE verwendet standardmäßig den UDP-Port 500 zur Protokollabwicklung. Im Folgenden wird IKEv1 etwas genauer beschrieben. IKEv2 (vgl. RFC 5996 <http://tools.ietf.org/html/rfc5996>) ist deutlich kompakter als die Version v1.

Die Version v2 benötigt in der Regel lediglich den Austausch von vier Nachrichten (bei v1 sind es 9 Nachrichten), um eine Verbindung aufzubauen. In v2 entfallen auch die Cookies, die in v1 ausgetauscht wurden, um DoS-Angriffe zu erschweren.

Erste Phase

Der erste Schritt des IKE-Protokolls besteht darin, unter Verwendung des Diffie-Hellman-Verfahrens einen geheimen Schlüssel, bzw. präziser gesagt, die Informationen, aus denen beide Partner den IKE-Schlüssel berechnen können, auszutauschen. Zunächst werden so genannte Cookies ausgetauscht, die dazu dienen, Denial-of-Service-Angriffe abzuwehren. Ein Cookie ist hierbei aber nicht zu verwechseln mit den Cookies, wie sie sonst aus dem WWW-Bereich bekannt sind (vgl. u.a. Seite 154). Ein Cookie im Rahmen von IKEv1 ist ein Hashwert, der aus einer geheimen Information, die der jeweilige Cookie-Erzeuger für jedes Cookie neu generiert, der IP-Adresse des Senders und aus einer Zeitmarke gebildet wird. Für einen Dritten erscheint ein Cookie als eine Pseudo-Zufallszahl, während es für die authentischen Beteiligten des Protokolls gezielt zu rekonstruieren ist. Die Protokollpartner müssen in einem Handshake das Cookie ihres Partners wieder an diesen zurücksenden und erst wenn beide die Korrektheit der von ihnen ursprünglich versandten Cookies überprüft haben, wird das Protokoll mit dem DH-Verfahren zur Schlüsselberechnung fortgesetzt¹¹. Beide Partner berechnen nun den gemeinsamen IKE-Schlüssel, mit dem sie in der zweiten Phase des Protokolls die zu vereinbarenden SA-Attribute vertraulich austauschen. Falls kein anderes Verfahren festgelegt ist, wird zur verschlüsselten Kommunikation der Tripel-DES im CBC-Modus eingesetzt. Die Cookies werden in den nachfolgenden Protokollschriften noch zur Identifikation der IKE-SA der betreffenden Kommunikationsverbindung verwendet.

Neben dem Cookie tauscht im so genannten *Main-Modus* der Partner, der die Kommunikation initiiert, auch noch die Menge von Krypto-Suiten aus, die er unterstützen kann. Diese enthalten Angaben über unterstützte Verschlüsselungsverfahren, Hashalgorithmen, Authentifizierungsmethoden sowie Diffie-Hellman-Parameter Werte. Der annehmende Partner wählt daraus eine, die er ebenfalls unterstützt, aus und teilt diese Wahl dem Initiator mit. Im so genannten *Aggressive-Modus* wählt der Initiator direkt die Verfahren aus, wodurch er keine Möglichkeit hat, beim nicht-Zustandekommen einer Verbindung zu erkennen, welche Verfahren sein Gegenüber kennt. Hier sind somit Angriffspunkte für Denial-of-Service Angriffe, falls der Initiator nach dem erstmaligen Scheitern eines Verbindungsaufbaus im Aggressive-Modus nicht auf den etwas langsameren, aber erfolgversprechenderen Main-Modus überwechselt. Der Initiator kann ja nicht sicher sein, dass

¹¹ Achtung, der einfache Cookie-Austausch ist keine Authentifikation!

die Nachricht, die den Verbindungsauftakt verweigert, tatsächlich vom ange- sprochenen Empfänger stammt.

Authentifikation

Da mit dem DH die Kommunikationspartner nicht authentifiziert werden, ist im nächsten Schritt dieser ersten Phase eine Authentifikation der IKE-Partner erforderlich. IKE sieht zum Nachweis der Identität der Kommunikationspartner im Wesentlichen drei Vorgehensweisen vor: (1) vorab ausgetauschte Schlüssel (Pre-Shared Master Keys), (2) digitale Signaturen unter Einsatz des DSA oder RSA-Verfahrens sowie (3) Public-Key Verschlüsselung mittels RSA oder El-Gamal. In dem Authentifizierungsschritt weisen die Partner jeweils nach, dass sie das Geheimnis (Pre-Shared Key, Signaturschlüssel oder privater Schlüssel zum öffentlichen Verschlüsselungsschlüssel) kennen. Die Authentifikationsvorgehensweise unterscheidet sich bei den drei Verfahren, jedoch wird im Prinzip vom jeweiligen Partner immer ein Hashwert über das Geheimnis, seine Identität, die gewählten Kryptoverfahren sowie die ausgetauschten Cookies berechnet und mit einem gemeinsamen Schlüssel verschlüsselt versendet. Der gemeinsame Schlüssel, der hierzu verwendet wird, berechnet sich mit einem Hashverfahren aus dem gemeinsamen DH-Schlüssel und den ausgetauschten Cookies. In Beispiel 14.6 wird beispielhaft die Authentifikation mittels Public-Key Verfahren erklärt.

Zweite Phase

Phase II

Nach einer erfolgreichen Authentifikation stimmen die beiden Partner in der zweiten Phase, auch bekannt als Quick-Mode, unter Verwendung der in der ersten Phase ausgehandelten Verfahren die für die SA-Erzeugung benötigten Sicherheitsattribute, wie die zu verwendenden Algorithmen oder die Schlüssellängen, aufeinander ab.

Der vergrößerte Ablauf des IKEv1-Protokolls ist in Abbildung 14.21 zusammengefasst.

Beispiel 14.6 (Authentifikation mit Public-Key Verfahren)

Abschließend wird anhand eines der drei möglichen Verfahren die Authentifikation (also die Schritte 5 und 6 aus Abbildung 14.21) noch etwas genauer beschrieben. Für das Folgende gehen wir von einer Authentifikation mittels Public-Key Verfahren im Main Modus aus.

Im Schritt (3) erzeugt A eine Nonce, wir nennen sie $nonce_A$, und generiert mittels eines Hashverfahrens einen Schlüssel K_A :

$$K_A = \text{hash}(nonce_A, cookie C_A).$$

Diese Nonce $nonce_A$ wird sodann mit dem öffentlichen Schlüssel von B verschlüsselt und zusammen mit den verschlüsselten DH-Werten und

Public-Key Verfahren

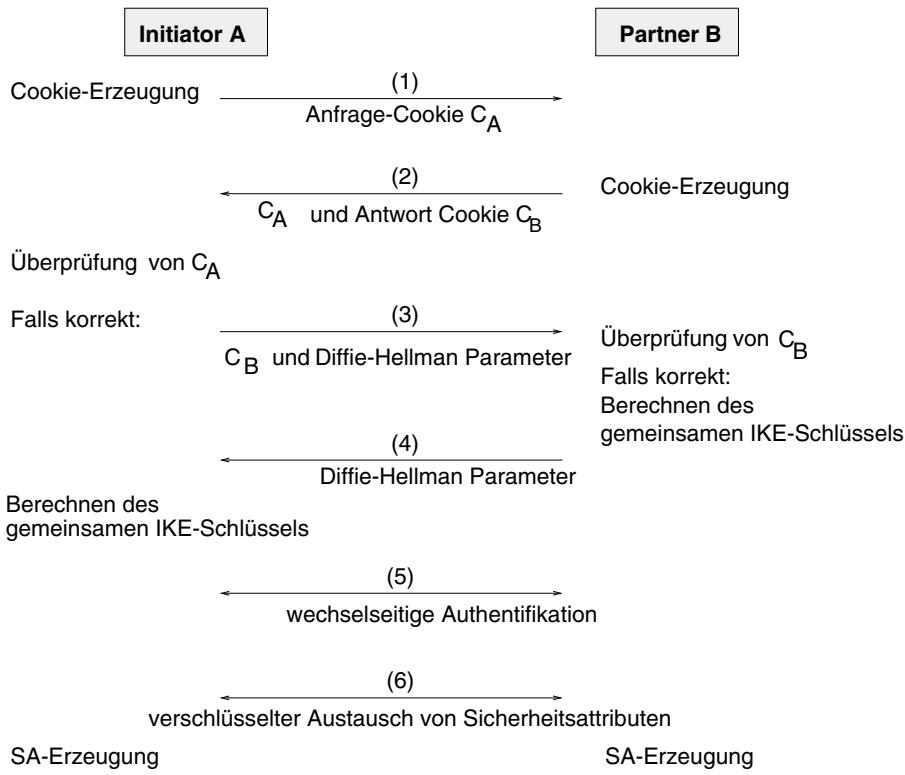


Abbildung 14.21: Vergröbertes IKEv1-Protokoll

der verschlüsselten Identität von A sowie mit dem Zertifikat von A zu B gesendet. Die Nachricht (3) hat damit folgendes Format:

$$\{nonce_A\}^{Public_B}, \{DH - Parameter\}^{K_A}, \{A\}^{K_A}, Zertifikat_A$$

Der Partner B entschlüsselt $\{nonce_A\}^{Public_B}$ und kann mit diesem Wert den gemeinsamen Schlüssel K_A berechnen und die weiteren Daten der Nachricht (3) entschlüsseln. Im Gegenzug sendet er A eine analoge Nachricht, bestehend aus einer von ihm gewählten Nonce, die er nun seinerseits mit dem öffentlichen Schlüssel von A verschlüsselt. Ebenso generiert er einen Schlüssel K_B in analoger Weise zum Schlüssel K_A , so dass die Nachricht (4) folgendes Format erhält:

$$\{nonce_B\}^{Public_A}, \{DH - Parameter\}^{K_B}, \{B\}^{K_A},$$

Partner A entschlüsselt seinerseits die $nonce_B$ und kann damit auch den Schlüssel K_B berechnen. Abschließend berechnet A aus seinen nunmehr vorliegenden Werten den gemeinsamen DH-Schlüssel $K_{A,B}$ und generiert einen Sitzungsschlüssel K , der ein Hash über den DH-Schlüssel sowie

über die beiden Nonces $nonce_a, nonce_B$ ist. Damit verschlüsselt A eine Nachricht und sendet diese als Nachricht (5) an B:

$$\{Das\ ist\ der\ Beweis,\ dass\ ich\ A\ bin\}^K.$$

B berechnet auf analoge Weise den Schlüssel K und sendet den Identitätsnachweis in Nachricht (6) zurück an A.

$$\{Das\ ist\ der\ Beweis,\ dass\ ich\ B\ bin\}^K.$$

Das Protokoll lässt, wie man sieht, offen, auf welche Weise der öffentliche Schlüssel von B in den Besitz von A gelangt. Auch das Versenden des Zertifikats von A nach B in Nachricht (3) ist im Protokoll nicht verpflichtend vorgeschrieben. Hierfür können also unterschiedliche Wege (und natürlich auch unterschiedliche Qualitätsstufen für die Authentizität der Schlüssel!) gewählt werden.



Zusammenspiel mit NAT

NAT

Wie bereits weiter oben angesprochen, definiert der RFC 3715 das Zusammenspiel zwischen NAT und IPSec während der IKE-Phasen. Während der Phase 1 werden Daten zwischen Absender-Port 500 sowie Empfangs-Port 500 ausgetauscht. Durch das NAT können diese Port-Adressen verändert werden, z.B. kann ein anderer Sende-Port eingetragen sein, so dass der Empfänger in der Lage sein muss, auch Pakete zu verarbeiten, die von einem Absender-Port ungleich 500 stammen. Damit der Empfänger eines IKE-Paketes das Vorhandensein einer NAT-Box erkennen kann, tauschen die Partner Nachrichten mit Hashes über ihre IP-Adressen und Ports aus¹². Falls kein NAT-Gerät auf dem Weg zwischen den beiden Hosts liegt, sind die Hashes gleich und die IKE-Kommunikation kann wie beschrieben abgewickelt werden. Wird ein NAT-Gerät entdeckt, so ist es die Aufgabe der Phase II durch eine spezielle UDP-Kapselung für eine Bearbeitungsmöglichkeit zu sorgen. Bei der Aushandlung der SAs wird festgelegt, dass ESP-verschlüsselte Pakete in ein UDP-Paket zu kapseln sind, so dass Checksummen von den NAT-Geräten berechnet und in diese neuen Header eingetragen werden können.

14.3.6 Sicherheit von IPSec

Die IPSec-Protokolle leisten keinen Beitrag zur Zugriffskontrolle und zur Verbindlichkeit. Das IPSec beschränkt sich auf Dienste zur Authentifikation, Vertraulichkeit und Integritätssicherung. Die Dienste zur Vertraulichkeit und Integrität sind verbindungslos und ohne Recovery-Fähigkeiten ausgestattet.

¹² Das ist grob mit dem Versenden von Probe-Nachrichten vergleichbar.

Der wesentliche Schritt bei IPSec besteht in der Kopplung der Netzwerkprotokolle mit den Konzepten der Sicherheits-Policy und der Security Associations, die auf der Anwendungsebene anzusiedeln sind. Das hat zur Folge, dass Informationen auf höheren Protokollebenen gewonnen werden können und für Kontrollen zur Verfügung stehen. Zu beachten ist jedoch, dass die erforderlichen Management- und Kontrollmaßnahmen außerhalb der eigentlichen IPSec-Protokolle AH und ESP durchgeführt werden. Auf diese Weise lassen sich die konzeptionellen Beschränkungen eines Schicht 3-Protokolls zwar umgehen, aber dafür wird die Forderung nach der Unabhängigkeit der OSI-Schichten aufgeweicht, da komplexe Wechselwirkungen zwischen Protokollen der Ebene 3 und 7 erforderlich sind.

Zur Authentifikation von Datenpaketen verwenden sowohl das AH- als auch das ESP-Protokoll einen Message Authentication Code. Bereits in Kapitel 8.1.4 haben wir darauf hingewiesen, dass ein MAC nicht die Funktion einer digitalen Signatur im Hinblick auf eine zweifelsfreie Zuordnung einer Aktion bzw. Nachricht zu einem Subjekt erfüllt. Da Sender und Empfänger den gleichen MAC-Schlüssel kennen, kann ein Sender stets seine Urheberschaft leugnen. Die Verwendung einer digitalen Signatur anstelle des MAC-Wertes im AH-Header würde ein sehr viel größeres Datenfeld erfordern, weil solche Signaturen in der Regel (außer bei schwacher Kryptografie aufgrund von Exportbeschränkungen) mehr als 1024 Bits umfassen, also z.B. wesentlich mehr Platz erfordern, als ein 160-Bit SHA-1-Wert. Die von der OSI-Sicherheitsarchitektur empfohlenen Verbindlichkeits- und Notariatsdienste werden mit IPSec nicht erbracht.

Die vorgestellten Protokolle weisen einige Besonderheiten auf, die aus Sicherheitssicht problematisch sind und im Folgenden erläutert werden.

Der Einsatz von IPSec erfordert die Spezifikation einer Policy-Datenbank auf jeder IPSec-Maschine. Die vollständige, konsistente und korrekte Formulierung der Datenbankeinträge ist keine einfache Aufgabe. Auf der Basis einzelner IP-Adressen, Ports etc. ist genau festzulegen, welche der Sicherheitsdienste in welcher Ausprägung einzusetzen sind. Eine korrekte Konfigurierung erfordert große Sorgfalt und ein tiefes Verständnis für die Wechselwirkungen und Zusammenhänge in vernetzten Systemen. Eine lückenhafte und unvollständige Strategiespezifikation eröffnet Sicherheitsprobleme. Entscheidet sich beispielsweise der zuständige Administrator bei der Absicherung einer Verbindung zwischen den Maschinen A und B für den Einsatz von ESP im Transport Modus ohne Authentifikationsanteil und ohne zusätzliche Absicherung mittels des Tunnel-Modus, so sind die übermittelten Pakete nicht gegen Spoofing-Angriffe oder unautorisierte Modifikationen geschützt. Dies mag ja für einige Applikationsdaten angemessen sein. Da aber über die spezifischen Anwendungen keine Informationen vorhanden

externe
Informationen

keine
Verbindlichkeit

schwierige
Konfigurierung

sind, kann dieser mangelhafte Schutz auch solche Anwendungsdaten betreffen, die authentifiziert und vor Modifikationen geschützt zwischen A und B ausgetauscht werden sollen.

AH-Protokoll

Stark umstritten ist die Rolle des AH-Protokolls. Das AH bietet nur Authentizität und Datenintegrität, aber keine Vertraulichkeit. Mit AH werden im Gegensatz zu den entsprechenden Sicherheitsdiensten des ESP auch Daten aus dem IP-Header der Pakete vor unberechtigten Manipulationen geschützt. Dies ist aber nur in solchen Fällen von Interesse, in denen IPSec im Transport-Modus angewendet wird. Dies wird in der Praxis jedoch selten der Fall sein, so dass ESP im Tunnel-Modus auch den Integritätsschutz von AH übernehmen kann. Die meisten Sicherheitsexperten sind sich darin einig, dass das AH-Protokoll lediglich dazu beiträgt, dass die IPSec-Lösungen komplizierter und aufwändiger zu administrieren sind, ohne dass damit ein höheres Sicherheitsniveau erreicht wird.

Design-Probleme

Da IPSec ein verbindungsloses Protokoll ist, sichern die Integritätsdienste nur einzelne Datenpakete ab, während die Integrität des gesamten Datenstroms einer Anwendung nicht überprüft wird. Hieraus ergeben sich Ansatzpunkte für Angriffe. Darüber hinaus ist zu beachten, dass die Authentifikation der Kommunikationspartner mittels MAC-Verfahren allein auf wissensbasierten Techniken beruht. Im Zusammenhang mit passwortbasierter Authentifikation (vgl. Abschnitt 10.2) haben wir aber bereits auf die Probleme, die mit einer solchen Technik einhergehen, hingewiesen. Für den Bereich der Vertraulichkeit ist anzumerken, dass das ESP-Protokoll die Verwendung von Blockchiffren im CBC-Modus zur vertraulichen Kommunikation zulässt. Der CBC-Modus verarbeitet Klartexte blockweise, so dass ein Angreifer einzelne Blöcke identifizieren und aus dem Klartextstrom herausfiltern kann. Die Fehlerausbreitung beim CBC beschränkt sich lediglich auf den nächsten Klartextblock. Diese beiden Eigenschaften ermöglichen so genannte Cut-and-Paste-Angriffe (s.u.). Dabei werden die Kryptotexte verschiedener Klartexte, die mit dem gleichen Schlüssel verschlüsselt wurden, so kombiniert, dass nur der Block, der direkt hinter dem Block, an dem die Vermischung erfolgt, beim Entschlüsseln inkorrekte Daten liefert. Das IPSec schreibt nicht vor, dass bei jeder Verbindung ein eigener, neuer Schlüssel zu verwenden ist, so dass insbesondere mit der hostbasierten Schlüsselvergabe ein und derselben Schlüssel für alle Datenpakete eingesetzt wird, die über eine gleiche Host-zu-Host-Verbindung übertragen werden. Damit ist die Voraussetzung zur Durchführung einer solchen Cut-and-Paste-Attacke geschaffen, da in diesem Fall sowohl für vertrauenswürdige als auch für nicht vertrauenswürdige Benutzer eines IPSec-Systems die gleichen Schlüssel verwendet werden.

Angriffe auf IPSec

Der Cut-and-Paste Angriff setzt einen autorisierten Benutzer X auf mindestens einem der beiden Endsysteme (Sender, Empfänger) voraus. Zur Erklärung des Angriffs betrachten wir folgendes Szenario: Zwischen zwei Endsystemen R_A und R_B wird nur das ESP-Protokoll und zwar mit einer hostbasierten Verschlüsselung und ohne Integritätskontrollen abgewickelt. Eine autorisierte Benutzerin Alice sendet nun von ihrem Endsystem R_A eine Nachricht M zu ihrem Partner Bob auf der Maschine R_B (vgl. Abbildung 14.22). Die grau hinterlegten Abschnitte repräsentieren die ver-

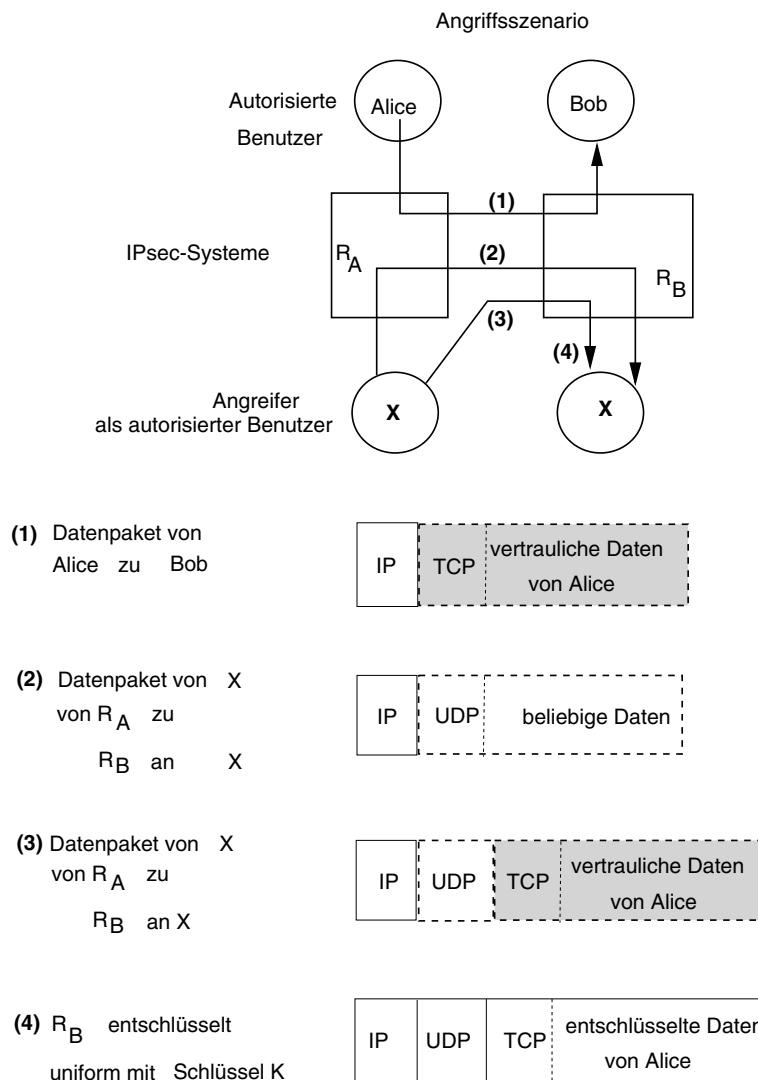


Abbildung 14.22: Cut-and-Paste-Angriff

schlüsselten Bestandteile des Datenpakets von Alice. Die Nachricht M wird vom Angreifer X abgehört und aufgezeichnet. Im nächsten Schritt verschickt der Angreifer nun seinerseits zwischen den beiden Endsystemen R_A und R_B eine UDP-basierte Nachricht, die an ihn selber gerichtet ist. Da X seine Nachricht vom gleichen Endsystem wie Alice absendet, werden seine Daten sowie die Daten von Alice aus der Nachricht M mit dem gleichen Host-Schlüssel K verschlüsselt.

Cut-and-Paste

Aus den abgefangenen, verschlüsselten Daten und der von ihm selber konstruierten Nachricht, von der er nur den Anfangsteil mit den Verkehrsdaten benötigt, erstellt der Angreifer nun ein neues Datenpaket. Dazu ersetzt er seine verschlüsselten Daten durch diejenigen von Alice und sendet dieses durch ein simples Cut-and-Paste neu zusammengestellte Paket direkt zur Maschine R_B , wobei wiederum der Angreifer selbst als Empfänger angegeben wird.

Aufgrund der Fehlerausbreitung des CBC-Modus wird nur der erste Block des TCP-Headers der ursprünglichen Nachricht M korrumptiert. Da alle Datenpakete, die vom System R_A beim System R_B eintreffen, durch dessen ESP-Protokoll vor der Weiterleitung an den Empfänger automatisch entschlüsselt werden, erhält X auf diese Weise die Klartextdaten, die Alice vertraulich an Bob gesendet hat. Zur Abwehr derartiger Angriffe sollte das ESP-Protokoll mit einer Authentifikation kombiniert und es sollte, soweit möglich, auf eine hostorientierte Schlüsselvergabe verzichtet werden.

Abwehr

Die gerade skizzierte Cut-and-Paste-Technik kann auch angewendet werden, um in einen Nachrichtenstrom zwischen zwei Partnern Alice und Bob Datenpakete einzuschleusen oder um die aufgebaute Verbindung gezielt zu übernehmen (engl. *session hijacking*). Wie bei der oben beschriebenen Cut-and-Paste-Attacke fängt der Angreifer im ersten Schritt eine Nachricht zwischen Alice und Bob ab und sendet in Schritt 2 eine eigene Nachricht vom System R_A zum System R_B . Diese Nachricht enthält die Daten, die der Angreifer in die Kommunikation zwischen Alice und Bob einschleusen möchte. Im einfachsten Fall kann er die ursprüngliche Nachricht von Alice löschen und anstelle der verschlüsselten Nutzdaten, die Alice versenden wollte, seine eigenen Daten an ihr Paket anhängen (siehe Abbildung 14.23), um damit zum Beispiel die Ausführung von Befehlen zu initiieren.

Chosen-Plaintext Angriff

Der automatisierte Einsatz eines hostorientierten Schlüssels, der für unterschiedliche Benutzer bei allen Datenpaketen, die das System verlassen, angewandt wird, bietet natürlich auch Ansatzpunkte für einen Angriff mit gewähltem Klartext. Der Ansatzpunkt zur Durchführung des Angriffs ist die Tatsache, dass auf der niedrigen Netzwerkebene Sicherheitsdienste uniform und undifferenziert angewendet werden. Die Sicherheitsdienste verschlüsseln bzw. entschlüsseln nämlich alle ihnen vorgelegten Daten unabhängig

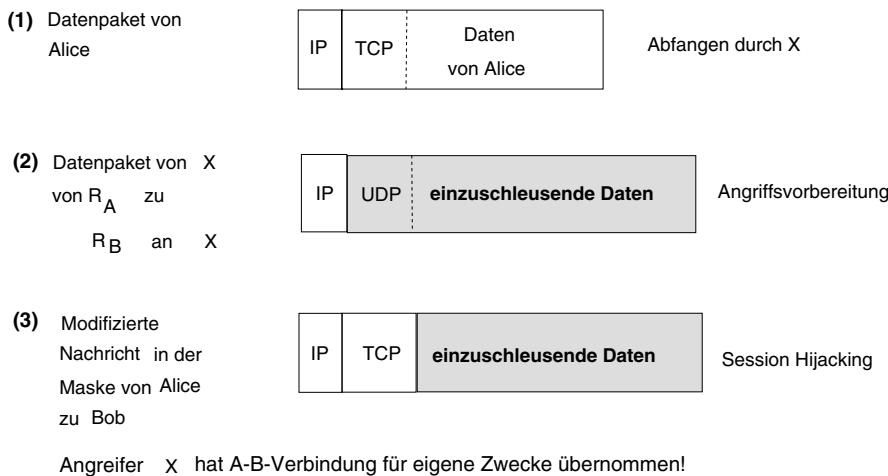


Abbildung 14.23: Session Hijacking Angriff

vom Absender und Empfänger. Dadurch ist es möglich, eine Nachricht mit einem gezielt gewählten Klartext, wie etwa einem Passwort, verschlüsseln zu lassen. Dazu muss man zum Beispiel nur eine entsprechend präparierte E-Mail an die Portadresse des Maildienstes desjenigen Zielsystems senden, dessen Host-Schlüssel attackiert werden soll. Der Angreifer hat nun nur noch seine eigene, verschlüsselte Nachricht abzuhören, um in den Besitz eines Klartext/Kryptotext-Paares zu gelangen. Mit diesem kann er anschließend eine Kryptoanalyse zum Brechen des verwendeten Schlüssels durchführen.

Wird nur ein 64-Bit DES-Schlüssel oder sogar nur ein 40-Bit Schlüssel verwendet, so ist eine Analyse beim heutigen Stand der Technik bereits mit vergleichbar niedrigen Kosten erfolgreich möglich. Mit einem gebrochenen Schlüssel ist der Angreifer dann in der Lage, den gesamten Netzverkehr, der über die mit dem Schlüssel geschützte Verbindung abgewickelt wird, zu entschlüsseln. Noch weitaus einfacher ist es aber, mit den gezielt verschlüsselten Klartexten und den zugehörigen Kryptotexten den Datenverkehr über die attackierte Verbindung zu belauschen und Pakete, die den gleichen Kryptotext enthalten, herauszufiltern. Auf diese Weise können auch ohne kryptoanalytische Ansätze sensible Informationen wie beispielsweise vertraulich übertragene Passworte offengelegt werden.

Entschlüsselung

Insgesamt verdeutlichen diese Angriffsszenarien, dass ein grobgranularer und uniformer Einsatz von Sicherheitsdiensten nur einen ungenügenden Schutz gewährt. Es besteht die Gefahr, dass Benutzer im Vertrauen auf einen gar nicht gewährleisteten Grad an Sicherheit sicherheitskritische Aktionen durchführen und damit lohnende Ziele für die oben geschilderten Angriffsszenarien werden.

14.4 TLS/SSL

Schichten-einordnung

Das Transport Layer Security Protokoll (TLS) ist aus dem Secure Socket Layer Protokoll SSL [66] hervorgegangen. TLS ist ein Industriestandard für verschlüsselte Kommunikation. Es setzt auf der Transportebene auf. TLS ist ein Internet-Standard u.a. für sichere HTTP-Verbindungen, der von allen gängigen Web-Browsern unterstützt wird. Dabei wird das auf der Anwendungsebene angesiedelte HTTP Protokoll über Dienste, die von dem darunterliegenden TLS Protokoll angeboten werden, abgesichert. Dieses gesicherte Protokoll wird üblicherweise als HTTP über TLS, kurz HTTPS¹³ bezeichnet. Hierbei wird, vereinfacht ausgedrückt, vom HTTP-Client ein sicherer TLS-Kanal zum Server aufgebaut, über den dann anschließend die HTTP-Daten transferiert werden. TLS ist jedoch nicht nur auf HTTP zugeschnitten, sondern bietet der Anwendungsebene (Schicht 7) Sicherheitsdienste an, die auf den Diensten der Transportebene aufsetzen, so dass über das TLS-Protokoll z.B. auch sichere FTP- oder E-Mail-Verbindungen aufgebaut werden können. Im Gegensatz zu SSL verwendet TLS das HMAC-Verfahren zur Berechnung der MAC-Werte. TLS verwendet ferner ein modifiziertes Schlüsselerzeugungsverfahren, wodurch eine stärkere Robustheit gegen Angriffe auf Hashwerte, die bei der Schlüsselgenerierung als Pseudozufallszahlengeneratoren eingesetzt werden, erreicht wird. TLS hat auch die Menge der Alert-Nachrichten erweitert, wobei alle Erweiterungen als fatale Warnungen eingestuft sind. Beispiele solcher Erweiterungen sind die Warnungen, dass eine unbekannte CA (Certification Authority) angegeben ist, dass die maximale Länge eines Record-Layer-Fragments überschritten worden ist, oder dass eine Entschlüsselungsoperation fehlgeschlagen ist.

Die aktuelle Version TLS 1.2 ist im RFC 5246 spezifiziert. Im März 2018 wurde der Internet-Draft von TLS 1.3 veröffentlicht¹⁴. Auf die Unterschiede zwischen TLS1.2 und TLS1.3 gehen wir am Ende der Ausführungen zu TLS noch kurz ein.

Implementierung

Implementierungen der Protokolle SSL und TLS stehen über Open Source Bibliotheken wie OpenSSL oder GnuTLS zur Verfügung. Des weiteren bieten viele Software-Plattformen, wie beispielsweise Windows in dem Secure Channel Package, Implementierungen beider Protokolle und SSL/TLS werden auch von den gängigen Browsern, wie Internet Explorer, Chrome, Firefox, Opera oder Safari unterstützt.

Abbildung 14.24 zeigt die Einordnung des TLS-Protokolls in die Schichtenarchitektur des TCP/IP-Modells.

¹³ HTTP Secure.

¹⁴ <https://datatracker.ietf.org/doc/draft-ietf-tls-tls13/> gültig bis September 2018

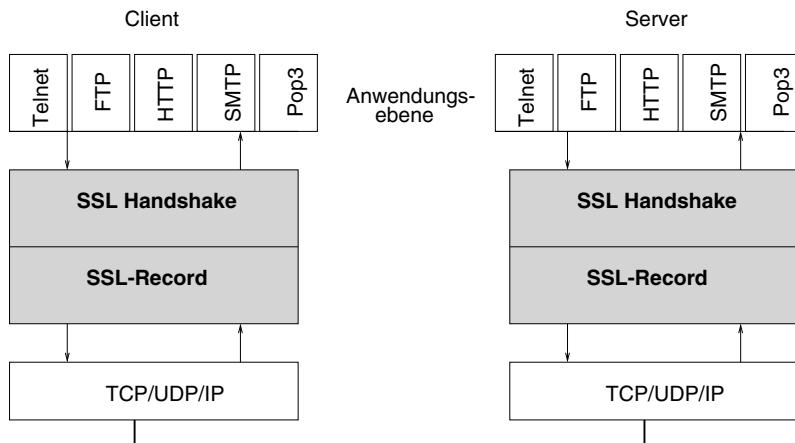


Abbildung 14.24: Schichteneinordnung des TLS-Protokolls

14.4.1 Überblick

Die Hauptaufgaben von TLS sind die Authentifikation der Kommunikationspartner unter Verwendung von asymmetrischen Verschlüsselungsverfahren und Zertifikaten, die vertrauliche Ende-zu-Ende-Datenübertragung unter Nutzung eines gemeinsamen Sitzungsschlüssels und schließlich auch die Sicherstellung der Integrität der transportierten Nachrichten unter Nutzung von Message Authentication Codes. Wie schon IPSec so verwendet auch TLS für die bidirektionale Verbindung zwischen Client und Server zwei unterschiedliche Sitzungsschlüssel.

Aufgaben

Die benötigten kryptografischen Verfahren und Hashfunktionen sind nicht a priori festgelegt, sondern werden pro Verbindung bzw. pro Sitzung, die auch mehrere Verbindungen umfassen kann, auf einfache Weise zwischen Client und Server abgesprochen. Anders als beim IPSec ist das Aushandeln der zu verwendenden Verfahren sowie der Austausch der benötigten Schlüssel ein integraler Protokollbestandteil. Wir werden darauf noch genauer eingehen. TLS legt ferner Rahmenvorgaben für die Verfahren zur Schlüsselerzeugung und der Übertragung authentifizierter Daten während der Phase des Verbindungsaufbaus fest. Beim Verbindungsaufbau sind Verfahren zur authentifizierten Verschlüsselung wie AES-GCM oder ChaCha20-Poly1305 zu verwenden und für den Bereich asymmetrischer Verfahren kann zwischen RSA und dem Diffie-Hellman-Verfahren gewählt werden.

Verfahren

Obwohl in TLS eine Kombination aus asymmetrischen und symmetrischen Verschlüsselungstechniken eingesetzt wird, ist es kein Hybridverfahren zum Austausch eines symmetrischen Schlüssels unter Verwendung asymmetrischer Verfahren. Hierbei wird der geheime Schlüssel zwar geschützt, aber dennoch über ein unsicheres Medium übertragen. TLS versucht dagegen,

Informations-austausch

möglichst wenig geheime Information über das nicht vertrauenswürdige Transportsystem, das Internet, zu transportieren. Deshalb wird lediglich eine Basisinformation zwischen Client und Server vereinbart, mit der die beteiligten Partner dann dezentral ihre weiteren Geheimnisse, wie den gemeinsamen Verschlüsselungsschlüssel und die MAC-Schlüssel, berechnen können. Die beiden pro Sitzung verwendeten Sitzungsschlüssel sowie die beiden MAC-Schlüssel werden unter TLS also nie über das unsichere Netz übertragen.

Sitzung

Die in Abbildung 14.24 angegebene Einordnung von TLS verdeutlicht, dass TLS Aufgaben der Sitzungs- und Präsentationsschicht (Schichten 5 und 6) des ISO/OSI-Modells übernimmt. Ein wesentlicher Vorteil der Sitzungsschicht gegenüber der TCP-Ebene besteht darin, dass Zustandsinformationen über einen längeren Zeitraum und über verschiedene Einzelverbindungen hinweg gespeichert und für die Verwaltung genutzt werden können. Für das zustandslose HTTP-Protokoll, das für jeden Zugriff auf eine Webseite eine neue TCP-Verbindung aufbaut, bedeutet das, dass mehrere solcher Verbindungen zu einer Sitzung gebündelt und damit effizienter als die jeweiligen Einzelverbindungen verwaltet werden können.

TLS-Teilschichten

Wie Abbildung 14.24 zeigt, besteht das TLS-Protokoll im Wesentlichen aus zwei Teilen, nämlich dem Record- und dem Handshake-Protokoll. Zu den Aufgaben der Recordschicht gehört die Fragmentierung der Daten der Anwendungsebene in TLS-Records, deren Kompression, die Berechnung von MACs und der gemeinsame Sitzungsschlüssel sowie die Verschlüsselung der TLS-Records. Auf der Handshake-Schicht findet die Authentifikation der Kommunikationspartner, das Aushandeln der zu verwendenden Verfahren und der Austausch benötigter geheimer Informationen statt.

ChangeCipher Spec

Neben diesen beiden Hauptprotokollen gehören zu TLS noch das Change Cipher Spec und das Alert Protokoll, die beide auch auf dem TLS-Record Protokoll aufsetzen. Das Change Cipher Spec Protokoll umfasst lediglich eine Nachricht bestehend aus einem Byte mit dem Wert 1. Mit dieser Nachricht werden die im Handshake ausgehandelten Verfahren als aktuell zu verwendende Verfahren übernommen.

Alert

Mit dem Alert-Protokoll können TLS-spezifische Warnungen an die Kommunikationspartner übermittelt werden. Eine Alert-Nachricht besteht aus zwei Bytes. Mit dem ersten Byte wird die Schwere der Warnmeldung angezeigt. Der Wert 1 beschreibt lediglich eine Warnung, während der Wert 2 einen fatalen Zustand signalisiert, was dazu führt, dass TLS die Verbindung sofort abbricht und auch keine neuen Verbindungen für diese Sitzung mehr eröffnet. Das zweite Byte beschreibt die Alarmsituation. Beispiele hierfür sind die Angabe, dass ein nicht korrekter MAC-Wert empfangen wurde, dass ein Zertifikat bereits abgelaufen oder zurückgerufen ist oder auch dass das

Handshake-Protokoll keine beidseitig akzeptierten Verschlüsselungsverfahren aushandeln konnte.

Um das TLS-Protokoll zwischen einem Client und einem Server abzuwickeln, müssen sich beide Partner zunächst über die Verwendung des Protokolls verständigen. Die erforderliche Information könnte zum Beispiel im Verlauf einer normalen TCP/IP-Sitzung übertragen und dabei der Einsatz von TLS vereinbart werden oder die Abstimmung über die Verwendung von TLS könnte ein integraler Bestandteil von Anwendungsprotokollen sein. Letzteres würde aber die explizite Änderung jedes Anwendungsprotokolls erfordern, was nicht praktikabel ist. In der Praxis wird ein sehr viel einfacherer Weg beschritten, indem für TLS-basierte Anwendungsdienste spezielle Portadressen reserviert sind, über die diese Anwendungsdienste abgewickelt werden. Tabelle 14.1 listet einige der wichtigsten, offiziell von der IANA reservierten Portadressen mit den daran gekoppelten Diensten auf.

reservierte Ports

Bezeichnung	Portadresse	Bedeutung
https	443	TLS-basiertes HTTP
ssmtp	465	TLS-basiertes SMTP
snntp	563	TLS-basiertes NNTP
telnets	992	TLS-basiertes Telnet
ftps	990	TLS-basierte FTP-Kontrollnachrichten
ftp-data	889	TLS-basierte FTP-Daten

Tabelle 14.1: Auswahl reservierter TLS-Portadressen

14.4.2 Handshake-Protokoll

TLS ist ein zustandsbehaftetes Protokoll, so dass, wie bereits erwähnt, Sitzungen zwischen Kommunikationspartnern etabliert werden können. Ein Client kann zu einem Zeitpunkt mehrere solche Sitzungen zum gleichen oder zu verschiedenen Servern unterhalten. Es gehört zur Aufgabe der Handshake-Schicht, die Sitzungsinformationen von Client und Server zu koordinieren und konsistent zu halten.

Handshake

In Tabelle 14.2 sind die Schritte des Handshake-Protokolls zusammengefasst. Im Verlauf dieses Protokolls werden die benötigten geheimen und

Protokollschritte

öffentlichen Informationen ausgetauscht, so dass damit die Kommunikationspartner in der Lage sind, die gemeinsamen MAC- und Sitzungs-Schlüssel dezentral zu berechnen.

Client	Server	Nachricht
1	→	ClientHello
2	←	ServerHello Certificate (optional) ServerKeyExchange (optional) CertificateRequest (optional) ServerHelloDone
3	→	Certificate (optional) ClientKeyExchange CertificateVerify (optional) ChangeCipherSpec Finished
4	←	ChangeCipherSpec Finished
5	↔	Anwendungsdaten

Tabelle 14.2: TLS-Handshake-Protokoll

ClientHello

Um eine Verbindung zum Server aufzubauen, muss der Client zunächst seine ClientHello-Nachricht absenden. Mit dieser Klartextnachricht werden bereits Informationen ausgetauscht, die in späteren Schritten zur Berechnung der gemeinsamen Geheimnisse erforderlich sind. Die wichtigsten dieser Informationen umfassen eine Datenstruktur R_c bestehend aus einem 32-Bit Zeitstempel und einer 28-Byte Zufallszahl, einem Sitzungsidentifikator, und einer Prioritätenliste (Cipher Suite) mit denjenigen kryptografischen und Kompressionsverfahren, die der Clientrechner unterstützt. Eine solche Cipher Suite könnte beispielsweise wie folgt lauten: AES-128Bit, SHA2-MAC. Das bedeutet, dass zur symmetrischen Verschlüsselung der AES mit einer Schlüssellänge von 128 Bit und zur Berechnung der Hashwerte ein SHA2-MAC verwendet werden soll. Für TLS sind eine Reihe solcher Cipher-Suites vordefiniert, so dass nicht beliebige Kombinationen von Verfahren abgestimmt werden können. Der Sitzungsidentifikator besitzt genau dann einen Wert ungleich Null, wenn eine schon bestehende Sitzung genutzt oder eine frühere Sitzung wieder aufgenommen werden soll. Der Random-

R_c

wert R_c hat die Aufgabe einer Nonce und dient somit zur Abwehr von Wiedereinspielungsangriffen beim Schlüsselaustausch.

Der Server antwortet seinerseits mit einer Hello-Nachricht, die ebenfalls eine Datenstruktur R_s bestehend aus einem 32-Bit Zeitstempel und einer 28-Byte Zufallszahl sowie einer Liste von kryptografischen und Kompressions-Verfahren enthält. Diese Verfahren wählt der Server aus der Liste des Clients aus, vorausgesetzt, dass er sie selber ebenfalls implementiert hat. Damit sind die für die Sitzung zu verwendenden Verfahren ausgehandelt. Die von Client und Server jeweils unterstützten Verfahren werden in Konfigurationsdateien festgelegt, wobei die Reihenfolge der Listeneinträge deren Priorität bestimmt. Der Server sucht gemeinsame Verfahren mit möglichst hoher Priorität aus. Möchte der Client eine bestehende Sitzung nutzen, so sucht der Server den entsprechenden Sitzungsidentifikator in seinem Sitzungs-Cache. Falls er ihn findet und zusätzlich auch bereit ist, unter diesem Sitzungsidentifikator eine weitere Verbindung aufzubauen, so antwortet er mit dem gleichen Identifikator. Gibt der Server keine Angabe zum Sitzungsidentifikator zurück, zeigt er dadurch an, dass diese Sitzung von ihm nicht gespeichert wird und damit zu einem späteren Zeitpunkt vom Client auch nicht wieder aufgenommen werden kann.

Soll der Server authentifiziert werden, so muss er dem Client sein Zertifikat zukommen lassen, das dem in der Hello-Nachricht abgesprochenen Format zu entsprechen hat. In der Regel werden hierzu X.509 Zertifikate eingesetzt. Falls der Server auch eine Authentifikation des Clients fordert (CertificateRequest-Nachricht), antwortet der Client mit seinem eigenen Zertifikat in analoger Weise. In der optionalen CertificateRequest Nachricht gibt der Server X.500 Namen von denjenigen Certification Authorities (CAs) an, denen er vertraut, und teilt dem Client gleichzeitig mit, welche Verfahren er beherrscht (RSA, DSA etc.). Versendet der Server kein Zertifikat, so übermittelt er dem Client in der ServerKeyExchange-Nachricht einen temporären öffentlichen RSA-Schlüssel. Ein solcher Schlüssel wird zum Beispiel benutzt, wenn aufgrund von Exportrestriktionen der zertifizierte öffentliche Schlüssel des Servers wegen seiner zu großen Länge nicht verwendbar ist. In diesem Fall darf für den temporären RSA-Schlüssel auch nur ein 512-Bit Modul eingesetzt werden. Temporäre Schlüssel können mehrfach genutzt werden, sollten aber häufig (z.B. täglich oder nach einer festgelegten Anzahl von Transaktionen) geändert werden. Dies gilt natürlich insbesondere dann, wenn nur ein 512-Bit Modul verwendet wird. Anzumerken ist jedoch, dass in nahezu allen TLS Implementierungen heutzutage eine Server-Authentifizierung erfolgt. Der Server schließt die Übertragung der Nachrichtenfolge des Protokollschriftes 2 mit seiner ServerHelloDone-Nachricht ab.

ServerHello

R_s

Zertifikat

temporärer
Schlüssel

Pre-Master Secret

Nach Erhalt der Daten aus Schritt 2 kontrolliert der Client zunächst die Gültigkeit des Server-Zertifikats. Weiterhin prüft er, ob die vom Server ausgewählte Liste von Verfahren tatsächlich mit seinen Angeboten verträglich ist. In diesem Fall übermittelt er mit der ClientKeyExchange-Nachricht dem Server eine geheime Basisinformation, das 48-Byte (384 Bits) Pre-Master Secret Pre . Haben sich die beiden Partner auf die Verwendung des RSA-Verfahrens verständigt, so verschlüsselt der Client das Geheimnis mit dem öffentlichen Schlüssel des Servers, $RSA(Pre, Public_S)$. Beim Einsatz des Diffie-Hellman-Verfahrens (DH) sendet der Client in dieser Nachricht nur seinen öffentlichen Schlüssel an den Server zurück. Man beachte, dass das DH-Verfahren in TLS nicht zur Berechnung des geheimen gemeinsamen Schlüssels, sondern zur dezentralen Berechnung des Pre-Master Secrets eingesetzt wird.

Master Secret

Das Pre-Master Secret Pre und die in den Hello-Nachrichten ausgetauschten Zufallszahlen R_c und R_s werden von Client und Server dazu verwendet, das 48-Byte Master Secret zu berechnen, aus dem dann die benötigten geheimen Schlüssel abgeleitet werden.

$$\text{master_secret} = \text{PRF}(Pre, \text{master secret}, R_c + R_s);$$

restliche Protokollschrifte

Falls der Client ein Zertifikat vorweisen muss, sendet er die CertificateVerify-Nachricht, um dem Empfänger die Überprüfung des Zertifikats zu ermöglichen. Die Nachricht ist ein MAC über alle bislang im Protokoll ausgetauschten Nachrichten und wird mit dem privaten Schlüssel des Clients signiert.

Mit der ChangeCipherSpec-Nachricht zeigen sowohl der Client als auch der Server an, dass sie ab jetzt die ausgehandelten kryptografischen Verfahren verwenden. Über die Finished-Nachricht signalisieren beide die erfolgreiche Beendigung ihres jeweiligen Anteils am Protokoll. Diese Nachricht ist ebenfalls eine Datenstruktur bestehend aus einem MAC über alle bislang im Protokoll ausgetauschten Nachrichten. Falls die MACs beider Partner übereinstimmen, wird ohne weitere Acknowledgements die etablierte Verbindung akzeptiert und beide können jetzt mit der Übertragung von Anwendungsdaten beginnen.

14.4.3 Record-Protokoll

Record-Protokoll

Die Record-Schicht hat die Aufgabe, Datenpakete in TLS-Records mit einer maximalen Größe von 2^{14} Byte zu fragmentieren, optional zu komprimieren, diese Daten mit einem MAC zu hashen und sie zusammen mit ihrem MAC zu verschlüsseln. Abbildung 14.25 fasst die wesentlichen Aufgaben des Record-Protokolls zusammen. Die Kompression muss verlustfrei sein und darf die Länge des Fragments nicht um mehr als 1024 Bytes vergößern. Bei

sehr kleinen Blöcken kann es vorkommen, dass durch die Kompression der Block vergrößert anstatt verkleinert wird.

Nach der Verschlüsselung des MACs zusammen mit der komprimierten Nachricht darf die Länge des Kryptotextes die des Klartextes um höchstens 1024 Byte übersteigen. Der TLS-Record Header enthält unter anderem Informationen über die verwendete TLS-Version, den Content-Type, der angibt, welches TLS-Protokoll (Alert, Handshake, ChangeCipherSpec) zu verwenden ist, sowie die Länge in Bytes des Klartextfragments. Es ist übrigens wichtig und sinnvoll, dass eine Kompression vor der Verschlüsselung erfolgt, da ein Kompressionsverfahren im Prinzip darauf basiert, in dem Originaltext nach Mustern zu suchen und Duplikate zu eliminieren. Existieren nur wenige Muster, was ja bei einem verschlüsselten Text der Fall ist, versagt das Komprimieren und führt ggf. sogar zu einer Vergrößerung des Datenvolumens.

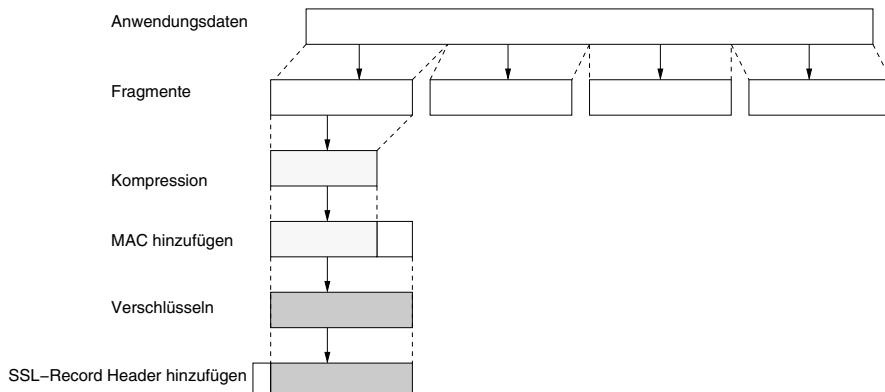


Abbildung 14.25: Aufgaben des TLS-Record Protokolls

Es bleibt somit zu klären, wie die benötigten Schlüssel, nämlich je ein MAC-Schlüssel für den Client und den Server sowie je ein geheimer Schlüssel für die Verbindung vom Client zum Server und umgekehrt, erzeugt und zwischen den Partnern abgestimmt werden.

Alle diese Schlüssel werden nach einem gleichartigen Schema sowohl vom Client als auch vom Server aus dem Master Secret (ms) abgeleitet. Dazu generiert das Protokoll so lange eine Folge von Schlüsselblöcken key_block, bis alle Schlüssel damit konstruiert sind.

Schlüsselerzeugung

Sitzungs- und Verbindungskonzept

Wie im Überblick bereits erwähnt und in den Protokollbeschreibungen gesehen, unterscheidet TLS zwischen Sitzungen und Verbindungen. Mit dem

bereits Erklärten können wir jetzt festhalten, durch welche Informationen eine Verbindung bzw. eine Sitzung unter TLS beschrieben ist.

Sitzungszustand

Eine Sitzung wird charakterisiert durch

- den Sitzungs-Identifikator: Das ist eine beliebige, durch den Server generierte Bytefolge, die eine aktive oder wieder aufnehmbare Sitzung charakterisiert.
- Zertifikat: Es handelt sich hierbei um ein X509.v3 Zertifikat des Partners.
- Kompressionsverfahren, wie z.B. zip,
- CipherSpec: Diese Spezifikation beschreibt die zu verwendenden Verschlüsselungs- sowie Hashverfahren und legt alle benötigten kryptografischen Attribute fest, wie zum Beispiel die Länge des Hashwertes.
- Master Secret, das zwischen Client und Server vereinbart ist.

Verbindungs- zustand

Eine Sitzung kann mehrere Verbindungen umfassen; jede Verbindung gehört zu genau einer Sitzung. Ein Verbindungszustand wird durch folgende Informationen beschrieben:

- Server- und Client-Randomzahlen der entsprechenden Verbindung,
- Server MAC-Schlüssel, der verwendet wird, wenn der Server Daten zum Client sendet,
- Client MAC-Schlüssel, der verwendet wird, wenn der Client Daten zum Server sendet,
- Serverschlüssel, der verwendet wird, wenn der Server Daten verschlüsselt zum Client sendet,
- Clientschlüssel, der verwendet wird, wenn der Client Daten verschlüsselt zum Server sendet,
- Initialisierungsvektor
- Sequenznummern: Jeder Partner verwaltet eigene Sequenznummern für gesendete und empfangene Datenpakete. Beim Senden bzw. Empfangen einer ChangeCipherSpec Nachricht werden die Sequenznummern der Verbindung auf 0 zurückgesetzt. Sequenznummern haben eine maximale Größe von $2^{64} - 1$.

An den Zuständen ist noch einmal abzulesen, dass die verwendeten Schlüssel nur jeweils für eine Verbindung gelten, während für alle Verbindungen einer Sitzung die gleichen Verfahren genutzt werden, so dass man bei der erneuten Verwendung einer bereits bestehenden Verbindung auf das Aushandeln der Verfahren im Handshake verzichten kann.

14.4.4 Sicherheit von TLS

TLS ermöglicht eine authentifizierte, vertrauliche und vor unbefugten Modifikationen geschützte Datenkommunikation zwischen zwei Kommunikationsendpunkten. Das Protokoll unterstützt verschiedene Authentifikations-schemata, die von einer wechselseitigen über eine nur einseitige Authentifikation bis hin zur vollständig anonymen Kommunikation reichen. Zu beachten ist, dass im letzten Fall auch die Vertraulichkeit der Kommunikation nicht notwendigerweise gewährleistet ist, da ggf. eine verschlüsselte Kommunikation zu einem Angreifer-Server aufgebaut wird. Die Daten sind dann zwar auf dem Transportweg verschlüsselt, so dass über einen passiven Angriff keine sensiblen Informationen erhältlich sind, aber die Verschlüsselung endet beim TLS-Endpunkt, also beim Angreifer-Server, bei dem die Daten dann im Klartext vorliegen. Diese Situation tritt beispielsweise ein, wenn bei einem Phishing-Angriff die Eingaben des Benutzers TLS-geschützt auf einen Angreifer-Server umgeleitet werden. Zu beachten ist ferner auch, dass die Art der Authentifikation allein vom Server bestimmt wird.

Authentifikation

Die Sicherheit des Handshake-Protokolls ist eine wesentliche Voraussetzung für die Sicherheit des gesamten TLS-Protokolls. Falls es nämlich einem Angreifer gelänge, dafür zu sorgen, dass die Kommunikationspartner schwache Verschlüsselungsverfahren oder schwache Schlüssel (z.B. 40 Bit) aushandeln, so könnte er anschließend mit großer Aussicht auf Erfolg versuchen, den verwendeten Kommunikationsschlüssel zu brechen. Für das Gelingen eines solchen Angriffs ist die Modifikation von Handshake-Nachrichten erforderlich. Das hätte aber zur Konsequenz, dass Client und Server in ihrer jeweiligen Finished-Nachricht unterschiedliche Hashwerte berechnen, da dieser MAC ja über alle von ihnen empfangenen und gesendeten Nachrichten gebildet wird. Die Finished-Meldungen würden deshalb wechselseitig nicht akzeptiert und der Verbindungsaufbau würde erfolglos terminieren. Der Angriffsversuch wäre somit abgewehrt.

Handshake-Protokoll

Um auch die Finished-Nachrichten gezielt zu modifizieren, benötigt der Angreifer das Master Secret, da dies zur Berechnung des MAC-Werts verwendet wird. Das Master Secret ist seinerseits aus dem Pre-Master Secret abgeleitet, so dass der Angreifer dieses Geheimnis kennen muss. Zu dessen Erlangung ist aber der private RSA- oder DH-Schlüssel des Servers notwendig. Bei geeignet großen Schlüsseln kann somit auch ein Angriff auf das Master Secret erheblich erschwert bzw. abgewehrt werden. Mit der Finished-Nachricht wird ein Hash-Wert über eine ganze Abfolge von Protokollschriften erstellt, so dass nicht nur die Integrität eines einzelnen Datenpakets, sondern vielmehr auch die der gesamten Nachrichtenabfolge sichergestellt ist. An dieser Stelle sei angemerkt, dass die eigentliche Authentifikation des Servers natürlich erst in diesem abschließenden Schritt

wirklich erfolgt ist. Denn mit dem korrekten Hash-Wert hat der Server die Kenntnis des korrekten geheimen Schlüssels nachgewiesen, der zu dem öffentlichen Schlüssel gehört, dessen Authentizität durch das Server-Zertifikat bestätigt wurde.

Perfect Forward Secrecy

Bereits in Kapitel 9.3 wurde auf die Bedeutung des Konzepts der Perfect Forward Secrecy (PFS) für die nachhaltige Sicherheit verschlüsselter Daten hingewiesen (vgl. Seite 418). Das TLS Protokoll ermöglicht die Umsetzung der Perfect Forward Secrecy, wenn im TLS-Handshake-Protokoll als Schlüsselaustauschverfahren das Diffie-Hellman-Verfahren verwendet wird. Da der für eine TLS-Verbindung benötigte Kommunikationsschlüssel auf der Basis temporär erzeugter DH-Schlüssel dezentral berechnet wird und der neue Kommunikationsschlüssel nicht von bereits etablierten Schlüsseln abhängig ist, wird gewährleistet, dass auch nach einer potentiellen Offenlegung des privaten Schlüssels, z.B. des Servers, daraus nicht nachträglich der Sitzungsschlüssel für verschlüsselte Kommunikation aus früheren TLS-Verbindungen, die aufgezeichnet wurden, entschlüsselt werden. Bis zum Sommer 2013 war jedoch die PFS Variante von TLS nur sehr wenig verbreitet, da diese Variante einen beträchtlichen zusätzlichen Overhead erfordert. Im Zuge der NSA-Affaire im Sommer 2013, bei der die massenhafte Aufzeichnung von unverschlüsselter, aber auch verschlüsselter Kommunikation, wie verschlüsselter TLS-Verbindungen, durch US amerikanische Geheimdienste auch einer breiten Öffentlichkeit bewusst wurde, gewann die Frage der Umsetzung des PFS-Konzepts bei TLS Verbindungen für Provider stark an Bedeutung. So fordert das BSI für die Bundesverwaltung als Mindeststandard die Verwendung von TLS 1.2 mit Perfect Forward Secrecy.

Klar ist auch, dass ein Man-in-the-Middle eine TLS-Verbindung übernehmen kann, wenn Client und Server sich nicht wechselseitig stark authentifizieren.

schwache Kryptografie

Die Sicherheit des Protokolls hängt zum einen von den verwendeten kryptografischen Verfahren ab, die die Kommunikationspartner im Handshake miteinander abstimmen, und zum anderen natürlich von der sicheren Implementierung der Funktionen, was im April 2014 durch den so genannten Heartbleed-Angriff auf OpenSSL-Software-Bibliotheken eindrucksvoll bewiesen wurde.

Heartbleed-Angriff auf OpenSSL

Sicherheitslücke

Anfang April 2014 wurde eine massive Sicherheitslücke in OpenSSL-Bibliotheken bekannt. Ursache für die aufgedeckte Schwachstelle ist ein Implementierungsfehler in der Heartbeat-Funktion von OpenSSL-Bibliotheken der Versionen 1.0.1 bis 1.0.1f. Die Schwachstelle besteht in einer fehlenden Überprüfung von Parametern. Die Verwundbarkeit wurde durch die so genannte Heartbleed-Attacke ausgenutzt. Da diese OpenSSL-Bibliotheken

sehr häufig als Basis von anderen Anwendungen genutzt werden, sind von der Schwachstelle neben Web-Servern wie Apache, auch Betriebssystemkomponenten oder Anwendungen wie IMAP, POP oder auch SMTP betroffen. Aufgrund der Vielzahl der angreifbaren Anwendungen verursacht die Beseitigung der Schwachstelle ganz erhebliche Aufwände, da private Schlüssel und Zertifikate erneuert, alte Zertifikate gesperrt, Passworte geändert und auf einer Vielzahl von Systemen Patches eingespielt werden müssen. Problematisch war zudem, dass die Sicherheitslücke bereits mit dem OpenSSL Release 1.0.1 vom 14.03.2012 verbreitet, aber erst mit Release 1.0.1.g am 07.04.2014 behoben wurde.

Ausgangspunkt für den Angriff ist die Heartbeat-Funktion in OpenSSL, die es ermöglicht, eine TLS Verbindung zwischen Client und Server aufrecht zu erhalten. Dazu sendet einer der Partner eine Nachricht mit beliebigem Inhalt an den jeweils anderen und der Empfänger sendet die Daten wieder zurück, um damit zu dokumentieren, dass die Verbindung noch in Ordnung ist. Um Daten des Anfragers (Client) in den Speicher des Empfängers (Servers) zu kopieren, wird das Kommando *memcpy(bp, pl, payload)* genutzt. Mit dem Kommando *memcpy* werden Daten kopiert, wobei der erste Parameter *bp* die Adresse angibt, wohin die Daten auf dem Server zu kopieren sind, der zweite Parameter *pl* spezifiziert die Adresse des zu kopierenden Bereichs und der dritte Parameter *payload* gibt die Länge der zu kopierenden Daten an. Abbildung 14.26 skizziert den Vorgang bei einem korrekten Heartbeat-Protokoll-Ablauf bezogen auf das Kopieren der Anfragedaten vom Client in den Pufferbereich des Servers. Der Server sendet die Daten in seinem Antwortpaket wieder an den Client zurück (vgl. Abbildung 14.26).

Heartbeat-Funktion

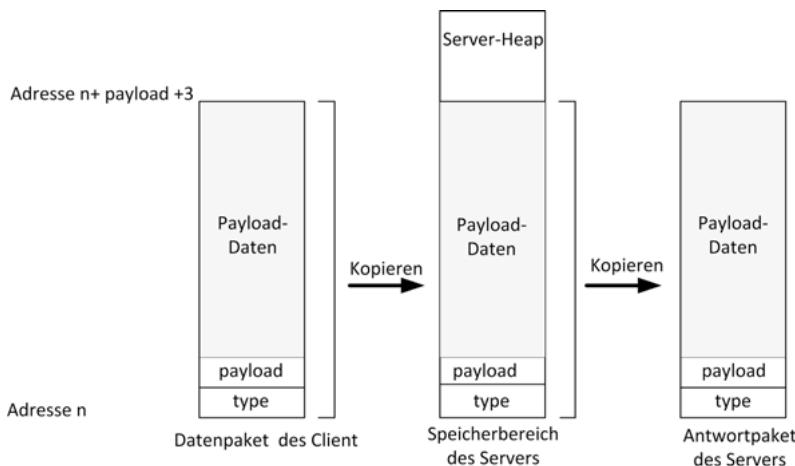


Abbildung 14.26: Korrekter Ablauf des Heartbeat-Protokolls

Die ausnutzbare Schwachstelle bestand darin, dass die Länge der übertragenen Daten nicht überprüft wird. Über den Parameter *payload* gibt der Sender zwar an, wie lang das gesendete Payload ist, aber der Empfänger prüft nicht, ob die empfangenen Daten dieser Längenangabe gehorchen. Der Empfänger vertraut also darauf, dass der Sender eine korrekte Angabe sowohl in Bezug auf die Länge der Eingabe als auch auf die tatsächliche Eingabe macht.

Hier setzte der unten beschriebene Heartbleed-Angriff an. Anders als bei klassischen Buffer-Overflow-Angriffen wurde jedoch nicht eine zulange Eingabe an den Empfänger gesendet, sondern die tatsächliche Eingabe war deutlich kürzer als die angegebene Payload-Länge. Im RFC 6520 für die Heartbeat-Funktion wird angegeben, dass das Payload eine Länge von 16 KByte (also maximal 2^{14}) nicht überschreiten sollte. Da aber auch diese Angabe von vielen OpenSSL-Implementierungen nicht überprüft wird, kann auch ein 64 KByte großer Bereich als Payload angegeben werden, indem die maximale Payload-Größe, also `0xFFFF`, verwendet wird.

Angriff

Bei dem Heartbleed-Angriff¹⁵ sendet der Angreifer ein Payload mit geringer Größe, z.B. 1 Byte, und behauptet gleichzeitig, dass er ein deutlich größeres Paket sendet, z.B. 16 KByte, d.h. der Parameter *payload* wird entsprechend groß angegeben. Der Empfänger reserviert in seinem Speicher einen Pufferbereich *P* mit einer Länge wie in *payload* gefordert. In dem Angriffsbeispiel reserviert der Empfänger also einen Bereich von 16 KByte. Er kopiert als nächstes die empfangenen Daten, im Beispiel ist das lediglich 1 Byte, in den reservierten Bereich *P* und sendet seine Antwort mit der Länge *payload* zurück. Im Angriffsbeispiel wird somit eine Antwortnachricht von 16 KByte gesendet. Der Server versendet hierfür die Daten aus dem allokierten Speicherbereich *P*, also Daten, die der Server ggf. gerade bearbeitet. Aufgrund der speziellen Speicherverwaltung von OpenSSL stehen an der Speicherstelle nicht allgemeine Daten des Servers, sondern spezifische Daten der OpenSSL Implementierung, also beispielweise auch Zugangsdaten für Verbindungen anderer Nutzer, Passworte, Server- und Verbindungsschlüssel etc. Abbildung 14.27 fasst die wesentlichen Schritte dieses Angriffs noch einmal zusammen.

Fehlerbehebung

Der Implementierungsfehler betrifft die Funktion *tls1_process_heartbeat* der Datei *t1_lib.c* der OpenSSL-Bibliothek. Behoben wurde der Fehler durch eine zusätzliche Kontrolle, bei der die Länge der Variable *payload* mit der tatsächlichen Größe des empfangenen Payloads, das in der SSL3_RECORD-Struktur (*s3->rrec*) abgelegt ist, verglichen wird. Ist die Payload-Länge zu groß angegeben, so wird das Paket durch den Server verworfen.

¹⁵ In der Schwachstellendatenbank CVE (Common Vulnerabilities and Exposures) unter <http://cve.mitre.org/> wird die Schwachstelle unter dem Namen CVE-2014-0160 geführt.

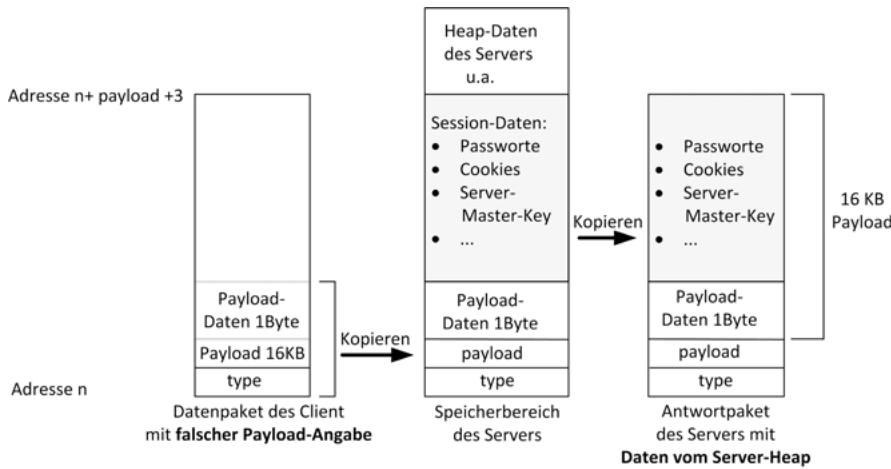


Abbildung 14.27: Ablauf eines Heartbleed-Angriffs

```
if (1 + 2 + payload + 16 > s->s3->rrec.length)
return 0; /* silently discard per RFC 6520 sec. 4 */
```

Durch den Heartbleed-Angriff wird, anders als bei den klassischen Buffer-Overflow-Angriffen, beim Server kein Speicherbereich unzulässig überschrieben. Der Angriff führt deshalb zu keiner Beeinflussung des Serververhaltens und bleibt für diesen verborgen. Der Angriff bzw. die Schwachstelle, die den Angriff ermöglichte, hat einmal mehr die Bedeutung der sicheren Programmierung mit u.a. Eingabeprüfungen verdeutlicht. Zudem hat der Angriff auch noch einmal den Mehrwert der Nutzung der Perfect Forward Secrecy (PFS) bei SSL/TLS verdeutlicht. Die Verwendung von PFS hätte die Schwachstelle zwar nicht geschlossen, aber das Schadensausmaß reduziert. Die Nutzung von PFS stellt sicher, dass ein mittels der Heartbleed-Attacke kompromittierter, aktueller privater Server-Schlüssel nicht genutzt werden kann, um damit dann auch verschlüsselte Verbindungen, die in der Vergangenheit aufgezeichnet wurden, im nachhinein zu entschlüsseln.

Lessons learned

Grenzen

Auf TLS treffen alle Vorteile aber auch die Einschränkungen von Ende-zu-Ende-Sicherheitsprotokollen unterhalb der Anwendungsebene zu, die wir ja bereits angeführt haben. Da die Sicherheit des Protokolls in erster Linie von der sicheren Verwaltung der Schlüssel, insbesondere natürlich des privaten Schlüssels des Servers abhängt, kommt dessen sicherer Verwaltung eine besondere Bedeutung zu. Die meisten TLS-Server sind General Purpose Rechner, die nicht als vertrauenswürdig einzustufen sind. Da bei jedem Verbindungsaufbau der Private-Key des Servers verwendet werden muss,

ist klar, dass er im Hauptspeicher zu verwalten ist. Damit das sensitive Datum nicht im Zuge eines normalen Swap-Vorgangs im Zusammenhang mit der virtuellen Speicherverwaltung auf die Festplatte geschrieben wird, muss durch geeignete Lock-Konzepte ein solches Auslagern verhindert werden. Neben den allgemeinen Problemen weist das Protokoll aber auch noch einige protokollspezifische Problembereiche auf, die im Folgenden angerissen werden.

keine
Verbindlichkeit

Da die übertragenen Daten nicht signiert werden, stellt TLS keine Verbindlichkeit von Aktionen sicher. Weiterhin werden auch keine Maßnahmen zur Abwehr von Verkehrsflussanalysen ergriffen, da nur die Nutzdaten in den TCP/IP-Paketen verschlüsselt werden.

Firewall

Das TLS-Protokoll kann nicht ohne weiteres zusammen mit Applikationsfilter-Firewalls betrieben werden. Ein dedizierter Proxy-Server eines Applikationsfilters wird von TLS als Mittelsmann eines Man-in-the-middle-Angriffs betrachtet, so dass der Verbindungsaufbau scheitern würde. Der TLS-Datenverkehr ist somit entweder zu tunneln oder es kann nur ein generischer Proxy-Firewall oder sogar nur ein Paketfilter verwendet werden, der die Nachrichten an die reservierten TLS-Ports ungehindert passieren lässt. Bei der Paketfilterlösung besteht natürlich die Gefahr, dass ein interner Angreifer diese reservierten Portadressen für andere Zwecke verwendet, ohne dass dies für den Paketfilter erkennbar ist. Ein TLS-Tunnel bedeutet, dass der Firewall den Datenverkehr nicht mehr analysieren und somit seinen Überwachungsfunktionen nicht mehr gerecht werden kann.

E2E
Verschlüsselung

TLS etabliert eine verschlüsselte Kommunikation zwischen zwei Endpunkten (Ende-zu-Ende). Dies ist für viele heutige Szenarien noch angemessen. Mit der Zunahme komplexerer Transaktionen, an denen mehrere Server beteiligt sind, zum Beispiel im SOA oder Cloud-Kontext, wird zunehmend der Bedarf auftreten, einzelnen Servern nur Teile der Nachrichten sichtbar zu machen. Dafür sind erweiterte Konzepte erforderlich, beispielsweise analog zum Onion-Routing, damit nicht Sicherheitslücken dadurch entstehen, dass die Daten stets vollständig auf den jeweiligen Servern entschlüsselt werden, um weiterverarbeitet werden zu können.

Vertrauenswürdigkeit von SSL-Zertifikaten

Das Protokoll https, das zur sicheren Kommunikation zwischen zwei Endpunkten auf TLS aufsetzt, ist der De-Fakto-Standard für das sichere Browsen im Internet geworden. Mit dem TLS-Protokoll wird eine Vertrauensbeziehung zwischen den beteiligten Endpunkten aufgebaut, die auf den verwendeten SSL-Zertifikaten der Server und bei einer wechselseitigen Identifizierung auch der Clients beruht. Die Vertrauenswürdigkeit der SSL-Zertifikate

und deren korrekte Validierung sind somit zentrale Eckpfeiler der Sicherheit in TLS-basierter Kommunikation. In heutigen Systemen wird diese Vertrauensbeziehung jedoch durch verschiedene Schwachpunkte in Frage gestellt. Im Folgenden werden wichtige Problembereiche deshalb abschließend diskutiert.

Ein TLS-Nutzer muss den Informationen, die ihm zur Prüfung der Zertifikate vorliegen und im Browserfenster angezeigt werden, vertrauen können. Im Zusammenhang mit Problemen des Web-Spoofings (siehe Seite 160) haben wir jedoch bereits darauf hingewiesen, dass hier in heutigen Systemen eine Sicherheitslücke klafft, da es Angreifern möglich ist, dem Benutzer gefälschte Seiten und damit auch gefälschte Zertifikatsinformationen vorzugaukeln.

Das Vertrauenskonzept, das SSL-Zertifikaten zugrunde liegt, wird durch die gängige Praxis in heutigen Systemen stark ausgeöhlt, da über die bei der Installation vorgenommenen Voreinstellungen standardmäßig eine Vielzahl von Zertifizierungsinstanzen eingetragen sind, deren Zertifikaten automatisch vertraut wird, ohne dass der Benutzer selbst die Vertrauenswürdigkeit des Zertifikats überprüfen kann. Im etwas weiteren Sinn kann man diese Vorgehensweise wiederum als einen Verstoß gegen das Erlaubnis-Prinzip betrachten.

Im Jahr 2011 gelang es Hackern, Zertifizierungsstellen, die SSL-Zertifikate ausstellen, zu kompromittieren, wodurch eine Vielzahl von gefälschten Zertifikaten für wichtige Domänen in Umlauf kam. Den Anfang machte ein Hacker-Angriff auf Registrierungsstellen des italienischen Zertifikat-Resellers Comodo. Dieser Angriff hatte zur Folge, dass der Angreifer sich über einen dieser kompromittierten Comodo-Registrare, den InstantSSL, gültige Zertifikate unter anderem für Microsoft, Mozilla, Yahoo, Skype und Google ausstellen lassen konnte. Diese Registrierungsstellen (Registrare) haben die Aufgabe, die Echtheit der Informationen im Zertifikatsantrag zu überprüfen, so dass sie nach erfolgreicher Prüfung von der Zertifizierungsstelle, in diesem Fall war das der Reseller Comodo, signiert werden.

Noch gravierendere Ausmaße hatte der im Frühjahr des gleichen Jahres durchgeführte Angriff auf die niederländische Zertifizierungsinstanz DigiNotar¹⁶. Den Hackern gelang es, in die Infrastruktur von DigiNotar einzubrechen und alle acht Zertifizierungsstellen (CAs) unter ihre Kontrolle zu bringen. Die Angreifer ließen sich von den kompromittierten CAs über 500 gefälschte SSL-Zertifikate ausstellen. Zu den betroffenen Domänen, für die gefälschte Zertifikate ausgestellt wurden, gehören neben google.com, microsoft.com, facebook.com, twitter.com, aol.com, android.com und skype.com

Eingespielte
Zertifikate

Vertrauenswürdige
CAs

Gefälschte
Zertifikate

DigiNotar

¹⁶ vgl. <http://www.rijksoverheid.nl/bestanden/documenten-en-publicaties/rapporten/2012/08/13/black-tulip-update/black-tulip-update.pdf>.

auch die Domänen einiger Geheimdienste, wie www.sis.gov.uk (MI6) oder www.cia.gov. und www.mossad.gov.il. Der Einbruch wurde erst Monate später entdeckt, so dass erst dann die kompromittierten Zertifikate zurückgezogen werden konnten. Mit den gefälschten Zertifikaten konnten die Angreifer dafür sorgen, dass beispielsweise eine Vertrauensbeziehung zwischen einem Client und einem von ihnen kontrollierten Server aufgebaut und sensitive Information an den vermeintlich authentischen Server übermittelt wurde.

Zertifikatprüfung

Ein weiteres sehr gravierendes Problem betrifft die SSL-Zertifikatsprüfung auf den beteiligten Rechnern bei Anwendungen, die nicht Browserbasiert ausgeführt werden. Derartige Anwendungen treten heutzutage in vielfältigen Ausprägungen auf. So wird TLS beispielsweise genutzt, um mittels Fernzugriffen Cloud-basierte Infrastrukturen zu administrieren und Daten in Cloud-Storage Bereichen abzulegen. Weitere Anwendungsbereiche betreffen die TLS-basierte Übermittlung von Bezahlinformationen von einem E-Commerce-Server zu beispielsweise PayPal oder Amazon zur Bearbeitung der Bezahlungsinformation. Aber auch Verbindungen von Authentisierungsservern zu mobilen Anwendungen auf beispielsweise Android oder iOS Rechnern werden in der Regel TLS-gesichert aufgebaut. Diese Anwendungen implementieren normalerweise keinen eigenen TLS-Stack, sondern nutzen verfügbare TLS-Bibliotheken, wie OpenSSL, GnuTLS, JS-SE,CryptoAPI oder auch Bibliotheken, die eine sichere Kommunikation auf einer höheren Netzwerkschicht anbieten und als eine Art Wrapper um die darunter liegenden TLS-Libraries fungieren. Beispiele sind cURL, Apache HttpClient oder auch urllib. Auch Web-Server Middleware, wie Apache Axis, Axis 2, oder Codehaus XFire bieten solche Wrapper auf einer höheren Abstraktionsschicht an

Sicherheitsanalyse

In [70] wurde ein breit angelegte Sicherheitsanalyse in Bezug auf die Validierung von SSL-Zertifikaten bei der Etablierung von authentisierten TLS-Verbindungen in nicht-Browserbasieter Software durchgeführt. Dabei wurden die gängigen Bibliotheken und Anwendungen für Linux, Windows, Android und iOS Systemen untersucht. Die Untersuchung hat ergeben, dass obwohl die genutzten TLS-Bibliotheken korrekt implementiert waren, ganz erhebliche Sicherheitsprobleme aufgetreten sind, da Software-Entwickler häufig die vielfältigen Optionen und Parameterkonstellationen dieser Bibliotheken missverständlich interpretieren. Als Beispiel nennen die Autoren die PHP Bibliothek des Flexible Payment Services von Amazon, der versucht, eine Verifikation des Hostnamen durchzuführen, indem der Parameter *CURLOPT_SSL_VERIFYHOST* der Bibliothek cURL auf *true* gesetzt wird. Der korrekte Parameterwert ist jedoch der Wert 2 und durch das Setzen des Wertes auf *true* wird der Parameter intern, ohne Fehlerhinweis nach außen, auf

Fehlerhafte Parameterwerte

1 gesetzt, wodurch keine Zertifikat-Validierung durchgeführt wird. Auch in der PHP Bibliothek des PayPal Payment-Standards wurde von den Forschern ein ähnlicher Bug entdeckt.

Als weiteres Beispiel für eine Fehlnutzung von Parametern in SSL-Libraries zeigen die Autoren Probleme mit der Bibliothek JSSE (Java Secure Socket Extension) auf, die unter anderem die API Funktion *SSLSocketFactory* zur Verfügung stellt. Diese API unterdrückt jedoch eine Zertifikatvalidierung, falls im Algorithmen-Feld des TLS-Client anstelle von der erwarteten Angabe https der Wert NULL oder ein leerer String steht. Zu den Software-Produkten, die eine solche Schwäche aufweisen, zählt der Apache HttpClient. Die Sicherheitsanalyse hat gezeigt, dass eine Vielzahl gängiger Software-Suiten, wie Apache Axis oder die EC2 Bibliothek von Amazon, Schwächen bei der Validierung von SSL-Zertifikaten aufweisen und damit anfällig gegen Man-in-the Middle Angriffe sind.

TLS1.2 versus TLS 1.3

TLS1.3 wurde im Vergleich zu TLS1.2 deutlich entschlackt und vereinfacht, so dass die Gefahr für Fehlkonfigurationen reduziert wurde. führt mit Verfahren wie Zero Round Trip Time (0-RTT) zu einer verbesserten Performanz, da in TLS1.3 nur noch ein Roundtrip an Nachrichten im Handshake benötigt werden, anstelle von zwei Roundtrips in TLS1.2. So können bei einer bekannten Seite direkt Daten zum Server übertragen werden (0-RTT). Der Client nutzt dafür einen bei der letzten Verbindung mit dem Server vereinbarten Schlüssel.

TLS1.3

Auch die Sicherheit wird unter TLS1.3. verbessert. Unterschiede zwischen den Versionen 1.2 und 1.3 sind u.a. dass in TLS1.3. keine proprietären symmetrischen Verschlüsselungsverfahren unterstützt werden. TLS1.3 hat die unsicheren Verfahren wie SHA-1, RC4, DES, 3DES, AES-CBC und MD5 entfernt. Alle verbleibenden Verfahren müssen zudem den AEAD Modus, siehe Seite 315, unterstützen. Damit wird der Design-Fehler aus TLS1.2, durch den erst der MAC über den Klartext berechnet und dann die damit erweiterte Nachricht verschlüsselt wird (MAC-then-Encrypt), behoben. Der Design-Fehler hat so genannte Padding-Oracle-Angriffe ermöglicht, bei denen ein Angreifer versucht, byteweise den Klartext zu erraten und die Integritätssicherung des Protokolls dem Angreifer verrät, ob er richtig geraten hat

In TLS1.3 ist ein Austausch des symmetrischen Schlüssels mittels RSA nicht mehr zulässig. Ein Schlüsselaustausch muss mit dem Diffie-Hellman-V erfahren erfolgen, dabei sollte DH mit elliptischen Kurven, also ECDH, genutzt werden. Alle verwendbaren Public-Key Verfahren unterstützen Perfect Forward Secrecy. Zudem wurden die Protokollnachrichten so geändert,

dass bereits nach der ersten Handshake-Nachricht, also nach dem ServerHello, die übertragenen Nachrichten verschlüsselt werden. ECC-Verfahren gehören in der Version 1.3. zu den Basisverfahren und es wurden neue Signaturverfahren, wie ed25519, spezifiziert, DSA wurde entfernt.

14.5 DNSSEC

In Abschnitt 3.4.1 wurden die Sicherheitsprobleme des Domain Name Service (DNS) diskutiert. An Sicherheitserweiterungen von DNS wird bereits seit 1997 unter dem Namen DNSSEC (Domain Name System Security Extension) gearbeitet. Seit 2005 existiert mit den RFCs 4033, 4034 und dem Update von 2013 im RFC 6840 eine Spezifikation von DNSSEC. Schutzziele von DNSSEC sind die Authentizität, sowie die Integrität von DNS-Einträgen, um Angriffe wie DNS-Cache Poisoning abzuwehren. Demgegenüber sind die Gewährleistung der Vertraulichkeit von DNS-Daten, wie beispielsweise der Verschlüsselung von IP-Adressen, um die Privatsphäre zu schützen, oder die Abwehr von Denial-of-Service Angriffen keine Ziele von DNSSEC. DNSSEC nutzt asymmetrische Kryptografie zusammen mit kryptografischen Hashfunktionen (vgl. Kapitel 7), um die Schutzziele zu erreichen.

14.5.1 DNS-Schlüssel und -Schlüsselmanagement

DNSSEC verwendet pro Domäne zwei Public-Key-Schlüsselpaare: den langlebigen Key-Signing-Key (KSK), bestehend aus einem privaten und einem öffentlichen Schlüssel, und den Zone-Signing-Key (ZSK), ebenfalls durch ein Schlüsselpaar festgelegt. Der private KSK sollte eine Länge von 2048 Bit besitzen; er ist 2 bis 4 Jahre lang gültig. Wie der Name bereits veranschaulicht, wird der private KSK dazu verwendet, andere Schlüssel zu signieren. Der private ZSK sollte eine Länge von mindestens 512 Bit besitzen; er ist lediglich 1 bis 2 Monate gültig. Mit seinem privaten ZSK signiert der Administrator einer Domäne alle DNS-Einträge in seiner Zonen-datenbank. Die ZSK-Schlüssel können völlig autonom von den jeweiligen Administratoren einer Domäne verwaltet und erneuert werden. Eine Schlüs-selerneuerung ist damit Domänen-lokal und einfach möglich. Dadurch kann man auch kürzere Schlüssel verwenden und damit die Effizienz der Validie-rungsoperationen deutlich gegenüber langen Schlüsseln verbessern. Da mit den ZSK-Schlüsseln alle DNS-Einträge signiert werden, die bei einem Loo-kup zu validieren sind, ist eine schnelle Validierung von großem Vorteil. Die KSK-Schlüssel werden demgegenüber domänenübergreifend verwendet, um Schlüsselmaterial zu signieren. Eine Schlüsselferneuerung erfordert eine do-mänenübergreifende Zusammenarbeit und ist damit aufwändig. Durch die

DNSsec

Schutzziele

Schlüssel

deutlich längeren KSK Schlüssel kann man die Erneuerungsintervalle ebenfalls deutlich verlängern, um diesem Aufwand Rechnung zu tragen. Da die KSK-Schlüssel lediglich zum Signieren anderer Schlüssel verwendet werden, ist der mit der Validierung verbundene Aufwand durch den langen Schlüssel tolerabel.

Zur Veranschaulichung der Vorgehensweise betrachten wir folgendes Szenario: Gegeben sei die Domäne $.de$ und deren Sub-Domäne $tum.de$. Im Folgenden erläutern wir zunächst, welche Schlüssel zum Signieren welcher Informationen verwendet werden, um die Integrität und Authentizität von DNS-Einträgen prüfbar zu machen. In einem zweiten Schritt erläutern wir dann, wie eine solche Prüfung abläuft und welche Informationen dafür erforderlich sind.

Wir gehen davon aus, dass für die beiden Domänen die beiden Schlüsselpaare ZSK_{de} und KSK_{de} bzw. $ZSK_{tum.de}$ und $KSK_{tum.de}$ bereits generiert worden sind. Wir beschreiben das generelle Schema aus Sicht der Domäne $tum.de$.

Signierte
DNS-Einträge

- Jeder DNS-Eintrag in der Zonendatenbank von $tum.de$ wird vom Administrator der Domäne mit dem privaten Zonenschlüssel $ZSK_{tum.de}$ signiert. Zur Ablage dieser Information wird in DNSSEC ein neuer Datentyp eingeführt, das sind die RRSIG-Einträge.
- Den zugehörigen öffentlichen $ZSK_{tum.de}$ Schlüssel, der für die Verifikation der signierten DNS-Datenbankeinträge der Domäne $tum.de$ benötigt wird, signiert der Administrator von $tum.de$ mit dem privaten $KSK_{tum.de}$ Schlüssel seiner eigenen Domäne. Die Zonenschlüssel werden also selbst signiert.
- Der Administrator stellt die beiden öffentlichen Schlüssel, also die öffentlichen Schlüssel von $ZSK_{tum.de}$ und $KSK_{tum.de}$, die zur Verifikation der Signaturen erforderlich sind, in speziellen DNS-Datenbank-Einträgen zur Verfügung. Dazu führt DNSSEC den Datentyp DNSKEY ein. D.h. die Zonendatenbank enthält zwei spezielle DNSKEY Einträge für die beiden öffentlichen Schlüssel. Zusätzlich wird die Signatur des selbst signierten, öffentlichen Schlüssels $ZSK_{tum.de}$ in einem RRSIG-Eintrag zur Verfügung gestellt.
- Damit bei einer DNS-Anfrage die so signierten DNS-Einträge von Dritten verifiziert werden können, wird ein Nachweis über die Vertrauenswürdigkeit des öffentlichen Schlüssels $KSK_{tum.de}$ benötigt, mit dem die Schlüssel-Signaturen verifiziert werden können. Hierzu kommt nun die hierarchische Struktur des DNS ins Spiel. In unserem Szenario ist $.de$ die übergeordnete Domäne von $tum.de$. Der Administrator von $.de$ signiert mit seinem privaten Zonen-Schlüssel ZSK_{de} den Hashwert

des öffentlichen Schlüssels $KSK_{tum.de}$ der untergeordneten Domäne $tum.de$. Damit bestätigt er die Vertrauenswürdigkeit dieses Schlüssels. Diese Information legt der Administrator der Domäne $.de$ in seiner Zonendatenbank ab. Hierfür wurde ebenfalls ein neuer DNS-Datentyp definiert, das ist der DS-Datentyp, der den Hashwert enthält. Die Signatur des Hashwertes ist wiederum in einem RRSIG-Eintrag abgelegt.

- Damit die Signatur, die mit dem privaten Schlüssel ZSK_{de} generiert wurde, geprüft werden kann, ist der zugehörige öffentliche Schlüssel ZSK_{de} der Domäne $.de$ notwendig. Der Schlüssel ZSK_{de} wurde seinerseits vom Administrator von $.de$ mit dem privaten KSK_{de} Schlüssel der $.de$ -Domäne signiert. Zur Verifikation der Signatur wird somit der öffentliche Schlüssel KSK_{de} benötigt. Dieser muss nun seinerseits vertrauenswürdig sein. Es wiederholt sich also die Situation. Die der Domäne $.de$ übergeordnete Domäne hat die Aufgabe, den öffentlichen Schlüssel KSK_{de} zu signieren. Die Kette endet gemäß der DNS-Hierarchie mit der Root-Zone. Diese muss die Echtheit und Vertrauenswürdigkeit der Schlüssel aller Top-Level Domains bestätigen. Hierzu gehört auch die Domäne $.de$. Der KSK dieser Root-Zone wird von der ICANN verwaltet; er soll alle 2 bis 5 Jahre ausgetauscht werden.

DNS-Einträge

Wie oben bereits beschrieben, werden in DNSSEC neue DNS-Einträge spezifiziert, um die erforderlichen zusätzlichen Informationen zum Nachweis und zur Überprüfung der Integrität und Authentizität von Datenbankeinträgen zur Verfügung zu stellen. Gegeben seien wieder die beiden Domänen $.de$ und $tum.de$. Die Zonendatenbank der Domäne $.de$ enthält folgende neue Einträge:

- Die Zonendatenbank der Domäne $.de$ enthält einen *DS-Eintrag* mit dem Hashwert des öffentlichen Schlüssels $KSK_{tum.de}$ der untergeordneten Domäne $tum.de$.
- Die Zonendatenbank der Domäne $.de$ enthält zu dem DS-Eintrag einen zugehörigen *RRSIG-Eintrag*, der den DS-Eintrag digital mit dem privaten Zonen-Schlüssel ZSK_{de} von $.de$ signiert.
- Die Zonendatenbank der Domäne $.de$ enthält zudem zwei *DNSKEY-Einträge*; je einen für den öffentlichen Schlüssel KSK_{de} und den öffentlichen Zonen-Schlüssel ZSK_{de} .
- Die Zonendatenbank der Domäne $.de$ enthält zu dem DNSKEY-Eintrag von ZSK_{de} den zugehörigen *RRSIG-Eintrag*, der den DNSKEY-Einträge digital mit dem privaten Schlüssel KSK_{de} signiert.

Für die Domäne $tum.de$ wird deren Datenbank ganz analog aufgebaut. Abbildung 14.28 veranschaulicht das skizzierte Szenario.

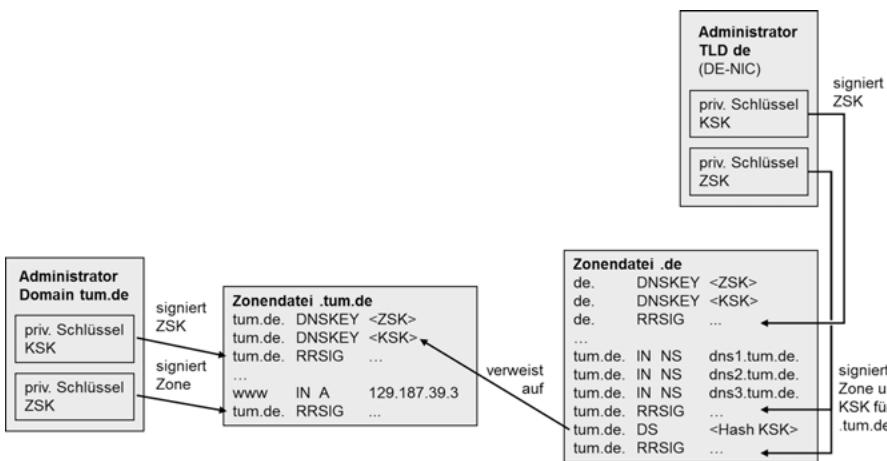


Abbildung 14.28: DNS-Datenbanken mit zusätzlichen DNSSEC Einträgen

14.5.2 DNS-Anfrage unter DNSSEC

Wir greifen das vorherige Szenario auf, um nachfolgend zu erläutern, welche Informationen bei einer DNS-Anfrage unter DNSSEC an den Anfrager geantwortet wird und wie der Anfrager diese Antwortdaten validiert.

Stellt ein Server eine DNS-Anfrage an die Domäne *tum.de*, so erhält er als Antwort neben dem DNS-Eintrag noch dessen Signatur, die in dem zugehörigen RRSIG-Eintrag zu finden ist, sowie die Daten aus den beiden DNSKEY-Einträgen, die die beiden öffentlichen Schlüssel $KSK_{tum.de}$ und $ZSK_{tum.de}$ der Domäne *tum.de* enthalten, sowie den zugehörigen RRSIG-Eintrag, der den öffentlichen Schlüssel $ZSK_{tum.de}$ mit dem privaten $KSK_{tum.de}$ signiert. Eine Verifikation der Authentizität der Antwortdaten erfolgt nun entlang einer Vertrauenskette in mehreren Schritten:

Verifikation von Antworten

- Um die Signatur des erfragten DNS-Eintrags zu prüfen, wird der öffentliche Schlüssel $ZSK_{tum.de}$ benötigt. Dieser ist der Antwortnachricht aus einem der beiden DNSKEY-Einträgen zu entnehmen. Mit diesem öffentlichen Schlüssel $ZSK_{tum.de}$ kann die Signatur der Antwortnachricht geprüft werden, also die Authentizität der Antwort.
- Dazu ist jedoch zuvor die Vertrauenswürdigkeit des öffentlichen Schlüssels $ZSK_{tum.de}$ zu prüfen. Dessen Signatur entnimmt er dem beigefügten RRSIG-Datum und verwendet den zweiten DNSKEY-Eintrag, der den öffentlichen Schlüssel $KSK_{tum.de}$ enthält, um nunmehr die Signatur des öffentlichen Zonenschlüssels, die ja selbst signiert ist, zu prüfen.
- Als nächstes muss also erst einmal die Vertrauenswürdigkeit des öffentlichen Schlüssels $KSK_{tum.de}$ geprüft werden. Dazu muss die Vertrauenskette durchlaufen werden. Der Hashwert des öffentlichen Schlüssels

$KSK_{tum.de}$ ist in der Datenbank der übergeordneten Domäne $.de$ in einem DS-Eintrag enthalten. Die Signatur des Hashwerts ist dem zugehörigen RRSIG-Eintrag zu entnehmen. Diese Daten sind von der Domäne $.de$ abzurufen.

- Zur Prüfung des signierten Hashwertes ist nunmehr der öffentliche Schlüssel ZSK_{de} erforderlich, da die Signatur mit dem zugeordneten privaten Schlüssel ZSK_{de} erstellt worden ist. Der öffentliche Schlüssel ZSK_{de} ist dem DNSKEY-Eintrag der Domäne $.de$ zu entnehmen und dessen Vertrauenswürdigkeit ist zu prüfen. Dies erfordert die nächste Hierarchiestufe.
- Der Ablauf setzt sich fort, bis die Wurzel-Instanz erreicht ist. Wenn alles korrekt validiert werden konnte, ist der Eintrag authentisch und wird verwendet.

Sicherheit

Durch die aufgebaute Vertrauenskette und der vertrauenswürdigen Wurzelinstanz, signieren die hierarchisch übergeordneten Domänen die öffentlichen KSK-Schlüssel der untergeordneten Domänen. Dies ist damit vergleichbar mit einer Zertifizierungshierarchie von Certification Authorities einer PKI (vgl. Kapitel 9.1). Da die DNS-Einträge digital signiert werden, können auch modifizierte Einträge erkannt werden: Authentizität und Integrität der Einträge ist also sicher gestellt. Dies gilt natürlich nur solange die jeweiligen privaten Schlüssel in den verschiedenen Domänen nicht kompromittiert sind. Die Administratoren haben also Sorge zu tragen, dass die privaten Schlüssel sicher verwaltet und dass die Schlüssel in den empfohlenen Zeitintervallen erneuert werden. Wie man an dem skizzierten Ablauf jedoch deutlich sieht, ist der Vorgang sehr aufwändig. Die Komplexität und Vielzahl der Prüfschritte birgt die Gefahr gezielt herbeigeführter DoS Attacken beispielsweise durch eingeschleuste falsche Antwort-Nachrichten.

Nutzung

Die deutsche Registrierungsstelle DeNIC hat am 31.5.2011 DNSsec in den Produktivbetrieb übernommen und DNSsec in der $.de$ -Zone eingeführt. Laut der Statistik unter <http://rick.eng.br/dnssecstat/> sind Stand 1-tes Quartal 2018 deutlich über 90 Prozent aller Top-level Domains mit DNSSEC signiert.

Rollover

Mit DNSSEC wird eine Vertrauenskette aufgebaut, deren Wurzel der Root-KSK der genutzt wird, um den ZSK der DNS-Root-Zone zu signieren. Der Root-Zone ZSK dient zur Validierung der Top-Level Domänen (TLD). Der Root KSK muss in allen DNSSEC-unterstützenden Resolvern eingetragen werden, damit die signierten Einträge entlang der Vertrauenskette validiert werden können. Der Root-Zone ZSK wird von Verisign verwaltet, während der Root-KSK von der Internet Corporation for Assigned Names and Numbers (ICANN) verwaltet wird.

Da der aktuelle Root KSK seit dem erstmaligen signieren der DNS Root Zone im Jahr 2010 genutzt wird, will die ICANN den Schlüssel erneuern. Dieser Rollover war bereits für 2017 vorgesehen, verschiebt sich jetzt jedoch nach jetzigem Planungsstand (erstes Quartal 2018) auf den 11. Oktober 2018, damit ein Schlüsselwechsel mit möglichst geringen Störungen für den Internet-Verkehr abgewickelt werden kann.

In den folgenden Abschnitten werden Sicherheitsprotokolle besprochen, die auf der Anwendungsebene angesiedelt und auf dedizierte Anwendungen zugeschnitten sind. Wir erklären standardisierte Dienste bzw. de facto Standards für wichtige Anwendungsbereiche wie E-Mail und Secure Messaging. Auch sichere, dezentrale Transaktionen, wie sie mit der Blockchain-Technologie ermöglicht werden und beispielsweise in der digitalen Währung Bitcoin eine weit verbreitete Anwendung besitzen, werden erläutert.

Es ist nicht das Ziel einen vollständigen Überblick über alle existierenden Ansätze zu geben. Vielmehr soll anhand der ausgewählten Dienste die generelle Vorgehensweise bei der Anreicherung existierender Dienste um Sicherheitsmaßnahmen verdeutlicht werden.

14.6 Elektronische Mail

Elektronische Mail ist eine der am häufigsten genutzte Anwendung in vernetzten Systemen. Ein Charakteristikum des Dienstes ist es, dass er interoperabel mit den unterschiedlichsten Architekturen und Systemen ist. In Kapitel 3.4.3 haben wir bereits die wesentlichen Sicherheitsprobleme erklärt, die sich bei der Nutzung des Simple Mail Transfer Protocols (SMTP) für einfache ASCII-Mails bzw. durch das Multipurpose Internet Mail Extension Protokoll (MIME) für multimediale Nachrichten ergeben. Die Mails werden unverschlüsselt übertragen und es besteht kein Schutz vor unautorisierten Modifikationen des Nachrichteninhalts oder vor Maskierungsangriffen durch eine unautorisierte Veränderung der Adressierungsdaten. Im Folgenden werden mit S/MIME und PGP Dienste zur Behebung dieser Sicherheitsmängel vorgestellt.

14.6.1 S/MIME

Der S/MIME-Standard (vgl. RFC 3850, 3851) ist eine Sicherheitserweiterung des MIME-Standards (Multipurpose Internet Mail Extension Protocol). Der MIME-Standard definiert Message Header Felder und Content Formate zur Repräsentation von Multimedia Content sowie Transfer Encodings, die dem Schutz des Inhalts vor Veränderungen durch das Mail System

S/MIME

selbst dienen. Ein Message Header Feld beschreibt die MIME-Version, den Content-Type, der die Daten im Nachrichten-Körper spezifiziert, das Content-Transfer-Encoding, die Content-ID sowie die Content-Description, die den Nachrichtenkörper beschreibt. Dies ist nützlich, wenn beispielsweise der Content des Körpers nicht lesbar ist wie z.B. bei Audio Daten. Beispiele für Content-Types sind text/plain, image/jpeg, image/gif, video/mpeg, oder auch multipart/mixed. Beispiele für Transfer Encodings sind 7bit, d.h. kurze Zeile mit ASCII Zeichen, 8bit, d.h. kurze Zeile mit non-ASCII Zeichen, oder auch base64 (radix 64), d.h. drei 8-bit Blöcke in vier 6-bit Blöcke codiert.

Im Folgenden ist ein Beispiel einer MIME-formatierten Mail angegeben, wobei die Text-Marke *boundary* zur Trennung von Objekten dient und durch *multipart/mixed* spezifiziert ist, dass die Mail aus mehreren Objekten besteht.

```
MIME-Version: 1.0
From: Renate Mustermann <mustermann@irgendwo.de>
To: Harry Irgendwo <hi@niergegenwo.com>
Date: Fr, 13 2002 16:15:05 -0700 (PDT)
Subject: A multipart example
Content-Type: multipart/mixed; boundary=boundary-1
This is the preamble area of a multipart message.
Mail readers that understand multipart format
should ignore this preamble.
--boundary-1
Content-type: text/plain; charset=US-ASCII "irgendein Text"
--boundary-1
Content-Type: multipart/parallel; boundary=unique-boundary-2
--unique-boundary-2
Content-Type: audio/basic
Content-Transfer-Encoding: base64
... base64-encoded 8000 Hz single-channel mu-law-format
      Audio Daten ab hier ...
--unique-boundary-2
Content-Type: image/jpeg
Content-Transfer-Encoding: base64 ... base64-encoded
      image Daten ab hier ...
--unique-boundary-2--
```

Mit S/MIME werden die MIME-Mails um folgende Sicherheitsdienste angereichert:

- Verschlüsselung von Nachrichteninhalten (envelop) zusammen mit dem verschlüsselten Nachrichtenschlüssel für die Empfänger,
- Signieren der Daten, indem der Hashwert der Nachricht signiert und mit dem privaten Schlüssel des Absenders verschlüsselt wird. Signatur und Nachrichteninhalt werden base-64 kodiert, so dass nur solche Empfänger

(eMail-Clients), die eine S/MIME Unterstützung haben, die Nachricht sowie die Signatur lesen können.

- Clear-Signed Daten, indem die digitale Signatur wie oben beschrieben erzeugt wird, jedoch nur die Signatur base-64 kodiert übertragen wird. Nicht S/MIME-fähige Empfänger können damit zwar die Nachricht lesen, aber nicht die Signatur verifizieren.
- Signieren und Verschlüsseln der Daten, indem zunächst die digitale Signatur erstellt und dann Signatur und Nachricht verschlüsselt werden.

Alle zur Bearbeitung einer S/MIME-Nachricht benötigten Daten werden in die Mail selbst integriert, so dass sie in einem Bearbeitungsschritt verarbeitbar ist. Man spricht hierbei vom one-pass processing. Abbildung 14.29 veranschaulicht die Einbindung der Sicherheitsdienste in eine MIME-Nachricht.

one-pass
processing

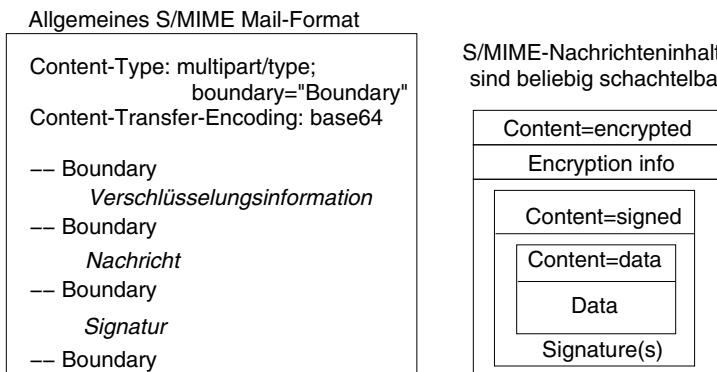


Abbildung 14.29: Allgemeines S/MIME Mail-Format

S/MIME nutzt zur Nachrichtenverschlüsselung ein hybrides Verfahren. Die Nachrichten werden symmetrisch verschlüsselt und mittels eines MACs geschützt sowie unter Nutzung des Public Key Cryptographic Standards (PKCS) (vgl. Seite 346) signiert. Sender und Empfängerseite müssen den DSA zum Signieren unterstützen und ebenfalls RSA anbieten, wobei ein Empfänger in der Lage sein sollte, RSA-Signaturen mit Schlüssellängen zwischen 512 Bits und 1024 Bits zu verifizieren. Zur Verschlüsselung des Verbindungsschlüssels müssen beide Partner das Diffie-Hellman-Verfahren unterstützen und sie sollten auch RSA verwenden können. Ein Sender sollte dabei in der Lage sein, RSA-Schlüssel mit Längen zwischen 512 Bits und 1024 Bits zur Verschlüsselung zu verwenden.

Verfahren

Da in S/MIME die zu verwendenden Verschlüsselungsverfahren nicht festgelegt sind, hat ein sender E-Mail-Agent zwei Entscheidungen zu treffen.

Auswahl von
Verfahren

Zunächst muss er einen symmetrischen Verschlüsselungsalgorithmus auswählen, der vom Empfänger unterstützt werden kann. Falls der Empfänger nur ein schwaches Verschlüsselungsverfahren unterstützt, muss der Sender-Agent entscheiden, ob er die Daten derart schwach verschlüsselt übertragen lassen möchte. Damit Sender und Empfänger die gleichen Verfahren verwenden, kann der Sender die ihm zur Verfügung stehenden Verfahren ankündigen. Empfänger können diese Information speichern, so dass sie sie beim Versenden einer Nachricht, die zurück an diesen Absender gesendet werden soll, direkt verwenden können. Hat ein Sender-Agent keine Kenntnisse über die unterstützten Verfahren seiner Empfängerseite, so wird empfohlen Tripel-DES zu verwenden, wenn man bereit ist, das Risiko in Kauf zu nehmen, dass die Nachricht vom Empfänger unter Umständen nicht entschlüsselt werden kann.

PKCS#7

S/MIME setzt hauptsächlich auf dem PKCS#7-Standard (siehe RFC2313) auf, der eine Syntax für verschlüsselte und signierte Nachrichten festlegt. Eine PKCS#7-Nachricht besteht aus einer Folge von Inhaltstypen, den Content Types. Der Standard unterscheidet an Inhaltstypen unter anderem zwischen Daten, signierten Daten, verschlüsselten und gehaschten Daten. Um PKCS#7 zu unterstützen, definiert S/MIME einen zusätzlichen Inhaltstyp, nämlich application/x-pkcs7-mime. Dieser Typ gibt an, dass eine MIME-Nachricht um Verschlüsselungen und Signaturen gemäß des PKCS#7-Standards angereichert wurde.

Die eingeführten Konzepte werden am Beispiel einer signierten E-Mail im Folgenden noch einmal zusammengefasst.

Beispiel 14.7 (S/MIME Signed Message Format)

Das Nachrichtenformat einer signierten S/MIME E-Mail ist in den RFCs 1847 und 2311 beschrieben und es wird, wie erklärt, der PKCS #7-Standard verwendet.

```
Content-Type: multipart/signed;
  protocol="application/pkcs7-signature";
  micalg=sha1; boundary=boundary1
--boundary1
Content-Type: text/plain
Das ist ein zu signierender Text.
--boundary1
Content-Type: application/pkcs7-signature; name=smime.p7s
Content-Transfer-Encoding: base64
Content-Disposition: attachment; filename=smime.p7s
ghyHhHUujhJhjH77n8HHGTrfvbnj756tbB9HG4VQpfyF467GhIGfH
4VQpfyF467GhIGfHfYT6jH77n8HHGhyHhHUujhJh756tbTrfv=
--boundary1
```

In dem ersten MIME-Objekt (Content-Type: multipart/signed) werden die Informationen eingetragen, die benötigt werden, um die Signatur zu verifizieren. Die Signatur gilt für den im zweiten MIME-Objekt angegebenen Text (Content-Type: text/plain). Die elektronische Signatur des Textes ist eine in base64 codierte PKCS#7 Datenstruktur des Typs *application/pkcs7-signature*. Das Objekt (Content-Type: application/pkcs7-signature) beschreibt diese Signatur. Die PKCS#7 Datenstruktur *application/pkcs7-signature* enthält ein ASN.1 SignedData-Objekt mit folgenden Informationen:

- version,
- digestAlgorithms: Identifikatoren von Hash-Algorithmen (OIDs).
- contentInfo: bei multipart/signed S/MIME Format leeres Feld, da Content in einem separaten MIME-Teil übertragen wird, sonst steht hier die signierte Nachricht.
- certificates: optional, enthält die Benutzer-Zertifikate von allen Signierern und ggf. von deren CAs bis zur Wurzel-CA
- crls: optional, Angabe von Certificate Revocation Lists (CRLs)
- signerInfos: Menge von SignerInfo Objekten, enthält jeweils u.a.: CA-Name des Zertifikatausstellers des/der Signierers, Hash-Algorithmen, verschlüsselten Hash der Nachricht (das ist hier das SignerObject), Verschlüsselungs-Algorithmus etc.

Mit diesen Informationen ist der Empfänger in der Lage, die Signatur des Mail-Textes zu verifizieren (falls die Zertifikate vorhanden und gültig sind).



Schlüsselzertifizierung

Auch S/MIME basiert auf einer X.509 Zertifikatinfrastruktur. Es ist möglich, dass auch kleine Arbeitsgruppen eine eigenständige Zertifizierungseinheit einrichten und betreiben können, ohne ein Teil einer globalen Zertifizierungshierarchie sein zu müssen.

Um sich ein X.509.v3 Zertifikat bei einer Zertifizierungsinstanz (CA) ausstellen zu lassen, sendet der S/MIME-Agent des Benutzers eine Nachricht mit dem Content-Type *application/pkcs10* als Anfrage an die CA. Die Anfrage enthält den Identifikator für das verwendete Signierverfahren und einen Anfrageblock. Dieser ist mit dem privaten Schlüssel des Absenders signiert und enthält dessen öffentlichen Schlüssel als Bitstring, sowie den Namen des Besitzers dieses Schlüssels. Die CA sendet ein Zertifikat und gegebenenfalls eine Zertifikatrückrufliste (CRL, Certificate Revocation List) zurück.

VeriSign

Zur Verifikation signierter Mails muss ein S/MIME Nutzer eine Menge von vertrauenswürdigen CA-Schlüsseln konfigurieren, d.h. er konfiguriert, welchen Zertifikaten von welchen CAs er vertraut. In der Regel sind sehr viele (zu viele!) CA-Schlüssel in heutigen Browsern als vertrauenswürdig voreingestellt. Einer der am weitesten verbreiteten CA-Dienste, der für S/MIME verwendet wird, ist VeriSign. VeriSign stellt ein X.509 Zertifikat aus, das Digital ID genannt wird. VeriSign unterscheidet drei unterschiedliche Vertrauensstufen für Zertifikate. Zertifikatanfragen nach Zertifikaten der Klasse 1 und 2 werden direkt online bearbeitet, während ein Zertifikat der Klasse 3 einen höheren Grad an Identitätsüberprüfung erfordert. Hierbei wird gefordert, dass der Benutzer persönlich bei der CA erscheint oder eine notarielle Beglaubigung vorlegt. Bei einer Klasse-1 Anfrage bestätigt VeriSign lediglich die Eindeutigkeit der E-Mailadresse des Anfragers und stellt eine Digitale ID, die diese Mailadresse enthält, aus. Bei einer Klasse-2 Anfrage werden automatisch die in der Anfrage übermittelten Daten gegen Einträge in einer Kundendatenbank geprüft und eine ID ausgestellt. Zusätzlich wird an die postalische Adresse des Anfragers eine Nachricht gesendet, die ihm mitteilt, dass eine Digitale ID auf seinen Namen ausgestellt wurde. Aufgrund der schwachen Identitätsprüfungen der Klassen 1 und 2 sind derartige Zertifikate sicherlich nicht geeignet, vertrauenswürdige vertrauliche Kommunikationsbeziehungen z.B. zum Internet-Banking oder zur Abwicklung elektronischer Geschäftsprozesse zu etablieren.

inoffizieller Standard

S/MIME ist auf dem besten Weg der zukünftige, im breiten Umfang auch akzeptierte E-Mail-Verschlüsselungsstandard zu werden. Dies ist darauf zurückzuführen, dass S/MIME sich an verbreitete Verschlüsselungsstandards anlehnt, etablierte Mail-Infrastrukturen nutzt und die Fähigkeit besitzt, neben einfachen ASCII-Texten auch beliebigen Binärkode zu verschlüsseln, zu authentifizieren und zu signieren. S/MIME wird von vielen kommerziellen und Open Source Anwendungen unterstützt.

14.6.2 Pretty Good Privacy (PGP)

PGP

Das Softwarepaket Pretty Good Privacy [68] wurde ursprünglich 1991 von Philip Zimmermann entwickelt. PGP steht mittlerweile als Public Domain Software¹⁷ und auch in kommerziellen Versionen für unterschiedliche Plattformen u.a. Unix- und Windows-Betriebssystemfamilien, aber auch MacOS zur Verfügung.

Dienste

PGP bietet, wie auch schon S/MIME, Dienste zur vertraulichen Mail-Übermittlung, zur Gewährleistung der Authentizität und Integrität einer Mail sowie zum Nachweis des Nachrichtenursprungs anhand digitaler Signaturen. PGP kann auch zur Dateiverschlüsselung eingesetzt werden und bietet wei-

¹⁷ Die neueste Version ist z.B. unter <http://www.pgpi.org/download> verfügbar.

tere Dienste, wie zum Beispiel ein Schlüssel-Recovery und die Möglichkeit, Dokumente, wie Faxe, zu signieren. PGP stellt lediglich die kryptografischen Dienste zur Verfügung, während der eigentliche Versand über das Netzwerk von den Standardprogrammen wie `sendmail` vorgenommen werden muss. Damit nutzt auch PGP die im Internet bereits vorhandenen Mail-Infrastrukturen.

PGP setzt RSA sowie seit der PGP Version 5.0 auch das Diffie-Hellman-Verfahren zum Austausch symmetrischer Schlüssel ein, und verwendet zur Mailverschlüsselung eine symmetrische Blockchiffre. Die Datenintegrität und Authentizität wird durch die Verwendung von Hashfunktionen bzw. MACs sichergestellt. Zum Signieren werden asymmetrische Verfahren, wie u.a. RSA, eingesetzt. Der Benutzer kann beim Erzeugen seines RSA-Schlüsselpaares die gewünschte Schlüssellänge angeben.

Nachrichten oder Dateien können auch komprimiert versandt bzw. gespeichert werden. PGP verwendet hierfür das ZIP-Verfahren. Um eine Interoperation mit E-Mailsystemen zu unterstützen, die nur ASCII-Texte verarbeiten können, ermöglicht PGP eine Radix-64 Konvertierung verschlüsselter Nachrichten in ASCII-Code.

Eine PGP-Nachricht besteht aus maximal drei Komponenten: der Nachrichtenkomponente, der Signatur- und der Verschlüsselungskomponente. Das Format ist in Abbildung 14.30 für eine Nachricht, die von einem Absender A zu einem Empfänger B verschickt werden soll, angegeben. Die Nachrichtenkomponente enthält die Daten, die übermittelt oder gespeichert werden sollen. Die Signaturkomponente enthält einen Zeitstempel mit dem Zeitpunkt der Signaturerstellung, sowie den verschlüsselten 160-Bit Hashwert und die Schlüssel-ID. Die Verschlüsselungskomponente enthält den verschlüsselten Verbindungsschlüssel sowie die Schlüssel-ID des verwendeten Empfängerschlüssels.

Soll eine Datei M vertraulich übermittelt werden, so generiert PGP einen symmetrischen Schlüssel K und verschlüsselt M unter Nutzung von K mit dem gewählten symmetrischen Algorithmus, $C1 = E_{symm}(M, K)$. Der Schlüssel K ist nur für diese Kommunikation gültig. Der symmetrische Schlüssel K wird anschließend mit dem öffentlichen Schlüssel K_E^B des Empfängers B unter Verwendung des RSA-Algorithmus verschlüsselt, $C2 = E_{asymm}(K, K_E^B)$, und $C1$ zusammen mit $C2$ werden an den Empfänger übermittelt. Beim Erhalt der Mail führt die PGP-Software des Empfängers die entsprechenden Entschlüsselungsschritte durch. Zur Entschlüsselung wird der private Schlüssel K_D^B des Empfängers benötigt, der verschlüsselt abgespeichert ist, so dass B sich zunächst authentifizieren und seinen privaten Schlüssel entschlüsseln lassen muss (s.u.), bevor er den Klartext der Nachricht M erhalten kann.

Verfahren

Format

Vertraulichkeit

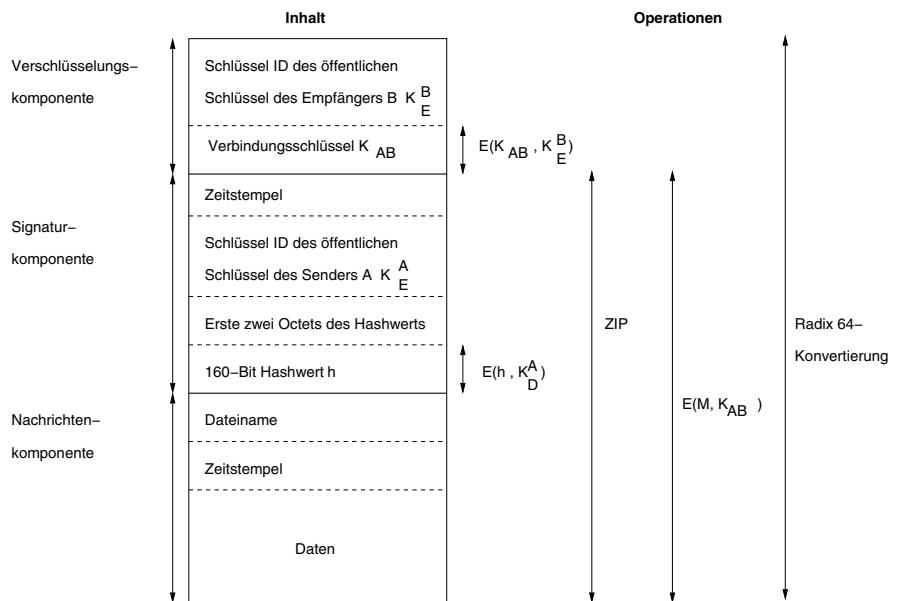


Abbildung 14.30: PGP-Nachrichtenformat

Signatur

Eine Nachricht wird digital signiert, indem zunächst ein Hashwert h über die Nachricht M sowie des Erzeugungszeitpunkts T der Signatur erzeugt wird, also $h = H(M \mid T)$. Durch die Einbindung des Zeitstempels T sind Wiedereinspielungen erkennbar. Der Absender signiert den Hashwert h unter Einsatz seines privaten Schlüssels. Die Signatur wird in der Regel an die Nachricht angehängt und zusammen mit dieser an den Empfänger übermittelt. PGP unterstützt jedoch auch die Möglichkeit, Signaturen entkoppelt von der Nachricht zu speichern bzw. zu übermitteln. Dies kann man zum Beispiel verwenden, um eine getrennte Logdatei aller Signaturen von Dateien, die man empfangen oder gesendet hat, anzulegen. Mit einer entkoppelten Signatur eines ausführbaren Programms kann man beispielsweise das Programm auf die Infektion mit einem Virus überprüfen. Eine entkoppelte Signatur kann man ferner dann verwenden, wenn das Dokument von verschiedenen Parteien unabhängig voneinander signiert werden muss. Anstelle von ineinander geschachtelten Signaturen, die bei einer Kopplung notwendig wären, können die einzelnen Signaturen unabhängig voneinander auf das Originaldokument angewendet werden.

Digitale Signatur und Nachrichtenverschlüsselung können auch zusammen auf eine Nachricht angewandt werden. Dazu wird zunächst, wie oben beschrieben, der Hashwert h der Nachricht M signiert und an die Nachricht angehängt. Im nächsten Schritt erfolgt dann die Verschlüsselung der so konstruierten Nachricht mit einem für diesen Transfer erzeugten symmetrischen

Schlüssel K . Im abschließenden Schritt wird K , wie beschrieben, mit dem öffentlichen Empfängerschlüssel verschlüsselt.

Schlüsselmanagement

Wie aus der voran stehenden Beschreibung hervorgeht, müssen unter PGP unterschiedliche Klassen von Schlüssel verwaltet werden. Im Einzelnen sind dies die symmetrischen Nachrichtenschlüssel, die asymmetrischen Schlüsselpaare sowie Schlüssel, um die privaten Schlüssel vertraulich zu speichern.

Um Dateien oder E-Mails verschlüsseln zu können, muss ein Anwender ein asymmetrisches Schlüsselpaar besitzen, das er sich unter Nutzung einer dafür vorgesehenen PGP-Bibliotheksfunktion zu erzeugen hat. PGP legt die erzeugten Schlüssel automatisch in einer eigens dafür angelegten Datei, dem so genannten Private-Key-Ring ab. PGP unterstützt die Möglichkeit, dass ein Benutzer mehrere Schlüsselpaare besitzen kann. Jedes dieser Paare wird im Schlüsselring verwaltet. Das ist zum Beispiel nützlich, wenn man für unterschiedliche Gruppen von Kommunikationspartnern jeweils eigene Schlüssel verwenden möchte, oder wenn Schlüssel erneuert werden sollen.

Zum Schutz des privaten Schlüssels wird dieser verschlüsselt gespeichert. Hierzu muss der Benutzer sein so genanntes Mantra angeben. Es handelt sich hierbei um eine Passphrase, also um ein erweitertes Passwort. Mittels eines Hashverfahrens wird der Hashwert des Mantras berechnet und als Schlüssel für die Verschlüsselung des privaten Schlüssels verwendet. Durch die Eingabe des Mantras authentifiziert sich der Benutzer als Eigentümer des privaten Schlüssels. PGP berechnet aus der Passphrase den Hashwert, entschlüsselt damit den privaten Schlüssel und entschlüsselt abschließend die verschlüsselte Mail oder Datei. Symmetrische Verschlüsselungsschlüssel werden von PGP pseudozufällig generiert. Der dazu verwendete Pseudozufallszahlengenerator PRNG erzeugt den initialen Zufallswert unter Einbeziehung der Verzögerungszeiten, die sich beim Messen der Tastatureingaben des Benutzers ergeben.

Ein Benutzer kann mehrere Schlüsselpaare besitzen, weshalb der Absender einer signierten und/oder verschlüsselten Nachricht dem Empfänger mitteilen muss, welcher seiner öffentlichen Schlüssel verwendet wurde. Da ein öffentlicher RSA Schlüssel mehrere hundert Dezimalstellen lang sein kann, ist das Anhängen des vollständigen Schlüssels an die Nachricht sehr aufwändig. Zur Reduktion dieses Aufwandes verwendet PGP anstelle des kompletten Schlüssels eine 64-Bit ID, die diesen eindeutig identifiziert. Die ID entspricht den 64 niedrigwertigsten Bits des öffentlichen Schlüssels. D.h. sei K_E^B ein öffentlicher Schlüssel des Mailempfängers B. Der zugehörige Schlüsselidentifikator ergibt sich durch $ID = K_E^B \bmod 2^{64}$. Das analoge Schema wird auch zur Identifikation der Signaturschlüssel von Absendern verwendet.

Private-Key-Ring

symmetrische
Schlüssel

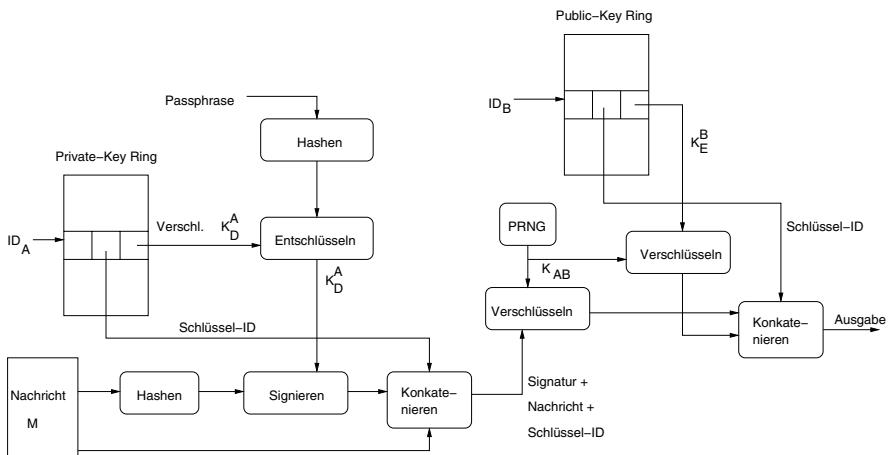


Abbildung 14.31: Verschlüsselte und signierte PGP-Nachricht von A nach B

Damit ist klar, welche Informationen in dem Private-Key-Ring eines Benutzers verwaltet werden. Jeder Eintrag in dieser Datei beschreibt ein asymmetrisches Schlüsselpaar des Benutzers. Der Eintrag enthält den Zeitpunkt der Schlüsselerzeugung, die Schlüssel-ID, den öffentlichen Schlüssel, den verschlüsselten privaten Schlüssel sowie die Benutzer-ID, die typischerweise die E-Mailadresse ist.

Public-Key-Ring

Die öffentlichen Schlüssel von Kommunikationspartnern werden ebenfalls in einer Datei, dem so genannten Public-Key-Ring gespeichert. Jeder Eintrag enthält den Zeitpunkt, wann der Schlüssel in der Datei eingetragen wurde, die Schlüssel-ID, den öffentlichen Schlüssel sowie die ID des Schlüsselbesitzers.

Abbildung 14.31 fasst die Schritte, die beim Versand einer verschlüsselten und signierten PGP-Nachricht zu durchlaufen sind, noch einmal zusammen. Durch Eingabe der Passphrase erhält der Benutzer A Zugriff auf seinen verschlüsselt gespeicherten privaten Schlüssel K_D^A . Die zu versendende Nachricht M wird gehasht und der Hashwert wird mit dem privaten Schlüssel K_D^A des Absenders signiert. Signatur, Nachricht und Schlüssel-ID werden konkateniert und mit dem neu erzeugten Verbindungsschlüssel K_{AB} verschlüsselt. Der Schlüssel K_{AB} wird seinerseits mit dem öffentlichen Schlüssel K_E^B des Empfängers verschlüsselt und zusammen mit dessen Schlüssel-ID mit der verschlüsselten und signierten Nachricht konkateniert.

Auf der Empfängerseite erfolgt die Entschlüsselung und Integritätsprüfung wie in Abbildung 14.32 skizziert. Mit der in der Nachricht enthaltenen Schlüssel-ID wählt der Empfänger aus seinem privaten Schlüssel-Ring den

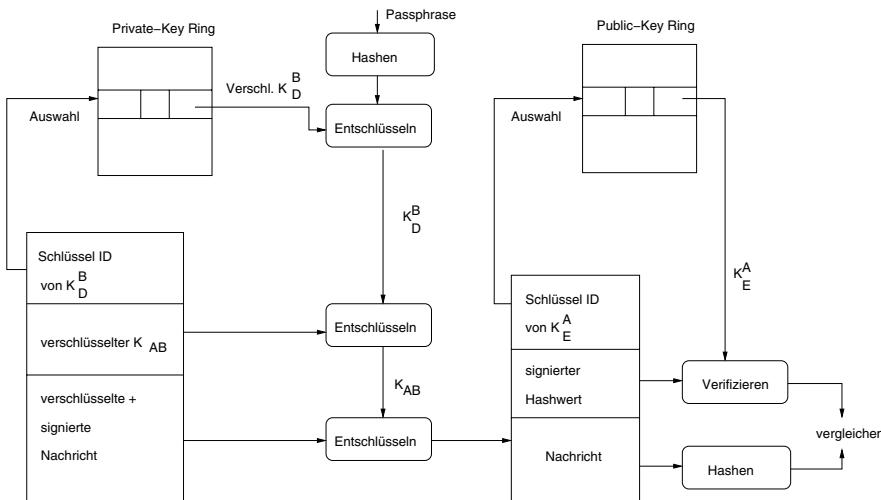


Abbildung 14.32: Empfang einer PGP-verschlüsselten und signierten Nachricht

benötigten privaten Schlüssel aus. Durch die Eingabe seiner Passphrase ermöglicht er dessen Entschlüsselung, so dass damit PGP den in der Nachricht enthaltenen Verbindungsschlüssel K_{AB} und damit schließlich die Nachricht selber entschlüsseln kann. Zur Überprüfung der Integrität und Authentizität der Nachricht wählt der Empfänger den öffentlichen Schlüssel K_E^A des Absenders aus dem öffentlichen Schlüsselring aus. Mit diesem kann der in der Nachricht mitgelieferte, signierte Hashwert h entschlüsselt werden. Die PGP-Software des Empfängers überprüft abschließend die Integrität der empfangenen Nachricht, indem diese gehasht und mit dem empfangenen Wert h verglichen wird.

Web of Trust

Zu klären bleibt noch, auf welche Weise unter PGP die öffentlichen Schlüssel der Kommunikationspartner authentifiziert werden. Ein wesentlicher Unterschied zu S/MIME besteht darin, dass die öffentlichen Schlüssel nicht unter Nutzung einer Zertifikatinfrastruktur verteilt werden müssen, sondern eine Bekanntgabe nach dem Prinzip des *Web of Trust* erfolgt. Dieses Vertrauensnetz beschreibt transitive Vertrauensbeziehungen. Um abgestufte Vertrauenslevels festlegen zu können, wird mit jeder Beschreibung eines öffentlichen Schlüssels im Public-Key Ring ein Eintrag verbunden, das Key Legitimation Field (KLF), das den Grad des Vertrauens beschreibt, das der Benutzer in diesen öffentlichen Schlüssel eines anderen besitzt. Ferner wird jeder Schlüsselbeschreibung eine Menge von Signaturen zugeordnet. Jede dieser Signaturen zertifiziert den öffentlichen Schlüssel und besitzt ihrerseits ein Attribut, das charakterisiert, bis zu welchem Grad der Benutzer

Vertrauens-
beziehungen

demjenigen vertraut, der die Signatur erstellt hat. Das Key Legitimation Field wird als gewichtete Summe der Attribute der Signaturen für den betreffenden öffentlichen Schlüssel berechnet, wobei die Gewichtungen durch den Benutzer konfigurierbar sind. Schließlich wird in der Beschreibung für den öffentlichen Schlüssel auch festgehalten, welchen Grad an Vertrauen der Benutzer in den Besitzer des öffentlichen Schlüssels hat. Beim Eintragen eines neuen Schlüssels in den Ring fragt PGP den Nutzer nach dessen Vertrauensschätzung. Die Skala reicht von (1) der Benutzer ist unbekannt, über (2) nicht vertrauenswürdig (usually not trusted to sign), (3) geringfügig vertrauenswürdig (usually trusted to sign), (4) vollständig vertrauenswürdig (always trusted to sign) bis hin zu (5) ultimativ vertrauenswürdig (eigener Schlüssel). Der Algorithmus zur Bestimmung des Trust-Levels läuft im Groben wie folgt ab.

- Mindestens eine Signatur muss als *ultimate trusted* eingestuft sein, das KLF-Feld des zugehörigen Schlüssels besitzt den Wert 1, KLF = 1
- Berechnung der gewichteten Summe:
 - Always Trusted Signaturen haben ein Gewicht von $1/X$,
 - Usually Trusted Signatures haben ein Gewicht von $1/Y$, wobei X, Y Benutzer-konfigurierbare Parameter sind.

Als Beispiel betrachten wir eine Belegung der Werte von X und Y mit: X=2, Y=4. Ein vollständiges Vertrauen in einen Schlüssel erfordert eine der folgenden Konstellationen:

- Einen ultimately Trusted Schlüssel, oder
- zwei always Trusted Schlüssel, oder
- einen always Trusted und zwei usually Trusted Schlüssel, oder
- vier usually Trusted Schlüssel.

Um den eigenen öffentlichen Schlüssel auch anderen Interneteilnehmern zur Verfügung zu stellen, empfiehlt sich dessen Verteilung über einen Schlüssel-Server, der allgemein zugänglich ist. Der Schlüssel kann aber auch als Anhang an E-Mails verschickt werden oder, wie der Name Web of Trust besagt, von vertrauenswürdigen, aber nicht zertifizierten Dritten weitergegeben werden.

Fazit

Bei Verwendung einer starken symmetrischen Verschlüsselung wie AES-128 Bit gilt PGP beim heutigen Stand der Technik als sicher gegen Brute-force-Attacken, so dass ein Angreifer auf anderem Weg versuchen wird, den Schlüssel zu knacken. Da der symmetrische Schlüssel mit dem öffentlichen RSA-Schlüssel des Empfängers einer E-Mail verschlüsselt wird,

also mit dem zugehörigen privaten Schlüssel wieder herstellbar ist, sind an die privaten RSA-Schlüssel und deren Verwaltung sehr hohe Sicherheitsanforderungen zu stellen. Wählt ein Benutzer bei der Erzeugung seiner RSA-Schlüssel einen großen Modul (> 1024 Bit), so gilt dies beim heutigen Stand der Technik als ausreichend, um die zurzeit schnellsten Faktorisierungsangriffe mittels des allgemeinen Zahlensiebs abzuwehren. Es sei hier aber noch einmal darauf hingewiesen (vgl. auch Abschnitt 7.6), dass beim Entdecken eines schnelleren Faktorisierungsverfahrens diese Sicherheit in Frage gestellt ist.

Ein erfolgversprechenderer Ansatzpunkt für einen Angreifer ist sicherlich eine Attacke auf den privaten Schlüssel, der in der Datei `secret.pgp` gespeichert ist. Zur Ver- und Entschlüsselung liegt der zu verwendende private Schlüssel im Klartext im Speicher, bzw. er liegt verschlüsselt auf einem Hintergrundspeichermedium. In Mehrbenutzersystemen, in Systemen mit einem allmächtigen Superuser oder auch in Systemen, die sich über die Anbindung an das Internet bedrohungen durch externe Eindringlinge aussetzen, sind die gespeicherten Schlüssel Spähangriffen ausgesetzt. Zu deren Abwehr bietet sich der Einsatz von Smartcards als Speicher- und ggf. auch als Verschlüsselungsmedium an.

Last but not least kann ein Angreifer natürlich auch versuchen, die Passphrase, mit der der Eigentümer seinen privaten Schlüssel schützt, aufzudecken. Hierbei treten die gleichen Probleme auf, die wir bereits bei der passwortbasierten Authentifikation in Kapitel 10.2.1 ausführlich diskutiert haben. Die Passphrase kann unter anderem durch das unbemerkte Belauschen der Passphasen-Eingabe beispielsweise über so genannte Keylogger-Programme, die die Tastatureingaben aufzeichnen, oder ganz einfach durch das Beobachten des Eingabevorganges aufgedeckt werden. Durch umsichtiges Verhalten der Benutzer sowie eine gute Systemadministration lassen sich entsprechende Angriffe begrenzen.

Abschließend sei angemerkt, dass die vorgestellten Protokolle zum Absichern von E-Mails einen wesentlichen Fortschritt im Hinblick auf die sichere Kommunikation bedeuten. Beim alltäglichen Einsatz dieser Technologie in einem Unternehmen, der öffentlichen Verwaltung etc. kommt es jedoch häufig zu Problemen. Wesentliche Managementprobleme, die sich ergeben, betreffen das Zusammenspiel zwischen der zentralen Firewall des Unternehmens, die bei einer verschlüsselten Mail keine Viren-Filterung oder Content-Filterung mehr vornehmen kann, so dass die unternehmerische Sicherheitsstrategie festlegen muss, wie mit verschlüsselten Mails im Unternehmen umzugehen ist. Ein weiterer Problembeispiel ergibt sich daraus, dass im betrieblichen Sekretariats- und Sachbearbeitungsbereich Vertretungsregeln im Krankheits- und Urlaubsfall von Mitarbeitern festgelegt sind. Häufig

privater Schlüssel

Passphrase

Praxisprobleme

tritt hierbei der Fall auf, dass die eingehenden E-Mails von den Vertretungen nicht bearbeitet werden können, da die benötigten Schlüssel nicht ausgetauscht wurden.

virtuelle Poststelle

Pragmatische Ansätze, die zur Lösung dieser Probleme verfolgt werden, bestehen darin, eine zentrale, virtuelle Poststelle einzuführen. Wie bei einer realen Poststelle laufen hier die elektronischen Mails ein, werden entschlüsselt, geprüft, ggf. auch gefiltert und wenn alles in Ordnung ist, an den Empfänger weitergeleitet. Ob dies unverschlüsselt oder erneut verschlüsselt zu erfolgen hat, ist in der Unternehmens-Policy zu regeln. Schließlich ist auch noch auf das Problem der mangelhaften Interoperabilität von Mail-Verschlüsselungsprotokollen wie z.B. zwischen PGP und S/MIME hinzuweisen. Ein reibungsloser E-Mailverkehr würde eigentlich erfordern, dass Benutzer verschiedene E-Mailclients mit den jeweils benötigten Schlüsselmanagementkomponenten konfiguriert haben.

14.7 Signal-Protokoll für Messaging-Dienste

Signal-Protokoll

Das 2013 Signal-Protokoll (früher als Axolotl-Protokoll bekannt) ermöglicht eine Ende-zu-Ende-Verschlüsselung in Instant-Messaging-Diensten und wird in vielen populären Messengern, wie WhatsApp, Facebook Messenger oder Signal¹⁸, genutzt. WhatsApp nutzt das Signaling-Protokoll, um Nachrichten-Schlüssel für die Ende-zu-Ende-Verschlüsselung auszutauschen. Für die eigentliche Verschlüsselung verwendet WhatsApp den AES256 im CBC-Modus sowie einen HMAC-SHA256 für die Nachrichtenauthentizität.

Protokolle

Das Signal-Protokoll kombiniert zwei verschiedene Protokolle, um Backward- und Forward-Secrecy zu garantieren. Zur wechselseitigen Authentisierung und erstmaligen Schlüsselvereinbarung wird das X3DH-Protokoll (vgl. Abschnitt 14.7.1) genutzt. Damit wird ein gemeinsames Geheimnis *SK* zwischen zwei Partnern berechnet, wobei zum Zeitpunkt der Schlüsselvereinbarung nur der sendewillige Partner A online sein muss. Das Signal-Protokoll generiert für jede zu versendende Nachricht einen eigenen Nachrichtenschlüssel. Hierfür verwendet es das Double-Ratchet-Protokoll (vgl. Abschnitt 14.7.2), das eine Kombination aus zwei Ratcheting-Protokollen¹⁹ ist und auf dem gemeinsamen Schlüssel *SK* basiert. Neben einer Ende-zu-Ende authentisierten Kommunikation ermöglicht das Signal-Protokoll auch die Aufdeckung von Wiedereinspielungen, das Erkennen geänderter Nachrichtenreihenfolgen sowie das Erkennen, dass Nachrichten gelöscht wurden.

¹⁸ vgl. <https://whispersystems.org/>

¹⁹ engl. für Ratsche, engl. to ratchet, etwas schrittweise verstärken.

Das Signal-Protokoll verwendet folgende kryptografischen Primitive:

Kryptoprimitive

- Es wird das Elliptic-Curve-Diffie-Hellman-Verfahren (ECDH) mit der Kurve Curve25519 verwendet.
- Die Authentizität der Nachrichten erfolgt über einen Keyed-Hash Message Authentication Code (HMAC) auf der Basis von SHA-256.
- Die Messenger-Nachrichten werden symmetrisch verschlüsselt, wofür der Advanced Encryption Standard (AES) im AEAD-Modus genutzt wird.

Im Folgenden stellen wir die Bestandteile des Signal-Protokolls, also das X3DH- sowie das Double-Ratchet-Protokoll detaillierter vor.

14.7.1 Extended Triple Diffie-Hellman (X3DH)

Das Extended Triple Diffie-Hellman (X3DH) Protokoll [113] ist eine Variante des ECDH-Protokolls zur wechselseitige Authentisierung zweier Partner A und B und zur Vereinbarung eines gemeinsamen, geheimen 32-Byte Schlüssels SK zwischen diesen. Zusätzlich wird noch ein Server S benötigt, an den aber nur minimale Anforderungen in Bezug auf dessen Vertrauenswürdigkeit gestellt werden. Der Server muss keine Trusted Third Party sein.

X3DH

Das Besondere des X3DH-Protokolls ist, dass es asynchron abgewickelt werden kann, das heißt, dass der Empfänger einer Nachricht zum Zeitpunkt der Schlüsselvereinbarung *offline* sein kann. Der Empfänger muss jedoch vorab auf dem Server S öffentliche Informationen in Form von Public-Keys abgelegt haben, so dass potentielle Kommunikationspartner auf der Basis dieser verfügbaren Information eine Schlüsselvereinbarung mit dem Empfänger durchführen können, ohne dass der Empfänger hierfür direkt interagieren muss.

asynchrone
Verarbeitung

Das X3DH-Protokoll wird vom Signal-Protokoll verwendet, um einmalig eine Authentisierung und einen Schlüsselaustausch zwischen den beteiligten Partnern A und B durchzuführen. Der erstellte Schlüssel SK wird dann als Basis im Double-Ratchet-Protokoll weiterverwendet.

Zunächst führen wir die wichtigsten öffentlichen Schlüsseltypen, die in dem Protokoll genutzt werden, kurz ein. Dafür wird im Wesentlichen die Notation aus dem Originalpapier[113] verwendet.

Public Key	Bedeutung
IK_A	Identity Key von A: Langlebige Identität
EK_A	Ephemeral Key von A: Flüchtiger DH Key für einen Protokollablauf
IK_B	Identity Key von B: Langlebige Identität
SPK_B	Signierter Prekey von B: Kurzlebiger, signierter Schlüssel. Dieser signierte Schlüssel wird regelmäßig erneuert, z. B. wöchentlich
OPK_B	One-Time Prekey von B: Einmal nutzbarer DH-Key für einen Protokollablauf

Protokollablauf

Schritt 1 Vorbereitung durch den Partner B

$B \rightarrow \text{Server S}: SPK_B, \{OPK_B^1, \dots, OPK_B^n\}, IK_B, sig.$

In einem initialen Schritt veröffentlicht der Partner B seine Prekeys, also sowohl den signierten, kurzlebigen Schlüssel SPK_B als auch eine Menge von einmal nutzbaren Prekeys $\{OPK_B^1, \dots, OPK_B^n\}$, sowie seinen Identity-Key IK_B auf dem Server S. Zusätzlich legt B auf dem Server auch die Signatur des signierten Schlüssels ab, $sig = XEdDSA(SPK_B, IK_B^{priv})$, wobei IK_B^{priv} der private Identity Schlüssel von B ist.

Schritt 2 Initieren einer Schlüsselvereinbarung durch den Partner A:

Server S \rightarrow A: $SPK_B, \{OPK_B^i, \dots, OPK_B^j\}, IK_B, sig.$

Um eine X3DH-Schlüsselvereinbarung mit dem Partner B zu initiieren, lädt A ein Bündel von Prekeys sowie die Signatur des signierten Keys von B, $IK_B, SPK_B, \{OPK_B^i, \dots, OPK_B^j\}, sig$, vom Server S. Der Server löscht diejenigen One-Time Prekeys, die von A geladen werden, also $\{OPK_B^i, \dots, OPK_B^j\}$. Sollten bei der Anfrage von A keine One-Time Prekeys von B mehr beim Server vorhanden sein, wird das Protokoll nicht abgebrochen, sondern ohne solche Einmal-Schlüssel fortgesetzt. Der Server kann B auffordern, weitere One-Time Prekeys zu generieren und hochzuladen.

Schritt 3 Signaturvalidierung und Berechnung des gemeinsamen, geheimen Schlüssels SK durch Partner A:

A prüft die Signatur sig und bricht das Protokoll ab, wenn ein Fehler auftritt. Ist die Signatur korrekt, generiert A ein flüchtiges (ephemeral) Schlüsselpaar (EK_A, EK_A^{priv}) mit dem öffentlichen Schlüssel EK_A für den durchzuführenden Protokollablauf.

Der gemeinsame, geheime Schlüssel SK wird aus Sicht von Partner A mittels einer 3- bzw. 4-maligen Anwendung des DH-Verfahrens wie folgt berechnet:

Fall 1: A besitzt keinen One-Time Prekey von B:

1. $DH1 = ECDH(IK_A^{priv}, SPK_B)$.
DH1 dient zur Authentisierung von A.
2. $DH2 = ECDH(EK_A^{priv}, IK_B)$.
DH2 dient zur Authentisierung von B.
3. $DH3 = ECDH(EK_A^{priv}, SPK_B)$.
DH3 wird für die Forward Secrecy benötigt.
4. $SK = KDF(DH1 \mid DH2 \mid DH3)$.
KDF ist eine festgelegte Key-Derivation-Funktion.

Fall 2: A besitzt einen One-Time Prekey von B. Dann fließt in die Berechnung von SK noch ein weiterer DH-Schlüssel ein:

1. $DH4 = ECDH(EK_A^{priv}, OPK_B)$
DH4 dient zur Abwehr von Replay-Angriffen.
2. $SK = KDF(DH1 \mid DH2 \mid DH3 \mid DH4)$.

Nach Berechnung von SK löscht A die flüchtigen Schlüssel und die DH-Schlüssel DH1-DH4.

Partner B berechnet seinerseits den Schlüssel SK auf der Basis der zu berechnenden Schlüssel DH1, DH2, DH3 und optional DH4. Dabei gilt

$$\begin{aligned} DH1 &= ECDH(IK_A, SPK_B^{priv}). \\ DH1 &= ECDH(EK_A, SIK_B^{priv}). \\ DH1 &= ECDH(EK_A, SPK_B^{priv}). \\ DH1 &= ECDH(EK_A, OPK_B^{priv}). \end{aligned}$$

Abbildung 14.33 veranschaulicht die drei bzw. vier DH-Schlüsselberechnungen.

Schritt 4 A sendet eine Initial-Nachricht an B:

Hierfür erstellt A zunächst eine Datenstruktur AD (Associated Data), die zumindest die öffentlichen Identitätsschlüssel beider Partner enthält, $AD = IK_A \mid IK_B$. Zusätzlich können der Datenstruktur auch weitere Informationen, wie beispielsweise der Nutzernamen, ein X509-Zertifikat oder andere Identitätsattribute, hinzugefügt werden.

A sendet eine Nachricht an B, mit:

$A \rightarrow B: IK_A, EK_A, C, LP,$

wobei LP die Liste der Identifikatoren der verwendeten Prekeys von B ist. $C = enc([M, AD], SK)$ ist eine AEAD-Verschlüsselung (siehe Seite 315) der Nachricht M . Zur Verschlüsselung wird entweder der gemeinsame Schlüssel SK direkt genutzt, oder es wird mittels einer festgelegten Ableitungsfunktion PRF aus SK ein Schlüssel abgeleitet.

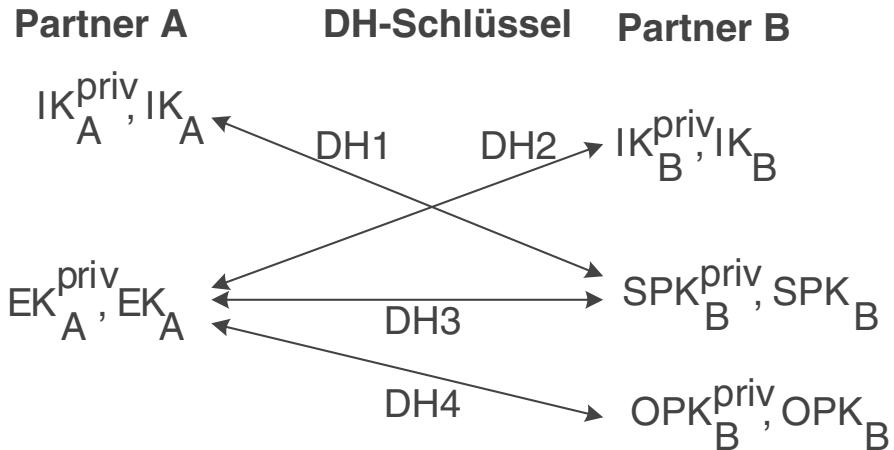


Abbildung 14.33: Schlüsselvereinbarungen im X3DH-Protokoll

Schritt 5 B empfängt und verarbeitet die Nachricht von A:

B empfängt

Message = IK_A, EK_A, C, LP .

B extrahiert die öffentlichen Schlüssel IK_A und EK_A von A aus der Nachricht, lädt seine privaten Schlüssel-Bestandteile der Prekeys, die A in der Liste LP aufgelistet hat, lädt seinen privaten Identity-Key IK_B^{priv} und berechnet dann mit seinen privaten Schlüsseln den gemeinsamen DH-Schlüssel SK . B konstruiert die Datenstruktur AD aus den ihm vorliegenden Informationen, $AD = IK_A \mid IK_B$, und entschlüsselt den Kryptotext C , $dec([C, AD], SK)$. Falls die Entschlüsselung fehlschlägt, wird das Protokoll beendet und der Schlüssel SK verworfen. Andernfalls kann der Schlüssel für weitere Protokollschrifte verwendet werden.

Sicherheitsanalyse:

Da die Sicherstellung der Authentizität der öffentlichen Identity-Keys nicht Bestandteil des X3DH-Protokolls ist, muss deren Authentizität vorab sichergestellt werden, beispielsweise durch manuell geprüfte Fingerprints der Schlüssel.

Replay

Wenn keine One-Time Prekeys von Bob verwendet werden, das X3DH also nur mit den DH-Schlüsseln DH1-DH3 arbeitet, so ist das Protokoll anfällig gegen Replay-Angriffe. Das bedeutet, die initiale Nachricht von A aus Schritt 4 kann wieder eingespielt werden, ohne dass B dies bemerkt. Das damit verbundene Risiko kann minimiert werden, wenn B und A direkt nach Beendigung des X3DH-Protokollablaufs einen neuen gemeinsamen Kommunikationsschlüssel aushandeln, der auf einem frischen Random-Input von B basiert, wie dies bei den Ratchet-Protokollen der Fall ist. So kann B bei-

spielsweise den SK mit einem neu berechneten DH-Schlüssel kombinieren, um einen randomisierten Kommunikationsschlüssel zu erzeugen. Alternativ kann B auch eine Liste von bereits gesehenen Nachrichten mit den von A dafür verwendeten, flüchtigen öffentlichen Schlüsseln EK_A speichern.

Da der Server S keine sensitiven Daten verwaltet, sind keine besonderen Anforderungen an dessen Vertrauenswürdigkeit zu stellen. Der Server kann jedoch DoS-Angriffe durchführen, indem er die Zustellung von Nachrichten verhindert oder verzögert. Zudem kann er dafür sorgen, dass A keine One-Time Prekeys erhält, so dass die Forward Secrecy von SK von der Gültigkeitsdauer des signierten Schlüssels SPK_B von B abhängt, bzw. eine Replay-Attacke möglich ist, gegen die die Partner sich aber mit post-X3DH-Protokollmechanismen, wie sie durch das Double-Ratchet-Protokoll implementiert werden, schützen können.

Server Trust

14.7.2 Double Ratchet-Protokoll

Die Kernidee des Double-Ratchet-Protokolls [141] besteht darin, zwei Ratchet-Varianten, das symmetrische und das DH-Ratchet, miteinander zu kombinieren. Durch Nutzung des symmetrischen Ratchet-Verfahrens wird für jede übertragene Nachricht ein eigener, symmetrischer Nachrichtenschlüssel berechnet. Mit dem DH-Ratchet-Verfahren werden neue Session-Keys, das sind die DH-Schlüsselpaare, etabliert, und es wird ein neuer, gemeinsamer DH-Schlüssel berechnet. Dieser fließt in das symmetrische Ratchet-Verfahren ein, wodurch er die Berechnung der neuen Nachrichtenschlüssel beeinflusst und die Eigenschaft der Folgenlosigkeit gewährleistet. Damit wird sichergestellt, dass die genutzten Schlüssel einem Angreifer keine Möglichkeit lassen, basierend auf der Kenntnis eines aufgedeckten symmetrischen Schlüssels zukünftige verschlüsselte Nachrichten zu entschlüsseln. Nachfolgend werden zunächst die beiden einzelnen Protokolle und dann deren Verzahnung vorgestellt. Das Vorgehen wird in Beispiel 14.8 illustriert.

Ratchet

Die Partner A und B verwalten jeweils drei Datenstrukturen, die sogenannten KDF-Ketten (KDF-chains). Es handelt sich um die Root-, Sender- und Empfängerkette, wobei die Senderkette des Partners A der Empfängerkette des Partners B entspricht und umgekehrt. Abbildung 14.34 veranschaulicht das generelle Vorgehen bei der Bildung einer KDF-Kette. Dabei ist KDF eine Schlüsselableitungsfunktion, die einen geheimen Schlüssel, den Chain-Key, sowie eine Dateneingabe erhält, und als Ausgabe den nächsten Chain-Key sowie einen Ausgabeschlüssel (Output-Key) erzeugt.

KDF-chain

Vorbemerkungen

Der initiale Chain-Key der Root-Kette ist ein gemeinsamer, geheimer Schlüssel SK der Partner. Im Signal-Protokoll wird SK mittels des X3DH-

Initialisierung

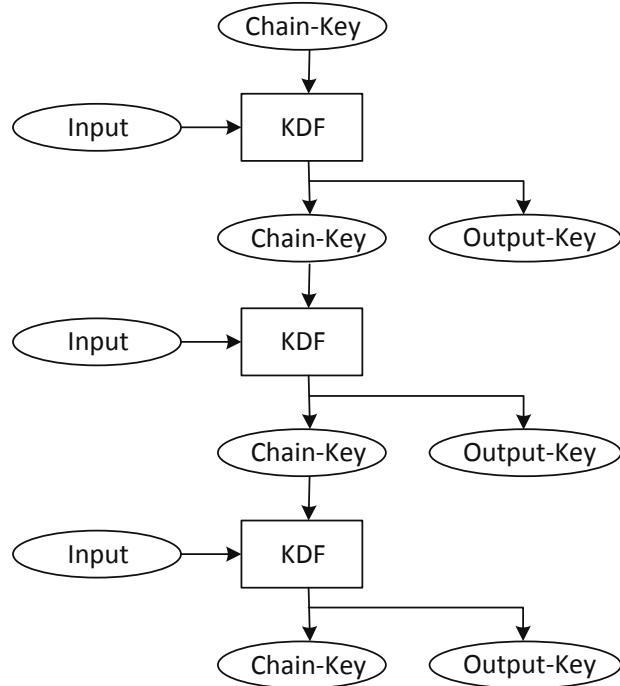


Abbildung 14.34: Konstruktionsprinzip einer KDF-Kette

Protokolls, wie unter 14.7.1 beschrieben, etabliert. Bei Initialisierung des Double-Ratchet-Protokolls erhält zumindest der Partner, der eine erste Nachricht senden möchte, auch den initialen, öffentlichen Ratchet-Public-Key des anderen Partners. Das ist der signierte Prekey SPK des Partners aus dem X3DH-Protokoll. Direkt nach der Initialisierung erzeugt der sendewillige Partner sein initiales DH-Ratchet-Schlüsselpaar (e, d) . Jede Nachricht, die einer der beiden Partner verschickt, enthält im Nachrichten-Header den aktuellen, öffentlichen Schlüssel e des Ratchet-Schlüssels des sendenden Partners.

In Beispiel 14.8 ist der Ablauf des Double-Ratchet-Protokolls in einem Ausschnitt aus einem Kommunikationsszenario zwischen den Partnern A und B veranschaulicht. In dem Beispiel sendet A drei Nachrichten und empfängt eine Nachricht von B, bevor A wiederum eine weitere Nachricht versendet.

Um auch mit Nachrichtenverlusten und geänderten Nachrichtenreihenfolgen umgehen zu können, fügt der Sender jedem Nachrichten-Header die laufende Nummer N , $N = 0, 1, \dots$, der Nachrichten aus seiner aktuellen Sende-

kette sowie die Anzahl PN der Nachrichtenschlüssel aus der vorherigen Sende-Kette hinzu. Der Empfänger einer Nachricht ist damit in der Lage, die Schlüsselerzeugung soweit fortzusetzen, bis der benötigte Nachrichtenschlüssel berechnet ist, der zur Entschlüsselung der aktuell empfangenen Nachricht erforderlich ist. Die erforderliche Anzahl an Schritten ergibt sich mit $s = PN - LR$, wobei LR die Länge der aktuellen Empfangs-Kette des Nachrichtenempfängers ist. Die Nachrichtenschlüssel, die zu übersprungenen Nachrichten gehören, werden gespeichert für den Fall, dass diese Nachrichten zu einem späteren Zeitpunkt doch noch eintreffen.

In Beispiel 14.8 ist in Abbildung 14.36 exemplarisch der Umgang mit verzögerten bzw. verlorenen Nachrichten aus Sicht des Empfängers B dargestellt.

DH-Ratchet

Jeder Partner besitzt zu jedem Zeitpunkt ein aktuelles DH-Ratchet-Schlüsselpaar (e, d) .

DH-Ratchet

Ablauf des Protokolls

Abbildung 14.35 im Beispiel 14.8 visualisiert einige der unten beschriebenen Aspekte.

1. Gegeben sei eine Situation, in der der Partner A den öffentlichen Ratchet-Schlüssel e_B seines Partners B und ein aktuelles eigenes Ratchet-Schlüsselpaar (e_A, d_A) besitzt, und seine drei KDF-Ketten über aktuelle Chain-Keys verfügen.

Senden

2. Partner A sendet eine Nachricht an B, die seinen aktuellen Ratchet-Public-Key e_A , sowie die Informationen N und PN im Nachrichten-Header enthält.

$A \rightarrow B: e_A, N, PN, c_i$, wobei $c_i = E(mA_i, A_i)$.

c_i ist die verschlüsselte Nachricht. Zur Verschlüsselung wird der Nachrichtenschlüssel²⁰ A_i verwendet, wofür A das symmetrische Ratchet-Verfahren genutzt hat (siehe unten).

3. Beim Empfang der Nachricht kontrolliert der Empfänger B anhand der in der Nachricht übertragenen Werte N und PN , ob er einen DH-Ratchet-Schritt oder symmetrische Ratchet-Schritte durchführen muss. Falls A seine Nachricht unter Einbeziehung eines neuen Ratchet-Keys generiert hat, so muss B einen DH-Ratchet-Schritt durchführen, um diesen neuen Schlüssel, dessen öffentlichen Bestandteil A ja in seiner Nachricht übermittelt hat, zu verwenden.

Empfangen

²⁰ Zum leichteren Verständnis bezeichnen wir in der folgenden Beschreibung den Nachrichtenschlüssel für die i-te Nachricht von Partner A mit A_i und die Nachricht von A mit mA_i , für B gilt das Analoge.

B wird nun zunächst im ersten Teilschritt des DH-Ratchet einen neuen CK Schlüssel für seine Empfangs-Kette berechnen. Dazu berechnet er den gemeinsamen DH-Schlüssel $DH = ECDH(d_B, e_A)$. Dieser Schlüssel dient als Eingabe in seine Root-Kette, um zusammen mit dem aktuellen Chain-Key RK der Root-Kette den Chain-Key CK der neu erzeugten Empfangs-Kette zu berechnen:

$(CK, RK_{neu}) = \text{KDF}(DH, RK)$, wobei RK der Chain-Key der Root-Kette von B ist, der nach Ausführung der Berechnung durch den neuen Chain-Key RK_{neu} ersetzt wird.

Der berechnete Schlüssel CK ist der Startschlüssel der neuen Empfangs-Kette. Damit hat B die Schritte, die A bereits vor dem Verschlüsseln seiner gesendeten Nachricht auf seiner Seite durchgeführt hat, ebenfalls ausgeführt. In den neuen CK -Schlüssel für seine Empfänger-Kette ist bereits der neue öffentliche Schlüssel von A eingeflossen, aber B selbst musste zur Berechnung noch einmal seinen alten privaten Schlüssel weiter verwenden, da A den dazu öffentlichen Schlüssel genutzt hatte, um auf seiner Seite CK zu berechnen.

Danach erzeugt B im zweiten Teilschritt des DH-Ratchet ein neues Ratchet-Schlüsselpaar (e_B, d_B) , berechnet damit erneut einen DH-Schlüssel, berechnet den neuen Chain-Key RK der Root-Kette sowie den Chain-Key CK einer neuen Sende-Kette. Damit ist eine neue Session etabliert, die Chain-Keys der beiden Sende- und Empfangs-Ketten sind generiert und können als Basis zur Erzeugung von neuen Nachrichtenschlüsseln für diese Session genutzt werden.

Symmetrisches Ratchet

Symmetrisches
Ratchet

Wenn einer der Partner A bzw. B eine Nachricht sendet oder empfängt, wird das symmetrische Ratchet-Verfahren ausgeführt, um den Nachrichtenschlüssel zur Ver- oder Entschlüsselung der Nachricht zu berechnen.

Ablauf des Protokolls

1. Betrachten wir erneut das Szenario: Partner A sendet eine verschlüsselte Nachricht mA_i an Partner B:

$A \rightarrow B: e_A, N.PN, c_i$, wobei $c_i = E(mA_i, A_i)$.

A berechnet dazu den Nachrichtenschlüssel A_i unter Rückgriff auf den Chain-Key CK der Sende-Kette:

$(CK, A_i) = \text{KDF}(\text{Konstante}, CK)$.

Die Konstante kann beispielsweise ein Byte 0x01 sein.

2. Partner A kann danach den Nachrichtenschlüssel A_i sowie auch den alten Chain-Key CK löschen.

Das nachfolgende Beispiel erläutert die Abläufe und Zusammenhänge an einem Kommunikationsszenario.

Beispiel 14.8 (Secure Messaging zwischen Partnern A und B)

Abbildung 14.35 veranschaulicht das Double-Ratchet-Verfahren aus Sicht des Kommunikationspartners A. Die Abbildung visualisiert die Ratchet-Schritte der beiden Ratchet-Varianten für ein Szenario, in dem A nach der Initialisierung den aktuellen öffentlichen Ratchet-Schlüssel e_B des Partners B besitzt, sein aktuelles Ratchet-Schlüsselpaar (e_A, d_A) erzeugt hat und über einen Chain-Key RK für seine Root-Kette verfügt. In dem Initialisierungsschritt erzeugt A zudem mittels des DH-Verfahrens und unter Nutzung der KDF-Funktion einen initialen Chain-Key CK für seine Sende-Kette. Die Abbildung visualisiert die Ratchet-Schritte, die durchgeführt werden, wenn A nun beispielsweise drei Nachrichten mA_1, mA_2, mA_3 an B sendet, danach eine Nachricht mB_1 von B empfängt und dann wieder selbst eine Nachricht mA_4 an B sendet.

Partner A

Der Nachrichtenschlüssel A_1 zur Verschlüsselung der Nachricht mA_1 wird durch die Anwendung des symmetrischen Ratchet-Verfahrens auf die aktuelle Sende-Kette von A, also unter Nutzung des aktuellen Chain-Keys CK , erzeugt. Die Nachrichtenschlüssel der weiteren gesendeten Nachrichten werden durch das wiederholte Durchführen des symmetrischen Ratchet-Schrittes generiert.

Senden

Wenn Partner A die Nachricht mB_1 von B empfängt, muss er den Nachrichtenschlüssel B_1 , mit dem Partner B seine Nachricht verschlüsselt hat, berechnen, um die Entschlüsselung durchzuführen zu können. Da er mit der Nachricht mB_1 auch den aktuellen Ratchet-Key von B erhält, berechnet er zunächst mit diesem und seinem eigenen privaten Ratchet-Schlüssel ein gemeinsames DH-Geheimnis, das B in analoger Weise vor dem Versenden seiner Nachricht berechnet hatte. Damit generiert A als nächstes den aktuellen Chain-Key seiner Empfangs-Kette und wendet danach das symmetrische Ratchet-Verfahren auf die Empfangs-Kette an. Auf diese Weise generiert A den gemeinsamen Nachrichtenschlüssel B_1 .

Empfangen

Der Partner A führt anschließend einen vollständigen DH-Ratchet-Schritt durch, indem er zum einen ein neues Ratchet-Schlüsselpaar und zum anderen zusammen mit dem öffentlichen Schlüssel (e_B) von B für seine neue Sende-Kette einen Startschlüssel CK generiert.

In dem Beispielszenario sendet A nun seine nächste Nachricht mA_4 und generiert dazu den Nachrichtenschlüssel A_4 , der den neu generierten CK der aktuellen Sende-Kette von A als Basis hat.

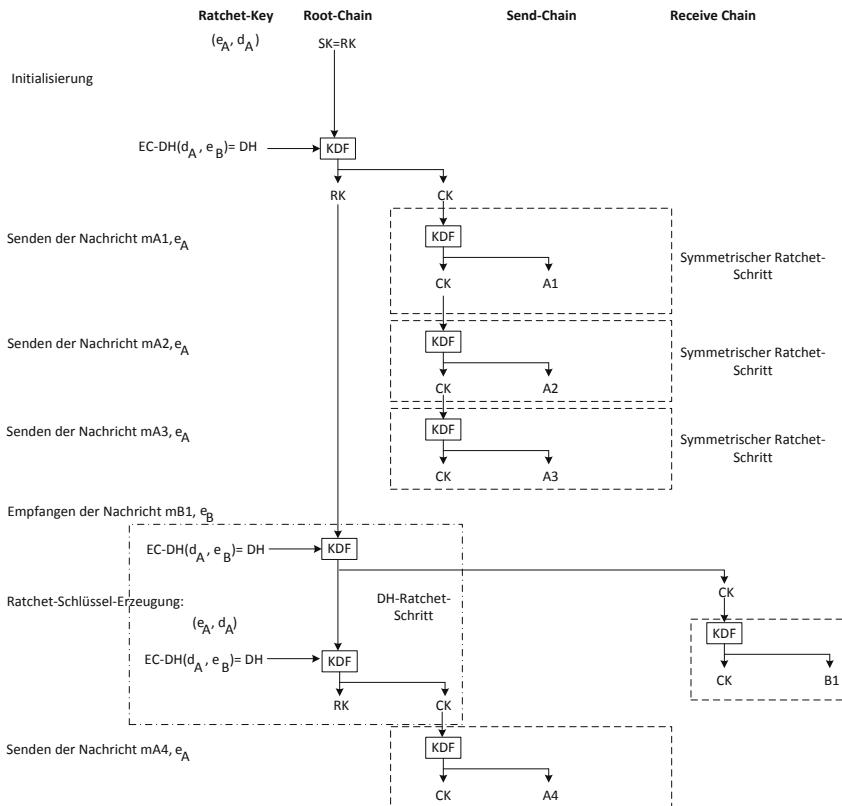


Abbildung 14.35: Double-Ratchet-Verfahren aus Sicht des Partners A

In dem beschriebenen Ablauf wurden bislang keine Nachrichtenverluste oder Veränderungen der Nachrichtenreihenfolge betrachtet. Abschließend wollen wir auch darauf noch kurz eingehen.

Out-of-order-Nachrichten

Betrachten wir erneut das beschriebene Szenario, diesmal aus der Sicht von B. Nehmen wir an, dass der Partner B nur die erste Nachricht mA_1 empfangen hat, bevor er seine erste Nachricht mB_1 sendet und dass er danach die Nachricht mA_4 von A empfängt. Das heißt, dass die Nachrichten mA_2, mA_3 entweder verloren gegangen sind, oder zu einem späteren Zeitpunkt noch eintreffen. B berechnet für alle Fälle die zur Entschlüsselung der noch ausstehenden Nachrichten erforderlichen Nachrichtenschlüssel und speichert

diese. Damit B überhaupt Kenntnis darüber hat, wieviele Nachrichten noch ausstehen und in welcher Reihenfolge diese gesendet wurden, sendet A mit seiner Nachricht, wie oben bereits angesprochen, die Informationen N und PN mit.

Partner B berechnet $PN - LR = 2$ und weiß, dass noch zwei Nachrichten von A ausstehen. Er führt deshalb mit seiner noch aktuellen Empfangs-Kette zweimal das symmetrische Ratchet-Protokoll durch, bevor er seinen Ratchet-Schlüssel mit dem DH-Ratchet erneut und Nachricht mA_4 verarbeitet, also A_4 berechnet (vgl. Abbildung 14.36).

Skippen von
Schlüsseln

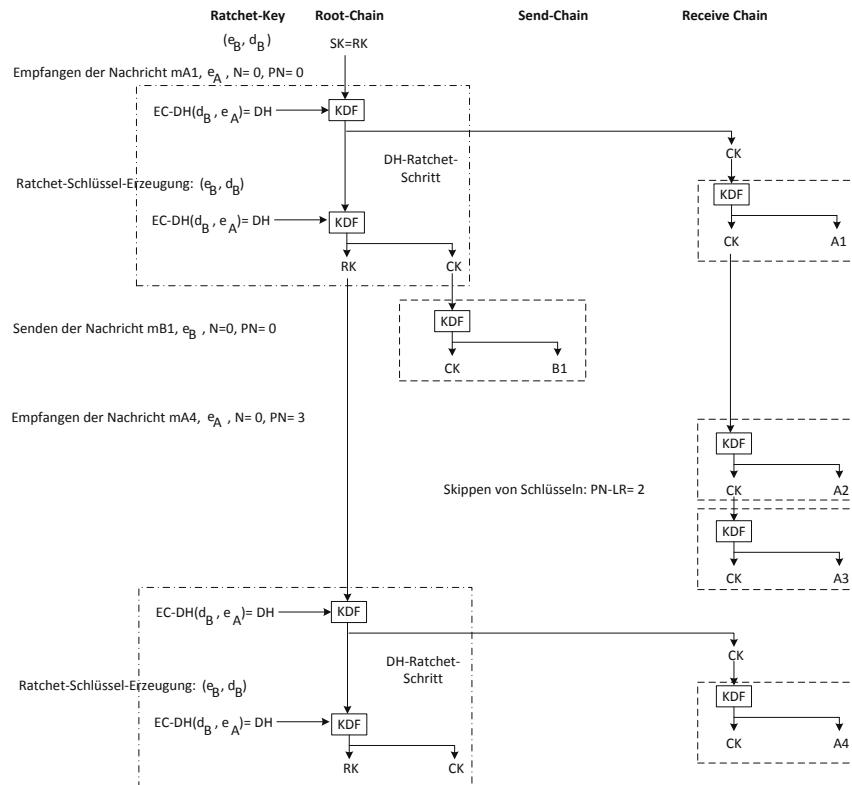


Abbildung 14.36: Ablauf aus Sicht des Partners B beim Überspringen der nicht genutzten Schlüssel A_2 und A_3

Partner B speichert die Schlüssel A_2 und A_3 für den Fall, dass die entsprechenden Nachrichten von A noch eintreffen.



Sicherheit

Das Signal-Protokoll wurde formal in Bezug auf seine Sicherheitseigenschaften verifiziert (vgl. [44]). Dabei wurden keine Schwachstellen entdeckt. Jedoch ist offensichtlich, dass die Speicherung von übersprungenen Nachrichtenschlüsseln A_i bei Partner B oder B_i bei Partner A eine mögliche Schwachstelle bedeuten kann, falls die Schlüssel nicht geschützt auf den jeweiligen Systemen abgelegt werden. Gespeicherte Nachrichtenschlüssel sollten deshalb nicht lange aufbewahrt sondern nach kurzen Zeitintervallen gelöscht werden.

14.8 Blockchain

Seit Satoshi Nakamoto [127] im Jahr 2008 in einem White Paper eine neue digitale Kryptowährung namens Bitcoin²¹ auf der Basis der Blockchain-Technologie veröffentlicht hat, ist um diese Technologie ein wahrer Hype entbrannt. Mit der Blockchain-Technologie versucht man Probleme in verteilten, kooperativen Anwendungen zu lösen, so dass die kooperierenden Teilnehmer selbst nicht vertrauenswürdig sein müssen, keine zentrale Vertrauensinstanz benötigt wird, keine Vertrauensinfrastruktur aufgebaut werden muss und dennoch ein akzeptables Vertrauenslevel geschaffen werden kann, so dass die Teilnehmer Werte, wie Geld oder Informationen, über diese Technologie verwalten können. Die Blockchain löst dieses Problem, indem eine spezifische Datenstruktur, das ist die Blockchain, aufgebaut wird, die durch die verteilten Teilnehmer im Konsens verwaltet wird. Die Blockchain-Datenstruktur besteht aus einer verketteten Liste von Blöcken, wobei jeder Block den kryptografischen Hash des vorhergehenden Blocks beinhaltet. Jeder Block enthält Informationen über durchgeföhrte Transaktionen, wie beispielsweise einen Geldtransfer von Partner A zu Partner B. Blöcke können nur im Konsens der Teilnehmer der Kette hinzugefügt werden. Zusammengefasst kann man sagen: Blockchain ist eine Technologie, mit der man Datenblöcke und damit auch Transaktionen in eine Reihenfolge bringen kann, die für alle nachvollziehbar und nicht veränderbar ist.

Während man die digitalen Kryptowährungen, wie Bitcoin, zu der ersten Generation von Anwendungen für die Blockchain zählt, nehmen verallgemeinerte Anwendungen an Bedeutung zu. Zu nennen sind dabei insbesondere so genannte Smart Contracts, die die Blockchain nutzen, um zwischen dezentral agierenden Organisationen oder technischen Systemen Regeln und Aktivitäten auszuhandeln und deren Ausführung zu initiieren. Ein Beispiel sind Smart Contracts für elektronische Türschlösser, die beispielsweise automatisch prüfen, ob der Nutzer die Nutzungsgebühr bezahlt hat.

²¹ <https://www.ethereum.org>

Grundlage für die schnelle Verbreitung der Blockchain-Technologie sind deren charakteristische Eigenschaften, die für viele verteilte Anwendungsszenarien interessant sind. Nachfolgend geben wir einen Überblick über diese Eigenschaften.

- Eine Blockchain erfordert keine zentrale vertrauenswürdige Instanz, sondern basiert auf Techniken zur verteilten Konsensbildung zwischen nicht notwendigerweise vertrauenswürdigen Teilnehmern. Konsensbildung
- Blockchains bilden die Grundlage für ein so genanntes Internet-of-value. Damit ist gemeint, dass Wertobjekte, wie beispielsweise digitales Geld, nachvollziehbar und nicht-abstreitbar von einem Nutzer an einen anderen transferiert werden können. Nachvollziehbarkeit
- In einer Blockchain werden revisionssichere Transaktionen, die nicht mehr nachträglich gelöscht oder geändert werden können, repräsentiert. In der klassischen Ausprägung der Chain sind die Transaktionen allen Teilnehmern sichtbar, so dass alle durchgeführten Transaktionen für alle Teilnehmer nachvollziehbar sind. Durch die Vergabe von Zugriffsrechten kann die Sichtbarkeit jedoch eingeschränkt werden; ein solches Vorgehen wird beispielsweise bei privaten Blockchains im Unternehmenskontext angewandt. Manipulations-sicher

Soll der Effekt einer Transaktion rückgängig gemacht werden, so kann dies nur im Konsens geschehen, indem eine Komplementär-Transaktion in der Blockchain abgelegt wird. Die ursprüngliche Transaktion wird jedoch nicht gelöscht.

Die meisten heutigen Einsatzfelder der Blockchain-Technologie sind noch immer im Finanzbereich zu finden mit Kryptowährungen wie Bitcoin und Ethereum²² oder Blockchain-basierten Anwendungen, um beispielsweise Kreditgeschäfte zwischen Privatkunden oder Finanztransaktionen zwischen Finanzinstituten abzuwickeln. Die Blockchain dringt seit einiger Zeit jedoch auch in ganz andere Branchen vor, zum Beispiel in das Supply-Chain-Management, um Transaktionen, Lieferungen und Rechnungen nachzuverfolgen, oder zur Nachverfolgung von Lieferketten bei Lebensmitteln.

Auch im Internet of Things und bei industriellen Wertschöpfungsprozessen wird ein großes Potential für Blockchain-Anwendungen gesehen, um in diesen verteilten Szenarien zentrale Instanzen, die meist nicht etabliert werden können, durch Blockchain-basierte Anwendungen zu ersetzen. So genannte Smart Contracts, auf die wir noch eingehen werden, bieten die Möglichkeit, Vereinbarungen zwischen Maschinen so festzulegen, dass deren Einhaltung

Anwendungsbereiche

²² <https://www.ethereum.org>

von allen beteiligten Parteien geprüft und Manipulationen und Zuwiderhandlungen automatisiert erkannt werden können. Durch die Nutzung von Blockchain können diese Kooperationen auch asynchron erfolgen. Das heißt, falls Partner zeitweise offline sind, erfolgt eine Synchronisation über die Blockchain-Datenstruktur. Maschinen können mit Blockchain-basierten verteilten Anwendungen Dienstleistungen anbieten, die von anderen Maschinen in Anspruch genommen werden. Die Abrechnung der nachweislich in Anspruch genommenen Dienste kann ebenfalls automatisiert und nachvollziehbar erfolgen.

Ein weiterer interessanter Anwendungsbereich besteht in der Erstellung von Herkunfts-nachweisen (Provenance) und der Nachverfolgung von Besitz-Historien für wertvolle Assets, wie beispielsweise Diamanten. Die Nachverfolgbarkeit ist auch für geregelte Szenarien, in denen Compliance-Anforderungen zu erfüllen sind, interessant, so dass Blockchain-Technologie auch zunehmend für das manipulationssichere Protokollieren von durchgeführten Aktionen im Unternehmen und als Compliance-Nachweis untersucht wird. Auch im Bereich der öffentlichen Verwaltung ist die Nutzung von Blockchain-Technologie ein Thema, um die Verwaltungsabläufe wie Grundbucheintragungen oder Beglaubigungen, die häufig noch eine vertrauenswürdige Instanz, wie einen Notar, erfordern, effizienter zu gestalten.

Im Folgenden werden wir die technischen Grundlagen der Blockchain-Technologie erklären und abschließend auf Bitcoin als bekanntes Fallbeispiel für die Nutzung einer Blockchain etwas genauer eingehen.

14.8.1 Technische Grundlagen

Ledger
Wie weiter oben bereits kurz angesprochen wurde, ist eine Blockchain eine Datenstruktur, die häufig als verteilt verwaltete Datenbank implementiert wird. Eine solche Datenbank repräsentiert ein Hauptbuch, wie es in der Buchführung bekannt ist. In einem Hauptbuch werden alle Aktivitäten der verschiedenen Grundbücher bzw. Tagebücher zusammengeführt und hinterlegt. Eine Blockchain entspricht dieser Sicht eines Hauptbuches, das alle verteilten Transaktionen in einer Struktur bündelt. Diese Struktur wird aber nicht nur einmal gespeichert, sondern existiert in Kopie auf vielen Rechnern, weshalb man auch häufig vom verteilten Hauptbuch, dem Distributed Ledger, spricht. Die Teilnehmer, die mit der Blockchain-Datenstruktur arbeiten, bilden ein virtuelles Netzwerk. Die Blockchain wird durch die Teilnehmer am Netzwerk organisiert. Technisch umgesetzt wird das virtuelle Netzwerk mittels entsprechender Kommunikationsprotokolle, wobei häufig, wie zum Beispiel bei Bitcoin, auf Peer-to-Peer-Netze zur Umsetzung zurückgegriffen wird. Anstatt von Teilnehmern spricht man dann auch häufig von Knoten im Netzwerk.

Die Aufgabe einer Blockchain ist es sicherzustellen, dass die von den Teilnehmern am Netzwerk durchgeführten Transaktionen revisionssicher, nachvollziehbar und nicht abstreitbar verwaltet werden. Dazu besitzt jeder Teilnehmer eine lokale Kopie der Blockchain-Datenstruktur. Um die genannten Eigenschaften zu garantieren, werden Hash-Funktionen, Merkle-Trees, Signaturen basierend auf Public-Key-Kryptografie sowie verteilte Konsens-Protokolle verwendet. Das Vorgehen bei der Nutzung der Blockchain besteht vergröbert aus folgenden Schritten.

1. Transaktion(en) erstellen, signieren und an alle Teilnehmer versenden.
2. Eine Menge von Transaktionen in einem Block zusammenfassen und einen Hashwert darüber berechnen.
3. Den Block mit dem letzten Block der Kette verlinken.
4. Den Block mit einem Konsensverfahren veröffentlichen (Block-Mining).
5. Abhängig vom Konsensus-Modell wird an denjenigen, der einen neuen Block erstellt und der Blockchain hinzufügt noch eine Belohnung ausgeschüttet.

Nachfolgend gehen wir auf die einzelnen Schritte etwas genauer ein.

1. Transaktions-Erzeugung

Ein dazu berechtigter Teilnehmer, in öffentlichen Blockchains kann das jeder sein, generiert eine Transaktion. Das kann beispielsweise eine Geldüberweisung sein, ein Eintrag in einem Arztbrief oder eine Grundbucheintragung. Der Ersteller der Transaktion signiert diese mit seinem privaten Signaturschlüssel und sendet diese Information an alle beteiligten Knoten bzw. Teilnehmer im Netzwerk. Die Transaktion muss jetzt bestätigt und in einem neuen Block in die Blockchain aufgenommen werden. Dies erfordert die Mithilfe von so genannten Miner-Knoten, das sind die Teilnehmer, die neue Blöcke für die Chain erzeugen (siehe unten), und den verteilten Konsens der Teilnehmer des Netzwerks. Solange die Transaktion nicht bestätigt ist, wird sie in einer entsprechenden Datenstruktur, einer Art Warteliste, verwaltet.

Transaktion

2. Block-Erstellung

Speziell berechtigte Knoten im Netzwerk, die Miner-Knoten, können mit einer Menge von erhaltenen Transaktionen einen neuen Block für die Chain erzeugen. In öffentlichen Blockchains kann jeder beteiligte Knoten auch ein Miner-Knoten sein.

Der Miner wählt eine Menge von Transaktionen, die er erhalten hat und die noch nicht bestätigt sind, aus. Um die Daten der einzelnen Transaktionen integritäts geschützt im Block zu speichern, berechnet der Miner-Knoten

Merkle-Tree

zunächst einen kryptografischen Hashwert für jede Transaktion (zum bei Bitcoin mit SHA256). Die Hashwerte der einzelnen Transaktionen werden im nächsten Schritt mittels eines Merkle-Trees zusammengefasst. Das Ergebnis dieser Zusammenfassung ist, dass alle Transaktionen des Blockes durch einen einzigen Hashwert, das ist der Hashwert des Wurzelknotens des Merkle-Trees, repräsentiert werden. Ein Merkle-Tree ist ein einfacher Binärbaum, in dem der Hashwert $h_{A,B}$ des Elternknotens der Knoten A und B als Hashwert der Konkatenation der beiden Hashwerte der Blattknoten A und B berechnet wird. Also $h_{A,B} = H(h_A \mid h_B)$, wobei H die genutzte Hashfunktion ist.

Um den Block mit einer prüfbaren Uhrzeit zu versehen, wird häufig auch noch ein Zeitstempel den Header-Informationen des Blocks hinzugefügt. Dies wird beispielsweise bei Bitcoin genutzt.

3. Verkettung

Verkettung

Ein neuer Block i wird mit der Blockchain verbunden, indem in den Header des neu erstellten Blocks der Hashwert des Vorgängerblocks $i - 1$ eingefügt wird (vgl. Abbildung 14.37). Damit wird sichergestellt, dass eine Veränderung an Transaktionen, die in einem Vorgängerblock repräsentiert sind, direkt aufgedeckt werden kann. Würde ein Teilnehmer versuchen, eine Transaktion aus dem Block j zu entfernen oder abzuändern und die so manipulierte Blockchain in Umlauf zu bringen, so müsste er den Hashwert des Merkle-Trees des manipulierten Blocks j neu berechnen. Dieser neue Hashwert wird dann jedoch nicht mehr mit dem Hashwert übereinstimmen, der in der ursprünglichen Blockchain in dem Nachfolgeblock $j + 1$ abgelegt worden ist. Weiterhin gilt, dass aufgrund der Verkettungskonstruktion der Blockchain auch in allen nachfolgenden Blöcken $j + 2, j + 3, \dots$ die Hashwerte nicht mehr passen. Der Angreifer müsste somit auch alle Nachfolgeblöcke in der Kette angleichen, damit eine konsistente Kette entsteht. Dies darf für den Angreifer nicht einfach möglich sein. Deshalb muss das Einfügen eines Blockes in die Kette im Konsens der Mehrheit der Teilnehmer des Netzwerks erfolgen und ein Manipulationsversuch muss für den Angreifer einen zu hohen Aufwand bedeuten.

Ein häufig genutztes Vorgehen zur Konsensbildung ist das so genannte Proof-of-Work (PoW), worauf wir weiter unten noch genauer eingehen werden. Die Erstellung des Proof-of-Work für einen Block erfordert eine Berechnung, die sehr rechenzeitintensiv und energieaufwändig ist. Will ein Angreifer einen Block fälschen, muss er für den gefälschten Block sowie auch für jeden Nachfolgeblock der Kette den Proof-of-Work neu berechnen. Das ist sehr aufwändig.

Abbildung 14.37 veranschaulicht einen Ausschnitt aus einer Blockchain. Sie zeigt den Aufbau eines Blockes i , bestehend aus einem Header-Bereich, der die Verwaltungsinformationen enthält, wie den Hashwert h_i der Wurzel des Merkle-Trees, den Hashwert des Vorgängerblocks, $H(\text{Block-}i-1)$ sowie Informationen, die für die Konsensbildung erforderlich sind. In der Abbildung wird von einem Proof-of-Work ausgegangen, so dass die berechnete Nonce mit aufgenommen ist. Der Datenteil eines Blocks enthält die im Block zusammengefassten Transaktionen zusammen mit dem Merkle-Tree über deren Hashwerte.

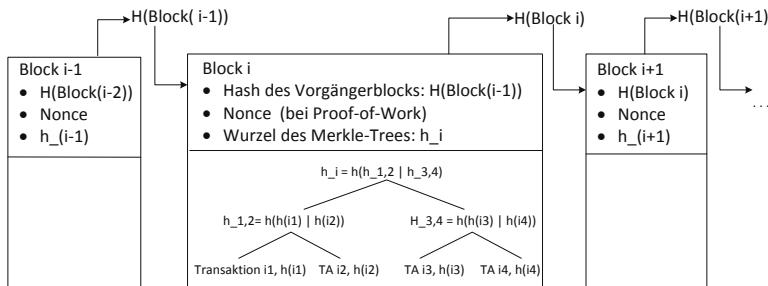


Abbildung 14.37: Verkettung von Blöcken in der Blockchain

4. Konsensbildung

Neue Blöcke werden im verteilten Konsens in die Blockchain aufgenommen. Hierfür sind unterschiedliche Vorgehensweisen im Einsatz.

4.1 Proof-of-Work

Ein weit verbreitetes Verfahren, das auch bei Bitcoin genutzt wird, arbeitet nach dem Prinzip des Proof-of-Work (PoW), das oben bereits kurz erwähnt wurde. Hierbei ist ein kryptografisches Rätsel zu lösen. Die Idee dabei ist, dass das Lösen des Rätsels sehr rechenaufwändig ist, während die Prüfung, ob das Rätsel korrekt gelöst wurde, effizient möglich ist. Das Ziel des PoW-Verfahrens ist es, einen Wert n , man spricht hier von einer Nonce, zu finden, so dass der Hashwert $h = H(H(\text{Block}(i-1)), n, h_i)$ mindestens eine festgelegte Anzahl an führenden Nullen besitzt. Präziser gesagt, es wird ein Schwellwert D festgelegt, und der berechnete Hashwert h muss kleiner als der Schwellwert D sein. Dabei ist $H(\text{Block}(i-1))$ der Hashwert des letzten Blocks der Kette. h_i ist der Hashwert der Wurzel des Merkle-Trees des erzeugten Blocks und repräsentiert in Form einer Prüfsumme die Transaktionen des neuen Blocks i . H ist die genutzte Hashfunktion, beispielsweise SHA-256. Je größer die geforderte Anzahl an führenden Nullen ist, desto schwieriger und Rechenzeit-intensiver wird

PoW

Mining

es, einen passenden Wert n zu finden. Das Suchen nach der passenden Nonce für einen Proof-of-Work nennt man Mining. Der Begriff stammt aus dem Umfeld von Bitcoin, da dort die Miner zur Belohnung für den zur Erstellung eines neuen Blocks geleisteten Rechenaufwand Bitcoins erhalten. In Anlehnung an das Schürfen von Gold schürfen die Miner-Knoten bei Bitcoin nach digitalen Münzen, eben den Bitcoins.

Veröffentlichung

Wurde eine Lösung, also eine passende Nonce, gefunden, so fügt der Miner-Knoten diese Nonce in den neuen Block ein, verkettet den Block mit seiner bestehenden Blockchain und veröffentlicht den neuen Block im Netzwerk, so dass die anderen Miner-Knoten das Ergebnis prüfen und übernehmen können.

Prüfung

Eine Prüfung, ob die gefundene Nonce n , die im versandten Block enthalten ist, korrekt ist, kann jeder Teilnehmer sehr einfach über eine Hashberechnung durchführen. Der Teilnehmer berechnet mit den vom Miner-Knoten versandten Hashwerten h_i des neuen Blocks i , dem Wert $H(\text{Block}(i-1))$, sowie der Nonce n den Hashwert $h = H(H(\text{Block}(i-1), n, h_i))$. Danach prüft er, ob die geforderte Anzahl an führenden Nullen in h vorhanden ist, also ob der Hashwert kleiner als der Schwellwert D ist. Wird ein neuer Block in die Kette aufgenommen, so wird automatisch damit ein neues kryptografisches Rätsel gestellt, da nun bei der Erzeugung des nächsten Blocks $i+1$ ein PoW für den versandten Block i und dessen Hashwert $H(\text{Block}(i))$ berechnet werden muss.

Schwellwert

Der Schwellwert D wird bei Bitcoin nach der Erzeugung von 2016 Blöcken bzw. nach ungefähr zwei Wochen, basierend auf den Plausibilitäts-geprüften Zeitstempeln in den 2016 Blöcken, automatisch hochgesetzt. Dadurch wird erreicht, dass stets ein Mindestaufwand zu betreiben ist, um einen neuen Block der Kette hinzuzufügen. Bei Bitcoin soll die Berechnung immer mindestens 10 Minuten dauern. Damit ist zum einen sichergestellt, dass es für einen Miner-Knoten erheblichen Aufwand bedeutet, das Rätsel zu lösen. Zum anderen soll damit die Wahrscheinlichkeit für Konflikte möglichst gering sein. Das bedeutet, dass man sicherstellen möchte, dass sich ein neuer Block eines Miner-Knotens mit hoher Wahrscheinlichkeit im Netzwerk verbreiten und alle Teilnehmer erreichen kann, bevor ein anderer Miner-Knoten parallel ebenfalls einen neuen Block erzeugt und veröffentlicht. Diese Konfliktfreiheit ist jedoch nicht gewährleistet, sondern es kann zu Konflikten und damit zur Verzweigung der Blockchain kommen. Deshalb ist eine Konsensbildung mit Konfliktauflösung erforderlich.

4.2 Konfliktlösung

Konsensus

Wie gesagt können Miner-Knoten parallel und unabhängig voneinander Transaktionen in neuen Blöcken zusammenfassen und für ihren Block

nach einer Nonce suchen. Es kann vorkommen, dass mehrere Miner-Knoten gleichzeitig zu einem Ergebnis kommen und einen neuen Block veröffentlichen. Die Blöcke enthalten dann in der Regel unterschiedliche Transaktionen. In einer solchen Situation, in der mehrere Blöcke veröffentlicht sind, tritt ein Konflikt auf und die Blockchain verzweigt (engl. fork). Abbildung 14.38 visualisiert diese Situation.

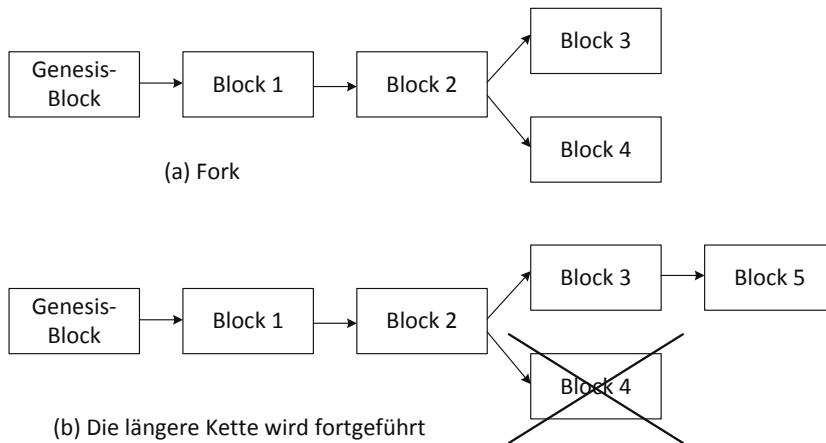


Abbildung 14.38: Verzweigung (fork) einer Blockchain

Im oberen Teil (a) der Abbildung ist die Verzweigung der Kette dargestellt. Ein Teilnehmer, der nahezu zeitgleich zwei unterschiedliche, neue Blöcke empfängt, wird den ersten empfangenen Block in seine lokale Kopie der Blockchain übernehmen und mit dieser Kette zunächst weiterarbeiten. Gleichzeitig wird er aber die anderen, etwas später eintreffenden Blöcke speichern, falls es sich herausstellen sollte, dass doch mit diesen Blöcken weiterzuarbeiten ist. Beim nächsten versandten Block, der eine der zuvor erhaltenen Ketten verlängert, ist dann klar, welche Kette die längere ist. Die längste Kette, oder präziser, die Kette, deren Erzeugung akkumuliert den größten Rechen- und Energieaufwand erfordert hat, wird vom Teilnehmer als aktuelle Blockchain übernommen. In Abbildung 14.38 ist diese Situation im unterem Bereich (b) dargestellt. Die Blöcke der Verzweigungen, die verworfen wurden, verfallen. Diejenigen Transaktionen, die nicht in dem neuen Block enthalten waren, werden in einen Pool der noch nicht bestätigten Transaktionen aufgenommen. Der Konsens besagt, dass die Partner Einigkeit über die zeitliche Reihenfolge der Transaktionen, die in verschiedenen Blöcken der Chain repräsentiert sind, haben.

Die Konsensbildung nach dem Proof-of-Work-Verfahren ist zeitlich sehr aufwändig. Bei Bitcoin beispielsweise dauert es im Durchschnitt 10 Minuten, bis ein kryptografisches Rätsel gelöst und ein neuer Block geschrifft (mining) wurde. Ein Teilnehmer an der Blockchain kann erst nach sechs Blöcken relativ sicher sein, dass seine Transaktion in der Kette akzeptiert wurde. Dieser Wert sechs ist jedoch nur ein Plausibilitätswert und nicht bewiesen, da es nie vollständig ausgeschlossen werden kann, dass ein Teilnehmer einen Fork der Kette durchführt, obwohl bereits sechs Blöcke die Transaktion bestätigt haben. Wird mit dem Fork eine längere Kette verteilt, in der die Transaktion nicht auftritt, so ist die Transaktion doch nicht bestätigt.

5. Belohnung

Belohnung

Die Miner, die das kryptografische Rätsel lösen, also die passende Nonce n berechnen, werden dafür belohnt, dass sie Rechenaufwand für die Suche nach der passenden Nonce aufgebracht haben. Im Bitcoin-System erhalten sie beispielsweise Transaktionsgebühren erstattet sowie digitale Münzen (Bitcoins) ausgezahlt. Da ein Miner selbst bestimmt, welche Transaktionen er in den neuen Block aufnimmt, kann es für einen Erzeuger einer Transaktion lohnend sein, eine Transaktionsgebühr auszuschütten, damit die Transaktion auch von Miner-Knoten verarbeitet wird. So kann in Bitcoin ein Teilnehmer eine Transaktion festlegen, in der mehr Bitcoins als Eingabe eingehen, also vom Konto des Teilnehmers abgebucht werden, als an angegebene Empfänger weitergeleitet werden. Der Differenzbetrag ist eine Transaktionsgebühr für den Miner-Knoten.

Blockchain-Varianten

öffentlich

Grob unterscheidet man zwei Blockchain-Varianten: die öffentlichen (engl. *public*), wie sie beispielsweise in Bitcoin [127] oder auch Ethereum²³ genutzt werden, und die privaten (engl. *private*) Blockchains, die zum Beispiel in Hyperledger²⁴ und MultiChain²⁵ verwendet werden. An öffentlichen Blockchains kann jeder, auch anonym, teilnehmen. Der Teilnehmer kann Transaktionen durchführen oder auch Mining betreiben. Die in öffentlichen Blockchains gespeicherten Daten sind in der Regel für alle Teilnehmer einsehbar, und auch die mit den Transaktionen verbundenen Identitäten (die Transaktionen sind digital signiert) sind für jeden Teilnehmer sichtbar. Eine öffentliche Blockchain kann den Zugriff auf Daten jedoch auch für Teilnehmer beschränken, dann ist es eine öffentliche, aber gleichzeitig genehmigungsbasierte Kette (siehe unten).

²³ <https://www.ethereum.org>

²⁴ <https://www.hyperledger.org/>

²⁵ <https://www.multichain.com/>

Bei privaten Blockchains können über ein Rollen- und Rechtemanagement Beschränkungen festgelegt werden. So kann der Teilnehmerkreis eingeschränkt, Zugriffe auf die Chain können beschränkt oder auch Berechtigungen zur Durchführung von Aktionen eingeschränkt werden. Beispielsweise lässt sich damit festlegen, wer die Berechtigung erhält, neue Blöcke zu erzeugen (mining) oder neue Transaktionen zu generieren. In privaten Blockchains, wie sie häufig im Unternehmensumfeld verwendet werden, sind somit alle Teilnehmer bekannt und der Zugriff auf die Daten der Blockchain ist auf eine Gruppe Berechtigter beschränkt.

privat

Blockchains kann man weiterhin darin unterscheiden, ob sie genehmigungsbasiert (engl. *permissioned*) oder genehmigungslos (engl. *permissionless*) sind. In einer genehmigungsbasierten Blockchain dürfen Teilnehmer nur dann Transaktionen erstellen oder auf Einträge in Blöcken lesend zugreifen, wenn sie dazu berechtigt sind. Das bedeutet damit auch, dass die Teilnehmer sich authentifizieren müssen, damit ihre Berechtigungen geprüft werden können. Die Konsensbildung erfolgt durch eine begrenzte Anzahl von dazu berechtigten Teilnehmern, die also eine spezielle Vertrauensstellung in dem Netzwerk einnehmen. Demgegenüber sind in einer genehmigungslosen Blockchain, wie der Name bereits sagt, keine Berechtigungen explizit vergeben und jeder Teilnehmer darf alles. Bitcoin ist ein Beispiel für eine Anwendung, die eine solche genehmigungslose Blockchain nutzt. In einer genehmigungslosen Blockchain muss ein Teilnehmer seine Identität nicht preisgeben. Das bedeutet, dass auch niemand verhindern kann, dass ein Teilnehmer eine Transaktion hinzufügt, so dass auf diese Weise einer möglichen Zensur durch eine zentrale Instanz entgegengewirkt werden kann. Eine private Blockchain ist stets auch genehmigungsbasiert, während öffentliche Ketten genehmigungsbasiert oder genehmigungslos sein können.

Genehmigungen

Die Varianten der Blockchain ermöglichen es, unterschiedliche Grade an Zentralisierung umzusetzen. So ist eine öffentliche Chain, die genehmigungslos arbeitet, völlig dezentral organisiert. Wenn eine öffentliche Blockchain Genehmigungen nutzt, benötigt sie Instanzen, die berechtigt sind, solche Genehmigungen auszusprechen, und denen die Teilnehmer vertrauen, so dass gewisse zentrale Komponenten integriert sind. In einer privaten Blockchain, die genehmigungsbasiert ist, werden die Berechtigungen zentral vergeben.

Alternative Ansätze zur Konsensbildung

Bei dem oben skizzierten Ansatz des Proof-of-Work muss ein Miner-Knoten unter Umständen einen erheblichen Aufwand in Bezug auf Rechenleistung und Energie erbringen, um neue Blöcke der Kette hinzuzufügen und möglichst der erste zu sein, der eine passende Nonce findet. Dieses Vorgehen ist nicht für alle Anwendungsszenarien sinnvoll. Wenn das Szenario kei-

nen Wettbewerb vorsieht, wie dies beispielsweise in privaten Blockchains meist der Fall ist, ist es deshalb sinnvoller, auf alternative Konsensusverfahren aufzusetzen, die zum Beispiel speicher- oder netzwerkbasiert sind. Bei speicherbasierten Ansätzen ist zum Lösen des Rätsels eine Anzahl an Speicherzugriffen erforderlich, und bei einem netzwerkbasierten Ansatz ist eine Kommunikation mit anderen Knoten erforderlich, um beispielsweise Informationen zu erhalten, die zur Lösung des Rätsels benötigt werden.

PoS

Eine andere Vorgehensweise zur Konsensbildung ist der so genannte Proof-of-Stake (PoS). Dieses Vorgehen wird häufig in privaten Blockchains genutzt. Die Knoten, die einen neuen Block validieren können, werden hierbei nicht nach ihrer Rechenleistung ausgewählt, sondern nach ihren Anteilen (engl. *stake*), beispielsweise dem Anteil an der Kryptowährung, also in diesem Fall nach ihrem Reichtum. Ein Konsens ist erzielt, wenn die Mehrheit der Anteilsinhaber zum gleichen Ergebnis kommt (Proof-of-Stake). Der Ansatz beruht darauf, dass man davon ausgehen kann, dass diejenigen, die einen hohen Anteil an den jeweiligen Werten haben, die in der Blockchain verwaltet werden, ein intrinsisches Interesse daran haben, dass das System weiterhin korrekt betrieben wird. Alternativ können spezielle Knoten als Miner für die Konsensfindung ausgezeichnet werden, oder es kann über ein Zufallsverfahren (Lotterie) eine Auswahl erfolgen. Eine Kombination von Proof-of-Work- mit Proof-of-Stake-Verfahren ist zusätzlich möglich.

14.8.2 Smart Contracts

Smart Contracts sind Programme, die meist in C++, Java, Python oder Go geschrieben sind, und mit denen Verträge zwischen Partnern programmiersprachlich beschrieben und durchgesetzt werden können. Ein solcher Smart Contract formuliert analog zu einem papierbasierten Vertrag die vertraglichen Pflichten sowie möglichen Strafen bei Zuwiderhandlung, aber auch die Vorteile, die sich für die beteiligten Vertragspartner aus dem Vertrag ergeben.

Smart Contracts ermöglichen es Partnern, die anonym sein können, ohne die Hilfe von Mittelsmännern Geschäftsprozesse über das Internet nachvollziehbar und revisionssicher abzuwickeln. Das Konzept wurde bereits 1997 von Nick Szabo [171] eingeführt, hat aber durch die Blockchain-Technologie erst seine Bedeutung erlangt. Jeder Knoten im Netzwerk agiert dabei als eine Art Treuhänder, der die Vertragsbestimmungen, die in Smart Contracts spezifiziert sind, automatisch ausführt. Ein Smart Contract ist Bestandteil einer Transaktion in der Blockchain, dessen Programmcode im Zuge der Validierung der Transaktion durch die Peers ausgeführt wird.

Smart Contracts können beispielsweise Copyright-Lizenzen abbilden oder sie können auch genutzt werden, um Service-Level-Agreements (SLAs) umzusetzen. Eine mögliche Anwendung von Smart Contracts im IoT-Umfeld

wäre die Verteilung von Software-Updates an Millionen von IoT Geräte. Dazu kann der Gerätehersteller einen Smart Contract programmieren, mit dem der Hash des Firmware-Updates in der Blockchain gespeichert werden kann. Die IoT-Geräte können den Hashwert des aktuellen Updates aus der Blockchain entnehmen, das Update über das Peer-to-Peer-Netz herunterladen und es auch weiteren Geräten zum Download zur Verfügung stellen.

Die Skripte, die in Bitcoin verwendet werden, um Transaktionen zu spezifizieren (siehe Abschnitt 14.8.4), sind bereits sehr einfache Ausprägungen solcher Smart Contracts. Die derzeit bekannteste Plattform zur Ausführung von komplexen Smart Contracts ist Ethereum (siehe unten). Auch die Hyperledger-Blockchain erlaubt die Ausführung von komplexen Smart Contracts, genannt Chaincodes, als Bestandteile von Blockchain-Transaktionen. Der Code der Contracts wird von so genannten Validating Peers in einem Docker-Container direkt auf dem Prozessor des Peers ausgeführt. Während der Ausführung hat der Code Zugriff auf die gespeicherten Daten in der Blockchain.

Auf der Basis von Smart Contracts lassen sich auch neue unternehmerische Strukturen etablieren, die dezentralen autonomen Organisationen (DAO). In einer DAO sind die Governance-Strukturen, Unternehmens-Policies sowie Verwaltungs- und Buchhaltungsabläufe, wie Cash-Flows, mittels Smart Contracts festgelegt. Eine DAO benötigt keinen zentralen Vorstand, sondern die Smart Contracts werden im dezentral organisierten Blockchain-Netz ausgeführt und das Unternehmen wird durch das Kollektiv der stimmberechtigten Anteilsinhaber gelenkt. Ein eher berüchtigtes Beispiel für eine DAO ist der Venture Capital Fund *The DAO*, der auf der Basis von Ethereum implementiert wurde (siehe unten).

DAO

Ethereum

Ethereum ist das derzeit bekannteste System, das die Ausführung komplexer Smart Contracts ermöglicht. Ethereum basiert auf einer öffentlichen Blockchain. Die Plattform, die seit 2015 im Einsatz ist, verwendet Ether als Kryptowährung, mit der sich die Teilnehmer Rechenleistung einkaufen können. Man spricht in diesem Zusammenhang auch häufig von Gas als Bezahlung. Zur Programmierung von Smart Contracts bietet Ethereum eine eigene, turing-vollständige Programmiersprache namens Solidity. Zur verteilten Konsensbildung wird derzeit²⁶ noch ein Proof-of-Work-Algorithmus verwendet, eine Umstellung auf Proof-of-Stake ist geplant.

Ethereum

Für viel Aufsehen hat im Juni 2016 der DAO-Hack²⁷ gesorgt, der sich gegen die Ethereum-basierte Blockchain der DAO namens *The DAO* rich-

DAO-Hack

²⁶ Stand 2017

²⁷ David Siegel. Understanding The DAO Attack, <http://www.coindesk.com/understanding-dao-hack-journalists/>

tete. Die Anteile an der DAO wurden 2016 an Investoren in Form von Token veräußert, die das Stimmrecht der Investoren repräsentieren. Bei diesem Token-Sale gelang es einem Angreifer, eine Code-Schwachstelle im Quellcode eines Smart Contracts auszunutzen, um Kapital in eine Art Tochtergesellschaft umzuleiten, so dass die anderen Parteien keinen Zugriff mehr darauf hatten. Dem Angreifer gelang es auf diese Weise ca. 3,6 Millionen Ether zu stehlen. Da jedoch ein integrierter Sicherheitsmechanismus in Ethereum dafür sorgte, dass auf das Geld in solchen Tochtergesellschaften erst nach 27 Tagen zugegriffen werden darf, konnte mit dem Fork der Blockchain erreicht werden, dass diese Angriffstransaktionen ins Leere liefen. Der Fork erforderte den Konsens der Mehrheit der Teilnehmer und war sehr umstritten. Nach dem harten Fork entstand neben der Ethereum-Währung eine weitere Kryptowährung namens Ethereum Classic. Beide Währungen existieren seit dem Fork parallel.

Ethereum zeigt, dass Blockchain-Plattformen, die komplexe Smart Contracts ausführen, in hohem Maße anfällig dafür sind, über fehlerhaften Code angegriffen zu werden. Solche Angriffe können nur sehr schwer in ihren Auswirkungen begrenzt und gestoppt werden.

14.8.3 Sicherheit von Blockchains

Integrität,
Vertraulichkeit

Das Blockchain-Prinzip ist darauf ausgerichtet, die Integrität der in der Chain gespeicherten Daten mittels kryptografischer Hashfunktionen zu sichern, und durch das Verteilen der Informationen und den verteilten Konsensus die Integrität zu validieren. Demgegenüber ist die Gewährleistung der Vertraulichkeit kein Designziel der Blockchain. In öffentlichen Blockchains ist es sogar erforderlich, dass zumindest Teile der Transaktionsinformation im Klartext vorhanden ist, damit die Knoten die Transaktionen verifizieren können.

Zuordenbarkeit

Transaktionen, die in den Blöcken abgelegt werden, werden von den Teilnehmern signiert, so dass das Blockchain-Prinzip die Nachvollziehbarkeit und Zuordenbarkeit von Aktionen ermöglicht. Es ist aber nicht prinzipiell ausgeschlossen, dass ein Angreifer eine durchgeföhrte Transaktion im Nachhinein abstreitet. Präziser gesagt, streitet der Angreifer die Aktion nicht ab, sondern er entfernt sie aus der Kette und macht sie somit ungeschehen. Auf solche Angriffe, die einen Fork der Kette nach sich ziehen, gehen wir im Folgenden noch weiter ein.

Identität

Bitcoin ermöglicht eine anonyme Mitwirkung, indem die Identitäten, das sind die öffentlichen Schlüssel, häufig gewechselt werden. Wird jedoch eine nachweisliche Zuordnung zwischen einer digitalen und der realen Identität gefordert, was für viele Blockchain-Anwendungen der Fall sein wird, so werden auch bei der Blockchain wieder klassische Ansätze wie die Etablierung einer PKI, oder aber auch ein Web-of-Trust-Ansatz erforderlich.

Anwendungen, die mittels Blockchains eine Verbindung zwischen der digitalen und der physischen Welt herstellen, erfordern aber weitergehende Schutzkonzepte als lediglich die Absicherung der Transaktionen in den Blöcken der Kette. So kann ein Smart Contract dazu genutzt werden, Daten, die das physische Gerät bereitstellt, zu verarbeiten und Aktionen auf dem physischen Gerät zu initiieren. Beispiele solcher Aktionen könnten das Öffnen eines physischen Schlosses, zum Beispiel die Autotür eines Mietwagens, oder das Abschalten eines technischen Systems sein. Sind die Daten, die das Gerät bereitstellt, manipuliert, so ist das fehlerhafte Verhalten der Hardware-Komponente in der Blockchain nicht abgebildet und über die Ausführung von Smart Contracts nicht nachvollziehbar. Die physischen Geräte müssen also auch gegen Manipulation geschützt werden, damit die Blockchain-basierte verteilte Anwendung für alle Parteien vertrauenswürdig abgewickelt werden kann.

Physische Assets

51% Angriff

Die Sicherheit der Blockchain beruht darauf, dass kein Teilnehmer mehr als die Hälfte der Rechenleistung des Netzes auf sich vereint. Nehmen wir an, ein Angreifer hat eine Transaktion getätigt, zum Beispiel eine Bezahlung durchgeführt, und möchte nun diese Transaktion wieder aus der Blockchain löschen, nachdem er den Gegenwert für die Bezahlung erhalten hat. Er kann sich nun zunutze machen, dass die Teilnehmer der Blockchain stets die längere Kette als die gültige akzeptieren. Der Angreifer kann somit eine Verzweigung der Kette herbeiführen (fork), genau an dem Block, der der Vorgänger des Blocks mit der getätigten Transaktion in der Kette ist. Die in Abbildung 14.38 dargestellte Verzweigung der Kette würde beispielsweise durch einen Angreifer gezielt herbei geführt worden sein. Er kann einen Block ohne seine Überweisungstransaktion erstellen, sowie weitere Blöcke schürfen, bis seine Kette länger ist als die aktuell im Umlauf befindliche Kette. Sobald er diese längere Kette veröffentlicht, werden die Teilnehmer diese Kette übernehmen, und die ursprünglich getätigte Transaktion ist nicht mehr in der Kette enthalten. Interpretiert man den Fork aus Abbildung 14.38 als einen solchen Angriff, dann generiert der Angreifer die Kette mit dem Block 4, macht diesen Block bekannt und generiert zusätzlich (ggf. bereits vorab parallel) die Kette mit den Blöcken 3 und 5. Wenn er diese längere Kette in Umlauf bringt, wird diese Kette von den Teilnehmern übernommen und der Block 4 wird nicht mehr weiter in der Blockchain vertreten sein. Es ist bekannt, dass PoW-basierte Blockchains besonders in ihrer Anfangsphase anfällig gegen solche Angriffe sind, da ein Miner-Knoten noch nicht über allzu große Rechenleistung verfügen muss, um die bestehende Kette abzuändern und seine längere Kette einzuspielen. Deshalb kann man zum Start einer neuen Blockchain kein hohes Vertrauen in die Persistenz der getätigten Transaktionen haben.

PoW-Problem

51% Angriff

Damit der Angreifer einen solchen Angriff erfolgreich durchführen kann, muss er über eine sehr hohe Rechenleistung verfügen, so dass er in der Lage ist, neue Blöcke schneller zu schürfen als alle anderen Knoten im Netz zusammen. Nakamoto hat in seinem Whitepaper [127] gezeigt, dass die Wahrscheinlichkeit, dass ein Angreifer, der weniger als 50% der Rechenleistung des Netzwerkes besitzt, einen erfolgreichen Angriff durchführen kann, mit zunehmender Länge der Kette exponentiell sinkt. Das bedeutet, dass die Wahrscheinlichkeit, dass es einem Angreifer oder einem Kartell von kooperierenden Angreifern gelingt, einen Fork der Tiefe n zu berechnen, von einer Komplexität $O(2^n)$ ist.

Confirmation

Der skizzierte Angriff und dessen Eintrittswahrscheinlichkeit hat zur Konsequenz, dass sich ein Teilnehmer nie ganz sicher sein kann, dass seine Transaktion persistent in der Kette aufgenommen ist. Blöcke, die an die Kette mit seiner Transaktion angefügt werden, bestätigen (confirm) die Transaktion. Ab einer Anzahl von angefügten Blöcken kann man mit einem ausreichend hohen Vertrauen (confidence) davon ausgehen, dass die Transaktion bestätigt ist, da der Rechenaufwand, der benötigt wird, um die Transaktion rückgängig zu machen, exponentiell in der Anzahl der angefügten Blöcke steigt. Wie bereits weiter oben angeführt, geht man in Bitcoin davon aus, dass nach sechs hinzugefügten Blöcken eine genügend hohe Zuschicherung vorhanden ist, dass die Blöcke persistent in der Kette bleiben. Das heißt je mehr Blöcke in der Kette nach einer Transaktion eingefügt sind, desto höher ist die Gewissheit, dass eine Transaktion nicht mehr rückgängig gemacht werden kann.

Problem: Pooling

Nakamoto zeigt auch, dass ein Angreifer, der über 10% der Rechenleistung des Netzes verfügt, nach sechs Erweiterungen der Kette um neue Blöcke (sechs Bestätigungen einer Transaktion), nur noch mit einer Wahrscheinlichkeit von $p = 0.00059$ eine erfolgreiche Attacke durchführen kann. Da auch bereits die hier angenommenen 10% der gesamten Rechenpower einen sehr großen Wert darstellen, hat Nakamoto die Wahrscheinlichkeit, dass der Angreifer das Wettrennen um die längste Kette gewinnen wird, als sehr gering eingeschätzt. Angesichts des sich unter Bitcoin abzeichnenden Trends zum Poolen von Rechenleistungen, um das Mining durch eine Gruppe von Minern effizienter durchzuführen, könnten sich jedoch in Bezug auf die Sicherstellung der Integrität der Blockchain neue Probleme ergeben, wenn Miner-Knoten gezielt an Angriffen auf die Kette zusammenarbeiten.

Herausforderung

Smart Contracts sind nicht ohne weiteres änderbar, so dass ein Updaten oder Patchen von Programmen, wie man es aus heutigen Systemen kennt, bei Smart Contracts kaum möglich ist, da dies ja eine nachträgliche Änderung von Einträgen in der Blockchain bedeuten würde. Die Contracts sind damit zwar revisionssicher, aber gleichzeitig kann auch Contract-Code, der Schad-

funktionalität enthält oder verwundbar ist, nicht entfernt oder korrigiert werden. So können im Code eines Smart Contracts auch Aufrufe externer Dienste enthalten sein, so dass der Code unerwünschte Funktionen ausführen kann, wie beispielsweise SPAM versenden. Der Code kann aber auch den ausführenden Peer direkt kompromittieren. Beispielsweise kann ein Contract-Code bei der Ausführung durch den validierenden Peer dessen CPU Ressourcen blockieren und zu einem Denial-of-Service auf den Peer führen. Die korrekte Programmierung solcher Contracts und formale Verifikation des Codes ist somit von hoher Relevanz für deren praktische Nutzbarkeit im Geschäftsumfeld. Ethereum bietet mit dem why3-Framework sogar schon eine solche Unterstützung für formale Verifikationen von Contracts, doch erfordert dessen Nutzung Expertenwissen und ist aufwändig, so dass es in der Praxis kaum genutzt wird.

14.8.4 Fallbeispiel: Bitcoin

Da Bitcoin [185] die mit Abstand bekannteste Anwendung der Blockchain-Technologie ist, stellen wir das Bitcoin-System nachfolgend kurz vor. Einige Aspekte sind ja bereits in der allgemeinen Darstellung zu Blockchain aufgegriffen worden.

Bitcoin ist eine digitale Kryptowährung, die auf einem Peer-to-Peer-Netz basiert. Bitcoin wurde von Satoshi Nakamoto 2008 in seinem Whitepaper [127] vorgestellt und ist seit 2009 als digitale Währung als Open-Source-Software umgesetzt und im Umlauf. Nachdem das Netz anfangs nur langsam gewachsen ist, haben die Bitcoins mittlerweile eine sehr große Verbreitung und werden auch zunehmend als Zahlungsmittel akzeptiert. An Geldautomaten können Bitcoins gegen Bargeld getauscht werden, mit Bitcoins kann an Bezahlterminals bargeldlos bezahlt werden und es gibt verschiedene Apps, die als Bitcoin-Geldbörsen dienen, um die Nutzung zu vereinfachen. Leider hat auch die organisierte Kriminalität bereits Bitcoin als anonymes Zahlungsmittel entdeckt, so dass beispielsweise Ransomware-Programmierer Bitcoins als Bezahlung einfordern.

Bitcoin kann man als ein Zustands-Transitions-System modellieren, wobei ein Zustand beschreibt, welche Bitcoins in wessen Besitz sind (Ownership). Präziser gesagt beschreibt ein Zustand die Menge aller Münzen der Eingaben und Ausgaben einer Transaktion (vgl. Abbildung 14.39). In Bitcoin spricht man auch von den nicht ausgegebenen Transaktions-Outputs (unspent transaction outputs bzw. UTXO). Jeder UTXO besitzt einen Wert und einen Eigentümer, der über eine 20-Byte Adresse identifiziert ist. Dies ist der Hashwert eines öffentlichen Schlüssels eines Public-Key-Paars.

Zustand-
Transitions-
System

Die Zustandsübergangsfunktion spezifiziert für einen gegebenen Zustand und eine spezifizierte Transaktion einen Nachfolgezustand. Eine Transak-

tion umfasst mindestens eine Eingabe. Jede Eingabe enthält einen Verweis auf die Ausgabe einer vorherigen Transaktion, und eine digitale Signatur (ECDSA) vom Eigentümer der UTXO über den Hashwert (SHA-256) des Eingabe-Records. Die Transaktion enthält zudem mindestens eine Ausgabe, wobei jeder Output einen UTXO umfasst und im einfachsten Fall den Hashwert des öffentlichen Schlüssels des designierten Empfängers des Geldes enthält. Abbildung 14.39 veranschaulicht eine Folge von Transaktionen. Geldtransfers erfolgen in Bitcoin somit nicht in Form von realen oder auch digitalen Münzen, sondern durch Eintragungen im verteilten Hauptbuch, der Blockchain von Bitcoin. Möchte ein Teilnehmer Geld ausgeben, dann muss er den entsprechenden Betrag über frühere Transaktionen virtuell überwiesen bekommen haben. Das erfolgt über Outputs früherer Transaktionen, die in der zu erstellenden Überweisungs-Transaktion als Input auftreten. Diese Überweisung wird unter Bitcoin dadurch realisiert, dass der Teilnehmer den Betrag, den er überweisen möchte, signiert und über eine Ausgabe in einer Transaktion formuliert. Der gewünschte Empfänger des Geldes (ohne dass Geld wirklich fließt), wird über den Hashwert des öffentlichen Schlüssels des Empfängers identifiziert. Bei Bitcoin wird eine Geldeinheit namens Satoshis verwendet. In Abbildung 14.39 wurde aus Gründen der Übersichtlichkeit darauf verzichtet, Signaturen und öffentliche Schlüssel aufzunehmen. Die Geldeingänge sind in den jeweiligen Inputs der Transaktion festgehalten, die Geldausgänge in den Outputs. So beschreibt die Transaktion 1 beispielsweise einen virtuellen Geldtransfer von 30k Einheiten, wobei, wie gesagt, in der Abbildung der öffentliche Schlüssel des Empfängers im Output nicht angegeben ist. Ebenfalls ist nicht angegeben, dass die Einheit 30k vom Geldgeber digital signiert ist.

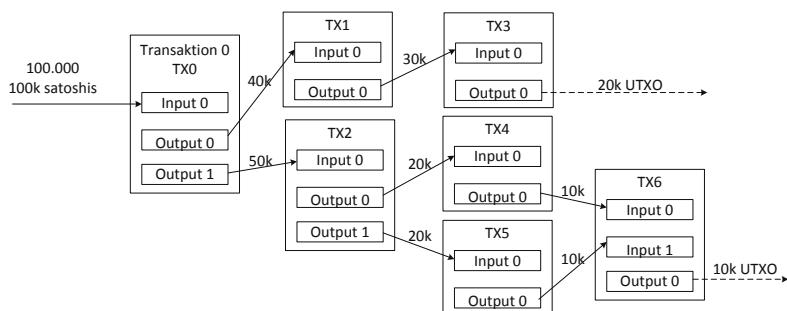


Abbildung 14.39: Transaktionen in Bitcoin

In dem Beispiel wird in jeder Transaktion eine Transaktionsgebühr von 10k entrichtet, da gilt: Summe der Output-Werte = Summe der Input-Werte - 10k.

Möchte der designierte Empfänger den Betrag im Output-Record verwenden (als Input in einer seiner nächsten Transaktionen), so muss er nachweisen, dass er den privaten Schlüssel besitzt, der zu dem gehaschten öffentlichen Schlüssel im Output der Transaktion gehört, durch die der Betrag überwiesen wurde.

Ein Output-Record kann noch weitere Festlegungen treffen, die gelten müssen, wenn der Ausgabewert weiterverwendet werden soll. Dies wird über Skripte spezifiziert. So kann ein Skript beispielsweise die Vorlage mehrerer Signaturen erfordern. Die in Blockchain genutzte Skriptsprache ist sehr einfach und unterstützt beispielsweise keine Schleifenkonstrukte, so dass die Ausführung der Skripte im Zuge der Validierung der Transaktion stets terminiert.

Skript

Die Zustandsübergangsfunktion prüft die Signatur, so dass also nur der berechtigte Besitzer den Geldbetrag transferiert, und prüft, ob die Summe der Eingabe-UTXO die Zahl der aktuell noch nicht ausgegebenen UTXO nicht übersteigt. Das heißt es wird geprüft, ob gegebenenfalls bereits ausgegebene Münzen erneut ausgegeben werden. Zudem wird geprüft, ob die Summe der UXTO in den Ausgaben nicht die Summe der Eingaben übersteigt. Im Nachfolgezustand der Transaktion sind die Eingabe-UTXO entfernt und die Ausgabe-UTXO hinzugefügt. Falls die Summe der Werte der Eingaben der Transaktion die Werte der Ausgaben übersteigt, dann ist der Differenzbetrag die Transaktionsgebühr, die derjenige behalten darf, der als erstes einen Block, der die Transaktion enthält, in die Blockchain einfügt, also den PoW erfolgreich berechnet.

In einem klassischen Bankenumfeld würde dann der Zustand die Bilanz darstellen. Eine Transaktion entspricht einem Überweisungsauftrag an die Bank. In dem Auftrag wird festgelegt, dass die Bank einen Wert x von A zu B transferieren soll, und durch die Zustandsübergangsfunktion wird das Konto A um den Wert x reduziert sowie das Konto B um den Wert erhöht.

Eine Eigenschaft, die diese digitale Währung auszeichnet ist die Möglichkeit, schnell und ohne hohe Transaktionskosten Geldtransfers und Bezahlungen über das Internet abzuwickeln. Dabei können die Teilnehmer weitestgehend anonym agieren und Transaktionen, wenn sie einmal bestätigt sind, können mit hoher Wahrscheinlichkeit nicht mehr geleugnet oder geändert werden. Es gibt keine zentrale Instanz, sei es eine Bank oder eine staatliche Stelle, die den Zugang zum Bitcoin-Netzwerk reglementiert. Bitcoin basiert auf einer öffentlichen, genehmigungslosen Blockchain. Die maximale Anzahl an Bitcoins, die jemals im Umlauf sein dürfen, ist mit 21 Millionen Bitcoins nach oben gedeckelt. Bitcoin's wichtiger Beitrag für digitale Währungen ist die Lösung für das Double-Spending-Problem und das

Eigenschaften

Problem der Byzantinischen Generäle [107]. Das Double-Spending-Problem ist eine zentrale Herausforderung digitaler Währungen, da sich Bitstrings leicht kopieren lassen und digitale Münzen mehrfach ausgegeben werden können. Mit dem Proof-of-Work und dem verteilten Konsensusverfahren (siehe oben), löst Bitcoin das Problem so, dass auch nicht vertrauenswürdige und anonyme Knoten im Netz agieren können.

Ledger

Der klassische Ansatz zur Lösung des Double-Spending-Problems besteht darin, in einem zentralen Hauptbuch (Ledger) festzuhalten, welcher Teilnehmer über welche Beträge verfügt, und Geldtransaktionen als Abbuchungen bzw. Zugänge auf die jeweiligen Konten zu verwalten; das heißt eine physische Repräsentation digitaler Münzen ist nicht erforderlich. Herkömmlich wird das Hauptbuch von einer zentralen Instanz verwaltet, die alle Transaktionen überwacht und damit das gesamte Netz kontrolliert.

Distributed Ledger

In Bitcoin wird diese zentrale Instanz vollständig ersetzt durch die dezentral agierenden Miner-Knoten, die über ein Peer-to-Peer-Netz kooperieren. Die Miner überwachen wechselseitig ihre Transaktionen im Hinblick auf die korrekte Einhaltung der Regeln. Das Hauptbuch wird als verteilte Datenbank durch die Blockchain implementiert. Die Datenbank wird dabei nicht aufgeteilt, sondern jeder Teilnehmer verwaltet eine lokale Kopie. Die Information ist damit massiv redundant auf allen Knoten gespeichert.

Bitcoin ist ein offenes System, jeder kann über das TCP-Protokoll und mit der Kenntnis von IP-Adressen mit anderen Teilnehmern des Netzes kommunizieren. Da jeder Teilnehmer eine Kopie der Datenbank verwaltet, müssen Transaktionen an alle Teilnehmer des Netzwerks versandt werden. Nachrichten werden dafür digital vom Absender signiert und die Empfänger prüfen die Gültigkeit der Signatur. Eine Konsequenz des Vorgehens ist, dass alle Bitcoin-Transaktionen öffentlich sind. Die Blockchain bei Bitcoin archiviert alle getätigten Geldtransaktionen seit der Start-Transaktion, dem so genannten Genesis-Block. Die in Blöcken gespeicherten Transaktionen sind so strukturiert, dass geprüft werden kann, dass ein Teilnehmer nur Bitcoins überweist, die er vorher einmal selbst erhalten hat.

PoW

Bitcoin verwendet das Proof-of-Work-Verfahren, siehe Seite 831, um neue Blöcke zu erzeugen. Das generelle Prinzip, wie der PoW bei Bitcoin berechnet wird, wurde bereits dargestellt. Bitcoin verwendet SHA-256 als Hashfunktion, so dass die Berechnung des PoW nur durch Try-and-Error erfolgen kann. Das heißt, ein Miner-Knoten wählt eineNonce n , berechnet den PoW und prüft, ob der berechnete Hashwert die Vorgaben des PoW erfüllt. Sollte das nicht der Fall sein, wird dieNonce inkrementiert und erneut die Berechnung durchgeführt. Sollten bei der parallelen Berechnung von neuen Blöcken durch Miner-Knoten Konflikte auftreten, so ist festgelegt, dass die längste Kette gewinnt. Damit wird ein Konsens herbeigeführt, welcher

Block gültig und welcher zu verwerfen ist. Falls eine Transaktion in einem Block der Blockchain enthalten ist, wird diese Transaktion indirekt durch die Nachfolgeblöcke in der Kette bestätigt, da die Nachfolgeblöcke mit dem Hashwert des Blocks der Transaktion arbeiten. Nach sechs Blöcken ist eine hohe Gewissheit erreicht, dass die Transaktion nicht mehr aus der Kette entfernt wird, weil sie in einer längeren Blockchain nicht mehr auftritt.

Da Bitcoin auf dem Proof-of-Work-Verfahren beruht, ist es aber auch anfällig gegenüber Angreifern, die mehr als 50% der Rechenleistung des Netzes auf sich vereinen. Diese sind in der Lage, eine Verzweigung der Blockchain zu erzeugen und zu einer längeren Kette als die aktuell von der Mehrheit der Miner genutzte Kette auszubauen. Eine Transaktion kann somit trotz erfolgter mehrfacher Bestätigung durch Nachfolgeblöcke nach der Veröffentlichung der längeren Kette aus der Blockchain bei Bitcoin entfernt worden sein.

Wie bereits ausgeführt, sind alle Transaktionen bei Bitcoin für alle anderen Teilnehmer sichtbar, Vertraulichkeit ist damit nicht gewährleistet. Teilnehmer können eine eindeutige Adresse und damit Identität für jede Transaktion generieren, um ihre wirkliche Identität zu verschleiern, jedoch haben Analysen gezeigt, dass eine vollständige Anonymisierung nicht möglich ist. Problematisch bei Bitcoin könnte auch werden, dass es SHA-256 hardcodiert nutzt. Falls Angriffe auf die Hashfunktion bekannt werden, ist keine Kryptoagilität gegeben.

Vertraulichkeit,
Anonymität

Das größte Problem bei Bitcoin ist sicherlich seine mangelhafte Skalierbarkeit und die immense Redundanz, würde wirklich jeder Teilnehmer eine vollständige Kopie der Blockchain speichern. Dies ist in der Praxis jedoch nicht erforderlich, da es leichtgewichtige Protokollvarianten gibt, die es Teilnehmern ermöglichen, nur Ausschnitte der Kette lokal zu speichern.

Für das korrekte Funktionieren von Bitcoin ist ein zuverlässiges Netz, in dem die erforderlichen Broadcasts mit geringem Delay versandt werden, sehr wichtig. Angreifer, die diese Kommunikation gezielt unterwandern, könnten dafür sorgen, dass die von ihnen erstellte und veröffentlichte Blockchain die längste ist. Damit könnten sie zum einen die Belohnung für die Erstellung erhalten, während die anderen Miner ihre Rechenzeit und Energie verschwendet haben und zudem auch nicht sicher sein können, ob die Transaktionen, die sie in den Blöcken zusammengefasst haben, bestätigt sind. Entsprechende Delay-Angriffe, die einen Fork zur Folge haben, sind sogar mit einer Rechenleistung kleiner als 50% (vgl. [47]) möglich.

Auch wenn der Bitcoin-Ansatz im Kern dezentral ausgelegt ist, ist ein Trend zu beobachten, dass das Schürfen von Bitcoins sich immer stärker auf kleine Miner-Gruppen (Pools) und Organisationen, die zusammen über immense Rechenleistung verfügen, konzentriert und damit die Gefahr der Mono-

Monopolisierung

polisierung und zentralen Steuerung steigt. Schliessen sich beispielsweise Miner in einem Kartell zusammen, um die Blockchain zu monopolisieren, so könnten sie die Veröffentlichung von Blöcken anderer Miner ignorieren, konsequent an der eigenen Blockchainteile weiter arbeiten und diese solange verlängern, bis sie die längste Kette ist, die dann veröffentlicht wird. Da alle kürzeren Ketten und deren Blöcke damit automatisch ungültig werden, kann ein solches Kartell nicht nur die Belohnung für das Schürfen der Blöcke einstreichen, sondern, was deutlich gravierender ist, ein solches Kartell kann bestimmen, welche Transaktionen persistent in der Kette aufgenommen werden und welche durch den gezielt herbeigeführten Fork als unbestätigte Transaktionen nicht mehr gültig sind.

Zur Berechnung von Proof-of-Work werden zudem ganz erhebliche Energie-Ressourcen eingesetzt. Diese fehlende Ressourcennachhaltigkeit ist ein zunehmendes Problem für die Blockchain. Es bleibt abzuwarten, wie sich diese digitale Währung weiter entwickeln wird.

14.8.5 Fazit und kritische Einordnung

Blockchain ist eine interessante Technologie, mit der man, etwas verkürzt dargestellt, Datenblöcke, die Transaktionen repräsentieren, in eine Reihenfolge bringen kann, die nachvollziehbar ist. Die Reihenfolge (Historie) und die gespeicherten Datenblöcke sind nicht veränderbar und es ist nicht erforderlich, dass die handelnden Akteure vertrauenswürdig sind. Kryptowährungen wie Bitcoin sind sehr bekannte Anwendungen der Blockchain-Technologie, aber Blockchain kann weit mehr Anwendungsfelder bedienen als lediglich das Feld der Kryptowährungen, insbesondere wenn die Technologie in einer genehmigungsbasierten (permissioned) Variante genutzt wird.

Man unterscheidet unterschiedliche Ausprägungen von Blockchains, die einen unterschiedlichen Grad an Zentralisierung bedeuten. So sind öffentliche und gleichzeitig genehmigungslose (permissionless) Blockchains vollständig dezentral organisiert und erfordern verteilte Protokolle zur Konsensbildung. Private Blockchains sind immer genehmigungsbasiert und erfordern eine zentrale Instanz. Eine Zwischenform bilden öffentliche Blockchains, die genehmigungsbasiert sind, die also die Berechtigungen für den Zugriff regeln, aber dennoch noch ein hohes Maß an verteilter Konsensbildung erfordern.

Gerade weil um die Blockchain-Technologie ein großer Hype entbrannt ist, und man sich durch diese Technologie viele neue Anwendungen und auch Geschäftsmodelle verspricht, sollte man den kritischen Blick darauf nicht verlieren. Unterzieht man beispielsweise die weit verbreitete öffentliche Blockchain Ethereum einem Fakten-Check, so wird man feststellen können, dass über 56 % aller Blöcke von lediglich drei Minern generiert

werden²⁸, was zu einer gewissen Zentralisierung führt und im Gegensatz zur postulierten Dezentralisierung der öffentlichen Kette steht. Die Ethereum-Chain ist aktuell (Quartal 1 2018) größer als 330 GB. Da das Herunterladen der ganzen Historie notwendig ist, wenn man Inhalte validieren will, wird dieser Aufwand sicherlich nicht von jedem potentiellen Teilnehmer geleistet werden können. Neue Teilnehmer, die dem Netzwerk als Knoten beitreten, werden wahrscheinlich zunehmen darauf verzichten, den erforderlichen Aufwand zu treiben, mit der Konsequenz, dass sie der Korrektheit der Daten vertrauen, ohne diese selber zu validieren. Dies ist eine Aufweichung des Prinzips der öffentlichen Kette, dass jeder Teilnehmer die Korrektheit der Kette validieren können muss. Auch das Postulat der Unveränderlichkeit der gespeicherten Historie ist kritisch zu bewerten. Weiter oben wurde bereits dargestellt, dass es schon Hard-Forks gegeben hat. Der Datenschutz kann bei öffentlichen Ketten ebenfalls zu Problemen führen, da konzeptuell jeder Teilnehmer alle Daten in der Kette sehen kann. Dabei könnte es sich auch um Daten handeln, die beispielsweise zum Zeitpunkt ihrer Aufnahme in die Blockchain noch keiner gesetzlichen Auflage unterlagen, die aber zu einem späteren Zeitpunkt neuen gesetzlichen Anforderungen zur z.B. Löschung unterliegen. Dies könnte ab Juni 2018 auf Datenblöcke zutreffen, die unter die neuen Anforderungen der EU-Datenschutzgrundverordnung (GDPR) fallen und deshalb beispielsweise gelöscht werden müssten. Ein Löschen ist aber konzeptuell wegen der Unveränderlichkeit der Historie nicht vorgesehen. Als Lösung bietet sich an, in der Blockchain keine sensiven Daten zu speichern, sondern nur Verweise auf (zentralen) Speichermedien. Dies zieht jedoch direkt das Problem nach sich, dass dann nicht mehr unmittelbar von den Teilnehmern geprüft werden kann, dass die eigentlichen Daten bei dieser externen Speicherung noch manipulationssicher sind.

In vielen Anwendungsbereichen ist der Einsatz von Blockchains zur Lösung eines Anwendungsproblems nicht sinnvoll. Abbildung 14.40 fasst in übersichtlicher Weise zusammen, unter welchen Rahmenbedingungen auf den Einsatz von Blockchains verzichten werden kann. Das dargestellte Flussdiagramm lehnt sich an das Papier von Wüst und Gervais an²⁹. Eine genehmigungsfreie, öffentliche Blockchain erscheint in solchen Szenarien sinnvoll einsetzbar zu sein, in denen eine Datenbank verwaltet werden muss, auf die mehrere Schreiber einen Zugriff benötigen, um neue Datenblöcke zu speichern, und wenn zudem keine zentrale, vertrauenswürdige Instanz kontinuierlich online verfügbar ist, aber gleichzeitig alle Schreiber unbekannt sind. Demgegenüber könnte die Nutzung einer öffentlichen, genehmigungsbasierten Blockchain sinnvoll sein, wenn zwar alle Schreiber bekannt sind,

Sinnvoller Einsatz?

²⁸ <https://etherscan.io>

²⁹ Do you need a Blockchain? ETH Zürich, 2017

diese aber nicht vertrauenswürdig sind und gleichzeitig eine öffentliche Validierung erforderlich ist. Hingegen erscheint eine private und damit auch genehmigungsisierte Blockchain sinnvoll, wenn letzteres nicht gilt, also keine öffentliche Validierung erforderlich ist. In allen anderen Fällen erscheint ein Einsatz von Blockchains wenig sinnvoll, insbesondere gilt dies auch, wenn die Anwendung einen hohen Datendurchsatz erfordert und nur geringe Verzögerungszeiten toleriert werden können.

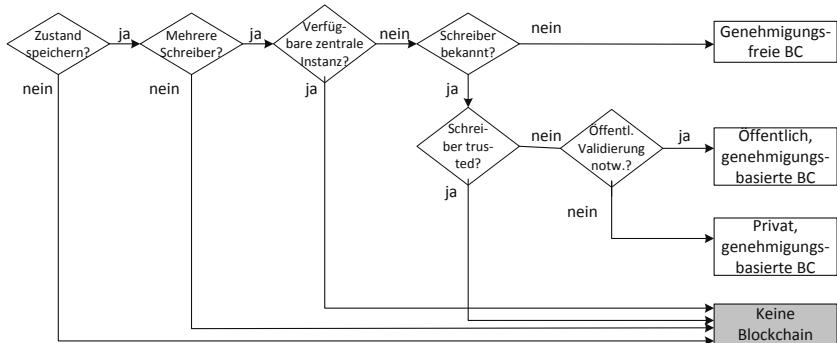


Abbildung 14.40: Wann ist eine Blockchain sinnvoll?

Missverständnisse

Abschließend gehen wir noch auf einige weit verbreitete Missverständnisse im Zusammenhang mit Blockchains ein.

- Häufig begegnet man der Aussage, dass alle Daten in der Blockchain vertrauenswürdig und korrekt seien. Da der Wahrheitsgehalt der Daten jedoch vor deren Speicherung in der Chain nicht notwendigerweise geprüft wird, kann so eine Annahme nicht per se getroffen werden.
- Auch die Aussage, dass Blockchains einen dezentralen Trust realisieren und kein Meinungsmonopol auftritt, wird durch die in der Praxis in der Regel sehr aufwändige Konsensbildung eher widerlegt, da diese Konsensbildungsvfahren dazu neigen, Monopole auszubilden. Diese Monopole, wie dies beispielsweise in der Finanzindustrie bei Kryptowährungen zu beobachten sind, agieren völlig unkontrolliert außerhalb staatlicher Strukturen und Ordnungsrahmen.
- Die Daten in der Blockchain sind permanent. Dies ist eine weitere häufig gehörte Aussage. Da die Miner entscheiden, welche Transaktionen langfristig übernommen werden, kann es sein, dass eine Transaktion doch irgendwann einmal nicht mehr in der längsten Kette steht und damit nicht persistent wird.

- Eng verknüpft mit der Blockchain ist häufig ein implizites Transparenzversprechen, das impliziert, dass die Blockchain-Daten von Jedermann validierbar sind. Eine Validierung durch beliebige Teilnehmer ist konzeptuell nur in öffentlichen, genehmigungsfreien Ketten möglich. Jedoch wird die schnell wachsende Größe von Blockchains, wie oben bereits ausgeführt, mittel- bis langfristig dazu führen, dass diese Validierungsmöglichkeit aus Gründen des damit verbundenen Aufwands nicht wahrgenommen werden kann.
- Vielfach wird eine Blockchain auch mit der Anonymität der Teilnehmer gleich gesetzt und postuliert, dass die Blockchain-Technologie die Privatsphäre garantiert. In einer öffentlichen, genehmigungsfreien Blockchain können Teilnehmer tatsächlich anonym agieren, weshalb ja auch Kryptowährungen wie Bitcoin zunehmend für kriminelle Aktivitäten wie Erpressungen verwendet werden. Bei genehmigungsbasierten Blockchains müssen jedoch die Teilnehmer notwendigerweise identifiziert werden können, um die Genehmigung zu erteilen oder zu verweigern.
- Noch weitergehender als die Aussage zum Privatsphärenschutz sind Aussagen, die postulieren, dass eine Blockchain die Sicherheit erhöht. Eine so weitreichende Aussage kann sicherlich nicht generell getroffen werden. Wie weiter oben ausgeführt, können fehlerhafte Smart Contracts sogar nicht unerhebliche neue Sicherheitsprobleme nach sich ziehen. Falls in einer genehmigungsbasierten Blockchain starke Identitäten und starke Authentisierungsprüfungen durchgeführt werden und auch die zentrale Instanz, die die Berechtigungen vergibt und deren Einhaltung kontrolliert, vertrauenswürdig ist, so liefert dies einen Sicherheitszugewinn. Wenn zusätzlich auch die in die Blockchain aufzunehmenden Datenblöcke vor ihrer Aufnahme auf Korrektheit, z.B. die Korrektheit des Codes, geprüft werden, ist noch ein höherer Sicherheitszugewinn möglich.

Zusammenfassen kann man festhalten, dass die Blockchain-Technologie viele alt bekannte Sicherheitskonzepte geschickt aufgreift und sicherlich viel Potential für interessante neue Anwendungen bietet, um zentrale Strukturen abzulösen. Die kritische Einordnung sollte aber auch gezeigt haben, dass eine Blockchain auch erheblichen Overhead verursachen kann, der sich nicht für alle derzeit diskutierten Anwendungen wirklich lohnt.

15 Sichere mobile und drahtlose Kommunikation

Die mobile und drahtlose Kommunikation ist aus dem heutigen beruflichen wie privaten Alltag kaum noch wegzudenken. Mobilfunknetze überspannen Länder und Kontinente und ermöglichen so eine nahezu ungebrochene Erreichbarkeit von Kommunikationspartnern. Für den Nahverkehrsbereich wird eine drahtlose Kommunikation eingesetzt, die die drahtgebundene Kommunikation ergänzt. Abschnitt 15.1 erläutert ausführlich den Aufbau und die Sicherheitsdienste des GSM-Systems und erklärt deren Schwächen. Abschnitt 15.1.6 geht anschließend kurz auf die GPRS-Architektur (General Packet Radio Service) ein, die auf der GSM-Architektur aufsetzt und eine ersten Schritt in die Richtung eines Paketvermittelnden (IP) Mobilfunksystems darstellt. Das GSM-System zählt zu den Mobilfunksystemen der 2ten Generation (2G), die durch die Systeme der dritten Generation (3G), wozu auch die UMTS-Systeme zählen, zwar noch nicht vollständig abgelöst, aber zumindest ergänzt werden. Abschnitt 15.2 erläutert die UMTS-Sicherheitsarchitektur. Abschnitt 15.3 geht anschließend auf den aktuellen Mobilfunkstandard der vierten Generation, dem LTE-Standard (Long Term Evolution) ein. LTE zielt darauf ab, das paketvermittelnde Kern-Netz der 2G bzw. 3G-Mobilfunknetze zu einem sehr leistungsfähigen All-IP-Netz weiterzuentwickeln.

Mit Abschnitt 15.4 erfolgt der Einstieg in die Thematik der Sicherheit in drahtlosen lokalen Netzen, den WLANs. Häufig benötigt man aber noch nicht einmal die Ausdehnungen und Entfernung eines LANs zur Kommunikation, sondern ist mit einer schnellen, spontan zu etablierenden drahtlosen Verbindung über kurze Distanzen (5 bis 10 Meter) zufrieden, wie beispielsweise bei der Datenübertragung zwischen einem Laptop und einem Drucker oder Mobiltelefon. Diese Funktionalität wird von dem Bluetooth-Protokoll bereitgestellt, auf das Abschnitt 15.5 detailliert eingeht. Das Internet of Things (IoT) nimmt zunehmend Gestalt an und erfordert energieeffiziente Kommunikationsprotokolle. Mit ZigBee steht ein Standard für die energiesparende Funkkommunikation über kurze Distanzen zur Verfügung, der für die Vernetzung von IoT-Sensoren und -Geräten nutzbar ist. Eine Darstellung

der ZigBee-Protokolle und deren Sicherheitsarchitektur findet sich in Abschnitt 15.6.

15.1 GSM

Hintergrund

Das GSM-System (Global System for Mobile Communication) war die erste weltweite Anwendung chipkartenbasierter Authentifikation. Das GSM-System kann man heutzutage als die größte Sicherheitsinfrastruktur weltweit betrachten. Das GSM ist ein paneuropäischer Mobilfunkstandard, der von 1982 bis 1990 entwickelt wurde. Die zellularen GSM-Netze überspannen heute bereits alle Kontinente. Im Juli 2018 verzeichnete die GSM Association¹ über 5.1 Milliarden registrierte GSM-Nutzer und über 8,6 Milliarden Verbindungen (einschließlich M2M) weltweit.

15.1.1 Grundlagen

Dienste

GSM bietet als Standarddienste die Sprachübertragung, wozu die Telefonie und der Notruf zählen, sowie Datendienste, wie den Short Message Service (SMS) mit Texten der Länge von 160 Zeichen, E-Mail oder FAX-Übertragung. Die Datenrate von GSM liegt bei 9.6 Kb/s. Durch fortgeschrittene Kanalcodierungsverfahren sind mittlerweile auch Raten bis 14.4 Kb/s möglich. Für heutige Anwendungen im Bereich der Multimediadaten ist dies aber noch immer zu wenig. Daneben gibt es eine ganze Reihe von Zusatzdiensten (engl. *supplementary services*), wozu u.a. die Anzeige der Nummer des Anrufers, die Anrufweiterleitung, Konferenzschaltungen oder verschiedene Sperren gehören. Ferner bietet GSM eine Vielzahl so genannter Mehrwertdienste (engl. *value added services*) wie u.a. Hot-Line, Reiseservice, Verkehrsinformationen oder auch Pannenhilfe. Das Herz eines jeden GSM-Mobiltelefons ist die SIM-Karte (Subscriber Identity Module). Das ist eine Smart Card, die in einen Kartenleser innerhalb des Mobiltelefons eingelegt wird und die für die Sicherheit wichtigen Daten und Algorithmen speichert.

Sicherheitsdienste

Die Integration von Sicherheitsdiensten war ein explizites Designziel bei der GSM-Entwicklung, wobei der Schutz der Luftschnittstelle im Mittelpunkt stand. Die Luftschnittstelle ist die Strecke, die per Funk überbrückt werden muss, bis die Daten in ein Festnetz eingespeist werden. Das Ziel war, einen Schutz vor unautorisiertem Telefonieren sowie vor illegalem Abhören zu gewährleisten. Da es in Mobilfunknetzen keine festen, im Voraus bekannten Verbindungen zwischen einer lokalen Vermittlungsstelle und einem mobilen Teilnehmer gibt, ist die korrekte Identifikation eines Teilnehmers eine wesentliche Voraussetzung dafür, Gebührenabrechnungen durchzuführen und

¹ Vgl. <http://www.gsmworld.com>

Verbindungen korrekt zu vermitteln. Zur Erfüllung dieser Anforderungen wird sowohl eine chipkartenbasierte Authentifikation unter Einsatz der SIM-Karte, als auch eine Verschlüsselung der Gesprächsdaten durchgeführt. Um die Aufenthaltsorte von GSM-Teilnehmern zu verschleiern und somit eine gewisse Anonymisierung zu erreichen, verwendet das GSM temporäre Teilnehmerkennungen. Die Sicherheitsdienste werden durch drei Systemkomponenten der GSM-Architektur realisiert, nämlich die SIM-Karte, das mobile Endgerät (z.B. ein Mobiltelefon) und das Network and Switching Subsystem. Im Folgenden wird deren Funktionsweise näher erläutert.

15.1.2 GSM-Grobarchitektur

Das GSM-Mobilfunknetz ist ein zellular strukturiertes Netz. Die Zellen bilden die kleinsten geografischen Einheiten, in denen die mobilen Endsysteme miteinander kommunizieren können.

Das Network and Switching Subsystem (NSS) fasst eine Menge von Verwaltungskomponenten der Netz-Betreiber zusammen. Dazu gehört das Mobile Services Switching Center (MSC), das die Aufgabe hat, die Basisstationen eines geografischen Bereichs zu verwalten, die Verbindungen zu kontrollieren und weiterzuleiten und den Übergang ins Festnetz (u.a. ISDN) zu realisieren. Außerdem unterhält das NSS das Heimatortregister HLR (Home Location Register), das alle registrierten Teilnehmer sowie deren aktuelle Aufenthaltsorte erfasst. Ein zweites Register, das Besuchsortregister VLR (Visitor Location Register), dient zur Speicherung von Informationen über Teilnehmer, die sich temporär in dem geografischen Bereich einer Funkzelle aufhalten. Zum NSS gehört ferner das Authentifizierungs-Zentrum AC (Authentication Center), das Informationen zur Identifikation und Authentifikation von Teilnehmern sowie zur Verschlüsselung der übertragenen Daten bereitstellt. Im Gerät-Identifikationszentrum EIC (Equipment Identification Center) des NSS werden schließlich die Identifikatoren der Geräte verwaltet. Dazu führt das EIC bis zu drei Listen, nämlich die schwarze für gesperrte Endgeräte, die gestohlen sind oder technische Mängel aufweisen, die graue, für fehlerhafte Geräte und die weiße mit den Seriennummern aller zugelassenen Geräte. Abbildung 15.1 veranschaulicht die GSM-Architektur. Im Folgenden werden die in der Abbildung angegebenen Informationen, die jeweils in den Subsystemen verwaltet werden, erklärt. Das Message Center ist für das Management von SMS-Nachrichten zuständig.

Mobile Endsysteme, MS (Mobile Station), wie Smartphones, können sich durch das zellulare System bewegen. In einem solchen mobilen Endgerät befindet sich eine SIM-Karte, die die Identifikations- und Authentifikationsdaten eines Mobilfunkteilnehmers enthält. Im Zustand detached ist die MS ausgeschaltet, aber eingehende Anrufe können auf dem Anrufbe-

NSS

MSC

HLR, VLR

AC

Mobile Station

antworter gespeichert werden. Wird die MS eingeschaltet, so sucht sie nach einer Zelle mit ausreichender Sendeleistung und bucht sich dort ein. Die aktuelle Ortsinformation wird in den zuständigen Registern HLR und VLR nachgehalten. Eine eingebuchte Station kann sich in zwei unterschiedlichen Zuständen befinden. Im Zustand `dedicated` ist der MS ein Kanal zur Gesprächübertragung zugeordnet. Ist die Station `idle`, so ist sie zwar eingeschaltet, aber ihr ist kein Verkehrskanal zugeordnet. Die MS ist dennoch fortwährend aktiv, so dass der Short Message Dienst genutzt werden kann und die Basisstation (s.u.) regelmäßig aktuelle Informationen über den Aufenthaltsort der mobilen Station erhält.

Basisstation

Die Funkübertragung erfolgt durch das Basisstationssubsystem (BSS), das das mobile System mit dem NSS verbindet. Das Basisstationssubsystem besteht aus zwei Teilen. Dazu gehört zum einen die Sende- und Empfangseinheit einer Zelle, das ist die Base Station Transceiver Station (BTS) oder kurz Basisstation, und zum anderen eine Kontrolleinheit, der Base Station Controller (BSC). Mehrere Basisstationen können über einen gemeinsamen Controller in einem BSS zusammengefasst sein. Der Controller übernimmt die Kommunikation mit dem NSS. Die Basisstation fragt in ihrer Zelle in regelmäßigen Abständen (zwischen 6 Minuten und bis zu 25 Stunden) nach den mobilen Endgeräten, die sich eingebucht haben. Kann wiederholt kein Kontakt zu einem eingebuchten Gerät hergestellt werden, so wird das Gerät in dem VLR der Basisstation als `detached` geführt. Ist die mobile Station zu weit von der Basisstation entfernt oder sinkt die Signalqualität unter einen festgelegten Wert, so erfolgt ein Handover, d.h. die MS wird zu einer anderen Zelle weitergereicht und dort in das VLR eingetragen.

15.1.3 Identifikation und Authentifikation

IMSI

Jeder Teilnehmer besitzt eine bis zu 15 Ziffern lange Teilnehmerkennung IMSI (International Mobile Subscriber Identity), die weltweit eindeutig und auf der SIM-Karte (s.u.) gespeichert ist. Diese internationale Teilnehmernummer besteht aus einem Ländercodeteil sowie einer Netz- und einer Teilnehmerkennung und wird vom Netzbetreiber der SIM-Karte fest zugeordnet. Die Rufnummer des Teilnehmers im öffentlichen Netz, die Mobile Station ISDN-Nummer (MSISDN), ist ebenfalls auf der SIM-Karte zu finden. Die IMSI wird normalerweise nur beim ersten Einbuchen zur Teilnehmeridentifikation verwendet. Danach erfolgt die Identifikation in der Regel nur noch anhand temporärer Teilnehmerkennungen, den TMSIs (Temporary Mobile Subscriber Identifier). Das heißt, dass nur noch diese über die Luftschnittstelle übertragen werden. Es ist die Aufgabe des Visitor Location Registers, für jedes mobile Endsystem, das sich in seiner Zelle aufhält, eine TMSI zu erzeugen und sie verschlüsselt an das Endgerät zu senden.

TMSI/LAI

Dieser temporäre Identifikator wird ebenfalls zusammen mit einem Identifikator für den Aufenthaltsort, dem LAI (Location Area Identifier), auf der SIM-Karte sowie im VLR gespeichert. Der TMSI besitzt nur eine lokale Gültigkeit. Wird im Zuge eines Handovers die VLR gewechselt, so erhält die mobile Station einen neuen TMSI zugeordnet. Auf diese Weise soll die Erstellung von Bewegungsprofilen verhindert werden. Für jedes mobile Endgerät existiert eine ebenfalls weltweit eindeutige Gerätekennung IMEI (International Mobile Station Equipment Identification), die im EIC des Netzwerk-Betreibers sowie im Gerät selbst gespeichert wird.

Im GSM-Netz werden die zur Identifikation und Authentifizierung benötigten Daten auf die SIM-Karte, auf das Authentifizierungs-Zentrum sowie auf die beiden Register HLR und VLR aufgeteilt (vgl. Abbildung 15.1).

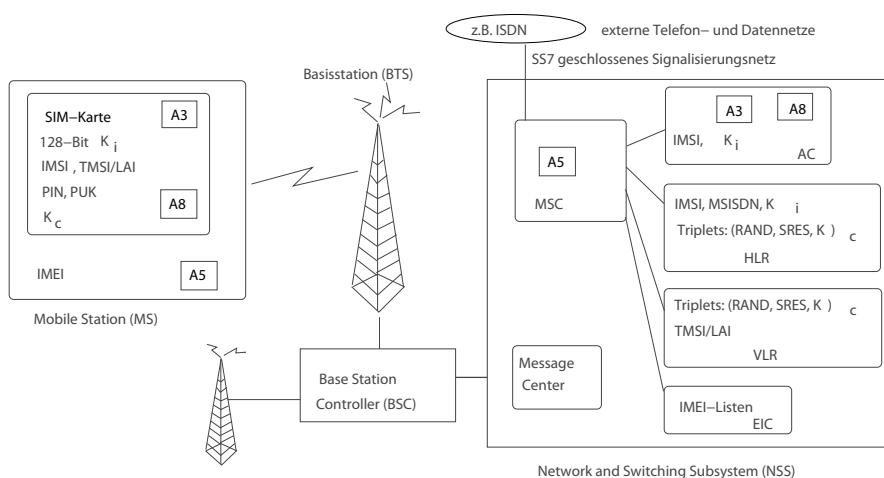


Abbildung 15.1: GSM-Grobarchitektur

SIM-Karte

Die SIM-Karte ist eine Chipkarte, die der Teilnehmer in sein mobiles Endgerät einlegen muss, um es nutzen zu können. Es gibt sie sowohl als Typ ID-1 Karte im Scheckkartenformat als auch als Plug-in Karte des Typs ID-000. Der EEPROM enthält die Teilnehmerkennung IMSI sowie die temporäre Kennung TMSI zusammen mit den Ortsinformationen LAI. Ferner befindet sich im EEPROM der so genannte Individual Subscriber Authentication Key K_i , der ein bis zu 128-Bit langer geheimer Authentifikationsschlüssel des Teilnehmers ist. Die Authentifikation eines Benutzers gegenüber der SIM-Karte erfolgt über eine vier- bis achtstellige PIN, die ebenfalls im EEPROM der Karte gespeichert ist. Zum Entsperren der Karte nach dreimaliger Falscheingabe dient ein achtstelliger PUK. Nach zehnmaliger Falscheingabe

SIM-Karte

be dieses PUKs wird die Karte endgültig gesperrt. Der ROM enthält den nicht kopierbaren Authentifikationsalgorithmus A3 sowie das Verfahren A8 zur Generierung von Schlüsseln für die Stromchiffre A5. A3 und A8 sind Einweg-Funktionen mit Schlüssel, die häufig durch den COMP-128 Algorithmus (siehe Abbildung 15.4) realisiert sind. Die SIM-Karte kommuniziert mit dem Mobiltelefon über das Protokoll T=0.

MS und NSS

A5

Die mobile Station enthält die Stromchiffre (vgl. Kapitel 7.5.2) A5 und den eindeutigen Gerätebezeichner. Die Chiffre A5 wird zur Gesprächs- bzw. Datenverschlüsselung benötigt, so dass auch der Netzbetreiber bzw. das Netzesubsystem eine Implementierung von A5 zur Verfügung stellen muss. Das NSS implementiert ferner auch die Algorithmen A3 und A8. Der A3 wird zur Generierung von Authentifizierungsinformationen benötigt und der A8 ermöglicht es dem NSS, Sitzungsschlüssel zu erzeugen, ohne dass diese über die Luftschnittstelle mit dem mobilen Endsystem ausgetauscht werden müssen. Die eindeutigen Teilnehmeridentifikatoren IMSI und die geheimen Kartenschlüssel K_i werden nicht nur auf der SIM-Karte, sondern zusätzlich auch im Authentifizierungs-Zentrum des Betreibers verwaltet.

A3, A8

Authentifikation

Triplets

Das AC erzeugt so genannte Authentication Triplets (in der Regel fünf Triplets auf einmal), bestehend aus jeweils einer Zufallszahl $RAND$, einer signierten Antwort $SRES$ und einem Sitzungsschlüssel K_c . Diese Triplets werden für die Challenge-Response-basierte Authentifikation und die Gesprächsverschlüsselung benötigt. Um das AC bei der Authentifizierung zu entlasten, werden diese vorab erstellten Triplets an das HLR und VLR weitergereicht, so dass diese ebenfalls über die notwendigen Authentifikationsinformationen verfügen und den Authentifikationsvorgang erheblich beschleunigen können. Bei einem Handover eines mobilen Geräts gibt das zuständige VLR die unverbrauchten Triplets dieses Geräts an dasjenige Mobile Station Switching Center (MSC) von dem Netzwerksystem weiter, in dessen Zuständigkeitsbereich sich dann das mobile Gerät befindet. Mit dem Tripletkonzept ist es möglich, dass beim Handover auch fremde Vermittlungssysteme (MSC) ein mobiles Gerät direkt authentifizieren können, ohne sich dazu noch einmal an das Authentifizierungs-Zentrum des Heimatnetzes des mobilen Teilnehmers wenden zu müssen.

Authentifikationsstrategie

Eine Authentifikation erfolgt stets beim Einbuchen der Mobilstation und beim Wechsel in einen neuen Verwaltungsbereich, also beim Handover. Eine wiederholte Authentifikation während eines Gesprächs ist aber auch möglich. Die geltende Authentifikationsstrategie wird vom Netzbetreiber festgelegt.

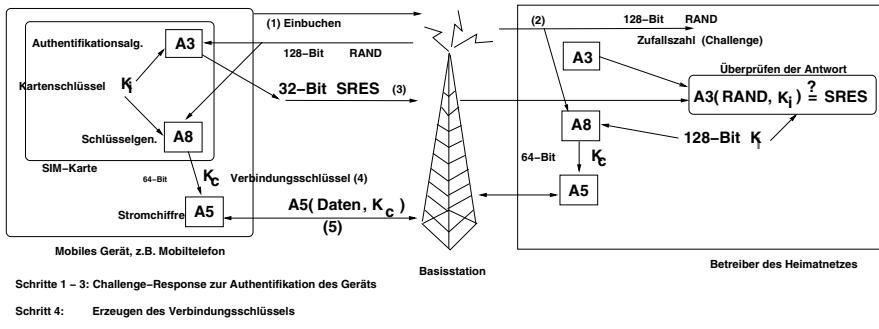


Abbildung 15.2: Authentifikation und Gesprächsverschlüsselung im GSM-System

Nachdem der Benutzer sich mittels seiner PIN gegenüber der Mobilstation authentifiziert hat, erfolgen die in Abbildung 15.2 zusammengefassten Schritte zur Authentifikation des Teilnehmers, hier jetzt der SIM-Karte. Dazu wird ein Challenge-Response-Protokoll mit symmetrischem Verfahren abgewickelt. Das MSC, sei es das vom Heimatnetz oder von einem Fremdnetz, das die Challenge einem Triplet entnimmt, sendet eine vom AC erzeugte 128 Bit lange Zufallszahl $RAND$ an die SIM-Karte. Unter Nutzung des Authentifizierungsalgorithmus A3 und des individuellen Schlüssels K_i berechnet die Karte einen 32 Bit langen Antwortwert,

$$SRES = A3(RAND, K_i), \text{ die so genannte Signed Response,}$$

und sendet diese zurück. Das MSC des Heimatnetzes überprüft die Antwort, indem es den ihm bekannten Teilnehmerschlüssel K_i aus seiner Datenbank AC liest. Es berechnet damit seinerseits

$$A3(RAND, K_i) = SRES'.$$

Das MSC eines Fremdnetzes greift zur Überprüfung wiederum auf das Triplet zurück und entnimmt diesem den zur Challenge $RAND$ vorab berechneten Antwortwert $SRES$. Falls gilt:

$$SRES = SRES',$$

ist der Teilnehmer authentifiziert und erhält Zugang zum Netz. Wichtig ist, dass der geheime Teilnehmerschlüssel K_i nie über die Luftschnittstelle übertragen wird.

Core Network

Zu klären bleibt, auf welche Weise die sensible Information, die in den Triplets im Klartext enthalten ist, sicher zwischen den Netzwerksystemen ausgetauscht wird. GSM verwendet hierfür kein öffentliches Netz, sondern die Kommunikation erfolgt über das so genannte Core Network, das das Signal System No.7 (SS7) Protokoll verwendet. SS7 [156] ist ein globaler Standard für die Interconnection von Telefonnetzen und anderen Dienstleistern.

Challenge-Response

SS7

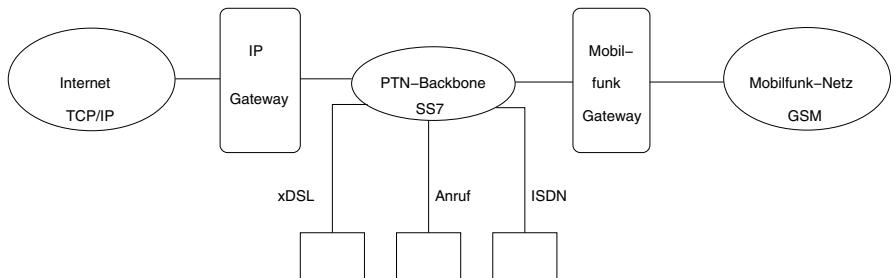


Abbildung 15.3: Vernetzungsstruktur eines öffentlichen Telefonnetzes

ler Standard für die Telekommunikation, der von dem Telecommunication Standardization Sector (ITU-T) der International Telecommunication Union (ITU) definiert wurde. Der Standard spezifiziert die Protokolle, über die die Komponenten eines öffentlichen Telefonnetzes (engl. *Public switched telephone network (PSTN)*) Informationen austauschen können, indem sie ein digitales Signalisierungsnetz nutzen.

Das SS7 ist ursprünglich für ein geschlossenes Netz von Telekommunikationsfirmen konzipiert worden und beinhaltet deshalb nur rudimentäre Authentifikationsmaßnahmen. Aufgrund der internationalen De-Regulierungs-bemühungen müssen Telekommunikationsfirmen heutzutage jedoch auch SS7-Verbindungen zu beliebigen anderen Teilnehmern herstellen. Abbildung 15.3 zeigt die Architektur eines öffentlichen Telefonnetzes bestehend aus dem Telefon-Backbonenetz, das primär auf SS7 basiert, der Anbindung an das Internet, sowie der Anbindung an Mobilfunknetze, die u.a. das GSM-Protokoll verwenden. Die Abbildung verdeutlicht, dass SS7-Systeme untereinander auch über das Internet verbunden sein können, wodurch sie den üblichen Gefährdungen ausgesetzt sind. Darüber hinaus können Benutzer über ISDN-Leitungen direkt Daten in das SS7-Netz einspeisen oder Nachrichten abhören. In [111] werden die möglichen Schwachstellen und Angriffspunkte auf das SS7 klassifiziert und beschrieben. Die Hauptangriffspunkte sind das Einschleusen gefälschter Datenpakete aufgrund der mangelhaften Authentifikation, das Sniffen der SS7-Verbindungen sowie Angriffe durch Viren, Würmer oder Trojanische Pferde auf die Service-Knoten des SS7-Systems, die sich aus der Anbindung an das offene Internet ergeben.

Probleme

15.1.4 Gesprächsverschlüsselung

Zur Verschlüsselung der Gesprächsdaten wird zunächst vom Heimatnetz ein gemeinsamer geheimer 64-Bit Sitzungsschlüssel K_c erzeugt. Diesen generieren sowohl die SIM-Karte als auch das AC des Heimatnetzes mit

dem Algorithmus A8 unter Rückgriff auf die Zufallszahl $RAND$ und den geheimen Kartenschlüssel K_i :

$$A8(RAND, K_i) = K_c.$$

Auf diese Weise wird gewährleistet, dass auch der Schlüssel K_c nicht über die Luftschnittstelle übertragen wird. Wie bereits erwähnt, erstellt ein AC für jeden mobilen Teilnehmer eine Menge von Triplets, so dass die benötigten Informationen ggf. bereits vorgefertigt zur Nutzung vorliegen. Da die Triplets jeweils einen vorab berechneten Sitzungs-Schlüssel enthalten, sind auch fremde MSCs, also die Switching Center anderer Netzbetreiber, die im Besitz der Triplets sind, in der Lage, die verschlüsselten Daten des mobilen Geräts zu verarbeiten. Da die Ver- und Entschlüsselung in den Basisstationen (BTS) erfolgt, werden die Sitzungsschlüssel an diese weitergereicht. Weil sowohl der A3 Algorithmus zur Authentifikation als auch der A8 zur Schlüsselgenerierung auf der SIM-Karte enthalten sind und nur vom Heimatnetz benötigt werden, können Netzprovider diese Verfahren selber festlegen. Fremdnetze benötigen keine Kenntnis davon.

Das mobile Gerät überträgt die verschlüsselten Daten nur bis zur nächsten Basisstation (Funkschnittstelle), das die Daten entschlüsselt und zur weiteren Bearbeitung an das Netzwerksubsystem weiterleitet.

Wie bereits erwähnt, werden die Algorithmen A3 und A8 häufig durch ein gemeinsames Verfahren, den COMP-128 Algorithmus, realisiert. Dessen Arbeitsweise fasst Abbildung 15.4 zusammen. Unter Eingabe einer Challenge $RAND$ sowie des Kartenschlüssels K_i berechnet er sowohl die signierte Antwort $SRES$ als auch den Sitzungs-Schlüssel K_c , also

$$(SRES, K_c) = \text{COMP-128}(RAND, K_i).$$

COMP-128

Die Verschlüsselung der Nutzdaten M einer Kommunikation erfolgt mittels der standardisierten Stromchiffre A5:

$$A5(M, K_c).$$

Stromchiffre

Für jedes zu übertragene Frame wird die Chiffre erneut mit dem Schlüssel K_c und der zu übertragenden Framenummer (ein 22-Bit-Wert) initialisiert, so dass für jedes Frame ein eigener 114-Bit-Schlüsselstrom generiert wird. Unter GSM wird ein Kommunikationsschlüssel K_c so lange wiederverwendet, bis durch das Message Switching Center des Heimatnetzes eine erneute Authentifikation der mobilen Station gefordert wird. Eine Schlüsselerneuerung ist also von der jeweiligen Authentifizierungsstrategie des Netzproviders abhängig.

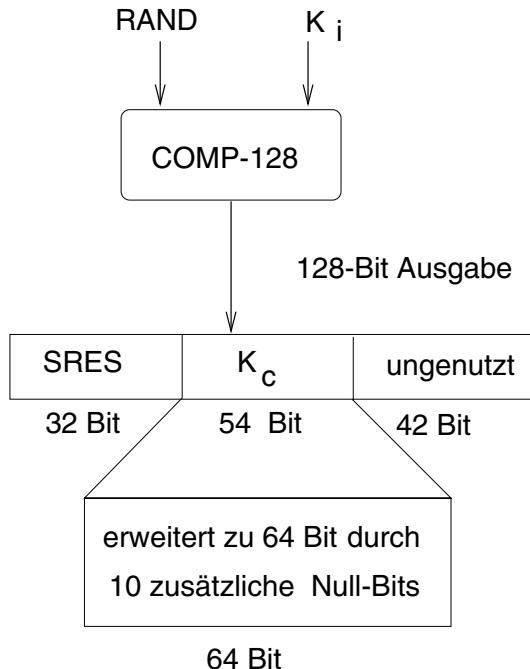


Abbildung 15.4: Der COMP-128 Algorithmus

A5

Obwohl der A5-Algorithmus offiziell geheim ist, wurde er über inoffizielle Kanäle bekannt gemacht und kryptografisch analysiert (u.a. [71]). Von A5 gibt es unterschiedliche Implementierungen, die mit A5/1, A5/2 etc. bezeichnet werden. Der A5/1 soll eine Funkverschlüsselung für Sprachübertragung garantieren. Die Chiffre A5/1 besteht aus drei Schieberegistern mit linearer Rückkopplung, die mit K_c und der Framenummer initialisiert werden. Die Schieberegister haben eine Länge von 19, 22 und 23 Bit, die zusammen die Schlüssellänge von 64 Bit ergeben. Von den generierten 228 Bit Schlüsselbits werden die ersten 114 Bit dazu verwendet, den Klartext des Frames, das von der mobilen Station zur Basisstation gesendet werden soll, mittels XOR zu verschlüsseln. Die zweiten 114 Bit werden verwendet, um ein Frame zu verschlüsseln, das in umgekehrter Richtung, also von der Basisstation zum mobilen Gerät gesendet werden soll. Das Verfahren A5/2 ist schwächer als A5/1 und war für den Einsatz in Ländern außerhalb von Europa gedacht. Aufgrund seiner Schwächen wurde der A5/2 ab 3GPP² Release 6 aus dem Standard herausgenommen. Das im Jahr 2002 entwickelte Verfahren A5/3 ist im Gegensatz zu seinen Vorgängern keine Strom- sondern eine Blockchiffre, die einen 64 Bit Block mit Hilfe eines 128 Bit Schlüssels verschlüsselt. Der A5/3 wurde durch die ETSI unter dem Markenzeichen 3GPP8 veröffentlicht.

² 3rd Generation Partnership Project

und basiert auf dem KASUMI-Verfahren (siehe Seite 874), das unter UMTS eingesetzt wird. Eine beachtenswerte Ausnahmerolle unter den verschiedenen A5-Varianten nimmt der A5/0 ein, der gar keine Verschlüsselung bietet.

15.1.5 Sicherheitsprobleme

Obwohl die in GSM-Netzen eingesetzten Sicherheitsmechanismen auf den ersten Blick gut und ausreichend erscheinen, weisen sie doch eine ganze Reihe gravierender Mängel auf. Die Authentifikation erfolgt nur einseitig. Das hat zur Folge, dass sich eine mobile Station gegenüber jedem Anfrager authentifiziert, sei es eine autorisierte Basisstation oder ein Angreifer. Da ein mobiles Gerät immer demjenigen antwortet, der das stärkste Signal verwendet, kann das Signal einer berechtigten Basisstation durch ein stärkeres Signal eines Angreifers ausgeschaltet werden. Eine solche gespooftete Station könnte zum Beispiel vorgeben, keinen Verschlüsselungsalgorithmus zu beherrschen, so dass das mobile Endsystem zur Kommunikation des A5/0 einsetzen wird, wodurch der Angreifer alle kommunizierten Daten direkt mitlesen kann.

einseitige
Authentifikation

Dieses Antwortverhalten von mobilen Stationen wird von so genannten IMSI-Catchern für Angriffe ausgenutzt. IMSI-Catcher sind Geräte, die sich gegenüber einem in der Nähe befindlichen Teilnehmer wie einer festen Basisstation des Mobilfunknetzes verhalten und die mobile Station zwingen, ihre IMSI anstelle der temporären Identität (TMSI) zu übertragen. Jedes eingeschaltete Handy im Empfangsbereich bucht sich automatisch bei diesem IMSI-Catcher ein, ohne dass der Teilnehmer dies bemerken kann. Zusammen mit den regelmäßigen Informationen, die eine mobile Station der Basisstation über ihren aktuellen Aufenthaltsort liefert, ist somit die Erstellung von Bewegungsprofilen auch für solche Personen möglich, die dazu nach dem Telekommunikationsgesetz gar nicht ermächtigt sind. Derartige Geräte besitzen zwar in Deutschland keine Betriebsgenehmigung der Bundesnetzagentur, da es sich aber im weitesten Sinne um Messinstrumente handelt, sind sowohl die Produktion als auch der Export zulässig. Erst im Juli 2002 hat der Bundestag ein Gesetz erlassen (im Zuge der Anti-Terrorpakete), das Behörden nach der Ausstellung einer richterlichen Genehmigung die Nutzung von IMSI-Catchern zum Abhören von Mobiltelefonen gestattet. Damit ist erstmals eine gültige Rechtsgrundlage für das Abhören von Mobilkommunikationsverbindungen gelegt worden.

IMSI-Catcher

Bewegungsprofile

Die zur Gesprächsverschlüsselung verwendeten 64-Bit Sitzungsschlüssel K_c sind nach dem heutigen Stand der Technik zu kurz, um einen Angriff mit einem vollständigen Durchsuchen des Schlüsselraums (Exhaustive Search Attack) auszuschließen. Anders als es der Name vermuten lässt, wird nicht

Sitzungsschlüssel

für jede neue Kommunikationsverbindung (Sitzung) ein neuer Schlüssel berechnet, sondern die Strategie, wann ein Schlüssel erneuert wird, legt das MSC des Heimatnetzproviders fest. Es ist in heutigen Systemen durchaus üblich, dass ein Kommunikationsschlüssel über mehrere Tage hinweg beibehalten wird. Mit der Kenntnis von K_c kann ein Angreifer die übertragenen Daten einfach dechiffrieren, da die Framenummer, die ebenfalls in die Schlüsselgenerierung einfließt, keine Zufallszahl ist, sondern sequentiell bei der Framecodierung erzeugt wird.

COMP-128

Der von vielen GSM-Netzbetreibern zur Erzeugung des Sitzungsschlüssels verwendete COMP-128 Algorithmus bedeutet in mehrfacher Hinsicht ein Sicherheitsrisiko. So verkürzt er, wie in Abbildung 15.4 deutlich zu erkennen, den sowieso schon kurzen Schlüssel auf magere 54 Bits, indem einfach 10 Bits künstlich auf Null gesetzt werden, und ermöglicht so einen Angriff mit gewähltem Klartext, der lediglich die Komplexität $\mathcal{O}(2^{18})$ besitzt. Für den Angriff nutzt man wiederum die fehlende wechselseitige Authentifizierung aus. Der Angreifer benötigt nur den physischen Zugriff auf die SIM-Karte, ein Chipkartenlesegerät sowie einen handelsüblichen PC. Nach dem Einlegen der SIM-Karte in das Lesegerät können mittels des PCs Authentifikationsanfragen an die Karte gesendet werden; es wird also eine Challenge eines ACs simuliert. Aus der Antwort der Karte kann bei genügend großem Stichprobenraum (bis zu 150.000 *RAND*-Anfragen sind erforderlich), der geheime Teilnehmerschlüssel K_i berechnet werden. Mittels der Techniken der differentiellen Kryptoanalyse (vgl. Kapitel 7.8.3) konnte bei einem konkret durchgeführten Angriff der Kartenschlüssel nach ca. 8 Stunden geknackt werden. Man beachte, dass mit dessen Kenntnis nicht nur ein einzelner Sitzungsschlüssel K_c , sondern alle derartigen Schlüssel, die die MS bereits verwendet hat und auch die, die sie erst noch erzeugen wird, im Prinzip kompromittiert sind. Der Angreifer benötigt lediglich die jeweilige Challenge *RAND* des realen ACs. Da passive Angriffe auf ein Funknetz sehr einfach durchführbar sind, ist dieses Problem für einen Angreifer sehr einfach zu lösen.

Seitenkanalangriff

Im Mai 2002 haben Forscher bei IBM einen neuen Angriff auf COMP-128 veröffentlicht, der auf einer speziellen Variante eines Seitenkanal-Angriffs beruht. Voraussetzung für die Durchführung des Angriffs ist, dass der Angreifer einen physischen Zugriff auf die SIM-Karte besitzt, um Stromaufnahme und elektromagnetische Abstrahlungen zu messen. Ausgenutzt wird eine Implementierungsschwäche von COMP-128. Da SIM-Karten eine geringe Prozessorleistung besitzen, werden bei der Schlüsselgenerierung voraus berechnete Tabellen verwendet. Der Zugriff auf diese Tabelle weist bei der Berechnung des Kommunikationsschlüssels K_c aus dem Kartenschlüssel K_i typische Verhaltensmuster auf, was zu physikalisch messbaren

Mustern führt. Diese Muster geben Informationen über den verwendeten Schlüssel K_i preis. Laut den Forschern von IBM benötigt der Angriff nur sieben Berechnungen des COMP-128, so dass ein erfolgreicher Angriff in Minuten schnelle durchführbar ist. Die Lösung von IBM sieht vor, den einzelnen Zugriff auf die Tabelle durch eine Folge von Tabellenzugriffen zu ersetzen, um auf diese Weise die Mustererkennung deutlich zu erschweren.

Ein weiterer Angriff besteht darin, die SIM-Karte zu clonen. Das GSM-System erkennt jedoch, wenn zwei SIM-Karten mit der gleichen IMSI gleichzeitig eine Verbindung aufbauen möchten. In diesem Fall überprüft das GSM-System den Aufenthaltsort der mobilen Geräte und bricht die Verbindungen ab, falls die Aufenthaltsorte unterschiedlich sind. Das hindert einen Angreifer jedoch nicht daran, mit einem geclonten Gerät Kommunikationsverbindungen abzuhören.

SIM-Clon

Auch der zur Gesprächsverschlüsselung ursprünglich verwendete Algorithmus A5 selbst muss als kryptografisch sehr schwach eingestuft werden, da unter anderem in [71] mehrere Angriffe beschrieben werden, die eine Komplexität von höchstens $\mathcal{O}(2^{40})$ besitzen. Dies stellt, wie wir bereits an anderer Stelle gesehen haben, keine nennenswerte Hürde mehr für Angreifer dar. Zudem wurden in verschiedenen Projekten Rainbow-Tables³ zum schnellen Knacken von A5/1-Schlüsseln berechnet und öffentlich verfügbar gemacht (u.a. <http://reflextor.com/trac/a51>). Im Januar 2010 wurde unter Nutzung solcher Rainbow-Tables ein verteilter Angriff auf den A5/1 durchgeführt. Der Angriff erfolgte sehr effizient, indem er die Tables mittels File-Sharing-Software auf verschiedene Rechner verteilt hat. Mit dem Angriff gelang es einen Schlüssel mit nur 40 Rechnern in drei Monaten geknackt zu knacken. Der Algorithmus A5 wurde aufgrund seiner bekannten Schwächen überarbeitet, um eine höhere Sicherheit zu gewährleisten. Wie bereits weiter oben ausgeführt, wird seit Juli 2002 die A5-Variante A5/3 genutzt, die auf dem KASUMI-Verfahren basiert. Zu beachten ist jedoch, dass laut Spezifikation das A5/3-Verfahren, obwohl es nominell 128-Bit Schlüssel verwendet, real nur 64 frei wählbare Bits nutzt, da die zweiten 64 Bit lediglich eine Kopie der ersten darstellen. Damit ist die Schlüssellänge nach wie vor nur 64 Bit und für heutige Anforderungen viel zu kurz. Dies trifft nicht auf die Chiffre KASUMI zu. Diese wurde veröffentlicht und analysiert und gilt als eine starke Chiffre. Zu beachten ist aber dennoch, dass die Gesprächsdaten nur auf der Verbindung zwischen dem mobilen Endgerät und der Basisstation verschlüsselt werden. Die Daten liegen somit in der Basisstation offen und werden bei ihrer weiteren Übermittlung ungeschützt übertragen.

A5

³ Eine Rainbow-Table ist eine Tabelle mit vorab berechneten Werten, um die Umkehrfunktion für kryptographische Hashfunktionen effizient(er) zu berechnen.

Integrität?

Schließlich sollte nicht übersehen werden, dass der GSM-Standard keine Vorkehrungen trifft, die Integrität von Daten mittels Hashfunktionen abzusichern oder sie zu signieren. Diese Dienste sind in einem System, das vorwiegend Gesprächsdaten übermittelt, nicht dringend erforderlich. Durch die stärkere Verschmelzung von Gesprächs- und Datenkommunikation sowie den Einsatz multimedialer Daten beim derzeitigen Übergang zu Mobilfunknetzen der dritten Generation, kann das Fehlen solcher Dienste jedoch gravierende Sicherheitsprobleme aufwerfen. Es ist klar, dass die Mobilfunknetze der nächsten Generation hierfür Vorkehrungen zu treffen haben.

VPN-Lösung

Wie die Sicherheitsprobleme von GSM verdeutlichen, sollte man sich nicht auf die Qualität der integrierten Sicherheitsdienste verlassen, wenn man über das GSM-Netz vertrauliche und sensible Daten übermitteln möchte. Wird beispielsweise das Mobiltelefon nur als Modem verwendet, um eine mobile Datenübertragung zwischen einem Laptop und einem Endgerät z.B. dem Mailserver des Unternehmens zu ermöglichen, so sollte man sich der bereits in Kapitel 14.2.2 diskutierten Technik eines VPNs bedienen, um eine sichere Ende-zu-Ende Kommunikation zwischen Laptop und Server zu etablieren. Das bedeutet, dass die zur sicheren Datenübertragung eingesetzten Verfahren unter Nutzung beispielsweise von IPSec von dem Unternehmen, das seinen Mitarbeiter einen solchen mobilen Zugriff ermöglicht, festgelegt werden können. Bevor die Daten in das GSM-Netz eingespeist werden, erfolgt bereits die Verschlüsselung sowie ggf. die Sicherung der Absender-Authentizität und der Integrität der Daten. Das unsichere GSM-Netz wird auf diese Weise getunnelt und der Nutzer ist damit nicht mehr auf die Dienste der GSM-Sicherheitsarchitektur angewiesen. Die GSM-Verschlüsselung bildet dann lediglich eine weitere Schutzschicht um die Daten, die aber für die Sicherheit der Daten nicht essentiell ist.

Zukunft von GSM

GSM-R

Der GSM-Standard wird auch in Zukunft noch breitflächig im Einsatz sein. So basiert der GSM-R (Rail) Standard als Basistechnologie des European Train-Control-System (ETCS), das wiederum Bestandteil des Europäischen Eisenbahnverkehrsleitsystems ist, auf GSM. GSM-R wird zur drahtlosen Übertragung von Nachrichten im Bahnverkehr eingesetzt. GSM-R-basiert werden hierbei Daten übertragen, die den Zuglauf, die Geschwindigkeiten oder die Positionen von Zügen überwachen. Zudem wird eine GSM-R-basierte Kommunikation auch zur Zugfernsteuerung wie beispielsweise Zwangsbremsungen eingesetzt. Da GSM-R auf GSM basiert und keine zusätzlichen Sicherheitsfunktionen integriert sind, ist der GSM-R Datenverkehr in analoger Weise z.B. durch das Einschleusen von gefälschten Nachrichten angreifbar und verwundbar wie das ursprüngliche GSM-System. Um eine höhere Sicherheit anzubieten, nutzt GSMD-R den

Standard EURORADIO als Overlay. Die in dem Standard verwendeten Verfahren und Sicherheitsprotokolle entsprechen jedoch nicht dem Stand der Technik, u.a. wird der 3DES für die verschlüsselte Kommunikation verwendet, so dass noch erheblicher Bedarf besteht, die Sicherheit der drahtlosen Kommunikation im Bahnverkehr zu erhöhen.

Im Kontext der Fahrzeug-Kommunikation werden ebenfalls Standard-GSM-Module eingesetzt. Sie ermöglichen eine Kommunikation zwischen dem Fahrzeug und den Backend-Systemen des OEMs (Original Equipment Manufacturer), also den Automobilherstellern. Sie ermöglichen zudem auch das Tracking von Autos via GPS und GSM. Das bedeutet, dass auch für alle diese Anwendungen die Sicherheitsprobleme und Schwachstellen des GSM-Systems zu beachten sind.

Automotive

15.1.6 GPRS

Ein Schritt in Richtung auf die Mobilkommunikation der dritten Generation (UMTS) ist der paketvermittelnde Dienst GPRS, der theoretisch eine Datenrate bis zu 115Kb/s erlaubt. In der Praxis muss man sich aber mit weit geringeren Datenraten von 14.4Kb/s im Upstream und 40Kb/s im Downstream zufrieden geben. Wie bei seinem Vorgängerprotokoll, dem HSCSD (High Speed Circuit Switched Data), das mehrere GSM-Kanäle bündelt, um eine höhere Bandbreite zu erzielen, so werden auch bei GPRS mehrere Kanäle, nämlich bis zu acht, für einen einzelnen Benutzer oder für mehrere Benutzer gemeinsam genutzt. Die Paketvermittlung ist im Gegensatz zur Leitungsvermittlung, wie sie GSM kennt, sehr viel besser für den Datentransfer geeignet. Die Kanäle werden bei GPRS dynamisch alloziert, woraus sich eine flexible Ressourcennutzung ergibt, aber die Datenraten auch variieren können. GPRS ist gut geeignet, um häufig kleine Datenmengen oder nicht zusammenhängend Datenströme zu übertragen. Beispiele sind E-Mails, E-Commerce oder auch ortsabhängige Dienste. GPRS (General Packet Radio Service) ist ein Dienst zur direkten Anbindung mobiler Endgeräte, wie Mobiltelefone, an das Internet. Das heißt, dass die über GPRS angebundenen Geräte eine IP-Adresse besitzen. GPRS baut auf dem GSM-Standard auf. Abbildung 15.5 veranschaulicht die Grobarchitektur eines GPRS-Systems.

Architektur

Der GGSN (Gateway GPRS Support Node) stellt die Verbindung zum Internet her und der SGSN (Service GPRS Support Node) vermittelt IP-Pakete über GSM an das Endgerät. IP-Pakete werden dazu, wie üblich, vor dem Versenden in eine Folge von Datenframes aufgespalten und beim Empfänger wieder zum kompletten Paket zusammengefügt. Um einen hohen Datendurchsatz zu erzielen, sendet ein mobiles Gerät unterschiedliche Datenframes in parallelen Timeslots an die Basisstation oder auch an unter-

GGSN

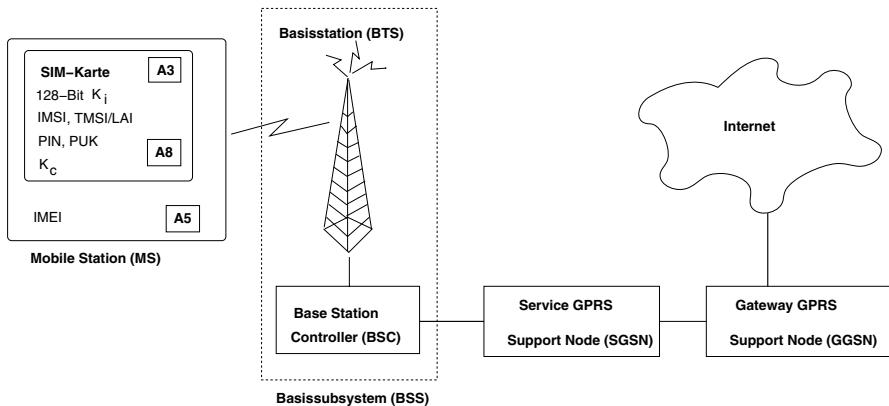


Abbildung 15.5: GPRS-Grobarchitektur

schiedliche Basisstationen im gleichen Basisstationen-Subsystem, falls ein Handover eintritt. Die verschiedenen Frames werden erst vom SGSN wieder zu einem Nachrichtenpaket vereinigt. Im Gegensatz zu GSM werden unter GPRS die einzelnen Datenframes aber nicht bereits von der Basisstation ver- und entschlüsselt, sondern erst durch das SGSN. Dies ist notwendig, da die Basisstation durch das parallele Senden des mobilen Geräts keine Kenntnisse darüber hat, welche Framenummer das zu entschlüsselnde Frame besitzt. Da die Framenummer aber als Parameter in den Algorithmus A5 zur Erzeugung des zur Verschlüsselung benötigten Schlüsselstroms eingeht, muss auch der Empfänger eines verschlüsselten Frames diese Framenummer kennen, um seinerseits den korrekten Schlüsselstrom zur Entschlüsselung generieren zu können.

SGSN

Beim Einbuchen sendet das mobile Gerät über die Funkschnittstelle wie bei GSM eine Anfrage an das Netz. Die Anfrage enthält die eindeutige Teilnehmerkennung IMSI des Geräts. Der SGSN empfängt derartige Anfragen und leitet sie an das Home Location Register bzw. Authentifizierungszentrum des Heimatnetzes weiter. Wie bei GSM werden vom AC Authentifizierungs-Triplets, bestehend aus einer Challenge RAND, einer signierten Antwort SRES und dem GPRS Schlüssel K_c , erzeugt. Mit dem Schlüssel K_c werden alle Daten, die zwischen dem mobilen Gerät und dem SGSN ausgetauscht werden, verschlüsselt. AC sendet die Triplets zum SGSN, das eine der Challenges RAND an das mobile Gerät weiterversendet. Die Authentifikation zwischen mobilem Gerät und SGSN erfolgt dann in analoger Weise wie bereits bei GSM beschrieben. Im Unterschied zu GSM verwaltet hierbei jedoch die SGSN die sicherheitsrelevanten Informationen, wie die Teilnehmer-Triplets und die Kommunikationsschlüssel K_c . Die Sicherheit von GPRS hängt somit sehr stark von der sicheren Konfiguration und Administration derartiger SGSN Komponenten ab.

Analog zur temporären Teilnehmeridentifikation TMSI unter GSM wird auch bei GPRS eine temporäre Kennung, die TLLI (Temporary Logical Link Identifier) verwendet. Jede SGSN verwaltet eine Datenbank, die eine Zuordnung zwischen der IMSI und der TLLI der Teilnehmer enthält. Die Schlüsselgenerierung und das Verschlüsseln der Daten erfolgt auf die gleiche Weise wie bei GSM, jedoch wird in GPRS eine neue Implementierung des A5-Algorithmus verwendet, der dem A5/3 aus dem GSM-System entspricht. Aus der Analogie ist klar, dass auch GPRS keine Ende-zu-Ende-Sicherheit unterstützt. Die Veränderungen im GPRS-Sicherheitsmodell gegenüber dem von GSM, nämlich eine Verschlüsselung zwischen dem mobilen Gerät und dem SGSN, sowie eine Neuimplementierung des A5 führen jedoch nicht notwendig zu einer höheren Systemsicherheit. Es sind lediglich einige der aus dem GSM-Umfeld publik gewordenen Angriffe nicht direkt auf GPRS übertragbar.

Zu beachten ist, dass ein GPRS-Gerät eine eigene IP-Adresse besitzt und sich somit direkt im Internet befindet. Entsprechend angreifbar sind diese Geräte dann auch. GPRS ermöglicht es mobilen Teilnehmern, jederzeit online (always on) zu sein. Das bedeutet aber natürlich auch, dass die mobilen Teilnehmer in sehr starkem Maß Angriffen aus dem Internet ausgesetzt sind. Angriffe könnten beispielsweise dann für Dritte interessant sein, wenn ein mobiler Teilnehmer mittels seines GPRS-Geräts direkt auf die lokalen Unternehmensdaten zugreift. Eine Etablierung eines VPN zur sicheren Ende-zu-Ende-Kommunikation ist hier somit ebenfalls mit Sicherheit sinnvoll.

Sicherheitsdienste

always on

15.2 UMTS

Mit UMTS (Universal Mobile Telecommunication System) wurde das Mobilfunksystem der dritten Generation basierend auf GSM als weltweiter Standard etabliert. Mit dem Harmonisierungs- und Globalisierungsprozess für den Mobilfunk im Jahr 1998 wurden die Standardisierungsbemühungen der ETSI (European Telecommunications Standards Institute) durch ein weltweites Standardisierungsgremium, der 3GPP (3rd Generation Partnership Projekt) abgelöst. Die Mitglieder von 3GPP setzen sich aus Standardisierungsgremien aus verschiedenen Ländern und Kontinenten, wie Europa, Nord-Amerika, Japan und Korea zusammen. Mobilfunksysteme der dritten Generation (3G) sind wesentlich komplexer als die der zweiten Generation und ermöglichen neben Content- und Diensteanbietern als zusätzlichen Teilnehmern auch eine Vielzahl zusätzlicher Operationen und komplexer Roaming-Szenarien. UMTS bietet in Pikozellen, die üblicherweise nicht größer als 50 Meter sind, Datenraten von bis zu 2Mb/s. Diese Rate nimmt aber mit der Größe der Zelle drastisch ab, so dass bei Makrozellen mit einer Grö-

UMTS

ße von einigen Kilometern nur noch bis zu 384Kb/s und bei Zellgrößen von einigen 10 Kilometern nur noch Datenraten von bis zu 144Kb/s erreichbar sind.

Die 3G-Systeme verwenden unterschiedliche Netzwerktypen und insbesondere verschiedene drahtlose Netze zur Kommunikation, so dass die Frage der UMTS-Sicherheit bereits frühzeitig ein zentraler Aspekt bei der Entwicklung des Standards war.

15.2.1 UMTS-Sicherheitsarchitektur

Basis-
Sicherheitsdienste

Die UMTS-Sicherheitsarchitektur übernimmt einige der Basis-Sicherheitsdienste von GSM und erweitert diese. Zu den Basisdiensten gehören die Vertraulichkeit der Teilnehmeridentität, die Authentifizierung des Teilnehmers gegenüber dem Netz, die verschlüsselte Kommunikation auf der Luftschnittstelle, die SIM-Karte als persönliches Sicherheitsmodul sowie die Authentifikation des Teilnehmers gegenüber der SIM-Karte. Als ein Analogon zur SIM-Karte wird das USIM (UMTS Subscriber Identity Module) eingeführt. Die Aufgaben des aus GSM bekannten Network and Switching Subsystem (NSS) werden unter UMTS aufgeteilt auf die Heimatumgebung, die von dem Netzbetreiber betreut wird, der die USIM des mobilen Benutzers ausgestellt hat, und auf das Service Netzwerk (SN). Das SN verwaltet die Basisstationen eines geografischen Bereichs, kontrolliert Verbindungen, leitet sie weiter und verwaltet das Besuchsortregister VLR (Visitor Location Register). Wie bereits bei GSM, so enthält das VLR auch unter UMTS Informationen über Teilnehmer, die sich temporär in dem geografischen Bereich einer Funkzelle aufhalten. Zur Heimatumgebung gehört das Authentifizierungs-Zentrum AC, das das Aufgabenspektrum der entsprechenden GSM-Komponente noch um zusätzliche Dienste ergänzt.

Core Netzwerk

Unter UMTS erfolgt der Austausch der Authentifizierungs-Vektoren über IP-basierte Netze, wie GTP, dem Tunneling Protokoll für GPRS. IP-basierte Netze können durch die Verwendung von IPSec (vgl. Abschnitt 14.3) und IPSec Tunnels zwischen Netzwerkkomponenten abgesichert werden. Zur Sicherung der im GSM-Kontext verwendeten MAP (Mobile Application Part) Nachrichten wurde ein sicheres MAP vorgeschlagen, das analog zu IP-Sec einen speziellen Header mit Sicherheitsinformationen an die Nachricht anfügt. Analog zu IPSec wird ferner vorgeschlagen, die Sicherheitseigenschaften der Verbindungen mittels Security Associations zu definieren. Da es sich bei diesem Protokoll aber noch nicht um einen Standard handelt, wird darauf nicht näher eingegangen.

Als erweiterte Sicherheitsarchitektur (vgl. Abbildung 15.6) bietet UMTS folgende Dienste an:

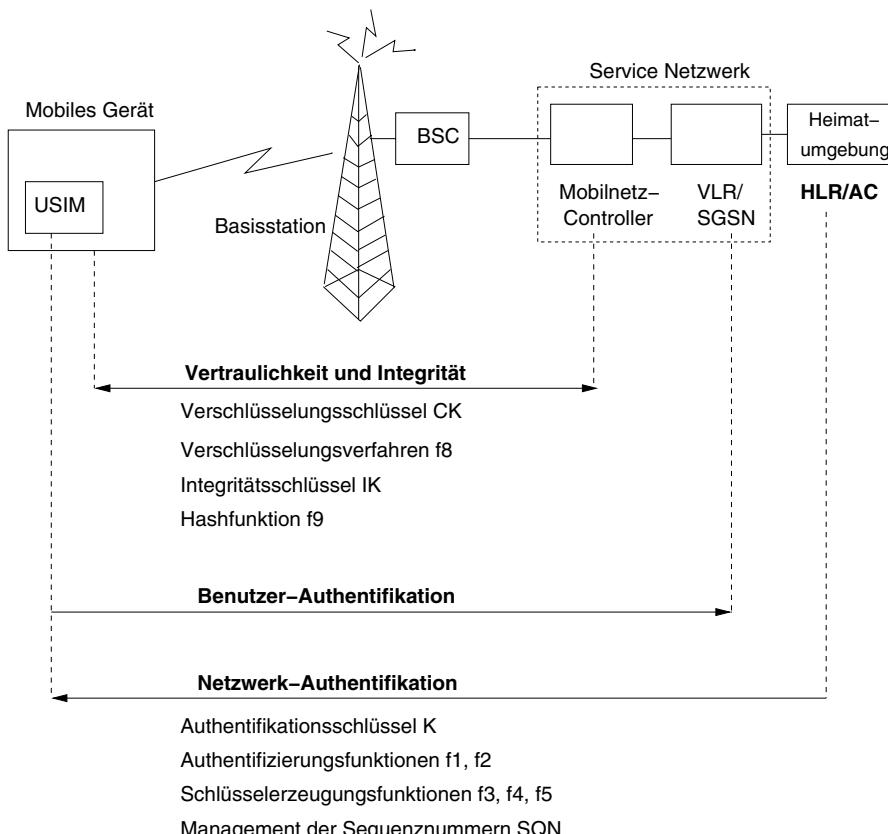


Abbildung 15.6: UMTS-Sicherheitsarchitektur

Erweiterte UMTS Authentifikation und Schlüsselvereinbarung:

Unter UMTS authentifiziert sich nicht nur das mobile System gegenüber dem Netz, sondern auch das Netz, in das man sich einbucht sowie das Heimatnetz⁴ authentifizieren sich gegenüber der USIM. Anzumerken ist aber bereits an dieser Stelle, dass sich ein Fremdnetz nur über einen Vertrauenspfad zum Heimatnetz des mobilen Teilnehmers und mittels der Kenntnis von Informationen, die es vom Heimatnetz erhalten hat, authentifiziert. Zur Erkennung von wiedereingespielten Authentifizierungsdaten, also zur Abwehr von Replay-Attacken, verwendet UMTS zusätzliche Sequenznummern. Damit die Partner einer UMTS-Verbindung in der Lage sind, die Integrität der übermittelten Signalisierungsdaten zu überprüfen, generiert das AC des Heimatnetzes einen gemeinsamen Integritätschlüssel IK. Weitere Erweiterungen und Verbesserungen der Schlüsselvereinbarungsprotokolle sind die Schritte, durch die sich sowohl das Service Netzwerk als auch das mobi-

UMTS-
Erweiterungen

⁴ Im UMTS-Kontext wird das häufig als Heimatumgebung (HE) bezeichnet.

le Gerät wechselseitig vergewissern, dass die verwendeten Schlüssel auch wirklich frisch, also nicht wiedereingespielt sind.

Integrität der Kontrollssignale:

Zum sicheren Aufbau von Verbindungen werden die Kontrollssignale mit einem Message Authentication Code (MAC) und dem Integritätschlüssel IK gesichert.

Kontrollierte Nutzung von Schlüsseln:

Das USIM kontrolliert, wieviele Daten bereits mit einem Verschlüsselungsschlüssels gesichert wurden. Es veranlasst eine erneute Authentifikation beim nächsten Verbindungsaufbau, falls das verschlüsselte Datenvolumen einen bestimmten Schwellenwert überstiegen hat.

Kontrollierte Nutzung der Schlüssellebenszeit:

Das Service Netzwerk erneuert die verwendeten Schlüssel, um deren Nutzungsdauer zu beschränken und zu kontrollieren.

Integrität und Vertraulichkeit der Kommunikationsdaten:

Die Kommunikation zwischen dem mobilen Gerät und dem Mobilnetz-Controller, also die Kommunikation über die Luftschnittstelle, erfolgt verschlüsselt und wird mit Prüfsummen zur Integritätskontrolle versehen. Im Gegensatz zu den mit 64 Bit bereits bei heutiger Technologie viel zu kurzen und damit zu schwachen GSM-Verschlüsselungsschlüsseln, verwendet UMTS 128 Bit Schlüssel für die vertrauliche Kommunikation.

Anzeige von fehlerhaften Authentifikationen:

Bei einer fehlerhaften Authentifikation wird dem Heimatnetz eine Fehlermeldung mit Fehlergrund zugestellt. Durch diesen neuen Dienst ist es unter UMTS insbesondere möglich, einen Maskierungsangriff eines nicht legitimen Netzes zu erkennen.

Im Folgenden gehen wir auf die UMTS-Sicherheitsdienste etwas genauer ein. Detailliertere Beschreibungen finden sich u.a. in den technischen Berichten der 3GPP Gruppe (vgl. <http://www.3gpp.org/>).

15.2.2 Authentifikation und Schlüsselvereinbarung

AKA

Die UMTS-Authentifikations- und Schlüsselvereinbarungsprotokolle AKA (Authentication and Key Agreement) setzen symmetrische Challenge-Response-Verfahren ein. Bei ihrem Entwurf wurde großer Wert auf eine weitestgehende Kompatibilität mit den GSM-Protokollen gelegt. Das AKA basiert darauf, dass zwischen dem Authentifizierungs-Zentrum (AC) der Heimatumgebung des mobilen Benutzers und dessen USIM ein geheimer

Schlüssel K vereinbart ist⁵ und beide Parteien die Authentifizierungsfunktionen f_1, f_2 , sowie die Schlüsselgenerierungsfunktionen f_3, f_4 und f_5 verwenden. Die genauen Verfahren sind nicht vorgeschrieben, jedoch hat die 3GPP in Anforderungen festgelegt, die die genutzten Verfahren erfüllen müssen.

Da die USIM und die Heiumgebung vom gleichen Netzwerk-Provider verwaltet werden, kann dieser die verwendeten Verfahren f_1, \dots, f_5 bestimmen. Die 3GPP empfiehlt die Verwendung eines Standardverfahrens genannt MILLENAGE, das als Basis den AES (siehe Abschnitt 7.5.5) einsetzt.

f_1, \dots, f_5

Authentifizierungs-Vektor

Bucht sich ein mobiler Teilnehmer bei einem Service Netzwerk ein, so muss dieses in die Lage versetzt werden, sich selbst und die Teilnehmer zu authentifizieren. Zu diesem Zweck verwendet UMTS die Authentifizierungs-Vektoren. Die Authentifizierungs-Vektoren erfüllen eine ähnliche Aufgabe wie die Triplets, die unter GSM verwendet werden. Liegen beim Einbuchen dem Service Netzwerk keine Authentifizierungs-Vektoren für den Teilnehmer vor, so sendet es eine Anfrage an das Authentifizierungs-Zentrum des Heimatnetzes des Teilnehmers.

Beim Empfang einer Anfrage nach Authentifizierungsdaten vom Service Netzwerk generiert das AC ein geordnetes Feld von n (typischerweise $n = 5$) Authentifizierungs-Vektoren (AV), das an das Service Netzwerk zurück gesendet wird. Ein AV enthält eine Challenge RAND, eine erwartete Antwort XRES, einen Verschlüsselungsschlüssel CK, einen Integritätsschlüssel IK und ein Authentifizierungs-Token AUTN. Diese Daten werden unter Nutzung des teilnehmerspezifischen 128-Bit-Schlüssels K und einem operatorspezifischen 16 Bit langen Authentication Management Field AMF, das z.B. die Lebensdauer der Schlüssel festlegen kann, wie folgt erzeugt.

Authentifizierungs-Vektor

1. Für jede Verbindung wird eine neue Sequenznummer SQN sowie eine 128-Bit lange Zufallszahl RAND erzeugt und mittels des Message Authentication Codes f_1 wird daraus der 64-Bit MAC-Wert

$$MAC = f_1(K, (SQN \mid RAND \mid AMF))$$

berechnet.

2. Mit dem Message Authentication Code f_2 wird die erwartete Antwort

$$XRES = f_2(K, RAND)$$

berechnet.

⁵ Die USIM wird von der Heiumgebung konfiguriert und der Schlüssel wird auf der USIM abgelegt.

3. Mit der Schlüsselerzeugungsfunktion f_3 wird der 128-Bit-Schlüssel

$$CK = f_3(K, RAND)$$

bestimmt.

4. Mit einer weiteren Schlüsselerzeugungsfunktion f_4 wird der 128-Bit-Schlüssel $IK = f_4(K, RAND)$ bestimmt.
5. Zur Verschleierung der Sequenznummer, von der auf die Benutzeridentität geschlossen werden kann, wird schließlich noch ein 48-Bit langer Anonymitätsschlüssel $AK = f_5(K, RAND)$ erzeugt.
6. Das Authentifikationstoken wird wie folgt konstruiert:

$$AUTN = (SQN \text{ xor } AK) \mid AMF \mid MAC$$

7. Mit den berechneten Daten wird abschließend der Authentifikations-Vektor AV, der an das Service Netzwerk gesendet wird, wie folgt zusammengestellt:

$$AV = RAND \mid XRES \mid CK \mid IK \mid AUTN.$$

Abbildung 15.7 fasst die beschriebenen Berechnungsschritte noch einmal zusammen.

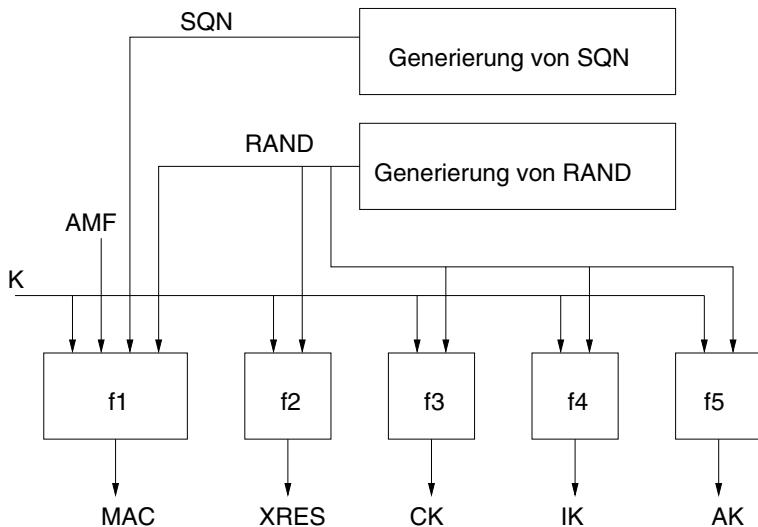


Abbildung 15.7: Erstellung eines Authentifikations-Vektors

Location Register (VLR) oder der Serving GPRS Support Node (SGSN) den nächsten Authentifikations-Vektor AV dieses Benutzers und sendet die darin enthaltene Zufallszahl RAND und das Token AUTN an das mobile Endsystem des Benutzers weiter. Die USIM des Geräts muss nun zur Challenge RAND die korrekte Antwort XRES berechnen und an das Service Netzwerk zurücksenden. Dazu arbeitet die USIM die folgenden Schritte ab:

1. Die USIM berechnet zunächst den Anonymitätsschlüssel

$$AK = f5(K, RAND)$$

und extrahiert damit die Sequenznummer SQN aus dem Token AUTN.

2. Als Nächstes wird der MAC-Wert

$$MAC' = f1(K, (SQN \mid RAND \mid AMF))$$

berechnet und mit dem in AUTN enthaltenen MAC-Wert verglichen. Bei Nichtübereinstimmung sendet die USIM eine Reject-Nachricht an das VLR bzw. den SGSN und teilt den Grund für den Abbruch des Authentifikationsprozesses mit.

3. Bei Übereinstimmung überprüft die USIM, ob die Sequenznummer SQN einen gültigen Wert besitzt. Dazu verwaltet eine USIM ein Feld von bereits verwendeten Sequenznummern (typischerweise ein Feld der Größe 32) und vergleicht die aktuelle mit den bereits erhaltenen. Details dieser Überprüfung sind in [2] zu finden. Ist die Sequenznummer ungültig, so wird ein Synchronisationsfehler an das VLR bzw. SGSN gemeldet. In diesem Fall wendet sich das Service Netzwerk an die Heimatumgebung des zu authentifizierenden mobilen Teilnehmers, um ein Feld von frischen Authentifikations-Vektoren zu erhalten.
4. Die USIM berechnet $RES = f2(K, RAND)$ und fügt diesen Wert in die Antwortnachricht an das Service-Netzwerk ein. Dieses vergleicht den Wert RES mit dem Wert XRES aus dem zugehörigen Authentifikations-Vektor. Bei Gleichheit extrahiert das Netzwerk die beiden Schlüssel CK und IK aus dem Vektor, um eine vertrauliche und gegen unauthorisierte Modifikationen geschützte Kommunikation aufzubauen.
5. Abschließend berechnet auch die USIM die zur Kommunikation zu nutzenden Schlüssel CK und IK:
 $CK = f3(K, RAND)$ und $IK = f4(K, RAND)$.

Die Berechnungsschritte der USIM fasst Abbildung 15.8 noch einmal zusammen.

Mit dem beschriebenen Protokoll werden die benötigten Schlüssel CK und IK zwischen dem USIM und dem Netzwerk vereinbart und der Benutzer

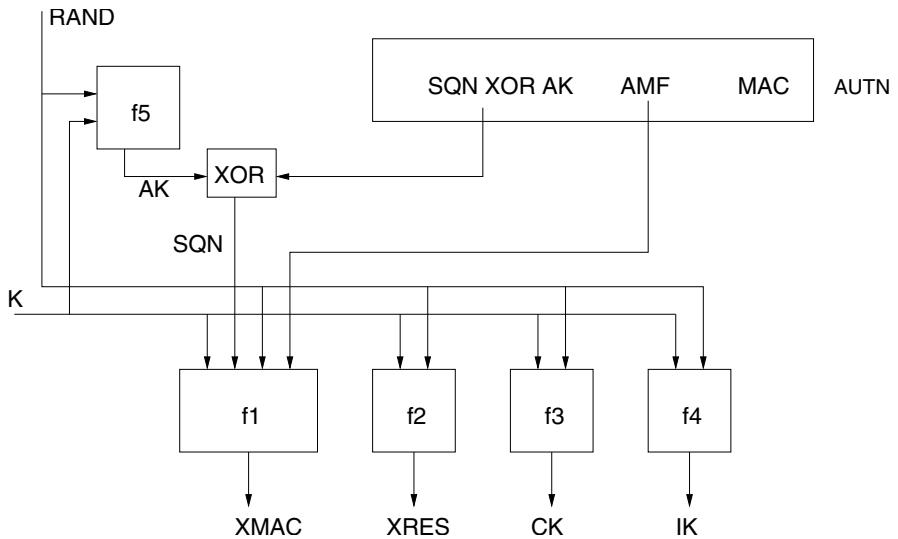


Abbildung 15.8: Berechnung der USIM-Antwort

authentifiziert sich gegenüber dem Netz. Mittels der Daten im AUTN authentifiziert sich die Heimatumgebung gegenüber dem mobilen Gerät, da zur Berechnung von AUTN die Kenntnis des geheimen Schlüssels K erforderlich ist. Der Schlüssel wird ja, wie oben beschrieben, sowohl bei der Erstellung des MAC-Wertes als auch bei der Erzeugung des Anonymitätsschlüssels AK benötigt. AUTN enthält darüber hinaus die zeitabhängige Sequenznummer SQN, so dass eine Wiedereinspielung erkannt werden kann. Mit der MAC-Verifikation ist es dem mobilen System möglich zu prüfen, ob die vom Netzwerk gesendete RAND auch tatsächlich in der Heimatumgebung generiert wurde, und ob die Heimatumgebung dem Netzwerk vertraut.

15.2.3 Vertraulichkeit und Integrität

f_8, f_9

Die Verfahren zur Verschlüsselung und zur Integritätsprüfung werden bei der Kommunikation zwischen dem USIM und den Netzwerk Controllern verwendet, die aber natürlich von verschiedenen Service Netzwerk-Providern verwaltet werden können. Die 3GPP hat hierfür festgelegt, dass Standardverfahren, die die von 3GPP spezifizierten (vgl. <http://www.3gpp.org/ftp/Specs/html-info/35201.htm>) Anforderungen erfüllen, oder das speziell entwickelte KASUMI-Verfahren⁶ [1] zu verwenden sind. Laut der 3GPP Forderung muss die Verschlüsselungsfunktion f_8 eine Stromchiffre und die Integritätsfunktion f_9 ein Message Authentication Code sein.

⁶ Kasumi ist japanisch und bedeutet Nebel.

KASUMI ist eine 64-Bit Blockchiffre mit einem 128-Bit Schlüssel. Um zu einer Stromchiffre zu gelangen, wird eine Variante des Output Feedback Modus (OFB) (siehe Seite 311) für den KASUMI verwendet. Zur Abwehr von Chosen-Plaintext-Angriffen (siehe Seite 359) wird der Initialisierungsvektor IV verschlüsselt und nur der verschlüsselte Wert IV' übertragen. Es gilt:

$$IV' = \text{KASUMI}(CK \ xor \ KM, IV),$$

wobei CK der vereinbarte Verschlüsselungsschlüssel und KM ein Wert zur Veränderung von CK ist. Die Integritätsfunktion f9 ist ein 32-Bit CBC-MAC (siehe Seite 371) mit KASUMI als zugrunde liegender Blockchiffre.

Fazit

Bei der Entwicklung der UMTS-Sicherheitsarchitektur baute man einerseits auf guten Erfahrungen auf, die mit den GSM-Sicherheitsdiensten gemacht wurden, und hat andererseits Lehren aus den bestehenden Sicherheitsschwächen von GSM gezogen. So trägt beispielsweise der Übergang von 64-Bit-Schlüsseln bei GSM auf 128-Bit-Verschlüsselungsschlüssel unter UMTS den beim heutigen Stand der Technik gestellten Anforderungen an geeignete Schlüssellängen Rechnung.

Schlüssel

Neben dem kurzen Schlüssel war die einseitige, nämlich nur geräteseitige Authentifikation ein weiteres wesentliches Defizit der GSM-Architektur. Diese hat in der Vergangenheit immer wieder zu Sicherheitsproblemen durch erfolgreiche Spoofing-Angriffe geführt (z.B. IMSI-Catcher-Angriffe). Mit den modifizierten Authentifizierungs-Vektoren und der Verwendung von Message Authentication Codes findet unter UMTS nun auch eine wechselseitige bzw. mehrseitige Authentifikation statt. Durch die korrekten AV-Daten authentifiziert sich nämlich nicht nur die Heimatumgebung gegenüber dem mobilen Benutzer, sondern es ist auch möglich, anhand der versendeten Daten die Vertrauenswürdigkeit des gerade genutzten Netzbetreibers zu überprüfen. Zu beachten ist aber, dass nicht der mobile Teilnehmer die Vertrauenswürdigkeit eines Netzbetreibers überprüft und bewertet, sondern dass dies durch den Betreiber des Heimatnetzes erfolgt. Als UMTS-Nutzer muss man sich hierbei also auf die Vertrauenspolicy seines Netzbetreibers verlassen.

Authentifikation

Mit der konsequenten Verwendung von Message Authentication Codes beim Verbindungsaufbau ist es unter UMTS auch möglich, Datenintegrität beim Austausch von Signalisierungsnachrichten zu gewährleisten. Damit kann verhindert werden, dass ein Angreifer beim Verbindungsaufbau Signalisierungsnachrichten gezielt modifiziert, um schwache Verfahren anstelle der gewünschten starken Verfahren zum Einsatz zu bringen. Eine erfolgreiche Wiedereinspielung von Authentifizierungsdaten wird durch die Verwendung

Integrität

offener Entwurf

der benutzerspezifischen Sequenznummern wirksam verhindert. Hervorzuheben ist schließlich auch noch, dass im Gegensatz zu GSM, das über lange Zeit geheime Verschlüsselungsverfahren verwendet hat, die 3GPP bei der Entwicklung von UMTS einen offenen Entwurf verfolgte. Alle Spezifikationen und Evaluationsberichte sind öffentlich zugänglich, was einerseits zur Erhöhung der Sicherheit beigetragen hat, andererseits aber auch zu einer Stärkung des Vertrauens in die tatsächliche Qualität der Sicherheitsdienste von UMTS führt.

Probleme

Problematisch für die nahe Zukunft wird sicherlich die Interoperabilität zwischen Mobilfunknetzen der zweiten Generation (GSM) und denen der dritten Generation (UMTS) sein. In einem Mischbetrieb, in dem beispielsweise das Netz UMTS und die Basisstation aber nur GSM unterstützt, kann es zu erfolgreichen Man-in-the Middle Angriffen kommen, da die Basisstation keine Integritätsprüfungen durchführt. Dadurch ist es einem Angreifer möglich, sich in einen Verbindungsaufbau einzuschalten und die Verwendung eines schwachen Verschlüsselungsverfahrens zu erzwingen.

Zu beachten ist auch, dass die Sicherheit der UMTS-Protokolle natürlich fundamental darauf basiert, dass die sensiven Daten der Authentifizierungsvektoren auf einem sicheren und vertrauenswürdigen Weg über das Core-Netzwerk ausgetauscht werden. Da dies unter UMTS über IP-Netze erfolgen wird, sind natürlich erhebliche Anstrengungen notwendig, um sowohl die Kommunikationspartner, das sind in diesem Szenario die Betreiber von Servicenetzen und die Betreiber von Heimatnetzen, wechselseitig zu authentifizieren, und einen sicheren Kommunikationskanal zu etablieren. Hier sind noch weitere Standardisierungsschritte notwendig.

15.3 Long Term Evolution (LTE) und SAE

SAE

Die System Architecture Evolution (SAE) Architektur wird bereits seit 2008 im Rahmen des 3GPP (3rd Generation Partnership Project) erarbeitet (vgl. u.a. [134]). Das Ziel ist es, das paketvermittelnde Kern-Netz der 2G bzw. 3G-Mobilfunknetze zu einem sehr leistungsfähigen All-IP-Netz weiterzuentwickeln. SAE stellt dazu mobile Breitbanddienste bereit, vereinheitlicht unterschiedliche paketbasierte Dienste wie GPRS/UMTS und das US-amerikanische CDMA2000, führt eine Konvergenz von Zugangstechnologien herbei, bietet ein Handover-Konzept, um transparent zwischen 3GPP und Nicht-3GPP-Netzen, wie beispielsweise WLAN, umschalten zu können, und definiert ein einheitliches Sicherheits-Framework mit einheitlichen Accounting- und Billing-Konzepten. SAE/LTE ist die 4te Generation der Mobilfunknetze, durch die ein mobiler Internet-Zugang von hoher Leistungsfähigkeit (mobiles Internet) unabhängig vom Zugangsnetz (WLAN, WiMAX, GERAN/UTRAN, etc.) zur Verfügung gestellt wird.

Herzstücke der 3GPP-Arbeiten sind das weiter entwickelte Kernnetz, der **EPC** (Evolved Packet Core), das mit dem 3GPP Release 8 im Jahr 2008 eingeführt wurde, sowie das weiter entwickelte Funkzugangsnetz **LTE** (Long Term Evolution). Beide Komponenten sind Bestandteil der neuen Mobilfunk-Gesamtarchitektur, dem **EPS** (Evolved Packet System). EPS ist ein Ende-zu-Ende-System, das neben den Endgeräten der Nutzer UE (User Equipment) sowohl 3GPP-Funkzugangstechniken wie LTE, GSM, GPRS oder UMTS, als auch Nicht-3GPP-Zugangstechniken, wie WLAN oder Wi-MAX unterstützt (siehe Abbildung 15.9).

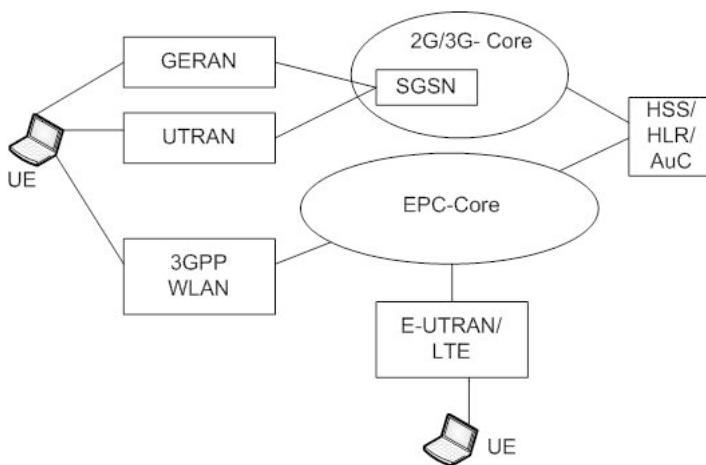


Abbildung 15.9: Grob-Architektur des Evolved Packet System

Wie in der Abbildung 15.9 verdeutlicht, umfasst das Evolved Packet System die GERAN⁷-, UTRAN⁸-, sowie E-UTRAN⁹-Schnittstellen. E-UTRAN ist das drahtlose Funkzugriffsnetz (Radio Access Network (RAN)), das die LTE-Technologie implementiert. Es verbindet die Endgeräte mit dem Mobilfunknetz. Der Evolved Packet Core (EPC) entspricht den SSS und OMC-B-Subsystemen von UMTS und stellt die Verbindung zu drahtgebundenen IP-Netzen her.

Das neue Kernnetz unterstützt damit, anders als die früheren Mobilfunkgenerationen, verschiedene Zugangsnetze und ermöglicht zudem ein flexibles Interworking zwischen den Technologien. So kann beispielsweise ein mobiler Nutzer von LTE zu WLAN und von dort ohne Unterbrechung beispielsweise zu UMTS wechseln.

⁷ GSM-Funknetz

⁸ Funknetz für WCDM/HSPA, wobei WCDM die Luftschnittstelle für UMTS implementiert.

⁹ Evolved UMTS Terrestrial Radio Access Network

Die EPS-Architektur umfasst mehrere Domänen.

1. Der **Circuit Core** stellt Netzkomponenten bereit, um verbindungsorientierten Diensten ein GSM/GPRS/UMTS-Netz zur Verfügung zu stellen.
2. Der **Packet Core** stellt Netzkomponenten für paketvermittelnde Dienste von GSM/GPRS/UMTS/LTE-Netzen bereit und unterstützt insbesondere auch Quality-of-Service-Anforderungen.
3. Das **IMS** (IP Multimedia Subsystem) enthält die Netzkomponenten, die benötigt werden, um Multimedia-Sessions, die über SIP (Session Initiation Protocol) abgewickelt werden, zu unterstützen.
4. Das **User-Management** verwaltet die Daten registrierter Nutzer.

15.3.1 EPC und LTE

LTE

Bereits 2004 wurde mit der Entwicklung des LTE-Standards für die Funk-schnittstelle begonnen. LTE ermöglicht sehr hohe Datengeschwindigkeiten von bis zu 100 Mb/s (theoretisch bis zu 326,4 Mb/s) beim Downlink und bis zu 50 Mb/s (theoretisch bei bis zu 86,4 Mb/s) beim Uplink. LTE nutzt das Frequenzspektrum von 1,4 MHz bis 29 MHz und unterstützt sowohl FDD als auch TDD. Um eine robuste Datenübermittlung zwischen LTE-Basisstation und Endgerät anzubieten, nutzt LTE die OFDM-Technik (Orthogonal Frequency Division Multiplexing). Ein Schwerpunkt von LTE liegt in der guten Unterstützung von Mobilität, also dem Handover. Zudem unterstützt LTE die flexible Funknetzplanung, so dass Zellen mit Signalisierungsreichweiten von bis zu 6 km sehr gute Leistung aufweisen und bei geringerer Leistung eine Reichweite von bis zu 100 km möglich ist. LTE ist vollständig IP-basiert, also durchgängig paketvermittelnd, und ist interoperabel mit 2G und 3G-Standards wie GSM oder auch UMTS.

LTE ist somit nicht einfach eine Erweiterung von UMTS, so dass die Einführung von LTE mit erheblichen Kosten verbunden ist, da ein neues Funkzugriffsnetz (RAN) etabliert werden muss. Das bedeutet, dass neue Endgeräte und neue Antennen erforderlich sind, um die neue Funkschnittstelle zu unterstützen. Seit Sommer 2010 haben deutsche Netzbetreiber damit begonnen, LTE auszurollen, jedoch ist auch im Jahr 2018 noch keine flächendeckende, bundesweite Verfügbarkeit von LTE gewährleistet. LTE-Endgeräte sind in Deutschland bereits seit Ende 2010 auf dem Markt.

EPC-Komponenten

Die Architektur des EPC und des LTE-Zugangsnetzes besteht aus folgenden Komponenten (vgl. Abbildung 15.10).

eNodeB

1. **eNodeB** (eNB): diese Komponente spezifiziert die LTE-Basisstation, die Dienste für die drahtlose Kommunikation zwischen dem Mobilfunknetz

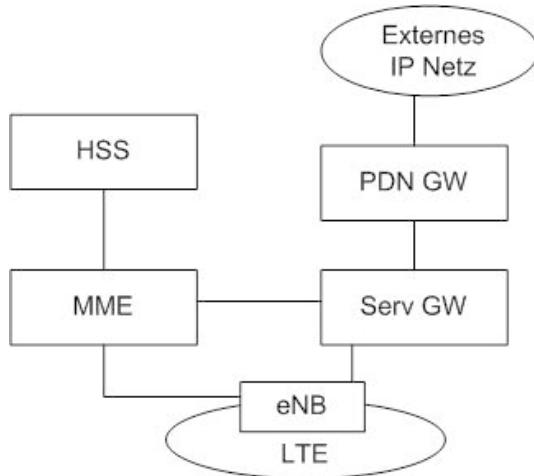


Abbildung 15.10: Architektur des EPC und LTE-Zugangsnetzes

und den Endgeräten zur Verfügung stellt. Das heißt, Endgeräte nutzen das drahtlose LTE-Zugangsnetz über eine eNodeB, um Zugang zum Mobilfunknetz zu erhalten. In einem LTE-Netz können sich mehrere tausend Basisstationen befinden; ein Handover zwischen den Stationen erfolgt über die X2-Schnittstelle. Die LTE-Komponente ist der Endpunkt der Verschlüsselung und des Integritätsschutzes bei der Kommunikation über die Luftschnittstelle. Die erforderlichen Schlüssel erhält die Komponente von der MME (s. u.). Jede LTE-Basisstation ist mit einer MME verbunden.

2. Die **MME** (Mobile Management Entity) entspricht dem SGSN und zum Teil auch dem VLR bei UMTS. Eine MME hat die Aufgabe, LTE-Basisstationen, die eNodeBs, zu steuern, die Endgeräte (UEs), die in den Zellen der verwalteten eNodeBs eingebucht sind, zu verwalten (Tracking Area Update¹⁰) und Signalisierungsnachrichten auf der Kontroll-Ebene zu verarbeiten. Zudem übernehmen sie das Mobilitäts- und Sicherheitsmanagement innerhalb von LTE, aber auch übergreifend mit anderen Netzen. Dazu authentisieren sie Endgeräte beim Einbuchen (optional auch zu späteren Zeitpunkten, wie bei UMTS), und erzeugen die erforderlichen Sitzungsschlüssel für Verschlüsselung und Integritätsschutz. Hierauf wird weiter unten noch genauer eingegangen.

MME

Jede MME ist mit einem HSS verbunden, so dass ein Zugriff auf relevante Daten von registrierten Nutzern, wie Authentisierungs- und Autorisierungs-Credentials möglich sind.

¹⁰ Dies entspricht dem Location Update bei UMTS.

HSS

3. Das **HSS** (Home Subscriber Service) entspricht dem HLR und dem AuC bei UMTS. Es verwaltet die sicherheitskritischen Daten registrierter Nutzer, wie die IMSI und die autorisierten Dienste des Nutzers, sowie auch den SIM/USIM-Schlüssel. Das HSS kommuniziert in SAE nicht mehr mittels SS7 mit den anderen Komponenten des Netzes, sondern es wird das IP-DIAMETER-Protokoll (vgl. RFC 3588) verwendet.

Service-Gateway

4. Die eigentlichen IP-Pakete, die die Nutzdaten transportieren, werden durch das **Service-Gateway** und das **Packet Data Network** (PDN) mit dem PDN-Gateway verarbeitet. Das Service-Gateway ist für die Ressourcenverwaltung und das Routen der Nutzerdatenströme in einem geographischen Gebiet zuständig. Zudem puffert es Daten, falls Endgeräte *idle* sind, unterstützt das Handover sowie Roaming der eigenen Nutzer beim Besuch eines Gastnetzes inklusive Accounting und sorgt für die Einhaltung von QoS Anforderungen.

Das PDN-Gateway verbindet die Endgeräte mit anderen, externen paketvermittelnden Datennetzen und stellt insbesondere die Internet-Verbindung über IP zur Verfügung (vgl. Abbildung 15.11). Zu einem Zeitpunkt kann ein Endgerät mit verschiedenen PDNs und darüber mit unterschiedlichen IP-Netzprovidern verbunden sein. Die PDN-Gateway-Komponente vergibt IP-Adressen, unterstützt sowohl IPv4 als auch IPv6 und führt Paketfilterungen und Policy-basierte Kontrollen durch.

Die IP-Verbindung des Endgerätes mit IP-Netzen ist getrennt von der IP-Verbindung, über die die Netzbetreiber innerhalb des EPS Managementdaten für das Signalisieren, Roaming etc. austauschen. Damit können die Datenverbindungen innerhalb des EPS als ein separat geschütztes, privates IP-Netz betrieben werden.

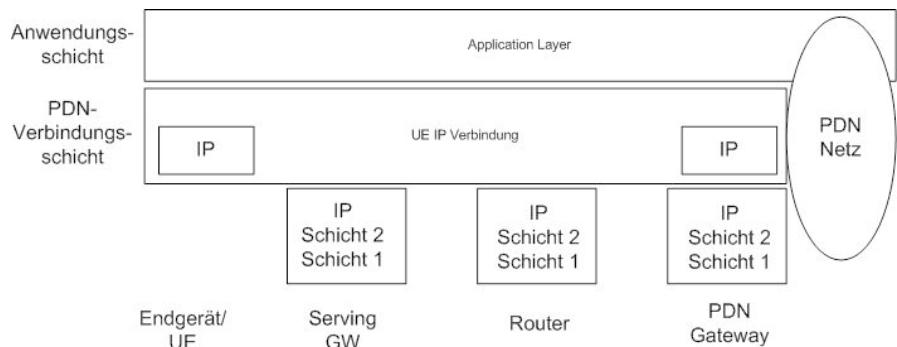


Abbildung 15.11: IP-Verbindung über das PDN-Gateway

15.3.2 Interworking

Da ein Hauptziel von SAE/LTE ein nahtloses Interworking zwischen verschiedenen Technologien ist, definiert der Standard Interworking-Szenarien und beschreibt, wie die LTE-Komponenten mit den 2G/3G-Komponenten bzw. WLAN/WiMAX-Komponenten zusammen arbeiten (vgl. Abbildung 15.12). Aus Sicherheitssicht ist hierbei beispielsweise das Zusammenwirken zwischen dem Home Location Register (HLR), das in 2G/3G die nutzerspezifischen Daten verwaltet, und dem HSS unter LTE von Interesse. Bei GSM/GPRS/UMTS greifen die SGSN-Komponenten für das Roaming direkt auf die Informationen aus dem HLR zu. Bei einer Zusammenarbeit mit LTE-Komponenten müsste somit ein Zugriff auf die HSS-Information erfolgen und HLR und HSS müssten konsistent gehalten werden. Hierfür sieht der Standard jedoch keine Vorgaben vor, so dass offen bleibt, wie eine einheitliche Sicht auf die sicherheitsrelevante Information hergestellt wird. Die Spezifikation geht davon aus, dass das HLR ein Teil des HSS ist. Weiterhin ist aus sicherheitstechnischer Sicht bemerkenswert, dass beim Roaming zwischen LTE und anderen Netzen das Roaming über den Service-Gateway stattfinden muss. Damit ist das Service-Gateway eine zentrale Komponente des Netzbetreibers, an der der Betreiber seine definierten Policies durchsetzen und Policy-basierte Kontrollen durchführen kann, oder aber an dem auch ein Monitoring des Netzverkehrs möglich ist.

Interworking

Beim Interworking zwischen 3GPP und Nicht-3GPP-Netzen wird zwischen vertrauenswürdigen und nicht vertrauenswürdigen Nicht-3GPP-Netzen unterschieden, wie beispielsweise ein öffentliches WLAN. Zudem unterscheidet man zwischen der Unterstützung von Netzwerk-basierten und von Client-basierten Mobilitätsdiensten. Bei Client-basierten Ansätzen ist es erforderlich, dass die Endgeräte Mobile-IP implementieren und damit einen IP-Tunnel zum PDN-Gateway aufbauen und das Zugangsnetz tunneln. Bei netzbasierten Ansätzen erfolgt das Mobilitätsmanagement von Mobile-IP transparent für die Endgeräte durch dedizierte Netzdienste. Bei einer Verbindung zu einem nicht vertrauenswürdigen, externen Netz wird eine IPSec-Verbindung zwischen dem jeweiligen Endgerät und der vom Netzbetreiber bereit gestellten ePDN-Komponente (evolved PDN) aufgebaut. Innerhalb des Betreibernetzes ist diese Komponente mit den anderen Komponenten des Netzes verbunden. Abbildung 15.12 stellt vergröbert die Gesamtheit der Komponenten und deren Verbindungen dar. Bei der Beschreibung sicherheitsrelevanter Prozesse gehen wir weiter unten noch auf einige Komponenten und deren Zusammenspiel genauer ein.

Nicht-3GPP

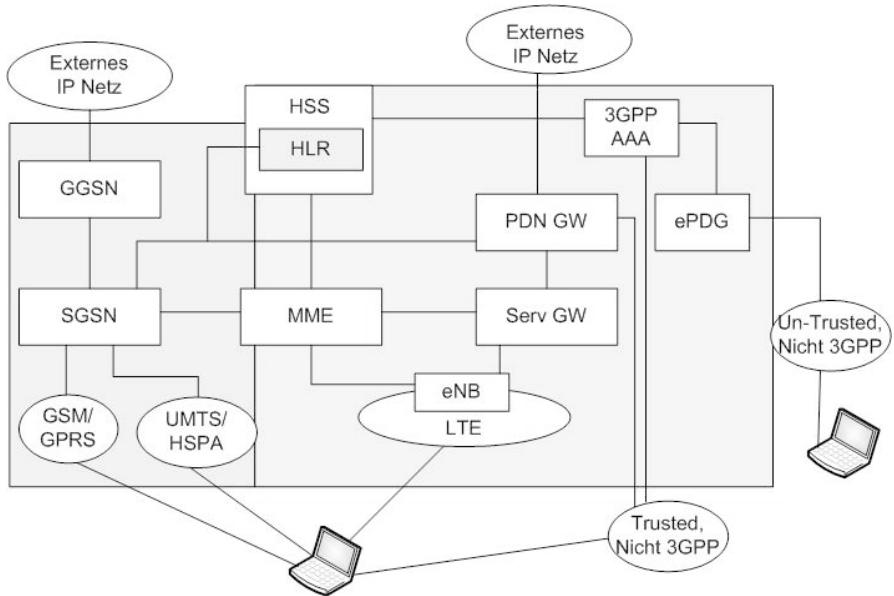


Abbildung 15.12: Überblick über die SAE/LTE-Komponenten

15.3.3 Sicherheitsarchitektur und Sicherheitsdienste

Sicherheitskonzepte

SAE/LTE übernimmt im Wesentlichen die Konzepte der UMTS-Sicherheitsarchitektur (vgl. Abschnitt 15.2.1), wobei die aus UMTS bekannten Verfahren zur Authentisierung und Schlüsselvereinbarung (AKA) auch für ein sicheres Interworking zwischen 3GPP und Nicht-3GPP-Netzen in einer adaptierten Form verwendet werden. Zum Schutz der Teilnehmeridentität verwendet SAE/LTE ebenfalls temporäre Identitäten, wie die M-TMSI im MME, und erzwingt bei jedem Wechsel der Basisstation (der eNodeB) oder auch beim Übergang vom *idle* in den *active*-Status das Aushandeln frischer Schlüssel, um einen stärkeren Schutz gegen kompromittierte Basisstationen anzubieten.

Schutz-Domänen

Domänen

Entsprechend der 3GPP-Spezifikation werden verschiedene Schutzbereiche (Domains) identifiziert. Sie entsprechen den bereits für UMTS eingeführten Domänen. In jeder Domäne werden spezifische Sicherheitskonzepte eingesetzt. Abbildung 15.13 veranschaulicht die Sicherheitsdomänen in der SAE/LTE-Architektur.

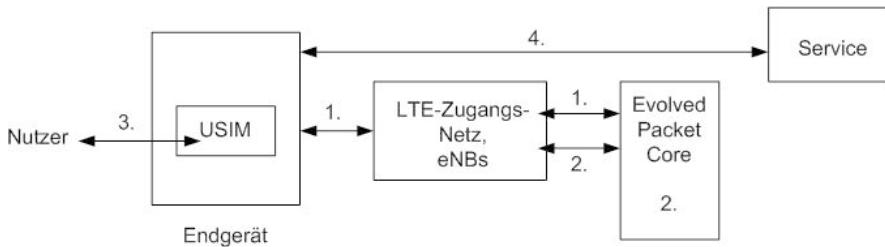


Abbildung 15.13: Sicherheitsdomänen der SAE/LTE-Architektur

- Zugangsdomäne:** Sie umfasst die Sicherheitskonzepte, um den Zugang von Endgeräten zum EPS abzusichern. Dazu gehören eine wechselseitige Authentisierung von Nutzer und Netzwerk, Konzepte zur Wahrung der Privatsphäre des Nutzers, Integritäts- und Vertraulichkeitsschutz sowohl der Nutzdaten des Benutzers als auch der Signalisierungs- und Managementdaten auf der Luftschnittstelle. Darauf wird weiter unten noch genauer eingegangen.
- Netzwerk-Domäne:** Sie umfasst diejenigen Sicherheitskonzepte, die Netzbetreiber einsetzen, um eine sichere Kommunikation zwischen den Komponenten des Netzes, sowohl in seinem eigenen Netz als auch Betreiber-übergreifend zu gewährleisten.
- Nutzer-Domäne:** Die Domäne umfasst Konzepte, um Nutzer gegenüber dem Endgerät zu authentisieren und einen abgesicherten Zugang des Nutzers zu seinem Endgerät zu gewährleisten, z. B. durch Nutzung von PINs.
- Anwendungs-Domäne:** In dieser Domäne werden anwendungsspezifische Sicherheitsdienste bereit gestellt, um eine Ende-zu-Ende-Absicherung zwischen dem Endgerät des Nutzers und dem Service-Anbieter, dessen Service er nutzt, umzusetzen.

Schlüssel

Ein wichtiges Ziel der SAE/LTE-Sicherheitsarchitektur ist es, differenzierte Schlüssel zu generieren, so dass für unterschiedliche Aufgaben konsequent unterschiedliche Schlüssel verwendet werden. Schlüssel werden zudem auch regelmäßig erneuert, d. h. dass jeder Schlüssel nur eine begrenzte Lebensdauer besitzt. Beim Netz-Zugang, dem so genannten E-UTRAN-Zugang, erfolgt deshalb nicht nur eine wechselseitige Authentisierung durch das EPS-AKA-Protokoll, sondern es werden auch die benötigten fünf Schlüssel generiert. Die wechselseitige Authentisierung erfolgt analog zu UMTS

Schlüssel

zwischen der USIM und dem Netz unter Rückgriff auf Authentisierungsvektoren (AVs), die das HSS generiert. Darauf gehen wir weiter unten noch etwas genauer ein.

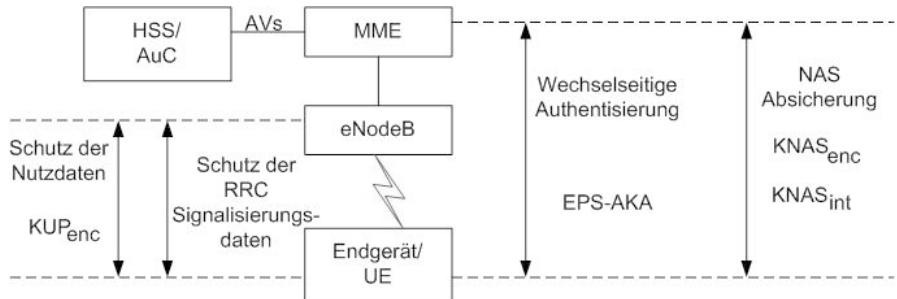


Abbildung 15.14: Überblick über die Sicherheitsarchitektur von LTE

Wie Abbildung 15.14 verdeutlicht, werden Schlüssel zu Absicherung von Signalisierungsdaten zwischen dem Endgerät (UE) und dem MME verwendet. Hierzu werden der Integritätschlüssel $KNAS_{int}$ sowie der Verschlüsselungsschlüssel $KNAS_{enc}$ benötigt¹¹.

Die Signalisierungsdaten, die über die Luftschnittstelle zwischen dem Endgerät und der eNodeB-Komponente (Basisstation) ausgetauscht werden, werden mit den Schlüsseln $KRRC_{int}$ bzw. $KRRC_{enc}$ geschützt¹².

Wie zuvor schon bei GSM/GPRS und UMTS werden auch bei SAE/LTE die Nutzdaten auf der Luftschnittstelle (user plane) zwischen UE und eNodeB verschlüsselt, wozu der Schlüssel KUP_{enc} verwendet wird. Die Spezifikation empfiehlt einen geschützten Transport dieser Daten zwischen den Komponenten eNodeB und Service-Gateway, beispielsweise durch die Etablierung eines IPSec-Tunnels; eine verschlüsselte Übertragung ist jedoch nicht vorgeschrieben. Ebenso wie bei UMTS werden auch bei LTE nur Signalisierungsdaten integritätsgeschützt, und zwar auf den beiden Ebenen der Netzwerk-Kontrolle (network access plane), also auf der Luftschnittstelle sowie auch bei der Verbindung zwischen UE und MME. D.h. Nutzdaten auf der User-Plane werden aus Performanzgründen auch in SAE/LTE nicht integritätsgeschützt.

Wechselseitige Authentisierung und Schlüsselvereinbarung: EPS-AKA

EPS-AKA

Analog zu UMTS wird ein Endgerät authentisiert, wenn es sich bei einem Provider einbucht. Das EPS-AKA-Protokoll baut auf dem UMTS-AKA-Protokoll auf und wird im Folgenden skizziert:

¹¹ NAS: Non-Access Stratum

¹² RRC: Radio Resource Control

1. UE → MME: Übertragen der IMSI zum MME.
2. MME → HSS: Anfordern von Authentisierungsvektoren AV vom HSS unter Angabe der IMSI, der Serving-Network-Identity (SN-ID) und des Netzwerktyps (z. B. E-UTRAN).
3. HSS prüft, ob die MME überhaupt berechtigt ist, AVs für das Netzwerk mit der Identität SN-ID anzufordern. Wie diese Überprüfung zu erfolgen hat, ist in der Spezifikation aber nicht festgelegt.

Das HSS erzeugt Authentisierungsvektoren unter Rückgriff auf den Schlüssel K , der zur IMSI gehört und sowohl im HSS als auch auf der USIM gespeichert ist, einer Sequenznummer, einer Zufallszahl $RAND$ und der Identität des Netzes SN-ID, in dem sich der Nutzer eingebucht hat. Abbildung 15.15 verdeutlicht die Schritte des HSS. Bemerkenswert ist, dass im Unterschied zu UMTS nicht die beiden Schlüssel CK bzw. IK übertragen werden, die zur Verschlüsselung und zur Integritäts sicherung benötigt werden, sondern das HSS erzeugt einen weiteren Schlüssel, den K_{ASME} . In dessen Berechnung fließen CK , IK und insbesondere auch die Netzwerk-Identität des Providers mit ein, so dass für unterschiedliche Netzbetreiber verschiedene Schlüssel erzeugt werden, um das Missbrauchspotential von Schlüsselmaterial zu begrenzen.

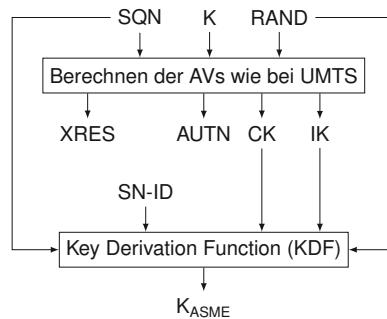


Abbildung 15.15: Erzeugung der Schlüssel und Authentisierungsvektoren in EPS-AKA

4. HSS → MME: Das HSS sendet die berechneten Authentisierungsvektoren an die MME zurück: $\text{EPS-AV} = (\text{AUTN}, \text{RAND}, \text{XRES}, K_{ASME})$.

Durch ein so genanntes Separation-Bit im AMF-Feld des $AUTN$ -Tokens zeigt das HSS an, dass es sich um einen Zugang zu einem EPS-Netz handelt. In den nachfolgenden Schritten muss das zu authentisierende Endgerät ebenfalls den Schlüssel K_{ASME} berechnen, um die erforderlichen fünf Schlüssel für die Kommunikation ableiten zu können.

5. MME → Endgerät: Übertragen von *AUTN* und *RAND* aus dem EPS-AV zum Endgerät (Challenge).
6. Das Endgerät prüft wie bei UMTS die Authentizität des Netzes unter Nutzung der *AUTN*-Informationen. Ist das Netzwerk authentisch, berechnet das Endgerät auf der Basis seines USIM-Schlüssels *K*, der Zufallszahl *RAND* und der Sequenznummer aus *AUTN* einen Wert *RES* sowie den Schlüssel *KASME*.
7. Endgerät → MME: Übertragen der berechneten Antwort *RES* zur MME (Response).
8. MME: Die MME vergleicht *RES* und *XRES* aus dem EPS-AV. Bei Übereinstimmung ist das Endgerät authentisiert, anderenfalls wird ein Fehler angezeigt und das MME kann eine erneute Authentisierung initiieren.

Bei einem Handover zwischen MMEs eines Netzproviders werden die nicht verbrauchten EPS-Authentisierungsvektoren weitergegeben. Bei einem Handover von einem MME zu einem SGSN (UMTS) können die AVs jedoch nicht weitergegeben werden, da sie nicht die erforderlichen Schlüssel *CK*, *IK* enthalten; d.h. ein SGSN kann mit den EPS-AVs nicht arbeiten. Umgekehrt können aber beim Handover von einem SGSN zu einem MME des gleichen Netzbetreibers die UMTS-AVs an den LTE-MME weitergegeben werden, und beim Wechsel zurück an den gleichen SGSN auch wieder zurückgegeben werden. Eine Weitergabe der UMTS-AVs innerhalb des SAE/LTE-Netzes an andere MMEs ist jedoch gemäß der Spezifikation nicht zulässig.

Schlüsselhierarchie und Verschlüsselungsverfahren

Schlüsselhierarchie

Durch die in SAE/LTE eingeführten, aufgabenspezifischen Schlüssel ergibt sich eine Schlüsselhierarchie (vgl. Abbildung 15.16). Die Schlüssel werden jeweils mit der Key-Derivation-Funktion (KDF) abgeleitet.¹³

Verfahren

Die Spezifikation schreibt vor, welche Verschlüsselungs- und Integritätskontrollverfahren durch die verschiedenen SAE/LTE-Komponenten unterstützt werden müssen bzw. sollten. Die Komponenten UE, MME und eNodeB sollten die Verfahren EEA0, 128 EEA1, 128-EEA2 bzw. 128 EIA1 und EIA2 unterstützen. Dabei entspricht EEA0 dem NULL-Verfahren, der EEA1 und der EIA1 dem Verschlüsselungsverfahren SNOW 3G¹⁴, während EEA2 dem 128 Bit AES im CTR-Modus und das Integritätsverfahren EIA2 dem 128-Bit AES im CMAC-Modus entspricht. SNOW 3G ist eine Stromchiffre, die von der 3GPP zur Umsetzung der UMTS-Verfahren f8 (Verschlüsselung) und f9 (Integrität) spezifiziert wurde. Diese Imple-

¹³ Die KDF ist in <http://www.3gpp.org/ftp/specs/html-info/33401.htm> spezifiziert.

¹⁴ Siehe u. a. http://www.gsmworld.com/documents/snow_3g_spec.pdf

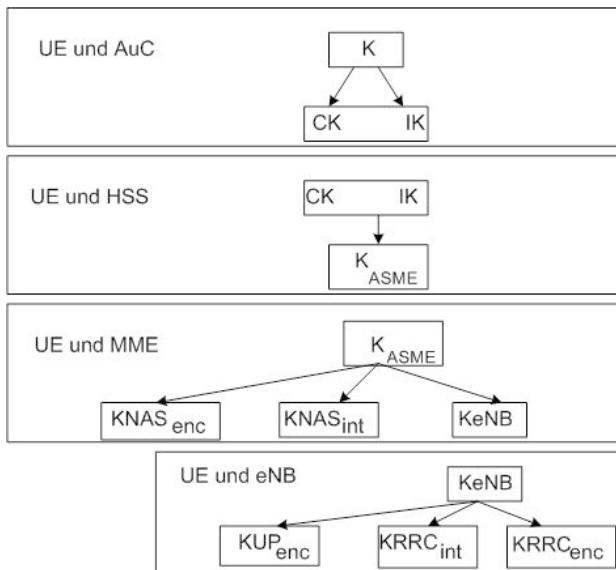


Abbildung 15.16: Schlüsselhierarchie in SAE/LTE

mentierungen sind als UEA2 bzw. UIA2 bekannt; sie nutzen SNOW zur Generierung des Schlüsselstroms. SNOW 3G arbeitet auf 32-Bit Blöcken und kann sowohl 128-Bit als auch 256-Bit-Schlüssel verwenden. UIA2 erzeugt einen 32-Bit MAC-Wert des Eingabestroms, ebenfalls basierend auf SNOW.

Die SAE/LTE-Spezifikation der Sicherheitsarchitektur fordert ergänzend, dass in Zukunft auch Schlüssellängen bis 256 Bit unterstützt werden müssen. Das UE sollte zudem das NULL-Integritätsverfahren unterstützen, das dann jedoch nur bei nicht-authentisierten Notfall-Anrufen für NAS bzw. RRC-Signalisierungsnachrichten verwendet werden darf.

Zur Absicherung der NAS-Verbindungen zwischen UE und MME legt das MME bzw. der Betreiber des MME eine Liste von Verfahren fest, die das MME unterstützt. Bei einer Signalisierungsanfrage vom UE wählt das MME die Verfahren aus, die zu verwenden sind, und teilt diese der UE mit. Bei einem Handover kann es deshalb ggf. zu Problemen kommen, wenn die neue MME die gewählten Verfahren nicht unterstützt; in einem solchen Fall müssen neue Verfahren verabredet werden. In analoger Weise wählt die eNodeB-Komponente die Verfahren aus, die zur Absicherung der Signalisierungsverbindung zwischen UE und eNodeB zu verwenden sind.

Analog zu UMTS wird auch in der SAE/LTE-Spezifikation keine Vorgabe zur sicheren Kommunikation zwischen den Netzkomponenten im Core-Netz gemacht. Auch wenn hier nach wie vor noch häufig proprietäre Netze verwendet werden, so sollten dennoch die IP-Verkehrsströme auch innerhalb

des Core-Netzes geschützt übertragen werden. Innerhalb der Netzdomäne eines Betreibers können IPSec-ESP-Verbindungen zwischen den Komponenten etabliert werden, und Betreiber- und Domänen-übergreifend sollten die Gateways der Netze mittels IPSec-Tunnel sicher verbunden werden.

Ein Nutzer soll laut Spezifikation in die Lage versetzt werden, sein Endgerät so zu konfigurieren, dass die Nutzung von Diensten nur dann möglich ist, wenn spezifizierte Sicherheitskonzepte zur Anwendung kommen. Empfohlen wird, dass es dem Nutzer ermöglicht werden soll, die USIM-Authentisierung zu aktivieren bzw. explizit auch zu deaktivieren.

15.3.4 Sicheres Interworking

Interworking zwischen GERAN/UTRAN und E-UTRAN

Interworking 3GPP

Ein Hauptziel von SAE/LTE ist es, die Delay-Zeiten beim Handover zwischen den verschiedenen Technologien so gering wie möglich zu halten, um eine nahtlose Kommunikation aufrecht zu erhalten. Eine einfache Situation liegt vor, wenn das mobile Endgerät bereits in einem SAE/LTE-Netz eingebucht war, von dort in einen GERAN/UTRAN-Netzbereich gewechselt ist und wieder in das E-UTRAN-Netz zurückwechselt. Für einen solchen Fall wird der etablierte Sicherheitskontext, der insbesondere den gemeinsamen Schlüssel K_{ASME} enthält, gecacht und kann einfach wieder aufgenommen werden.

Liegt hingegen beim Wechsel aus einem GERAN/UTRAN-Netz noch kein E-UTRAN-Sicherheitskontext vor, so wird der Sicherheitskontext des 3G-Netzes auf einen E-UTRAN-Kontext abgebildet. Dazu sendet der SGSN-Knoten des 3G-Netzes die Schlüssel CK und IK zum MME und diese berechnet daraus unter Nutzung einer festgelegten Mappingfunktion den Schlüssel K_{ASME} . In der umgekehrten Richtung erfolgt ein analoges Mapping, d. h. das MME berechnet zunächst aus dem K_{ASME} mit der Mappingfunktion die Schlüssel CK und IK und sendet diese beim Handover an den SGSN. Zu beachten ist, dass bei diesem Vorgehen ein kompromittiertes GERAN/UTRAN-Netz auch das E-UTRAN kompromittieren kann, so dass die Spezifikation dringend empfiehlt, möglichst schnell nach dem Handover einen vollständigen EPS-AKA-Protokoll durchlauf durchzuführen.

Interworking zwischen Nicht-3GPP und E-UTRAN

3GPP,
Nicht-3GPP

Als Beispiel wird das Interworking zwischen einem nicht vertrauenswürdigen WLAN-Zugang und E-UTRAN betrachtet. Bucht sich ein Teilnehmer über ein WLAN in ein E-UTRAN-Netz ein, so kommen spezielle Komponenten zum Einsatz, um den Teilnehmer zu authentisieren. Um Zugriff auf das interne EPS-Netz zu erlangen, muss sich der Teilnehmer aus dem nicht vertrauenswürdigen WLAN an die ePDG-Komponenten des EPS wenden, die als sicherer Zugangspunkt zum Netz agiert. Dazu wird ein IPSec-Tunnel

zwischen dem UE und der ePDG-Komponente etabliert. Eine wechselseitige Authentifikation zwischen dem UE und dem AAA-Server des Netzbetreibers und ein Schlüsselaustausch werden mittels IKEv2 initiiert, wobei sich die ePDG-Komponente mit einem Zertifikat ausweist, während die UE sich mit den Informationen aus der USIM in üblicher Weise authentisiert, also mit dem geheimen Schlüssel K aus der USIM.

Zur Authentisierung wird ein adaptiertes EAP-AKA-Protokoll zwischen dem 3GPP-AAA-Server und der UE abgewickelt. Hierfür erhält der AAA-Server den ESP-AV vom HSS/AuC. Das EAP-AKA-Protokoll wird als Bestandteil des IKEv2-Protokolls ausgeführt und nach erfolgreicher Authentisierung wird ein IPSec-Tunnel zwischen dem UE und dem ePDG etabliert, so dass das unsichere WLAN getunnelt wird. Abbildung 15.17 veranschaulicht im oberen Teil die Authentisierung mittels IKE und EAP-AKA und im unteren Teil den etablierten IPSec-Tunnel zwischen UE und ePDG.

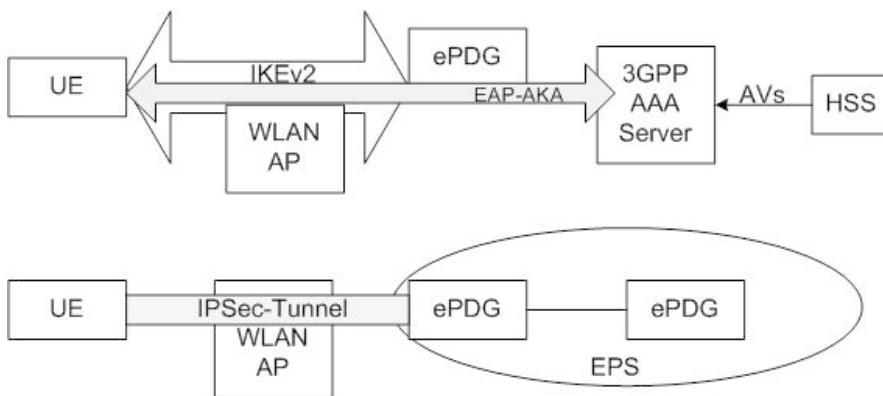


Abbildung 15.17: IPSec-Tunnel zwischen UE und ePDG

Fazit

Mit SAE/LTE ist ein sehr guter Schritt zu einer langfristig funktionsfähigen, performanten, mobilen Breitbandkommunikation gemacht worden. Das sehr ambitionierte Ziel wird erreicht, eine leistungsstarke und gleichzeitig möglichst sichere Infrastruktur für die Mobilkommunikation der 4. Generation zu etablieren, die 3GPP und Nicht-3GPP-Netze vereint und ein nahtloses Roaming und Handover zwischen den unterschiedlichen Technologien ermöglicht. Auch im Bereich der Sicherheitskonzepte hat eine Weiterentwicklung stattgefunden, wobei gleichzeitig bewährte 3G-Konzepte, wie die UMTS-Authentisierung und Schlüsselvereinbarung (AKA), die Nutzung von temporären Identitäten, Pre-Shared-Secrets in USIM-Chipkarten etc. beibehalten wurden. Die aufgabenspezifischen fünf verschiedenen Schlüssel stellen zudem eine Verbesserung der Sicherheit dar.

Fazit

Das Schadenspotential wird durch die konsequente Auftrennung verkleinert, die Schlüsselhierarchie erlaubt ein differenziertes Schlüsselmanagement mit Schlüsselerneuerungsprotokollen. Die eingesetzten Schlüssel und Verfahren entsprechen dem Stand der Kunst. Weitere Verfahren können konfiguriert werden, und ein Übergang auf Schlüssel, die länger als 128 Bit sind, ist bereits vorgesehen. Um Tracking und Tracing zu erschweren, werden die Signalisierungsnachrichten verschlüsselt und mittels Hashwerten geschützt, so dass Manipulationsversuche aufgedeckt werden können. Durch konsequente und frühzeitige Schlüsselerneuerung und Authentisierung wird die Gefahr manipulierter Basisstationen deutlich reduziert.

Schwächen der Spezifikation liegen nach wie vor in dem Fehlen von verbindlichen Anforderungen zur Absicherung der Nutzdaten im Kernnetz oder auch beim Austausch von Signalisierungs- und Managementdaten im Kernnetz. Nutzdaten bleiben auf der Luftschnittstelle aus Performanzgründen nach wie vor ohne Integritätsschutz, was für Sprach- und Multimediatenströme akzeptabel erscheint, für die Datenkommunikation jedoch problematisch ist. Um ein effizientes, nahtloses Handover zwischen 2G/3G und E-UTRAN zu ermöglichen, werden abgeschwächte Authentisierungen toleriert, wodurch eine Verwundbarkeit entsteht, wenn nicht sehr schnell nach dem Handover eine vollständige EPS-AKA-Authentisierung durchgeführt wird.

Ausblick

Als Ausblick ist festzuhalten, dass das Netz der Zukunft, das Future Internet, durchgehend IP-basiert sein wird, selbstorganisierende Funknetze und adaptive softwarebasiert (re)konfigurierbare Luftschnittstellen bereit stellen und zur Erhöhung der Leistungsfähigkeit über optische Netzkomponenten, intelligente Antennen und Sensoren verfügen wird. Die erhöhten Anforderungen an die Flexibilität und Anpassungsfähigkeit des Netzes der Zukunft wird durch neue Protokolle und adaptive Dienst- und Managementarchitekturen erfüllt werden. Zur Verbesserung der Abdeckung und Versorgung werden verstärkt neue Konzepte wie multi-hop, ad-hoc und Sensor-Netze, die keine aufwändigen Infrastrukturmaßnahmen erfordern, integriert werden. Auch die Endgeräte, die die Nutzung des Netzes der Zukunft ermöglichen sollen, müssen sehr viel flexibler und leistungsfähiger werden. Mit neuen multimodalen Schnittstellen, neuen Werkstoffen für die Displays und mit flexibel anpassbaren Software- und Hardwarearchitekturen werden zukünftige Endgeräte eine Vielzahl von Funktionen in sich vereinen, die heute noch auf verschiedene Geräte aufgeteilt sind.

offene Fragen

Herausforderungen für die Akzeptanz und Nutzbarkeit des Future Nets liegen in der Lösung der Probleme, die sich aus dem nahtlosen Interworking über Netz-, Technologie- und Organisationsgrenzen hinweg ergeben. Auch wenn für das technologische Zusammenwachsen von Netzen bereits Lösungsvorschläge erarbeitet wurden, wird in der Praxis nicht jede technisch

machbare Netz-Kopplung sinnvoll einsetzbar sein. Hier spielen weitere Faktoren wie Kosten, Sicherheit und Qualität von Diensten eine Rolle. Im Gegensatz zu funktionalen Anforderungen, bei denen man beim Zusammenführen verschiedener Technologien durchaus einen Leistungsabfall oder Abstriche im Leistungsumfang in Kauf nehmen kann, ist dies bei Qualitätseigenschaften nicht akzeptabel. Um den Nutzern die geforderten und ausgehandelten bzw. auch bezahlten Sicherheits- und sonstigen Qualitätsattribute durchgehend garantieren zu können, sind noch weitere Forschungs- und Entwicklungsarbeiten notwendig.

15.4 Funk-LAN (WLAN)

Das Konzept der Funk-LANs (engl. *Wireless LAN*, *Wave-LAN*, kurz WLAN) wurde bereits in den achtziger Jahren entwickelt. Den Durchbruch im Hinblick auf eine breite Anwendbarkeit brachte aber erst der im Jahr 1997 festgelegte IEEE Standard 802.11, der seitdem kontinuierlich weiter entwickelt wurde.

15.4.1 Einführung

Der 802.11-Standard (siehe [86]) operiert auf dem lizenzierten gebührenfreien ISM (Industrial, Scientific, Medical) Funkspektrum zwischen 2,4 und 2,48 GHz. Bei der Nutzung dieses Spektrums kann es zu Störungen beispielsweise durch andere elektromagnetische Quellen kommen, die das gleiche Spektrum verwenden. Derartige Störungen können zum Beispiel durch medizinische Geräte, Haushaltsgeräte (u.a. Mikrowelle) oder elektronische Wegfahrsperren sowie durch Bluetooth-fähige Geräte verursacht werden. Um einen gewissen Grad an Unempfindlichkeit gegenüber derartige Störungen zu erzielen, werden im Standard (und auch seinen Weiterentwicklungen s.u.) spezifische Bandspreizverfahren verwendet. Die Signalreichweite beim 802.11-Standard beträgt ca. 30–150 Meter und die Signale durchdringen auch Mauern und feste Gegenstände.

Aufgrund der kostengünstigen Installation von WLANs, ihrer relativ großen Signalausbreitung und der hohen Datenraten, die mit den Weiterentwicklungen des Standards angeboten werden, setzt man WLANs heutzutage bereits vielfältig ein. Wichtige Einsatzgebiete sind neben Privathaushalten und Unternehmens-Infrastrukturen, in denen WLANs meist als Zugangsnets zum internen Netz eingesetzt werden, oder auch öffentliche Hot Spots in Geschäften, Hotels, Flughäfen oder Bahnhöfen. In jüngerer Zeit werden Funknetze zunehmend auch in Produktionsanlagen, bei der Vernetzung von medizinischen Geräten oder aber auch in der Fahrzeug-zu-Fahrzeug-Kommunikation eingesetzt. So wurde beispielsweise im Kontext

802.11

Einsatzgebiete

intelligenter Verkehrssysteme die Erweiterung 802.11p Als Standard für PKWs spezifiziert.

Durch den zunehmenden Einsatz von Funk-Netzen in sicherheitskritischen Umgebungen kommt der Frage nach der IT-Sicherheit in diesen Netzen eine bedeutende Rolle zu.

Die unterschiedlichen 802.11-Standards

Der Ausgangspunkt der Protokoll-Familie ist der 802.11-Standard, der aber mit nur 1-2Mb/s eine zu geringe Datenrate bot, so dass er nicht in der Breite Akzeptanz fand. Im Folgenden gehen wir kurz auf die wichtigsten Weiterentwicklungen dieses Grundstandards ein.

IEEE 802.11b

Der IEEE 802.11b-Standard arbeitet genau wie der Grundstandard 802.11 auf dem lizenfreien ISM-Band und ist daher mit diesem kompatibel. Die spezifizierte Brutto-Datenrate wurde gegenüber dem Grundstandard erhöht und beträgt bis zu 11 Mb/s. Netto werden jedoch nur ca. 5 Mb/s erreicht. Die zulässige Höchstleistung beträgt in Europa 100 mW, in USA bis zu 1000 mW.

IEEE 802.11a

Der IEEE 802.11a arbeitet auf lizenpflichtigen Frequenzbändern im 5 GHz Bereich. Der Standard ermöglicht eine Brutto-Datenrate von 6 - 54 Mb/s. Neben höheren Datenraten bietet er auch geringere Interferenzen als die auf dem ISM-Band arbeitenden Varianten. Der Standard IEEE 802.11a war nach seiner Verabschiedung im Jahre 1999 vorerst auf die USA beschränkt. Erst mit der Freigabe der Frequenzbänder im November 2002 durch die RegTP konnte er auch in Deutschland genutzt werden. Jedoch unterliegt seine Nutzung Beschränkungen. Anstelle der sonst üblichen 100 mW ist die Leistung auf 30 mW eingeschränkt und der Standard darf auch nur im Inhouse-Betrieb eingesetzt werden. Der Grund für die Beschränkungen sind mögliche Störeinflüsse für andere Geräte wie beispielsweise Radar- oder Satellitenübertragungsgeräte aus dem militärischen Bereich.

IEEE 802.11g

Der im Juni 2003 verabschiedete Standard IEEE 802.11g arbeitet ebenfalls im 2,4 GHz ISM-Band und ist daher abwärtskompatibel zu IEEE 802.11b. Er ermöglicht eine Brutto-Bandbreite von bis zu 54 Mb/s. Weiterhin bietet er gegenüber den beiden Standards a und b einen verbesserten Quality-of-Service (QoS), wodurch er gut für Multimedia- und Videostreaming-Anwendungen einsetzbar ist.

IEEE 802.11h

Die Erweiterung IEEE 802.11h wurde September 2003 veröffentlicht. Sie ist die europäische Variante des IEEE 802.11a und arbeitet ebenfalls in dem Spektrum von 5 GHz mit einer maximalen Bandbreite von 54 Mb/s. Besonderheiten dieser Erweiterung sind die dynamische Frequenzauswahl (DFS = Dynamic Frequency Selection) und die variable Sendeleistung (TPC = Transmit Power Control), die den IEEE 802.11h von 802.11a

unterscheidet. Als Sendeleistung sind max. 200 mW erlaubt. Ebenso wie der a-Standard, so ist auch 802.11h auf den Indoor-Bereich beschränkt.

Die Erweiterung IEEE 802.11i spezifiziert die Sicherheitsdienste für die IEEE 802.11-Familie. Der IEEE 802.11i wurde 2004 verabschiedet. Da dieser Standard die auch in den aktuellen WLANs gültigen Sicherheitsdienste spezifiziert, gehen wir im Folgenden darauf genauer ein.

Der Standard 802.11n erhält keine weiteren Sicherheitsfunktionen. Er ermöglicht eine höhere Datenrate von bis zu 589 MBit/s.

IEEE 802.11i

802.11n

15.4.2 Technische Grundlagen

Bevor wir vertiefen auf die Sicherheitsdienste des 802.11i Standards eingehen, werden in diesem Abschnitt einige technische Grundlagen eingeführt, die für das Verständnis der Sicherheitsproblematik rund um WLAN hilfreich bzw. zum Teil auch notwendig sind.

Funkübertragung

IEEE 802.11 spezifiziert zwei grundlegende Übertragungstechniken: eine auf Infrarot und eine auf Funkwellen basierende. Die Infrarotübertragung nutzt entweder diffuses Licht, welches an Wänden und Möbel reflektiert wird und eine Reichweite von ca. 10 m ermöglicht, oder gerichtetes Licht, falls eine Sichtverbindung zwischen Sender und Empfänger vorhanden ist. Infrarot als Übertragungstechniken hat sich jedoch nicht durchgesetzt, so dass hier nicht weiter darauf eingegangen wird. Für die Übertragungstechnik Funkwellen sind folgende zwei Varianten der Bitübertragungsschicht spezifiziert.

- FHSS (Frequency Hopping Spread Spectrum) Frequenzsprungverfahren: Beim Frequenzsprungverfahren FHSS wird die Trennung der Netze durch verschiedene Sprungfrequenzen erreicht. Der ursprüngliche Standard sah 79 Frequenzbänder in Europa und Nordamerika und 23 in Japan vor. Jedes Band hat eine Bandbreite von 1 MHz im 2,4-GHz-ISM-Band. Die Auswahl erfolgt mit Hilfe einer pseudo-zufälligen Sprungsequenz. Den Wechsel der Trägerfrequenz bezeichnet man als einen Hop.
- DSSS (Direct Sequence Spread Spectrum) Bandspreizverfahren: DSSS ist das heute am Häufigsten eingesetzte Verfahren. Die Bandspreizung DSSS wird im IEEE 802.11 durch den Einsatz eines so genannten Barker-Codes erreicht. Dessen wesentliche Merkmale sind die relativ hohe Störunempfindlichkeit und Toleranz gegenüber der Mehrwegausbreitung. Das zugrunde liegende Prinzip der Frequenzspreizung besteht darin, die Daten mit einer Pseudozufallsfolge (Pseudo-Random Numerical Sequence) mittels XOR zu verknüpfen.

Die Protokolle mit den niedrigeren Datenraten des IEEE 802.11b nutzen noch immer den Barker-Code zur Spreizung, die neueren Standards mit den Datenraten von 5,5 und 11 Mbit/s nutzen demgegenüber den Complementary Code Key (CCK) oder optional das Packet Binary Convolutional Coding (PBCC). IEEE 802.11b arbeitet auf festgelegten Frequenzen im ISM-Band. Die Frequenzen hängen von nationalen Regelungen ab. Insgesamt wurden 14 DSSS-Kanäle festgelegt, je nach Land werden jedoch nur 11 (USA/Kanada), 13 (Europa) oder aber 14 Kanäle (Japan) genutzt.

Infrastruktur- und Ad-hoc Modus

Infrastruktur-
Modus

802.11-WLANs können in zwei Modi betrieben werden. Im Ad-hoc-Modus kommunizieren die Teilnehmer mittels WLAN-Controllern über Punkt-zu-Punkt Verbindungen direkt miteinander. Es wird dazu ein spontaner Netzwerkverbund aufgebaut. Ein typisches Szenario hierfür ist die schnelle Kommunikation zwischen einigen wenigen Laptops während eines Projektmeetings.

Im Infrastruktur-Modus (vgl. Abbildung 15.18) kommuniziert ein Endgerät (WLAN-Client) drahtlos mit einem Access Point (AP), der eine Brücke zwischen dem funk- und dem drahtgebundenen Netz (z.B. dem lokalen Ethernet) schlägt. Beispiele für WLAN-Clients sind Laptops, Mobiltelefone

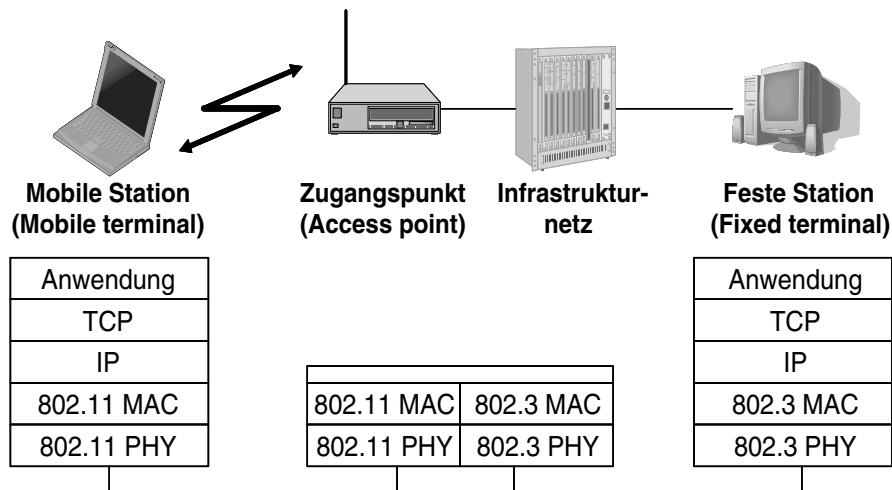


Abbildung 15.18: WLAN Netz im Infrastrukturmodus

oder stationäre PCs. Eine Kommunikation zwischen zwei WLAN-Clients in einem WLAN im Infrastrukturmodus erfolgt stets über den Access Point als Vermittler. Die Brückenfunktion, also die Umsetzung des Funkprotokolls in

ein drahtgebundenes Protokoll, wird durch Protokolle der ISO-OSI Schicht 2, dem Data Link Layer (vgl. Seite 100), erbracht.

Jeder Access Point gehört zu dem Verwaltungsbereich eines BSS (Basic Service Set), das mehrere Access Points verwalten kann. Ein WLAN-Client kann immer nur mit einem Access Point verbunden sein. Access Points haben jedoch nur eine begrenzte Sende- und Empfangsreichweite, die innerhalb von geschlossenen Räumen ca. 30 Meter und außerhalb bis ca. 150 Meter beträgt. Um eine größere Abdeckung zu erzielen, kann man verschiedene Access Points über Funk oder Kabel miteinander verbinden.

Medienzugang, Zugriffskontrolle

Bevor ein WLAN-Client im Infrastrukturmodus mit seiner Datenübertragung beginnt, muss er eine Assoziation zu einem Access Point herstellen. Dies erfolgt in drei Schritten. Im ersten Schritt wird der Kontakt aufgenommen. Jeder Access Point besitzt einen SSID (Service Set Identifier), der ihn identifiziert. Auf welche Weise eine Verbindung vom Endgerät zum Access Point hergestellt wird, d.h. ob das Endgerät die korrekte SSID seines AP kennen muss oder nicht bzw. auf welche Weise er sie erfährt, ist abhängig davon, in welchem Betriebsmodus das WLAN betrieben wird. Man unterscheidet die Modi *any* und *eingetragene SSID* bzw. *closed network*. Im Modus *any* akzeptiert der Access Point beliebige SSIDs, d.h. der WLAN-Client benötigt keine Kenntnisse über die SSID. Man spricht in diesem Fall auch von so genannten offenen WLANs (*open network*). Im Modus *eingetragene SSID* muss der WLAN-Client die SSID des Access Points kennen, um am WLAN teilnehmen zu können. In diesem Fall gibt es wiederum mehrere Varianten. Hat der Access Point seinen Broadcast-Modus aktiviert, so sendet er in festen Intervallen, so genannte Beacon-Frames, als Broadcast an alle potentiellen Empfänger des Sendebereichs. WLAN-fähige Endgeräte lauschen auf diese Frames und können den SSID des Access Points aus dem Frame extrahieren und verwenden.

Ein Client kann seinerseits aktiv nach den SSIDs eines bestimmten Access Points fragen, indem er Probe-Nachrichten versendet, die eine gesuchte SSID enthalten. Falls der Access Point entsprechend konfiguriert ist, wird er gemäß dem Standard mit einer Probe-Response antworten und in dieser seine SSID mitteilen.

Will man jedoch ein geschlossenes WLAN betreiben, ist es sinnvoll, dass der Access Point die Aussendung der Broadcastnachrichten unterdrückt; eine solche Option, sie wird als *cloaked mode* bezeichnet, wird von den meisten Herstellern angeboten, ist jedoch nicht standard-konform. Access Points, die sich im *cloaked mode* befinden, versenden ihr SSID aber nicht per Beacon-Nachrichten und antworten auch nicht auf Probe-Requests von

SSID

geschlossenes
WLAN

Endgeräten. Die SSID kann in diesem Fall als eine sehr einfache Form eines gemeinsamen Geheimnisses betrachtet werden; dies ist natürlich als einziger Authentifikationsmechanismus nicht ausreichend, wenn man wirklich unautorisierte Netznutzungen ausschließen möchte. Der Modus *cloaked mode* stellt nämlich keine ernsthafte Hürde für einen Angreifer dar, da die SSID des Access Points auch in weiteren Management-Nachrichten im Klartext enthalten ist. Ein Angreifer muss nur den Nachrichtenverkehr mithören, um die SSID in Erfahrung zu bringen.

stärkstes Signal

In der Regel verwendet ein Client denjenigen Access Point, der mit der größten Signalstärke sendet. Nach der Identifikation des AP erfolgt eine Authentifikation. Nach dem Authentifikationsschritt, dieser kann bei einem als offen (open) konfigurierten WLAN auch trivial ausfallen, wird das Endgerät als Knoten im Funknetz aufgenommen, wofür der AP spezielle Assoziations-Nachrichten versendet. Danach kann eine Kommunikation erfolgen.

ACL

Neben der Kenntnis des korrekten SSID des Access Points kann man den Zugang zum WLAN auch dadurch beschränken, dass Zugriffskontrolllisten (ACLs) konfiguriert werden. Um die Menge der berechtigten Benutzer einzuschränken, erlauben Access Points den Zugriff nur für diejenigen mobilen Teilnehmer, deren Netzwerkadresse, also die MAC-Adresse der Netzwerkkarte, dem Access Point bekannt ist. Diese Möglichkeit des Zugriffsschutzes eignet sich jedoch nur für kleine Installationen mit einem oder sehr wenigen Access Points, da diese MAC Adressliste auf jedem Access Point einzeln gepflegt werden muss. Somit steigt der Administrationsaufwand sehr stark an, wenn zusätzliche Access Points installiert werden. Der erreichbare Schutz durch diese Maßnahme ist zudem gering, da zum einen die MAC-Adresse im Klartext übertragen wird und zum anderen viele WLAN-Karten das Ändern der MAC-Adresse mittels Software ermöglichen. So ist es ein Einfaches für einen Angreifer, eine gültige MAC-Adresse abzuhören¹⁵, diese MAC-Adresse in seine WLAN-Karte zu übernehmen und damit erfolgreich einen Spoofing-Angriff durchzuführen.

Hot-Spot Zugang

Der drahtlose Zugang zu einem öffentlichen Hot Spot erfolgt heutzutage in der Regel über die Universal Access Method (UAM). Hierbei etabliert der WLAN-Client eine Verbindung zum Access Point (vgl. Abbildung 15.19) und erhält über einen DHCP-Server eine dynamische IP-Adresse zugewiesen. Mit dem Start eines Web-Browsers wird der Benutzer dann automatisch auf eine Startseite weiter geleitet, die ein Login unter Angabe des Benutzernamens und Passwortes erwartet. Die Verbindung vom Client zu dieser Startseite wird mittels TLS (siehe Seite 778) abgesichert, so dass die vom Benutzer übertragenen Daten verschlüsselt übertragen werden.

¹⁵ Hierfür stehen Tools, die Netzwerksniffer, zur Verfügung.

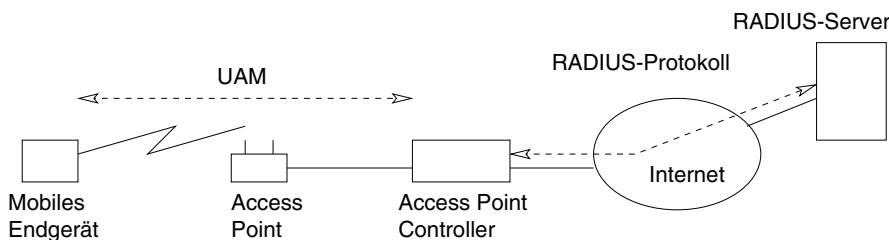


Abbildung 15.19: Hot-Spot Zugang

In einem solchen Hot-Spot-Szenario übernimmt der Access-Point in der Regel sowohl die Aufgaben des DHCP-Servers als auch die des Web-Servers, der das Login mit dem Benutzer durchführt. Die Credentials des Benutzers, also Kennung und Passwort, werden anschließend zu einem zentralen RADIUS-Server weitergeleitet, der die tatsächliche Authentifikation durchführt. Der Web-Server, der das Login abwickelt, authentifiziert sich hierbei in der Regel über die Vorlage seines Zertifikats.

Bei der Nutzung von UAM wird lediglich der Netzzugang mittels TLS abgesichert, der weitere Datentransfer über das drahtlose Netz erfolgt unverschlüsselt. Das bedeutet natürlich insbesondere, dass die Zugangspasswörte von mobilen Benutzern, die diese Anbindung nutzen, um auf unternehmensinterne Mail-Server mittels z.B. POP3 oder SMTP zuzugreifen, im Klartext übertragen werden. In derartigen Fällen sollte also unbedingt ein VPN zum unternehmenslokalen Netz aufgebaut werden, über das dann alle nachfolgenden Zugriffe abgesichert durchgeführt werden können.

Abgesicherter
Netzzugang

Bereits in den vorangegangenen Ausführungen wurden einige Sicherheitsprobleme kurz angerissen, die sich bei der Nutzung eines WLAN ergeben können. Im Folgenden werden wir auf allgemeine Sicherheitsprobleme im Zusammenhang mit der Nutzung von WLANs eingehen, bevor wir in Abschnitt 15.4.4 auf die 802.11i-Sicherheitsdienste eingehen werden.

15.4.3 WLAN-Sicherheitsprobleme

Die drahtlose Kommunikation mit der ungeschützten Funkübertragung birgt eine Reihe von Sicherheitsrisiken. Beim Einsatz von Funknetzen muss man sich stets darüber im Klaren sein, dass dies Broadcastmedien sind. Das bedeutet, dass das Abhören von Funkverbindungen im Gegensatz zu drahtgebundenen Netzen deutlich vereinfacht ist, da bereits eine herkömmliche Antenne dazu ausreicht, um den Datenverkehr auch durch Lauscher, die sich außerhalb des mit WLAN vernetzten Gebäudes aufzuhalten, abzufangen. Je nach Gelände und Stärke des Signals, haben die WLAN-Funkwellen eine Ausbreitung von ca. 150 Metern für die normale Nutzung. Mit Spezialanten-

Abhören

nen und -equipment ist es aber möglich, schwache Signale noch in 12,87 km (8 Meilen) Entfernung aufzuzeichnen¹⁶.

Zu beachten ist auch die Ausbreitung in vertikaler Richtung. Das ist dann von Relevanz, wenn das Gebäude mehrstöckig ist und evtl. auch andere Firmen oder Institutionen im gleichen Gebäudekomplex untergebracht sind. Ein Auffangen der Funksignale ist nur dann nicht möglich, wenn im Gebäude eine elektromagnetische Abschirmung vorgesehen ist, was aber höchstens in Ausnahmen der Fall sein dürfte.

Signalausbreitung

Das Eindringen und Mithören, aber auch das Verändern von Datenpaketen in einem Funknetz ist für Angreifer somit sehr viel einfacher als es in herkömmlichen drahtgebundenen Netzen der Fall ist. Während in drahtgebundenen Netzen der Angreifer direkten Zugriff auf das Kabel oder eine Anschlussbuchse besitzen muss, also bei einem LAN physisch in das entsprechend vernetzte Gebäude vorgedrungen sein muss, reicht es bei einer Funkkommunikation aus, sich in der Nähe des vernetzten Gebäudes aufzuhalten. Der Funkverkehr wird durch Mauern zwar behindert, aber nicht abgeschirmt, so dass ein Angreifer von der schon fast sprichwörtlichen nahen Parkbank oder aus dem Auto auf dem Firmenparkplatz etc. unbemerkt in das lokale Funknetz eindringen kann. Anzumerken ist dazu, dass sich ein Angreifer, der ein Funknetz abhört, für das keine weitergehenden Sicherheitsmaßnahmen getroffen sind, in Deutschland nicht strafbar macht, da der Angreifer keine Zugriffshürde zu überwinden hat.

Man-in-the-middle

Da WLAN-Endgeräte meist auf das stärkste Signal reagieren und zu dem sendenden Access Point eine Verbindung aufzubauen versuchen, ist es einfach, einen gefälschten AP zu installieren. Ein solcher Access Point besitzt die vollständige Kontrolle über die Kommunikationsverbindungen und es ist ihm ein Leichtes, Daten abzuhören, Daten einzuschleusen, zu modifizieren oder auch DNS-basierte Angriffe durchzuführen. Es sei hier bereits darauf hingewiesen, dass auch Verschlüsselungsdienste, die auf der Verbindungsebene ansetzen, natürlich keine wirkungsvolle Abwehr gegen derartige gefälschte APs darstellen, da damit lediglich der Verbindungs weg abgesichert, aber dem Endpunkt vertraut wird.

UAM

Bei öffentlichen Hot Spots, die die Universal Access Method als Zugangsprotokoll implementieren, kann das zur Konsequenz haben, dass der mobile Benutzer seine Credentials, also seine Kennung und das zugehörige Passwort diesem gefälschten AP preisgibt. Mittels TLS wird ja nur der Kommunikationsweg abgesichert, beim Empfänger, unter UAM ist dies der Web-Server, zu dem umgeleitet wurde, liegen diese Daten dann im Klartext vor. Auch wenn kein gefälschter AP als Man-in-the-Middle fungiert, ist diese Offenle-

¹⁶ siehe <http://www.spiegel.de/international/world/a-941262.html>

gung von Zugangsdaten bei Gastnetzen natürlich ein Problem. Die Gefahr, dass ein Benutzer unter UAM seine Daten an einen gefälschten AP weiterleitet, ist relativ groß, da der AP zwar sein Zertifikat vorweisen muss, aber Zertifikate in der Regel von Benutzern nicht manuell geprüft werden. Auf diese Problematik, die ja auch im Zusammenhang mit SSL bzw. TLS zu Tage tritt, hatten wir in dem entsprechenden Kapitel bereits hingewiesen.

Schließlich sind auch noch die Bedrohungen durch Denial-of-Service Angriffe zu benennen. Mit einem herkömmlichen, entsprechend konfigurierten Adapter (z.B. in einem Mobiltelefon oder Notebook) lassen sich Funknetze so stören, dass die angeschlossenen Endgeräte davon ausgehen, dass Interferenzen aufgetreten sind. Die Geräte werden in einem solchen Fall ihre Übertragung für eine festgelegte Zeitspanne verzögern. Falls der präparierte Adapter kontinuierlich seine Störsignale aussendet, kann das Funknetz seine normale Datenübertragung nicht fortsetzen. Werden drahtlose Netze für zeitkritische Datenübertragungen eingesetzt, so kann dies natürlich zu gravierenden Problemen führen.

DoS

Im Rest des Abschnitts gehen wir nun ausführlich auf die Sicherheitsdienste eines Funknetzes gemäß eines der 802.11 Standards ein.

15.4.4 WEP und WPA

Mit dem 802.11i-Standard wurden die Sicherheitsdienste spezifiziert, die unter dem Namen WPA2 (Wi-Fi Protected Access)¹⁷ in aktuellen Access Points und WLAN-Controllern umgesetzt werden. Der Standard hat aus den Schwächen der Vorläufer WEP und WPA gelernt und stellt eine deutliche Verbesserung der Sicherheit im Vergleich zu diesen dar. Um hierfür ein gewisses Verständnis zu wecken, gehen wir im Folgenden ganz kurz auf diese beiden Vorgänger-Standards ein, die aber nicht mehr von WLAN-Geräten unterstützt werden. Sie dienen hier lediglich zur Anschauung, um aufzuzeigen, wie auch internationale Standardisierungsorganisationen vor klassischen Sicherheitsfehlern nicht gefeit sind.

WEP

Integraler Bestandteil des IEEE-Standards 802.11 von 1997 war das Wired Equivalent Privacy-Protokoll (WEP) für eine vertrauliche Punkt-zu-Punkt-Kommunikation. Das Ziel bei der WEP-Spezifikation war es, wie der Name ja auch schon sagt, im Funknetz den gleichen Sicherheitslevel wie im drahtgebundenen lokalen Netz (z.B. Ethernet) zu erzielen. Es wurde also kein hohes Sicherheitsniveau angestrebt.

WEP

¹⁷ Siehe <http://csrc.nist.gov/publications/nistpubs/800-97/SP800-97.pdf>

Um eine vertrauliche Datenkommunikation zwischen einem WLAN-Client und dem Access-Point zu gewährleisten, setzte WEP eine RC4-basierte Stromchiffre ein, die einen 24-Bit langen Initialisierungsvektor IV zusammen mit einem geheimen Schlüssel K von 40 bzw. 104 Bit Länge verwendete. Der gemeinsame Schlüssel K war ein Pre-shared Secret, das manuell konfiguriert wurde und für alle Teilnehmer des Netzes gleich war. Ein Schlüsselmanagement war nicht vorgesehen. Zur Integritäts sicherung wurde ein CRC-Verfahren verwendet.

Protokollschrifte

Den Ablauf der WEP-Protokollschrifte zur verschlüsselten und integren Datenübertragung kann man wie folgt zusammenfassen:

1. Sender und Empfänger müssen ein gemeinsames Geheimnis, einen symmetrischen Schlüssel K kennen, um eine Nachricht M zu versenden.
2. Der Sender berechnet eine Prüfsumme von M , $\text{CRC}(M)$, die aber nicht von K abhängt, wählt einen 24-Bit langen Initialisierungsvektor IV und erzeugt mit der Stromchiffre RC4 und dem erweiterten Schlüssel $K' = IV \mid K$ einen Schlüsselstrom $\text{Key} = RC4(K')$.
3. Die Nachricht M sowie deren Prüfsumme $\text{CRC}(M)$ werden mit dem Schlüsselstrom per XOR verknüpft:

$$C = (M \mid \text{CRC}(M)) \oplus \text{Key}.$$
4. Der Kryptotext C und der Initialisierungswert IV werden schließlich über die Funkschnittstelle übertragen.
5. Der Empfänger verwendet seinerseits den gemeinsamen Schlüssel K und den erhaltenen IV, um mittels RC4 den Schlüsselstrom Key zu generieren.
6. Zur Entschlüsselung berechnet der Empfänger:

$$C \oplus \text{Key} = M \mid \text{CRC}(M').$$

 Das heißt, er erhält einen Klartext M und eine Prüfsumme $\text{CRC}(M')$, die aber nicht notwendig zum Klartext M passen muss, so dass dies noch geprüft werden muss.
7. Abschließend prüft der Empfänger die Prüfsumme, also

$$\text{CRC}(M') \stackrel{?}{=} \text{CRC}(M).$$

WEP-Schwächen

Authentizität

Die mit WEP spezifizierten Sicherheitsdienste waren sehr schwach. Zur Authentisierung wurden keine paarweisen Authentisierungsnachweise genutzt, sondern der allen WLAN-Clients bekannte Schlüssel K . Die Kenntnis dieses Pre-shared Secrets musste ein WLAN-Client in einem Challenge-Response Verfahren nachweisen. Da sich eine sehr große Zahl von Rechnern den ge-

meinsamen Schlüssel teilten, konnte man keinen der Rechner, geschweige denn einzelne Benutzer individuell authentifizieren. Bereits der Besitz einer WLAN-Karte mit dem korrekten Schlüssel ermöglichte einen Zugang zum Funknetz.

Durch die fehlende Authentifikation des Access Points gegenüber den Endgeräten war es Angreifern zudem sehr einfach möglich, Denial-of-Service (DoS) Angriffe gegen Endgeräte durchzuführen. Am einfachsten war es, durch ein stärkeres Signal (jamming) den gesamten Funkverkehr zu stören.

WEP sah kein Schlüsselmanagement vor, so dass eine Erneuerung des Schlüssels bei einer Kompromittierung, z.B. durch den Diebstahl einer WLAN-Karte, nicht über ein entsprechendes Protokoll automatisiert erfolgen konnte.

Zur Gewährleistung der Integrität einer zu übertragenen Nachricht M berechnete der Sender mittels des CRC-32-Verfahrens eine 32-Bit Prüfsumme der Daten. Das CRC-Verfahren ist ein sehr effizientes, lineares Verfahren, um Bitübertragungsfehler zu erkennen, aber es ist in keiner Weise dafür geeignet, die Integrität von Daten im Sinne kryptografischer Hashfunktionen (vgl. Abschnitt 8.1.1) zu gewährleisten. Unter WEP wurden die Nutzdaten zusammen mit der Prüfsumme unter Nutzung der Stromchiffre RC4 verschlüsselt. Die Linearitätseigenschaft des verwendeten Prüfsummenverfahrens zusammen mit den Eigenschaften einer Stromchiffre, die auch linear ist, führte dazu, dass ein Angreifer beliebigen Chiffretext manipulieren und gleichzeitig die Prüfsumme anpassen konnte, so dass der Empfänger der manipulierten Nachricht nicht mehr in der Lage war, die Veränderung zu erkennen.

Die unter WEP gesendeten Datenpakete wurden mit der Stromchiffre RC4 unter Nutzung eines 64 Bit bzw. 128 Bit symmetrischen Schlüssels verschlüsselt. Für beide Schlüssellängen galt aber, dass 24-Bit des Schlüssels als Klartext (das ist der Initialisierungsvektor IV) vorlagen, so dass nur ein 40-Bit bzw. 104-Bit-Schlüssel zum Einsatz kam. Da bei einer Stromchiffre niemals der gleiche Schlüsselstrom für verschiedene Nachrichten verwendet werden darf, wurde unter WEP der statische, gemeinsame Pre-Shared Schlüssel K für jedes Datenpaket um einen 24-Bit Initialisierungsvektor zu $K' = K \mid IV$ ergänzt. Da der IV aber nur 24 Bit lang war, dauerte es bei einem stark frequentierten Access Point mit zum Beispiel 11 MBit Datenrate nur wenige Stunden, bis Initialisierungsvektoren erneut verwendet wurden. Es wurde dann also ein Schlüssel wiederholt eingesetzt und die Vertraulichkeit ist nicht mehr gewährleistet.

Aus den Schwächen des WEP lassen sich unmittelbar die Anforderungen an einen guten Sicherheitsstandard ableiten:

DoS

Integrität

Vertraulichkeit

Lessons learned

- Es wird eine paarweise, wechselseitige Authentisierung von WLAN-Client und Access Point benötigt.
- Erforderlich ist ein Schlüsselmanagement, um Schlüssel nach Bedarf zu erneuern.
- Es ist eine starke Verschlüsselung nach dem Stand der Technik und mit standardisierten Verfahren vorzusehen.
- Es ist ein starker Integritätsschutz nach dem Stand der Technik und mit standardisierten Verfahren zu integrieren.

WPA

Um die im WEP-Standard aufgedeckten Sicherheitslücken zu schließen, wurde für den Nachfolgestandard 802.11i eine neue Sicherheitsarchitektur für WLANs spezifiziert. Mit der Einführung des 802.11i Standards sollte auch der AES zur Datenverschlüsselung integriert werden. Dies erforderte jedoch eine Änderung der Hardware, um die aufwändigen Ver- und Entschlüsselungen durchzuführen. Es musste deshalb eine Zwischenlösung konzipiert werden, um schnell und ohne Austausch von Hardware-Komponenten einen höheren Sicherheitsgrad zu erzielen als mit WEP.

WPA

Als Zwischenlösung wurde der Wi-Fi Protected Access (WPA) entwickelt. Die Zwischenlösung war sowohl aufwärtskompatibel zum neuen 802.11i-Standard als auch abwärtskompatibel mit bestehender Hardware. Das heißt, sie bot den großen Vorteil, auf der existierenden Hardware ausführbar zu sein, da lediglich ein Software-Update notwendig war.

TKIP

Hauptziele des WPA waren eine stärkere Verschlüsselung, ein verbessertes Schlüsselmanagement, eine Integritätssicherung einzelner Nachrichtenpakete sowie eine verbesserte Authentifikation. Zur Erfüllung der ersten beiden Ziele wurde das TKIP (Temporal Key Integrity Protocol) entwickelt. Bis zum Jahr 2013 durfte das TKIP-Verfahren auch im neuen Standard 802.11i als optionaler Modus genutzt werden.

Michael

Zur Verbesserung des Integritätsschutzes wurde eine spezielle Hashfunktion, das Michael-Verfahren, eingeführt, das aber nicht die Stärke von standardisierten Hashverfahren aufweist.

WPA endete 2013

Da WPA von Anfang an als Zwischenlösung konzipiert war und aufgrund seiner Abwärtskompatibilität nicht das erforderliche Sicherheitsniveau erreichen konnte, lief die TKIP Unterstützung in den WLAN-Geräten im Jahr 2013 aus. Im Folgenden gehen wir deshalb auf TKIP auch nicht mehr ein. Interessierte Leser seien auf frühere Auflagen dieses Buches verwiesen.

15.4.5 802.11i Sicherheitsdienste (WPA2)

Der IEEE Standard 802.11i (WPA2) umfasst folgende Sicherheitsdienste:

1. Der Standard 802.11i verwendet ein hierarchisches Schlüsselmanagement, dessen Ausgangspunkt ein paarweiser Masterschlüssel *PMK* ist, der individuell zwischen AP und WLAN-Client vereinbart wird.
2. Der Masterschlüssel kann manuell vereinbart oder dynamisch on-demand ausgetauscht werden. Zur dynamischen Vereinbarung eines solchen Schlüssels zwischen authentifizierten Partnern wird das 802.1X-Authentifizierungsframework verwendet, auf das wir in Abschnitt 15.4.6 noch kurz eingehen werden.
3. Aus dem paarweisen Masterschlüssel *PMK* werden Verbindungs-schlüssel *PTK* abgeleitet. Ein Bestandteil eines solchen Verbindungs-schlüssels ist ein temporärer Kommunikationsschlüssel *TK*, der zur Verschlüsselung der Datenpakete verwendet wird. Als Verschlüsse-lungsverfahren wird AES-CCMP mit einem 128 Bit Schlüssel und zur Integritätssicherung wird AES im CBC-Modus eingesetzt.

Schlüsselmanagement

Unter dem IEEE 802.11i Standard werden unterschiedliche Schlüssel verwendet, die in einer Schlüsselhierarchie organisiert sind. Man unterscheidet grob die Klasse der paarweisen Schlüssel, die individuell zwischen einem WLAN-Client und einem Access Point vereinbart sind, und die Klasse der Gruppenschlüssel. Die paarweisen Schlüssel werden zur Unicast-Kommunikation verwendet und die Gruppenschlüssel dienen zur Broadcast-Kommunikation. Abbildung 15.20 skizziert die Schlüsselhierar-chie unter WPA2. Auf die verschiedenen Schlüssel, sowie die erforderlichen Protokollschrifte zu deren Etablierung gehen wir nachfolgend noch genauer ein.

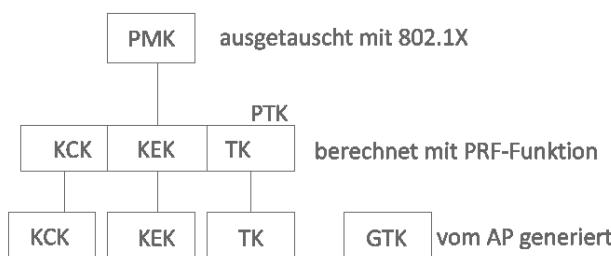


Abbildung 15.20: Schlüsselhierarchie unter WPA2

Abbildung 15.21 fasst die WPA2 Schlüssel und die Schritte zu deren Vereinbarung zusammen.

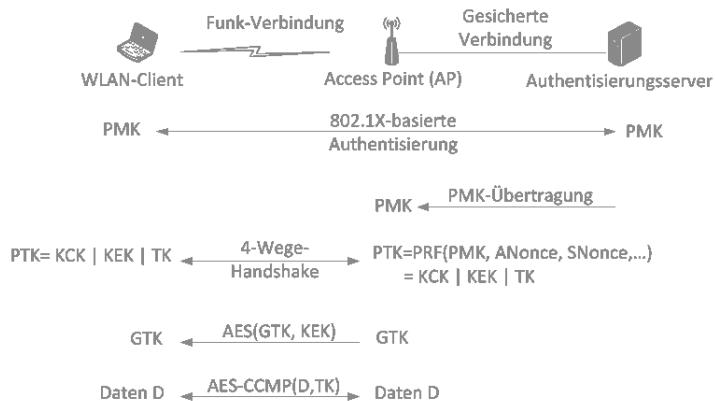


Abbildung 15.21: WPA2: Schlüsselberechnung, Verteilung und Nutzung

Pairwise Master Key (PMK)

PMK

Die Wurzel der Schlüsselhierarchie ist der so genannte Pairwise Master Key (PMK), der über einen Authentisierungs-Server während der Authentifikation (z.B. bei EAP-TLS) verteilt oder aber durch einen Administrator konfiguriert wird (Pre-Shared Key, PSK). Der PMK ist individuell für jedes Endgerät. Der 256 Bit lange Pre-shared Key kann entweder eine entsprechende Zufallszahl sein oder aus einer Passphrase berechnet werden, die durch den Administrator eingegeben werden muss und zwischen 8 und 63 Byte lang ist. Die Berechnung des Pre-Shared Keys aus einer Passphrase geschieht wie folgt:

$$\text{PMK} = \text{PBKDF2}(\text{passphrase}, \text{ssid}, \text{ssidLength}, 4096, 256),$$

wobei PBKDF2 ein Verfahren aus dem PKCS #5 v2.0, Standard, dem Password-based Cryptography Standard, ist. Mit dem PBKDF2-Verfahren werden die Passphrase, die SSID und die SSID-Länge durch 4096-maliges Anwenden einer Hash-Funktion zu einem 256 Bit langem Wert gewandelt.

802.1X

Wird kein Pre-Shared Key genutzt, so ist IEEE 802.1X (s.u.) zu verwenden, um den Master Key durch den Authentifizierungsserver zu generieren und sicher zum WLAN-Client zu transferieren. Der IEEE 802.11i-Standard schreibt hierzu kein Verfahren vor. Es können hierfür bekannte Protokolle, wie das RADIUS-Protokoll (vgl. Abschnitt 10.4.1) verwendet werden. Nach der erfolgreichen Authentisierung des WLAN-Clients überträgt der Authentisierungsserver den paarweisen Master-Key PMK auf gesichertem Weg zum Access Point (siehe Abbildung 15.21).

Pairwise Transient Key (PTK)

Bevor eine sichere Kommunikation zwischen dem WLAN-Client und dem Access Point aufgenommen werden kann, müssen noch Verbindungsschlüssel vereinbart und es muss sichergestellt werden, dass die Schlüssel korrekt und gültig sind. Weiterhin muß ein Gruppenschlüssel für die Abwicklung von Broadcasts verteilt werden.

WPA2 spezifiziert ein 4-Wege-Handshake-Protokoll zwischen dem WLAN-Client und dem Access Point, um den Verbindungsschlüssel PTK zu berechnen.

Der PTK besteht aus drei Einzelschlüsseln, $PTK = KCK \mid KEK \mid TK$. Der TK (Temporal Key) ist ein 128 Bit Schlüssel, der für die Datenverschlüsselung mittels AES-CCMP genutzt wird. Der 128 Bit KEK (Key Encryption Key) wird vom Access Point verwendet, um den gemeinsamen Gruppenschlüssel GTK des Netzes sicher an den WLAN-Client zu verteilen (vgl. Abbildung 15.21). Der 128-Bit Schlüssel KCK (Key Confirmation Key) wird zur Integritätssicherung und zum Authentizitätsnachweis beim Berechnen der PTK s während des 4-Wege-Handshakes verwendet.

Der temporäre Schlüssel PTK wird bei jedem Verbindungsauftakt neu berechnet. Die Berechnung basiert auf dem gemeinsamen Master-Key PMK sowie Nonces, die von den Partnern (hier der WLAN-Client und der Access Point) erzeugt und zur Schlüsselberechnung verwendet werden.

4-Wege-Handshake:

1. Der Access Point generiert eine Zufallszahl $ANonce$ und sendet diese an den WLAN-Client.
2. Der WLAN-Client generiert eine eigene Zufallszahl $SNonce$.
3. Der WLAN-Client berechnet den PTK unter Verwendung des PMK , der $ANonce$, der $SNonce$, der MAC-Adresse des AP ($addr_{AP}$) und seiner eigenen MAC-Adresse ($addr_{Client}$) mittels einer in der Spezifikation festgelegten Pseudo-Random-Funktion PRF :

$$PTK = PRF(PMK, ANonce, SNonce, addr_{AP}, addr_{Client}).$$

4. Der WLAN-Client sendet seine Zufallszahl $SNonce$ zusammen mit einem MIC-Wert (Message Integrity Code) als Nachweis, dass er im Besitz des gemeinsamen PMK s ist, an den Access Point zurück. Für den MIC-Wert gilt:

$$MIC = \text{AES-CBC}(ANonce, KCK)$$

Der KCK ist ein Bestandteil des PTK , in dessen Berechnung der PMK eingeht (s.o.).

PTK

4-Wege-Handshake

5. Dem Access Point liegen mit der $SNonce$ alle Daten vor, so dass er seinerseits den gemeinsamen $PTK = KCK | KEK | TK$ berechnen kann. Mit dem KCK kann der Access Point den MIC-Wert des WLAN-Clients und damit dessen Authentizität prüfen.
6. Der Access Point sendet eine PTK -Aktivierungsnachricht an den WLAN-Client und berechnet den MIC-Wert über diese Nachricht unter Nutzung seines KCK .
7. Der WLAN-Client prüft den MIC-Wert und bestätigt abschließend die Aktivierung des PTK .

Gruppenschlüssel

GTK

Um mit mehreren WLAN-Clients gleichzeitig kommunizieren zu können, wird ein Gruppenschlüssel benötigt, der zwischen dem Access Point und den Clients vereinbart wird, wozu die bereits ausgetauschten Schlüssel verwendet werden. Der AP erzeugt dazu einen temporären Gruppenschlüssel GTK und verschlüsselt diesen für jeden Client mit dem paarweisen Key Encryption Key (KEK), der ein Bestandteil des $PTKs$ ist, den er im 4-Wege-Handshake mit dem jeweiligen Client vereinbart hat. Den verschlüsselten GTK sendet der AP an den WLAN-Client. Die Gruppenschlüssel werden sowohl nach dem Ablauf eines bestimmten Zeitintervalls gewechselt, als auch beim Austritt eines Gruppenmitglieds aus der Gruppe. Der Gruppenschlüssel wird vom Access Point verwendet, um beispielsweise ARP-Broadcast-Nachrichten effizient an alle Clients des WLANs zu senden.

Kommunikationsschlüssel

TK

Mit den paarweise berechneten PTK -Schlüsseln werden auch die 128-Bit Kommunikationsschlüssel TK als Bestandteil des PTK berechnet. Für die vertrauliche Unicast-Kommunikation zwischen dem WLAN-Client und dem Access Point werden die zu übertragenen Daten D mittels AES-CCMP unter Verwendung des TK verschlüsselt (vgl. Abbildung 15.21).

AES-CCMP

CCMP

Der 802.11i-Standard schreibt vor, dass entweder TKIP oder der AES als Stromchiffre im CCMP (Counter-Mode-CBC-MAC Protocol) genutzt werden sollen. Seit 2012 wird durch die Erweiterung im 802.11ad auch der GCMP (Galois/Counter) Modus als zulässiger Modus akzeptiert. TKIP sollte jedoch aufgrund seiner Sicherheitsmängel heutzutage nicht mehr verwendet werden. Der AES-GCMP ist auf den Einsatz für kurze Entferungen im 60 GHz Band ausgerichtet. Die Umsetzung der 802.11ad Spezifikation ist auch unter dem Namen Wireless Gigabit bekannt. Der AES-CCMP ist eine 128-Bit Stromchiffre im Counter Mode (vgl. Seite 313) zusammen mit

dem CBC-MAC-Verfahren zur Berechnung eines Hashwertes. Der Klartext eines zu übertragenen Pakets wird in 128-Bit Blöcke aufgeteilt und die Blöcke werden mittels der Stromchiffre AES-CTR und dem Schlüssel TK verschlüsselt (vgl. Abbildung 15.22).

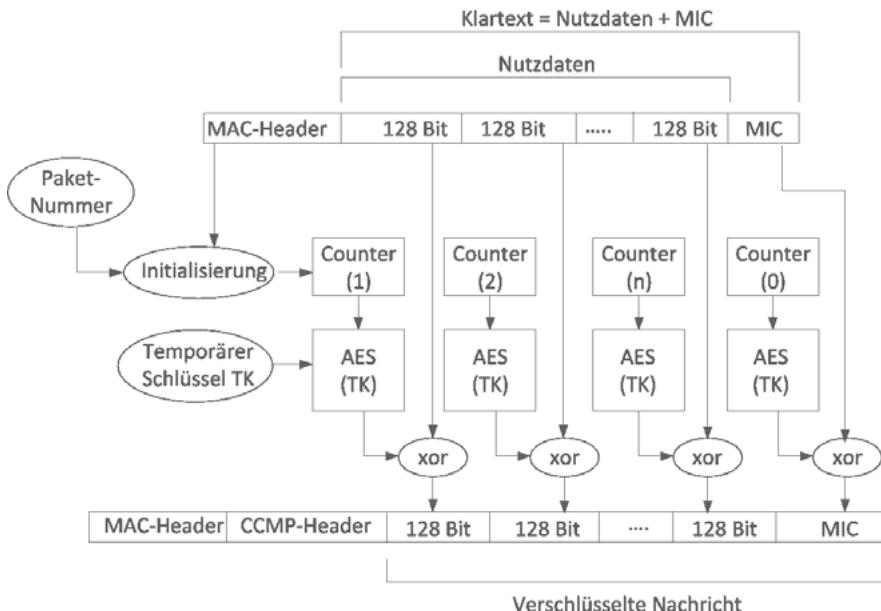


Abbildung 15.22: AES-CCMP Datenverschlüsselung

Der Schlüssel TK wird sowohl zum Verschlüsseln als auch zur Integritätsberechnung eingesetzt. Unter Nutzung des CBC-MAC-Verfahrens wird für das zu übertragende Paket ein Hashwert berechnet, in den der Header des Pakets, die Länge des Headers und das Payload des Pakets einfließen. Der so berechnete CBC-MAC-Wert wird mit dem Kryptotext konkateniert und an den Empfänger gesandt.

WPA2 Sicherheit

Der 802.11i Standard verwendet mit dem 802.1X/EAP-Framework ein Protokoll zur Ende-zu-Ende Authentifikation, in das auch starke Authentifizierungsmechanismen einfließen können. Im Gegensatz zum ursprünglichen WEP-Standard bietet WPA2 einen starken Integritätsschutz basierend auf AES-CBC mit einem 128 Bit MIC-Wert.

Mit der eingeführten Schlüsselhierarchie ausgehend von paarweisen Masterschlüsseln (PMK), über paarweise temporäre Schlüssel (PTK), die bei

jeder Verbindung erneuert werden, bis zu den eigentlichen Kommunikationsschlüsseln TK erfüllt WPA2 die Forderung nach der Nutzung von unterschiedlichen Schlüsseln für unterschiedliche Aufgaben. WPA2 sieht ein Schlüsselmanagement vor, das eine Schlüsselerneuerung nach dem Ablauf einer Zeitspanne oder auch abhängig vom Verschlüsselungsvolumen erlaubt.

Zur Verschlüsselung wird der Verschlüsselungsstandard AES-128 als Stromchiffre verwendet. Der eingesetzte Counter-Modus gewährleistet, dass kein Schlüssel erneut zum Einsatz kommt, da der TK erneuert wird, sobald der Zähler seinen Maximalwert erreicht hat.

Die Sicherheit des WPA2 gilt als sehr gut. ein Sicherheitsbeweis, siehe [78], hat keine Protokollschwächen offenbart. Umso bemerkenswerter war die Meldung über Sicherheitsschwächen im WPA2-Protokoll-Design, die im Oktober 2017 für einiges Aufsehen gesorgt hat. Darauf gehen wir weiter unten noch genauer ein. Die aufgedeckte Schwäche stellt jedoch nicht die Analysen in Fragen, die dem Sicherheitsbeweis zugrunde liegen. Es zeigt aber einmal mehr eindrucksvoll auf, wie schwierig es ist, vollständige Sicherheitsbeweise zu führen, da in der Regel dafür Annahmen getroffen werden, deren Umgehung, wie im vorliegenden Fall, dann doch eine Sicherheitslücke aufdecken kann.

Ein weiteres Problem bei WPA2 wurde bereits im Jahr 2010 publik. Es handelt sich um die so genannten Hole 196 Schwachstelle, die ein beträchtliches Risiko für WLAN-Netze darstellt. Durch Ausnutzen der Schwachstelle können Innentäter gezielt die verschlüsselte Kommunikation im WLAN abhören.

Hole 196-Angriff auf WPA2

Hole 196

Unter dem Namen Hole 196-Schwachstelle ist seit 2010 eine Schwachstelle im WPA2-Protokoll (IEEE 802.11i) bekannt, die durch interne Angreifer ausgenutzt werden kann. Die Schwäche wurde bereits in der Dokumentation des 1242-seitigen IEEE 802.11 Standards beschrieben, und zwar in der letzten Zeile auf der Seite 196. Das führte zu dem Namen der Schwachstelle.

Um die Schwachstelle auszunutzen, muss der Angreifer ein Innentäter, also ein authentisierter WLAN-Client sein, und auch der Opfer-Rechner ist ein authentisierter WLAN-Client. Jeder WLAN-Client besitzt einen paarweisen Schlüssel PTK , bzw. TK mit dem AP und alle Clients kennen den Gruppenschlüssel GTK des WLANs. Der Angreifer-Rechner besitzt also einen gemeinsamen Schlüssel $TK_{Angreifer}$ mit dem Access Point (AP). Der IEEE 802.11i Standard geht davon aus, dass nur der Access Point Broadcast-Nachrichten, die mit dem GTK verschlüsselt werden, versendet. Der Standard sieht jedoch kein Konzept vor, dass einen Client daran hindert, eigene mit GTK verschlüsselte Datenpakete zu versenden.

Hier setzt der Angriff an. Der Angreifer sendet ein mit dem gemeinsamen Gruppenschlüssel GTK verschlüsseltes, speziell präpariertes Datenpaket an die anderen WLAN-Clients. Der Angriff ist im Kern ein ganz klassischer ARP-Cache-Poisoning-Angriff (siehe Seite 127), um die Daten des Opfer-Rechners auf den Angreifer-Rechner umzuleiten. Diese Umleitung führt der Access Point durch, da der Opfer-Rechner in seinen Datenpaketen aufgrund der manipulierten ARP-Cache-Daten unwissentlich eine falsche MAC-Adresse als Empfänger-Adresse, nämlich die des Angreifers, angibt. Nachfolgend sind die Schritte, die die Beteiligten an dem Angriff jeweils durchführen, zusammengestellt.

Angriff

Angreifer-Rechner:

1. Der Angreifer erstellt ein gefälschtes ARP-Request-Datenpaket, das die IP-Adresse des Access Points mit der MAC-Adresse des Angreifers assoziiert.
2. Das Datenpaket wird mit dem GTK verschlüsselt und Over-the-Air zu den Opfer-Rechnern gesandt.

Opfer-Rechner:

1. Das Datenpaket wird mit dem gemeinsamen GTK entschlüsselt.
2. Mit den Informationen aus dem ARP-Request wird der ARP-Cache des Opfer-Rechners aktualisiert, d.h. der IP-Adresse des Access Points wird die MAC-Adresse des Angreifers zugeordnet.
3. Nachfolgende, beliebige Nachrichten, die ab diesem Zeitpunkt vom Opfer-Rechner versendet werden, werden gemäß dem WPA2-Protokoll für Unicast-Nachrichten mit dem TK-Schlüssel, TK_{Opfer} , den nur der Opfer-Rechner und der Access-Point kennen, verschlüsselt und zum Access Point gesandt. Aufgrund der manipulierten ARP-Cache-Einträge enthält jedoch das Datenpaket nun die MAC-Adresse des Angreifers als Empfangsadresse.

Access Point:

1. Die Datenpakete werden mit dem passenden TK_{Opfer} entschlüsselt.
2. Der AP verschlüsselt die Daten mit dem Schlüssel, den er sich mit dem Angreifer-Rechner teilt, also mit $TK_{Angreifer}$, und sendet die Daten zum Angreifer.

Auswirkungen

Der Angreifer kann also die Daten des Opfer-Clients entschlüsseln und auf diese Weise die gesamte Kommunikation des Opfer-Rechners abhören, wie beispielsweise Login-Daten, sensitive Mails etc.

Durch Ausnutzen der Schwachstelle kann ein authentisierter Nutzer die Daten anderer authentisierter Nutzer im Klartext abhören. In Unternehmensnetzen, in denen zum Beispiel auch Gast-Rechner einen Zugang bekommen können, entsteht damit ein beträchtliches Sicherheitsleck. Da für die Durchführung des Angriffs die Kenntnis des Gruppenschlüssels *GTK* erforderlich ist, kann der Angriff nur durch Innentäter erfolgen. Dies ist dennoch gravierend, da nach wie vor sehr viele Angriffe durch Innentäter durchgeführt werden. Der Angriff nutzt die Protokollabläufe von WPA2 aus; er zielt nicht darauf ab, geheime Kommunikationsschlüssel der WLAN-Clients zu knacken. Mit dem Angriff kann eine Man-in-the-Middle Attacke, so wie oben skizziert, oder eine Denial-of-Service-Attacke durchgeführt werden. Da der Angriff auf dem AP oder den Clients kein auffälliges Verhalten hervorruft, ist er nicht mit den üblichen Überwachungsverfahren zu erkennen.

Mögliche Abwehrmaßnahmen

Client-Isolierung

Der 802.11i Standard sieht kein Konzept vor, um die Schwachstelle zu beheben. Zur Beschränkung der möglichen Risiken wird vorgeschlagen, dass der Access Point die so genannte Client-Isolierung aktiviert.

Mit einer aktivierte Client-Isolierung wird erreicht, dass keine Kommunikation mehr zwischen den Clients über den Access Point erfolgen kann. Das bedeutet, dass zwar nach wie vor ein Angreifer gefälschte ARP-Requests an Clients senden und damit deren Caches manipulieren kann, aber die nachfolgende Umleitung der gesendeten Nachrichten dieser Clients auf den Angreifer-Rechner wird unterbunden, da der Access Point eine solche Weiterleitung nicht mehr durchführt. Das Konzept der Client Isolation, wie es von einigen Access Points und WLAN Controllern unterstützt wird, ist kein Bestandteil des Standards.

AP-Spoofing

Der Schutzmechanismus der Client-Isolierung greift jedoch zu kurz und kann zudem von Angreifer-Rechnern umgangen werden, indem ein gefälschter Router aufgesetzt wird und die WLAN-Clients mit der oben skizzierten ARP-Cache-Poisoning Attacke dazu gebracht werden, ihren Datenverkehr über den gefälschten Router zu leiten. Da der Access Point die Daten vor der Weiterleitung entschlüsselt, liegen sie im Klartext vor und das Schutzziel der Vertraulichkeit ist gefährdet.

Externer Client

Zudem kann ein angreifender WLAN-Client auch in seinem manipulierten ARP-Request die MAC-Adresse eines Clients ausserhalb des betrachteten WLANs angeben. Für diese externen Clients gilt die vom Access Point umgesetzte Client-Isolierung dann nicht mehr und die Daten des Opfer-

Rechners werden durch den Access Point an den vermeintlich korrekten Empfänger ausserhalb des WLANs - natürlich unverschlüsselt - weitergeleitet.

Eine wirksame Abhilfe gegen die Angriffe kann derzeit auf der Schicht 2 nur durch den Verzicht auf die Nutzung des gemeinsamen Gruppenschlüssels *GTK* erzielt werden. Da aber der Standard vorsieht, dass ein solcher Schlüssel beim 4-Wege-Handshake verteilt wird, kann man nur durch einen Software-Patch beim Access Point dafür sorgen, dass jeweils ein neuer, paarweiser *GTK* im Handshake ausgetauscht wird. Broadcast-Nachrichten sind dann nicht mehr möglich und müssen durch Unicast-Nachrichten ersetzt werden, was ggf. erhebliche Performanz-Einbußen nach sich zieht.

Derzeit ist noch offen, welche Lösung in der Weiterentwicklung des Standards integriert wird. Die Ursache des Problems liegt ja darin, dass nicht kontrolliert werden kann, dass der gemeinsame, symmetrische Gruppenschlüssel *GTK* ausschließlich vom Access Point zum Versand verschlüsselter Broadcast-Nachrichten verwendet wird. Durch die Verwendung von Public-Key Kryptographie könnte man dieses Problem lösen. Dazu müsste der Access Point ein asymmetrisches Schlüsselpaar als Gruppenschlüssel generieren und im Handshake den öffentlichen Schlüssel seinen Clients mitteilen. In einem solchen Setup könnte der private Schlüssel des AP zum Verschlüsseln von Broadcast-Nachrichten verwendet werden. Dies kann dann nur der Access Point durchführen. Alle authentisierten WLAN-Clients können die Nachricht entschlüsseln, aber selber keine Nachrichten mit dem Gruppenschlüssel erstellen. Diese Lösung erfordert aber grundlegende Änderungen der WLAN-Protokolle, da bislang aus Performanz-Gründen nur symmetrische Verschlüsselung vorgesehen ist.

Es ist klar, dass Schutzmaßnahmen auf der Schicht 2 unzureichend sind; eine Ende-zu-Ende Sicherheit ist damit nicht gewährleistbar. Dazu ist es erforderlich, zusätzliche Schutzmaßnahmen auf höheren Ebenen einzuführen und eine Ende-zu-Ende Verschlüsselung zu realisieren. So kann man die Nutzdaten, wie beispielsweise eMails, auf der Anwendungsebene stark verschlüsseln und die Vertraulichkeit der Kommunikation ist durch einen Hole 196-Angriff nicht gefährdet.

Unicast

Public-Key

E2E

WLAN-KRACK-Angriff

Im Oktober 2017 haben die belgischen Sicherheitsforscher M. Vanhoef und F. Piessens eine Schwäche im Protokolldesign des 4-Wege-Handshakes von WPA2 aufgezeigt (vgl. [174] und <https://www.krackattacks.com/>). Bei dem so genannten WLAN-KRACK-Angriff (key reinstallation attack) gelingt es, einen bereits etablierten PTK-Schlüssel erneut einzuspielen. Da es sich um eine Design- und nicht um eine Schwachstelle in ei-

ner bestimmten Implementierung des WPA2 Protokolls handelt, sind alle WLAN-Implementierungen betroffen, die die Spezifikation des 802.11i-Standards umsetzen.

Designproblem

Wie oben ausgeführt, (siehe Seite 905), wird beim Zugang zum WLAN-Netz zunächst der 4-Wege-Handshake ausgeführt, bei dem zwischen dem WLAN-Client und dem Access-Point ein gemeinsamer, geheimer Schlüssel PTK sicher vereinbart wird. Nach dem Austausch der Nachricht 3 des 4-Wege Handshakes, wird der vereinbarte PTK und damit auch der Kommunikationsschlüssel TK aktiviert und nachfolgend in der Stromchiffre¹⁸ zur verschlüsselten Kommunikation genutzt. WPA2 generiert für jede zu verschlüsselnde Nachricht einen neuen Schlüsselstrom, wofür ein Zählerwert inkrementiert und mit einer Nonce initialisiert wird. WPA2 stellt damit sicher, dass die Stromchiffre einen Schlüsselstrom nicht mehrfach generiert.

Da es bei der Nachrichtenkommunikation in einem WLAN zu Störungen bei der Übertragung kommen kann, überträgt der Access Point gemäß Spezifikation die Nachricht 3 erneut, sollte er keine Empfangsbestätigung vom Client erhalten haben. Bei jedem erneuten Empfangen der Nachricht 3 reinitialisiert der Client den PTK und setzt den Zähler, der mit der Nonce initialisiert wurde, sowie den Replay-Counter, zurück.

Angriff

Hier setzt der KRACK-Angriff an. Als erstes erfordert der Angriff die Positionierung eines Man-in-the-Middle. Um eine Man-in-the-Middle Attacke auszuführen, muss der Angreifer die gleiche MAC-Adresse wie der attackierte Access Point nutzen, da die MAC-Adresse in die Schlüsselgenerierung einfließt. Der Angriff erfolgt mittels eines geclonten Access Points, der auf einem anderen Funkkanal als der ursprüngliche Access Point kommuniziert. Durch einen Man-in-the-Middle Angreifer wird zunächst dafür gesorgt, dass die Bestätigungsrichtung 4 des Clients nicht an den Access Point weitergeleitet wird. Der Client hat davon jedoch keine Kenntnis, weshalb er den Schlüssel bereits zur verschlüsselten Kommunikation verwendet. Der Angreifer fängt diese Nachrichten ebenfalls ab. Da der Access Point keine Empfangsbestätigung (Nachricht vier) erhält, wird er spezifikationskonform seine Nachricht 3 erneut an den Client übertragen. Damit erreicht der Angreifer, dass der WLAN-Client den PTK erneut installiert (reinitialization attack) und dass der Client seine Zählerstände, die für die ggf. bereits durchgeföhrten Verschlüsselung geändert worden sind, zurücksetzt. Der Angreifer leitet nun die Bestätigungsrichtung des Client an den Access Point weiter. Client und Access Point nutzen den reinitialisierten TK, um zusammen mit der zurückgesetzten Nonce zur Initialisierung des Zählers und mit dem ebenfalls zurückgesetzten Replay-Counter die Schlüsselströme für die Stromchiffre (TKIP, AES-CCMP oder AES-GCMP) zu erzeugen.

¹⁸ Die Spezifikation sieht die Nutzung von TKIP, AES-CCMP und AES-GCMP vor.

Abhängig von der genutzten Stromchiffre hat der skizzierte Angriff unterschiedliche Auswirkungen. Bei TKIP können Datenpakete entschlüsselt, wieder eingespielt und auch manipuliert werden, da auch der MIC-Schlüssel wiederhergestellt werden kann. Am weitesten verbreitet in heutigen WLANs ist die Nutzung der Stromchiffre AES-CCMP. Mit dem Angriff können Datenpakete entschlüsselt und wieder eingespielt werden. Bei AES-GCMP ist sogar auch eine Manipulation der Pakete möglich. Wichtig ist, dass der Angreifer nach wie vor keine Kenntnis von dem ausgetauschten Kommunikationsschlüssels TK hat. Er besitzt abgefangene, verschlüsselte Nachrichten des Clients. Diese Nachrichten sind mit jeweils einem individuellen Schlüsselstrom mittels eines einfachen XORs verschlüsselt worden. Der Schlüsselstrom wird beim AES-CCMP basierend auf dem TK und der aktuellen Zählerstand des Counters, ausgehend von dem Nonce-Wert generiert. Nach der Reinstallation des TK wird der Client für neue Nachrichten gleiche Schlüsselströme verwenden, da auch die Noce und damit auch der Zähler des Schlüsselstrom-Generators zurück gesetzt worden ist. Zwei unterschiedliche Klartexte, die mit dem gleichen Schlüsselstrom mittels XOR verschlüsselt wurden, lassen sich schnell knacken, wenn man einen der Klartexte bereits kennt. Dies müsste also für einen direkten Angriff gegeben sein, was möglich, aber auch nicht trivial ist. Wenn es dem Angreifer mit einer solchen Attacke beispielsweise gelingt, TCP Sync-Pakete zu entschlüsseln, so erlangt der Angreifer Kenntnis über die TCP Sequenznummer und kann nachfolgend die unverschlüsselte TCP-Verbindung hijacken. Eine verschlüsselte TLS-Verbindung wird dadurch aber nicht gefährdet. Dass ein solcher Angriff möglich ist, wird in dem Papier [174] gezeigt.

Eine besondere Schwachstelle weisen Linux und Android-Implementierungen auf. Linux nutzt die Funktion *wpa_supplicant*, bei der ab der Version 2.4 eine erneute Übertragung der Nachricht 3 dazu führt, dass der Schlüssel TK als ein Schlüssel bestehend aus lauter Nullen reinstalled wird; dies ist als All-Zero-Key bekannt. Das Überschreiben des TK mit lauter Nullen war ursprünglich dazu gedacht, um den TK-Speicherbereich zu bereinigen. Durch den KRACK-Angriff wird ein All-Zero-Schlüssel reinstalled, was natürlich eine äußerst gravierende Sicherheitslücke darstellt. Besitzt der Angreifer ein aufgezeichnetes Paar bestehend aus Klartext und Kryptotext und ist ihm der Klartext bekannt, so kann er damit direkt aus dem Kryptotext die Bits der Stromchiffre extrahieren. Unter Nutzung der AES-CCMP Stromchiffre wurde dieser 128-Bit Schlüsselbit-Block als Ausgabe einer AES-Verschlüsselung unter Nutzung des Schlüssels TK generiert, wobei der verschlüsselte Klartext der aktuelle Zählerstand des Counters ist. Mit dem bekannten Schlüssel TK, der ja nur aus Nullen besteht, kann der Angreifer den extrahierten Schlüsselstrom eines entschlüsseln und den Zählerstand zurückgewinnen. Damit ist der Angreifer in der Lage, die Schlüsselströme, die

Auswirkungen

All-Zero Schlüssel

der Client zur Verschlüsselung generiert, ebenfalls zu generieren und sämtliche verschlüsselten Datenpakete des Clients zu entschlüsseln, solange der Client seinen Counter nicht mit einer neuen Nonce erneut initialisiert. Die All-Zero-Key Schwachstelle ist auch noch in der Version 2.6 vorhanden, so dass auch alle Android Geräte, die darauf aufsetzen, anfällig gegen All-Zero-Schlüssel-Angriffe sind.

Windows, iOS

Da weder Windows noch iOS Implementierungen von WPA2 eine erneute Übertragung der Nachricht 3 ermöglichen, dadurch also eigentlich nicht spezifikationskonform umgesetzt sind, ist dies in diesem Fall hilfreich, weil diese Implementierungen die Schwachstelle in Bezug auf die Wiedereinspielung des PTKs nicht aufweisen. Eine Verwundbarkeit bei der Nutzung von Gruppenschlüsseln besteht jedoch auch bei diesen Implementierungen.

Router-Verwundbarkeit

Schließlich ist noch anzumerken, dass auch Access Points auf die aufgezeigte Weise angreifbar sind, wenn sie das Fast BSS Transition (FT) Handshake unterstützen oder eine Repeater-Funktionalität (Client Funktionalität) anbieten. Der FT Handshake ist Teil der 802.11r-Spezifikation und wird in der Regel nicht von Heim-Routern sondern in Unternehmensnetzen unterstützt. Da auch die wenigsten Heim-Router eine Repeater-Funktionalität unterstützen, sind die Heim-Access Points meist nicht verwundbar, wohl aber die WLAN-Clients.

Abwehr

Eine Abwehr von KRACK-Angriffen auf WPA2-Implementierungen ist durch ein Patchen der Implementierung relativ einfach möglich, da lediglich dafür gesorgt werden muss, dass die an der Kommunikation beteiligten Entitäten protokollieren, ob sie bereits einen PTK installiert haben und nutzen und eine erneute Installation verhindert wird. Von vielen Gerät- und Softwareherstellern wie Intel, Dell oder auch Microsoft, Google, Apple und auch Linux wurden entsprechenden Patches zeitnah bereitgestellt.

Zu beachten ist, dass Protokolle, die losgelöst von den darunter liegenden WPA2-Protokollen, eine sichere Verbindung aufbauen, wie beispielsweise TLS oder IPSec, durch den Angriff natürlich nicht berührt sind.

Zu beachten ist weiterhin, dass der Angriff zum Teil nicht triviale Angriffs-schritte umfasst. Es ist zudem erforderlich, dass der Angreifer sich in der physischen Nähe des angegriffenen Clients befindet, wodurch sich die Angriffsrisiken insbesondere für den Heimbereich deutlich abschwächen.

15.4.6 802.1X-Framework und EAP

Wie bereits erläutert, wird in den WLAN-Erweiterungen häufig der 802.1X-Standard (vgl. [87]) verwendet, um Partner zu authentifizieren und um Masterschlüssel auszutauschen. Der Standard stellt ein Authentifikations-

framework bereit, in dem dedizierte Authentifikationsmechanismen genutzt werden können. Beispiele derartiger Mechanismen sind zertifikatsbasierte Authentifikation, Smartcards oder auch einmal benutzbare Passworte. Der 802.1X-Authentifizierungsstandard wurde ursprünglich für einen anderen Kontext, nämlich die Telefonie entwickelt. Im Folgenden erläutern wir jedoch die Nutzung des 802.1X-Standards im Zusammenhang mit dem 802.11i-Standard.

Der 802.1X-Standard unterscheidet drei Rollen, nämlich den Supplicant, den Authenticator und den Authentication Server. Ein Supplicant ist ein Subjekt (z.B. eine Funkkarte eines WLAN-Clients), das einen Dienst nutzen möchte, der über einen Port des Authenticators angeboten wird. Ein Authenticator ist beispielsweise ein Ethernet Switch oder ein Access Point. Ein Supplicant authentifiziert sich über den Authenticator gegenüber einem Authentication Server. Ein solcher Server ist in der Regel ein RADIUS-Server. Das heißt, der Access Point nimmt gegenüber dem RADIUS-Server die Rolle eines Einwahlknotens ein.

Rollen

Der Authenticator implementiert ein Dual-Port-Konzept. Das heißt, dass es zwei Ports für den Zugriff auf das Funknetz gibt, den unkontrollierten und den kontrollierten Port. Der unkontrollierte Port leitet jedes Datenpaket weiter, ohne die Autorisierung zu überprüfen, während der kontrollierte Port nur Pakete von authentifizierten Partnern passieren lässt. Der unkontrollierte Port wird deshalb nur bei der Abwicklung des Authentifikationsprotokolls verwendet, indem ein mobiles Endgerät zu Beginn eines Verbindungsaufbaus nur Zugriff auf den unkontrollierten Port des Access Points erhält. Typischer Weise wird hierbei dem Endgerät nur erlaubt, mittels EAP mit dem Authentifizierungsserver zu kommunizieren. Alle weiteren Netzdienste sind so lange gesperrt, bis die Authentifizierung erfolgreich abgeschlossen ist.

Dual-Port

Die Authentifizierung kann in diesem Framework von dem Netzprovider durchgeführt werden. Dabei kann es sich um das Gastnetz in einem Pre-Paid Kontext oder einem Pay-by-use Modell handeln oder aber auch um den Heimprovider in einem Abrechnungsmodell, das auf einem Subscriptions-Modell (z.B. GSM/GPRS) beruht.

Abbildung 15.24 gibt einen Überblick über den unter 802.11i genutzten Protokollstack. Die gestrichelt umrandeten Protokolle gehören nicht zum Standard-Protokollstack. Wie man sieht, wickeln die beteiligten Instanzen den Datenverkehr über den Access Point mittels des EAP-Protokolls ab. Die EAP-Nachrichten sind in weitere Datenpakete gekapselt, die unter Nutzung von Sicherheitsdiensten höherer Protokollschichten (z.B. EAP-TLS, also EAP unter Nutzung von TLS (siehe RFC 2716)) abgesichert sein können. Auf diese Weise lassen sich in EAP-Nachrichten auch Authenti-

Protokollstack

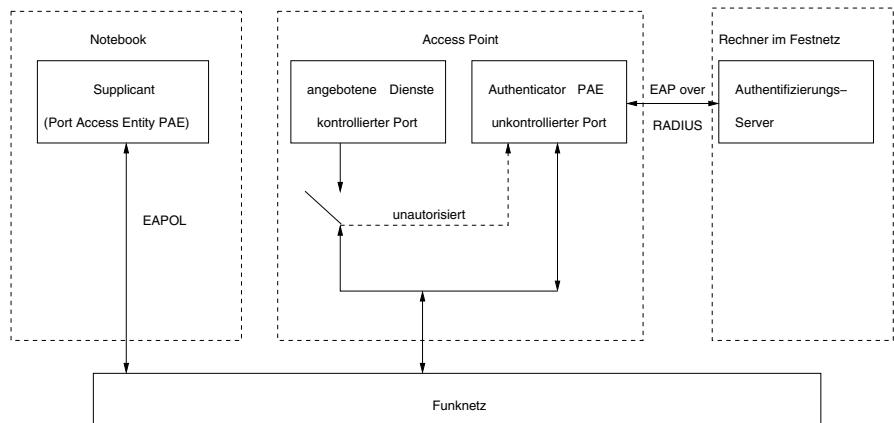


Abbildung 15.23: Dual-Port Konzept unter 802.1X

fizierungsdaten von Authentifikationsprotokollen der höheren Schicht, wie z.B. Kommunikationsschlüssel von TLS, sicher transportieren. Im Funknetz werden die EAP-Daten direkt in 802.11 Paketen verschickt, weshalb dieses Protokoll EAP over LAN (EAPOL) genannt wird. Da der Authentifizierungsserver in der Regel ein RADIUS-Server ist, muss der Authenticator (der Access Point) als RADIUS-Client konfiguriert sein und EAP-Pakete in RADIUS-Pakete transformieren bzw. kapseln. Das entsprechende Verfahren wird EAP over RADIUS genannt.

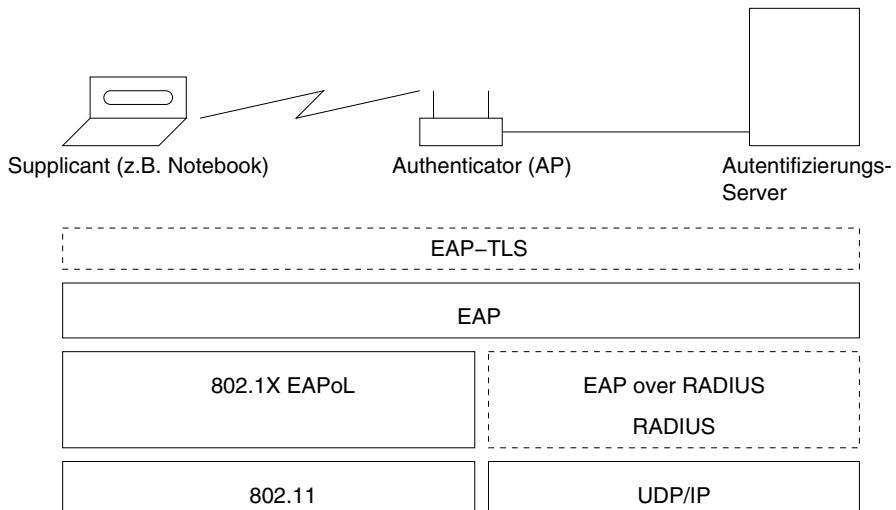


Abbildung 15.24: 802.11i Protokollstack

Extensible Authentication Protocol (EAP)

Der IEEE Standard 802.1X verwendet das Extensible Authentication Protocol (EAP) (vgl. Seite 750), um das notwendige Spektrum unterschiedlicher Authentifizierungsmechanismen (u.a. Passwort-basierte, Zertifikat-basierte, SIM/USIM-basierte Authentifikation) zu unterstützen. Die Authentifizierungsdaten von Benutzern werden an spezielle Authentifizierungsserver, wie RADIUS- oder Kerberos-Server, weitergeleitet, wodurch eine zentralisierte Teilnehmerauthentifikation ermöglicht wird.

Abbildung 15.25 gibt einen groben Überblick über den Authentifikationsablauf unter 802.1X zwischen den beteiligten Partnern.

EAP

Protokollablauf

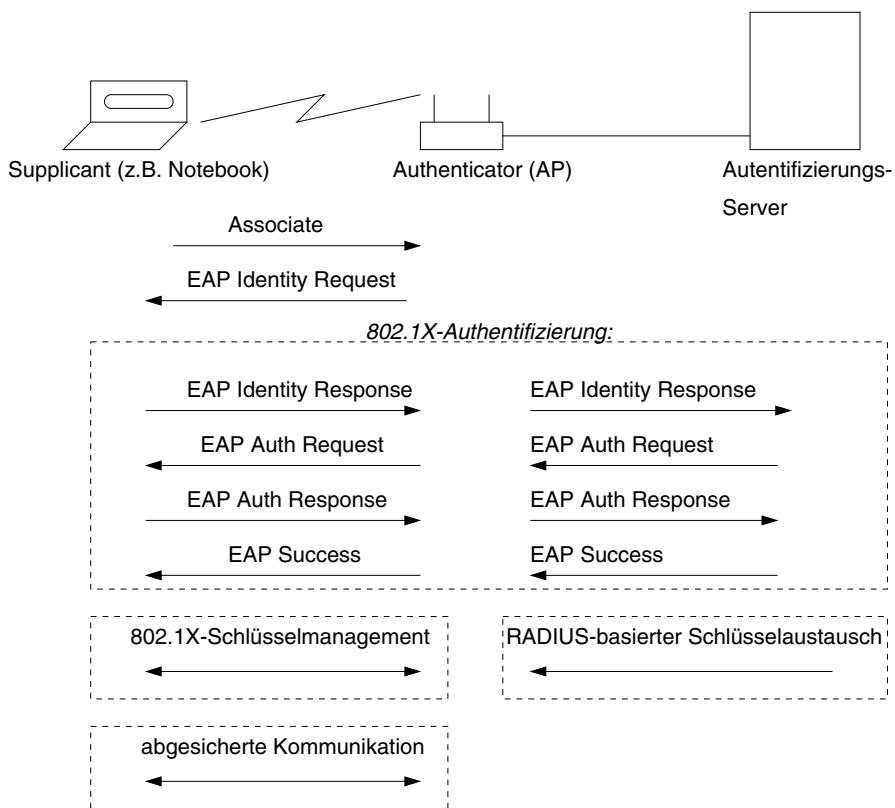


Abbildung 15.25: Typischer Protokollablauf unter 802.1X

In einem typischen Protokollablauf einer 802.1X Authentifikation sendet der Authenticator (also hier der Access Point) eine Challenge an den Supplicant (also die WLAN-Karte des Endgeräts), der darauf antwortet. Der Authenticator sendet daraufhin eine Zugriffsanfrage an den Server, der seinerseits eine Folge von Challenges über den Authenticator an den Supplicant schi-

cken kann, die dieser jeweils zu beantworten hat. Beispielsweise könnte hier ein SSL/TLS-Server vom Endgerät ein gültiges Zertifikat anfordern. In diesen Protokollschriften kann der Authentication Server auch Kommunikationsschlüssel an den Supplicant übermitteln. Der Authentication Server ist also eine vertrauenswürdige Instanz in diesem Modell. Das Protokoll terminiert, indem der Authentication Server dem Supplicant und auch dem Authenticator mitteilt, ob die Authentifikation erfolgreich oder fehlerhaft war.

Typischerweise erhält das mobile Gerät von dem Authentifizierungsserver auch seinen Master-Schlüssel, um anschließend den kontrollierten Port des Access-Points nutzen und Zugriff auf Dienste (z.B. Internet-Zugriff) in Anspruch nehmen zu können. Das 802.1X-Framework übernimmt dann weiterhin die Aufgabe des Schlüsselmanagements. Hierbei werden u.a. Protokolle abgewickelt, durch die sich die Beteiligten wechselseitig von dem Besitz vereinbarter Schlüssel überzeugen, und es werden für jede Verbindung frische Sitzungsschlüssel generiert.

Sicherheitsprobleme

Man-in-the-Middle

Sicherheitsanalysen (siehe [123]) weisen auf Probleme beim Einsatz von 802.1X im WLAN-Zusammenhang hin. Da sich der Supplicant nur gegenüber dem Authenticator, also dem Access Point in einem WLAN-Einsatzumfeld, authentifizieren muss, und Nachrichten ohne die Verwendung kryptografischer Prüfsummen zwischen dem Server, dem Authenticator und dem Supplicant übertragen werden, ergibt sich hier wieder die klassische Ausgangssituation für einen Man-in-the-Middle-Angriff. In [123] wird ein solcher Angriff sowie ein Angriff mit dem Ziel der Verbindungsübernahme (engl. *session hijacking*) erläutert. Die Kernidee dabei ist, dass der Angreifer versucht, durch eine eigene Nachricht, dem korrekten AP zuvor zu kommen. Das heißt, dass der Angreifer wartet, bis eine Authentifikation zwischen AP und Client erfolgreich beendet ist. Bevor nun der korrekte AP seine Success-Nachricht einschließlich ggf. des neuen Kommunikationsschlüssels an den Client senden kann (vgl. Abbildung 15.25), wird der Angreifer aktiv. Er sendet eine Disassociation-Nachricht an den Client, der damit glaubt, dass seine Verbindung zum AP abgebrochen ist, während der AP immer noch von einer bestehenden Verbindung zu dem Client ausgeht, also die Verbindung weiterhin Aufrecht erhält. Falls keine Verschlüsselung verwendet wird, was z.B. in öffentlichen Hot-Spots in der Regel der Fall ist, kann nun der Angreifer die etablierte Verbindung direkt übernehmen und über den gesicherten Port des AP die eigentlich geschützten Dienste nutzen. Dies ist ihm bis zum nächsten Time-out möglich, das aber in der Regel erst nach ca. 60 Minuten auftritt.

Eine Variante des Angriffs besteht darin, dass der Angreifer wiederum versucht, der Erfolgsmeldung des korrekten APs zuvor zu kommen. Diesmal schickt er aber selber eine Erfolgsmeldung, maskiert sich also wieder als AP. Mit dieser gefälschten Erfolgsmeldung schickt der Angreifer jedoch keinen Schlüssel mit, so dass der Client davon ausgeht, dass eine unverschlüsselte Kommunikation erwünscht ist und seine Daten entsprechend unverschlüsselt transferiert. Damit hat der Angreifer ein leichtes Spiel, diese Daten abzufangen, zu verändern, wieder einzuspielen etc.

Die Verwendung des allgemeinen 802.1X-Frameworks zur Verbesserung des Sicherheitsstandards bei 802.11i ist erneut ein typisches Beispiel dafür, dass man Sicherheitsprotokolle nicht einfach in andere Umgebungen übernehmen kann, wenn die Umgebung andere Randbedingungen aufweist als diejenigen, die für die ursprüngliche Entwicklungsumgebung des Protokolls gegolten haben.

veränderte
Einsatzumgebung

Das 802.1X-Framework wurde ursprünglich für einen Einsatz durch Service-Provider wie Telefongesellschaften konzipiert. In diesem Szenario war zum einen eine einseitige Authentifikation allein des Clients völlig ausreichend, damit die Telefongesellschaften eine korrekte Abrechnung der Dienste durchführen können. Die Telefoniesysteme waren geschlossene Systeme, in denen ein Man-in-the-Middle Angriff äußerst unwahrscheinlich und aufwändig ist. Die Authenticatoren in diesem Szenario, also die Service Provider, werden als vertrauenswürdig angesehen. Diese Randbedingungen lassen sich bei der Übertragung auf das WLAN-Umfeld nicht länger aufrecht erhalten. Die Authenticatoren sind beliebige Access Points, die natürlich nicht per se vertrauenswürdig sind, so dass auf dieser Basis das 802.1X unsicher ist. Das ist wie gesagt klar und auch bereits aus der IEEE Spezifikation des Protokolls direkt ableitbar, in der es heißt, dass 802.1X unsicher ist, falls es in einer Umgebung mit einem gemeinsam nutzbaren Zugriffsmedium¹⁹ verwendet wird, ohne dass zusätzliche Maßnahmen zur Etablierung von Sicherheits-Associations durchgeführt werden. Über die unsicheren Authenticatoren im WLAN-Umfeld werden unter anderem Zertifikate weitergereicht und Schlüssel ausgetauscht, die natürlich manipuliert sein können.

Telefonie-Kontext

Das heißt, will man nicht auf einen AP vertrauen müssen, sollten die erforderlichen Zertifikate des Authentifikations-Servers bereits vorab auf sicherem Weg an die mobilen Clients verteilt werden. Um eine Ende-zu-Ende Sicherheit zu etablieren, sollte anstelle des einfachen EAP besser EAP im geschützten Tunnelmodus (PEAP, protected EAP) eingesetzt werden. PEAP wurde von Microsoft, Cisco und RSA Security entwickelt. Es benutzt TLS zum Aufbau eines sicheren Tunnels, über den in einer zweiten Phase die eigentliche Authentisierung abläuft. Damit wird ein verschlüssel-

PEAP

¹⁹ Die Luftschnittstelle ist offensichtlich ein derartiges Medium.

ter Tunnel zwischen Endgerät und Authentifizierungs-Server aufgebaut und der Access-Point erhält keinen direkten Zugriff mehr auf die Credentials der Teilnehmer.

15.5 Bluetooth

Start

Bluetooth (u.a. [126]) ist ebenfalls eine Technologie zur drahtlosen Kommunikation. Die Anfänge von Bluetooth gehen zurück auf eine Studie aus dem Jahr 1994, in der die schwedische Telekommunikationsfirma Ericsson mögliche Wege erforschen ließ, um Kabelverbindungen zwischen Mobiltelefonen, PC-Karten, Headsets, Druckern und anderen Geräten überflüssig zu machen. Die Studie ermittelte den Bedarf an einer preiswerten Funkschnittstelle mit geringer Reichweite und geringem Stromverbrauch.

Namensgebung

Seit 1998 wird von einem internationalen Konsortium für drahtlose Übertragungsverfahren, der Bluetooth Special Interest Group (SIG), eine Bluetooth-Spezifikation entwickelt. Zu den Mitgliedern dieser Gruppe gehören führende Hersteller von Telekommunikationsprodukten, wie Ericsson, Nokia, Intel, 3Com, IBM und Microsoft. Heute gehören über 2500 Mitglieder dieser Gruppe an. Wie ein Mitglied der Gruppe bei einem Meeting einmal treffend bemerkte, gehören zur Gruppe „Everyone you can think of. Lots of companies you never heard of.“ Die neue drahtlose Technologie wurde zu Ehren des Wikingerkönigs Harald Blauzahn (Harald Blaatand), der im 10ten Jahrhundert vor Christus in Dänemark regierte, Bluetooth genannt. Dieser Wikingerkönig trug stark zur Einigung und zur Beilegung von religiösen Kämpfen in seinem Land bei. Die Harmonisierung und Vereinheitlichung waren auch wesentliche Ziele des Bluetooth-Projekts, was die Namenswahl augenscheinlich motiviert hat. Es sollte ein gemeinsamer Standard geschaffen werden, so dass unterschiedliche Geräte über eine drahtlose Verbindung miteinander kommunizieren können. Mit dem Standard IEEE 802.15.1 wurde eine erste Version des Bluetooth-Standards entwickelt. Der Standard wurde kontinuierlich weiterentwickelt und mit Bluetooth 5 liegt ein aktueller Standard vor, der stark auf den Einsatz im IoT Umfeld ausgerichtet ist.

Bluetooth-Versionen

Während die ursprüngliche Version von Bluetooth noch darauf ausgerichtet war, Geräte aus dem persönlichen Arbeits- und Lebensumfeld, wie Kopfhörer, Lautsprecher oder Smartphones zu verbinden, lag der Fokus bei der Einführung von Bluetooth 4 auf Geräten mit beschränkten Energieressourcen, wie Fitnessbänder oder Smartwatches. Bluetooth 5 geht weiter in die Richtung auf IoT und ermöglicht die einfache Vernetzung von Dingen des

täglichen Lebens wie Thermostate, Türschlösser oder Haushaltsgeräte oder auch von Gegenständen im Produktionsumfeld.

Das ursprüngliche Bluetooth-Protokoll wurde als IEEE 802.15.1-Standard spezifiziert und kontinuierlich weiter entwickelt. Mit der Version Bluetooth v2.0 in 2004 wurde die Datenrate auf bis zu 3 Mbit/s erhöht und der Energieverbrauch auf ca. die Hälfte des Bedarfs des ursprünglichen Standards verringert. Im August 2007 wurde der Standard Bluetooth 2.1 + EDR (Enhanced Data Rate) verabschiedet, der als wichtige Neuerung das Secure Simple Pairing Protokoll definiert. Bluetooth 3.0 + HS (Highspeed) wurde 2009 veröffentlicht. Die Version unterstützt auf Basis von u.a. WLAN einen Highspeed Kommunikationskanal mit einer theoretischen Übertragungsrate von 24 Mbit/s, indem die L2CAP-Protokollsicht erweitert wurde, um den zusätzlichen Highspeed-Kanal zu ermöglichen. Die Spezifikation Bluetooth 4.0 wurde Ende 2009 verabschiedet und unterstützt das Low Energy Profil., so dass die Version auch als Bluetooth LE (Low Energy) bekannt ist. Die Version V.4 ist abwärtskompatibel und unterstützt AES-128 Bit für die Verschlüsselung. Eine spezielle, nicht abwärtskompatible Variante ist Bluetooth Low Energy/Smart, die es ermöglicht, in weniger als fünf Millisekunden einen Kommunikationskanal über eine Entfernung von bis 100 Metern aufzubauen. Low Energy optimiert, wie der Name es bereits andeutet, den Energieverbrauch durch u.a. verringern des Overheads beim Verbindungsauflaufbau. Audiodaten lassen sich mit LE jedoch nicht mehr übertragen. 2013 wurde Version 4.1 und 2014 wurde die Version 4.2 veröffentlicht, deren Schwerpunkt die Erhöhung der Übertragungsgeschwindigkeit und die weitere Verringerung des Energieverbrauches waren. Der Standard Bluetooth 5 wurde in 2016 veröffentlicht, der ebenfalls abwärtskompatibel ist. Zentrale Verbesserungen liegen in der deutlichen Erhöhung der Reichweite, indem bei Sichtkontakt 200 Meter und in Gebäuden ca. 40 Meter möglich sein sollen, und in der Erhöhung der Datenrate (2 Mbit/s ohne EDR), sowie der Spezifikation neuer Dienste wie Standortübermittlung. Unter Bluetooth 5 können Datenpakete 255 Byte groß werden, wodurch zum Beispiel URLs in einem Paket übertragen werden können. Die Reichweiten und Bandbreitenerhöhung kann jedoch nur alternativ genutzt werden, so dass Bluetooth-Geräte zugeschnitten auf ihre Einsatzmöglichkeiten konfigurierbar sind.

Versionen

15.5.1 Einordnung und Abgrenzung

Im Gegensatz zu der WLAN-Technologie war Bluetooth ursprünglich vorrangig auf eine drahtlose, serielle Kommunikation zwischen Geräten über sehr kurze Distanzen bis maximal 10 Meter ausgerichtet. Bluetooth-Geräte nach dem Standard Bluetooth 5 verfügen aber bereits über Reichweiten bis zu 200 Metern. Während die ersten Versionen von Bluetooth noch eine ge-

Bluetooth

ringe Datenübertragungsrate aufwiesen, die deutlich niedriger als 802.11 war, ziehen die aktuellen Versionen gleich. Aufgrund seiner ursprünglich noch sehr geringen Reichweite zählte man Bluetooth zu den Personal Area Network-Protokollen (PAN). Mit den Weiterentwicklungen von Bluetooth in den Versionen 4 und 5 erhöhten sich Bandbreite und Reichweite, so dass diese Zuordnung nicht mehr ganz charakteristisch ist. Als Pluspunkt gegenüber dem 802.11-Standard kann Bluetooth aber einen sehr viel geringeren Stromverbrauch aufweisen sowie die sehr preisgünstige Fertigung von Bluetooth-Chips, so dass ein Masseneinsatz des Chips in einer Vielzahl von kleinen zum Teil sehr ressourcenschwachen Geräten Stand der Technik ist.

Interferenz

Bluetooth Sender und Empfänger verwenden das lizenzenfreie ISM (Industrial, Scientific, Medical) Funkspektrum bei 2,4 GHz-Band. Die genutzten Frequenzbänder sind allgemein verfügbar, so dass Bluetooth ohne nennenswerte geografische Einschränkungen weltweit betrieben werden kann, um Bluetooth-fähige Geräte, die in Reichweite sind, zu koppeln. Problematisch an dem genutzten 2,4 GHz-Band ist jedoch, dass auch andere Geräte in diesem Frequenzbereich senden. Dazu gehören neben Mikrowellenherden und weiteren Geräten im Heimbereich insbesondere die Funk-LANs nach dem IEEE 802.11 Standard. In Umgebungen, in denen beide Systeme eingesetzt werden, kann es somit zu Interferenzen kommen. Treffen Bluetooth-Verbindungen und Funk-LAN Verbindungen zusammen, so können diese Verbindungen ins Stocken geraten oder Übertragungsfehler auftreten. Beide Protokolle sehen spezielle Fehlerkorrekturmaßnahmen vor, um solche Fehler zu erkennen und zu beseitigen.

Bluetooth ist die am häufigsten eingesetzte Technologie zur Etablierung von ad-hoc Netzen, das heißt von Netzen ohne vorgegebener Infrastruktur²⁰. Typische Anwendungsszenarien sind die Datenübertragung zwischen Laptop und Smartphone, zwischen Head-Set und Smartphone oder auch im Auto zwischen Head-Unit und Smartphone.

15.5.2 Technische Grundlagen

Zum Verständnis der Sicherheitsarchitektur von Bluetooth ist es hilfreich, einige technische Grundlagen zu klären. Bluetooth verwendet eine Kombination aus Leitungs- und Paketvermittlung. Bei der Paketvermittlung unterscheidet man zwei Typen, zum einen die SCO-Pakete (Synchronous Connection-Oriented) für die Sprachübertragung mittels synchroner Verbindungen und zum anderen ACL (Asynchronous Connection-Less) für

²⁰ 802.11-Netze werden in der Regel im Infrastruktur-Modus eingesetzt.

asynchrone Verbindungen. Das allgemeine Format eines Bluetooth-Pakets ist wie folgt:

Paket-Format

72 Bit	54 Bit	0 – 2745 Bit
Zugangscode	Header	Payload

Der Zugangscode dient zur Synchronisation der kommunizierenden Geräte, der Header enthält Informationen zur Leitungssteuerung (engl. *link control*) und die Nutzdaten beinhalten im Fall eines ACL-Pakets eine 16 Bit CRC-Prüfsumme. Der Header enthält unter anderem 3 Bits als temporäre Adresse zur eindeutigen Identifikation einzelner Geräte in einem Bluetooth-Netz.

Um Interferenzeffekte zu begrenzen, verwendet Bluetooth ein spezielles Codierungsverfahren, das Spread Spektrum, das das zu sendende Signal zerlegt und über die gesamte Breite des verfügbaren Frequenzbereichs spreizt. Auf diese Weise entsteht ein robustes Signal, das weniger anfällig gegenüber Interferenzen mit gleichartigen Funkquellen ist, als dies bei der Verwendung des konventionellen Schmalbandfunks der Fall wäre. Beim Schmalbandfunk wird auf einer festgelegten Frequenz gesendet und empfangen, die gerade breit genug ist, um die jeweiligen Daten zu übertragen. Durch die Spreizung erhält man eine erhöhte Festigkeit gegen Störungen und auch eine höhere Abhörsicherheit.

Spread Spektrum

Die Spreizung erfolgt bei Bluetooth mit dem Frequenzsprungverfahren (engl. *frequency hopping*), bei dem der Sender mit einem pseudozufällig bestimmten Hop-Muster von einer Frequenz zur nächsten springt. Die Bandbreite ist dafür in Einheiten zu je 1MHz aufgeteilt. Ein Bluetooth-Gerät wechselt 1600 mal pro Sekunde (Hop-Rate) zwischen 79 Frequenzen. Jedes Bluetooth-Netz besitzt einen Master, der das Hop-Muster festlegt und den Mitgliedern des Netzes mitteilt. Andere Sender, die möglicherweise in unmittelbarer Gerätenachbarschaft senden, verwenden eine andere Hop-Rate sowie andere Hop-Muster, so dass dadurch Störungsmöglichkeiten verringert werden. Das Frequenzsprungverfahren liefert aber keine Datensicherheit, da natürlich auch 79 Frequenzen abhörbar sind, zum Beispiel mit 79 Empfängern bei einer Brute Force Vorgehensweise. In Spanien und Frankreich werden übrigens nur 23 Frequenzbänder verwendet.

Frequency hopping

Bluetooth-Netze

Mittels Bluetooth können sich Geräte (auch spontan) zu Netzen, den Piconetzen, zusammenfinden. In einem Piconetz teilen sich maximal acht aktive

Piconetz

Bluetooth-Geräte die Funkkanäle. Stets agiert eines der Geräte als Master und die restlichen Geräte sind Slaves. Neben den acht aktiven können noch mehr als 200 weitere Geräte in einem Stromspar-Modus (Park-, Hold- oder Sniff-Mode) passiv dem Piconetz angehören. Der Initiator der ersten Verbindung eines Piconetzes übernimmt eine Masterfunktion und die anderen Teilnehmer agieren als Slaves. Will ein Gerät einem Piconetz beitreten, so sendet es eine *inquiry*-Nachricht aus und alle erreichbaren aktiven Geräte antworten auf diese Anfrage. Abhängig davon, welche Sicherheitseinstellungen gelten, kann das Gerät nun eine direkte Kommunikation mit dem Master des Netzes aufnehmen. Die Verbindung ist stets Punkt-zu-Punkt (vgl. Abbildung 15.26 (a) oder (b)). Der Master legt den Takt und das Hop-Muster fest, die dann von allen Teilnehmern des Piconetzes verwendet werden, und teilt auch die Nutzung der Kanäle auf die Slaves auf.

Scatternet

Mehrere Piconetze können sich zu Scatternetzen zusammenfinden (vgl. Abbildung 15.26). Hierbei ist es dann möglich, dass ein Slave mit verschiedenen Mاستern in den verschiedenen Piconetzen verbunden ist (vgl. Teilnetz (c) in der Abbildung), oder ein Master eines Piconetzes kann gleichzeitig Slave in einem anderen Piconetz sein. Ausgeschlossen ist jedoch, dass ein Knoten Master in unterschiedlichen Piconetzen ist.

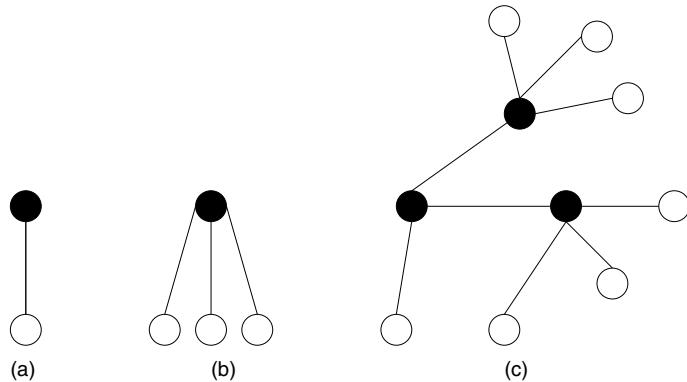


Abbildung 15.26: Scatternete bestehend aus Piconetzen

Ad-hoc vernetzte Geräte können auch als Router agieren, um Nachrichten an Geräte weiterzuleiten, die nicht direkt mit dem Absender verbunden sind. Das bedeutet natürlich auch, dass die Mitglieder eines solchen Netzes davon abhängig sind, dass die anderen Mitglieder Nachrichten tatsächlich weiterleiten. In einem solchen Kontext sind somit Denial-of-Service-Angriffe sehr einfach durchführbar, indem beispielsweise falsche Routing-Informationen eingeschleust oder der Funkverkehr gestört wird. Weitere Sicherheitsprobleme bei Ad-hoc-Netzen betreffen die Authentifikation der Teilnehmer,

da aufgrund der fehlenden Infrastruktur eine Identifikation der spontan hinzukommenden Partner schwierig ist. Damit ergibt sich direkt auch ein Vertraulichkeits- und Integritätsproblem. Da drahtlose Kommunikationsverbindungen auf einfache Weise abgehört werden können, ist eine Verschlüsselung unabdingbar. Fehlt jedoch eine vertrauenswürdige Authentifikation der drahtlos kommunizierenden Partner, so kann auch kein vertrauenswürdiger Schlüsselaustausch stattfinden. In den nachfolgenden Abschnitten werden die Lösungen vorgestellt und diskutiert, die Bluetooth für diese und allgemein für die Sicherheitsprobleme in drahtlosen Kommunikationsverbindungen bietet.

Protokollstack

Der Bluetooth-Protokollstack enthält mit dem Link Manager Protokoll (LMP) und dem Logical Link Control and Adaption Layer (L2CAP) sowohl Bluetooth-spezifische als auch mit OBEX (Object Exchange Protocol), UDP, TCP oder WAP (Wireless Application Protocol) allgemein verwendete Protokolle. Abbildung 15.27 gibt einen Überblick über den Bluetooth-Protokollstack.

Protokollstack

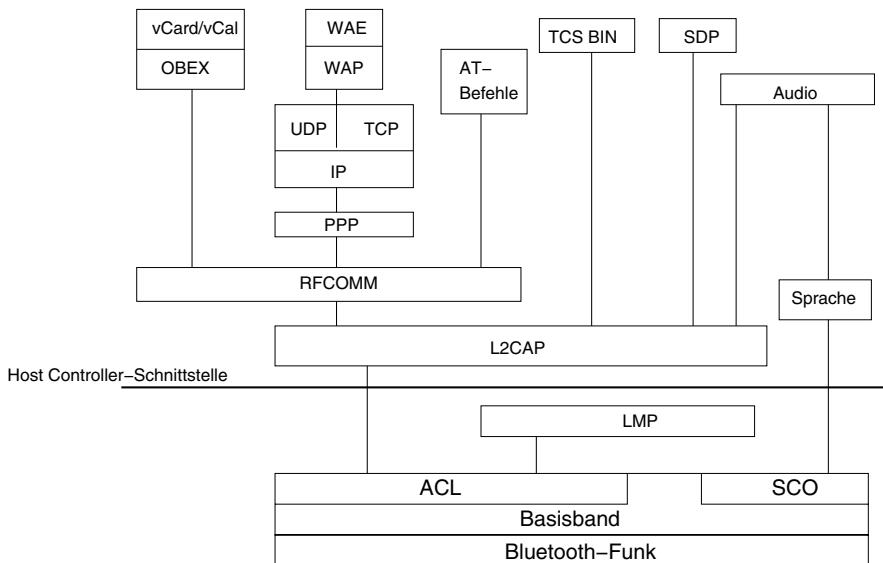


Abbildung 15.27: Bluetooth-Protokollstack

Das Basisband, das gemäß der ISO-OSI Schichtung das Protokoll der Bitübertragungsschicht definiert, ermöglicht sowohl physikalische SCO- als auch ACL-Funkverbindungen. Audiodaten wie Sprache können auch direkt, ohne die Beteiligung des L2CAP, vom Basisband übertragen werden. Die

Basisprotokolle

Protokolle LMP und L2CAP sind entsprechend dieser Schichtung auf der nächst höheren Schicht, der Sicherungsschicht (data link layer) anzusiedeln. Der Aufbau und die Steuerung der Verbindungen zwischen Bluetooth-Geräten gehört zu den zentralen Aufgaben des LMP. In ihm sind auch die wichtigsten Sicherheitsdienste (siehe Abschnitt 15.5.3), wie Authentifikation, Verschlüsselung und Schlüsselaustausch verankert. Das L2CAP setzt auch auf dem Basisband auf, das nur für ACL-Verbindungen definiert ist und Protokollen der höheren Ebene (u.a. TCP/IP) das Senden und Empfangen von Datenpaketen mit einer Größe von bis zu 64 KB ermöglicht. Es ist die Aufgabe des L2CAPs ein Protokoll-Multiplexing durchzuführen, da das zugrunde liegende Basisbandprotokoll nicht in der Lage ist, verschiedene Protokolltypen zu unterscheiden. Da die Größe der über das Basisband übermittelbaren Datenpakete auf maximal 341 Byte begrenzt ist, müssen vom L2CAP größere Datenpakete vor ihrer Übertragung über die Luftschnittstelle in kleinere zerlegt und wieder korrekt zusammengefügt werden. Um eine spezielle Dienste-Verbindung zu einem anderen Bluetooth-Gerät aufzunehmen, ist es notwendig, diese Dienste auffinden zu können. Eine solche Dienst-Findungsfunktionalität bietet das Service Discovery Protokoll (SDP), über das Geräteinformationen, Dienste und Leistungsdaten abgefragt werden können. Dieses Protokoll ist auf der Transportschicht anzusiedeln.

Kabelersatz-protokolle

Über die beiden Protokolle RFCOMM und TCS BIN (Telephony Control Specification Binary) werden herkömmliche Kabelverbindungen emuliert. Das RFCOMM wird Kabelersatzprotokoll genannt und emuliert den seriellen RS232 Anschluss über das L2CAP, d.h. RFCOMM unterstützt Anwendungen, die den seriellen Port von Geräten nutzen. Beispiele dafür sind direkte Bluetooth-Verbindungen zwischen einem PC und einem Drucker, oder die Verbindung eines Smartphones mit einem Gateway, das über Kabel mit einem nicht Bluetooth-fähigen Gerät, wie beispielsweise einem Netzwerk-Hub, kommuniziert. RFCOMM ermöglicht TCP/IP Verbindungen über das PPP-Protokoll, das ursprünglich für Modemverbindungen entwickelt wurde. TCS BIN ist ein bitorientiertes Telefonie-Protokoll, das die Rufsignalisierung zum Aufbau von Sprach- und Datenkommunikation zwischen Bluetooth-Geräten regelt. Mit diesem Protokoll lassen sich somit Bluetooth-Geräte wie schnurlose Telefone betreiben. Die von der Bluetooth-Gruppe zusätzlich noch spezifizierten AT-Befehle ermöglichen schließlich noch die Steuerung von Mobiltelefonen und Modems in unterschiedlichen Einsatzumgebungen.

übernommene Protokolle

Neben den Internetprotokollen UDP, TCP, IP, PPP, verwendet Bluetooth auch noch andere nicht Bluetooth-spezifische Protokolle. Zu nennen ist hier unter anderem das OBEX, das ursprünglich von der IrDA zur Infrarot-Kommunikation entwickelt wurde. OBEX ist im Prinzip eine vereinfachte Form des HTTP Protokolls zum Austausch von Objekten. OBEX übernimmt

Aufgaben eines Protokolls der Sitzungsschicht (session layer) gemäß ISO/OSI. Darauf setzen Protokolle zum Objektaustausch wie vCard zum Versenden von Visitenkarten, vCal zur Synchronisation von Terminkalenderdaten oder auch zur Synchronisation von E-Mail-Daten (vMessage, vNotes), die zur Anwendungsschicht zählen.

Der Aufbau und die Verwaltung von Verbindungen zwischen Bluetooth-Geräten ist die Aufgabe des Link Managers. Das ist eine Software, die auf jedem Bluetooth-Gerät läuft. Die Manager tauschen Nachrichten über das Link Management Protocol (LMP) aus. Bluetooth spezifiziert 55 verschiedene Nachrichtenformate, so genannte PDUs (Protocol Data Units), wie beispielsweise LMP_sres, die die Authentifizierungsnachricht enthält, oder LMP_unit_key, die den Geräteschlüssel (s.u.) enthält.

Link Manager

Im Folgenden erklären wir die Sicherheitsarchitektur und Eigenschaften des ursprünglichen 802.15.1 Standards, da diese auch in dem Nachfolge-Standard übernommen wurde. Abschließend gehen wir kurz auf das in der Version v2.0 hinzu gekommenen Protokoll des Secure Simple Pairings ein.

15.5.3 Sicherheitsarchitektur

Die Sicherheitsdienste unter Bluetooth sind auf der Schicht 2, der Datenübertragungsschicht (vgl. Abbildung 3.1), angesiedelt. Sie umfassen Maßnahmen zur einseitigen und auch zur wechselseitigen Authentifikation der Kommunikationspartner, die durch eine Bluetooth-Adresse identifiziert werden, sowie zur Verschlüsselung der transferierten Daten und zur Autorisierung von Dienstenutzungen. Zu beachten ist, dass die Subjekte in Bluetooth ausschließlich Geräte sind. Das heißt, dass sowohl die Autorisierung als auch die Authentifikation allein auf Gerätebasis und nicht für einzelne Dienste oder Benutzer durchgeführt wird. Die Objekte sind die Dienste (z.B. File Transfer, Dial-in), die ein Gerät zur Nutzung durch andere anbietet. An Zugriffsrechten kennt Bluetooth nur den erlaubten und den verbotenen Zugriff, wobei die Zugriffserlaubnis unbedingt oder bedingt vergeben werden kann. Eine Zugriffsbedingung kann zum einen von der Vertrauenswürdigkeit des zugreifenden Geräts und zum anderen von einer vorab erfolgten, korrekten Authentifizierung abhängen. Auf die Zugriffsrechte wird weiter unten noch etwas genauer eingegangen.

Sicherheitsmodell

Die zur Identifikation verwendete Geräteadresse (BD_ADDR) ist eine 48 Bit lange eindeutige Adresse, die von der IEEE definiert wird. Über die 48 Bit-Adressen lassen sich bis zu 281 Billionen Bluetooth-Geräte eindeutig identifizieren. Bluetooth Geräte-Adressen sind öffentlich bekannte Adressen, die über eine *inquiry* Nachricht von einem anderen Bluetooth-Gerät erfragt werden können. Geräte, die sich in der Reichweite befinden

Identifikation

und den Discoverable Modus aktiviert haben, antworten auf die Anfrage mit ihrer Geräteadresse. Mit der *inquiry* Nachricht sucht ein Bluetooth-Gerät auch nach Geräten, die sich in seiner Reichweite befinden. Über eine Page-Nachricht versucht das suchende Gerät sich mit einem der gefundenen Geräte zu verbinden. Abhängig von den geräteabhängigen Sicherheitseinstellungen kann diese Verbindungsaufnahme direkt erfolgen (z.B. es handelt sich um ein unbekanntes Gerät, aber eine Authentifizierung und Autorisierung ist notwendig) oder aber das verbindungsanfordernde Gerät muss Zugangs- und Zugriffskontrollen durchlaufen. Falls das anfordernde Gerät eine Geräteadresse besitzt, die beim gewünschten Verbindungspartner als vertrauenswürdig eingestuft wurde, ist die Verbindungsaufnahme ebenfalls möglich und die Dienste des Verbindungspartners sind ohne Einschränkungen nutzbar. Das suchende Gerät etabliert die Verbindung und wird damit Mitglied des Piconetzes des Partners.

Punkt-zu-Punkt

Da alle Sicherheitsmaßnahmen auf der Datenübertragungsschicht angesiedelt sind, wird keine Ende-zu-Ende-Sicherheit gewährleistet, sondern je zwei direkt miteinander verbundene Bluetooth-Geräte bauen einen, je nach gerätespezifisch festgelegten Sicherheitsanforderungen, authentifizierten, verschlüsselten Kommunikationskanal zwischen sich auf. Die vertrauliche Kommunikation basiert auf einem 128 Bit Verbindungsschlüssel, aus dem ein zwischen acht und 128 Bit langer Kommunikationsschlüssel abgeleitet wird. Die Verschlüsselung erfolgt mittels einer symmetrischen Stromchiffre. Ein Gerätepaar, das miteinander kommunizieren will, verwendet einen solchen gemeinsamen Verbindungsschlüssel, der beim erstmaligen Kontakt der beiden Geräte, dem so genannten Pairing, vereinbart wird.

Sicherheitsmanager

Abbildung 15.28 gibt einen Überblick über die Bluetooth-Sicherheitsarchitektur, die im Folgenden näher betrachtet wird. Die zentrale Komponente bildet der Sicherheitsmanager, der folgende Aufgaben wahrnimmt.

- Er verwaltet die Sicherheitsattribute von Diensten und Geräten (s.u.),
- überprüft die Berechtigung von Zugriffsanfragen beim Verbindungsaufbau,
- führt die Authentifikation durch und
- er ver- und entschlüsselt die Daten bevor eine Verbindung zur Anwendung hergestellt wird.
- Er initiiert bzw. bearbeitet die Eingabe einer ESCE (External Security Control Entity), wie z.B. einem Benutzer, der seine PIN eingibt, und
- er startet das Pairing zweier Geräte, die sich noch nicht kennen. Hierbei wird in der Regel eine PIN-Eingabe durch den Benutzer erforderlich. Eine derartige Eingabe wird ebenfalls vom Sicherheitsmanager angefordert.

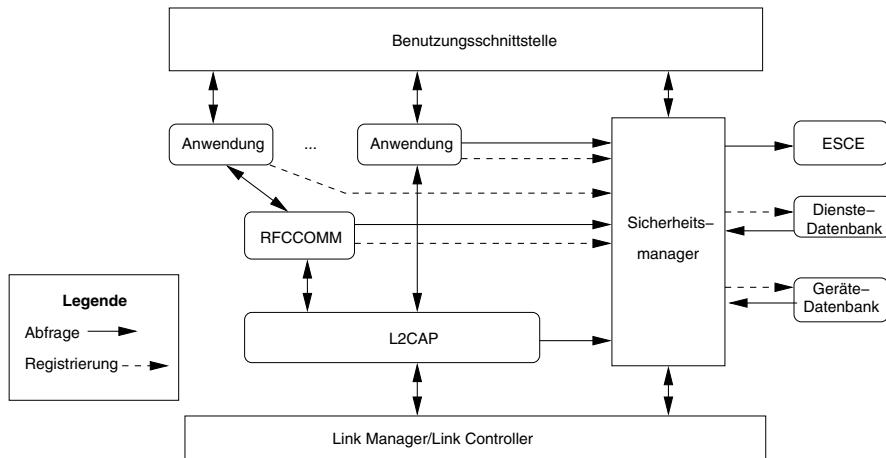


Abbildung 15.28: Bluetooth-Sicherheitsarchitektur

Sicherheitsmodi

Bei Bluetooth unterscheidet man vier Modi. Bei einer Anfrage zur Etablierung einer Bluetooth-Verbindung wird überprüft, in welchem Modus sich die Geräte befinden und abhängig davon werden unterschiedliche Sicherheitsdienste angestoßen. Abbildung 15.29 zeigt ein grobes Ablaufdiagramm mit den unterschiedlichen Aktionen, die in dem jeweiligen Modus durchzuführen sind.

Modus 1

Der Modus 1 ist der unsichere Modus, der keine Sicherheitsfunktionen vorsieht. Die Sicherheitsdienste der Datenübertragungsebene werden in diesem Modus vom Gerät vollständig ignoriert. D.h. in diesem Modus werden vom Gerät selber keine Sicherheitsdienste initiiert. Jedoch reagiert ein Gerät in diesem Modus auf eine Authentifizierungsanforderung eines anderen Geräts seinerseits mit einer Authentifizierung. Aufgrund der fehlenden Sicherheitsdienste sollte der unsichere Modus nur für sicherheitsunkritische Anwendungen eingesetzt werden (z.B. zum elektronischen Austausch von Visitenkarten).

Der Modus 2 bietet Sicherheitsdienste auf der Dienstebene, so dass unterschiedliche Anwendungen mit unterschiedlichen Sicherheitsfunktionen ausgestattet werden können. Im Modus 2 muss zunächst eine L2CAP-Verbindung aufgebaut werden, bevor ein sicherheitskritischer Dienst genutzt werden kann. Bevor eine solche L2CAP-Verbindung etabliert wird, muss im Modus 2 sichergestellt werden, dass der gewünschte Zugriff auf den Dienst bzw. das Gerät erlaubt ist. Die dazu benötigten Informationen stehen in der Gerätedatenbank und ergeben sich aus dem Vertrauenslevel (s.u.), die das Gerät bzw. der Dienst besitzt. So ist beispielsweise bei einem als ver-

Modus 2

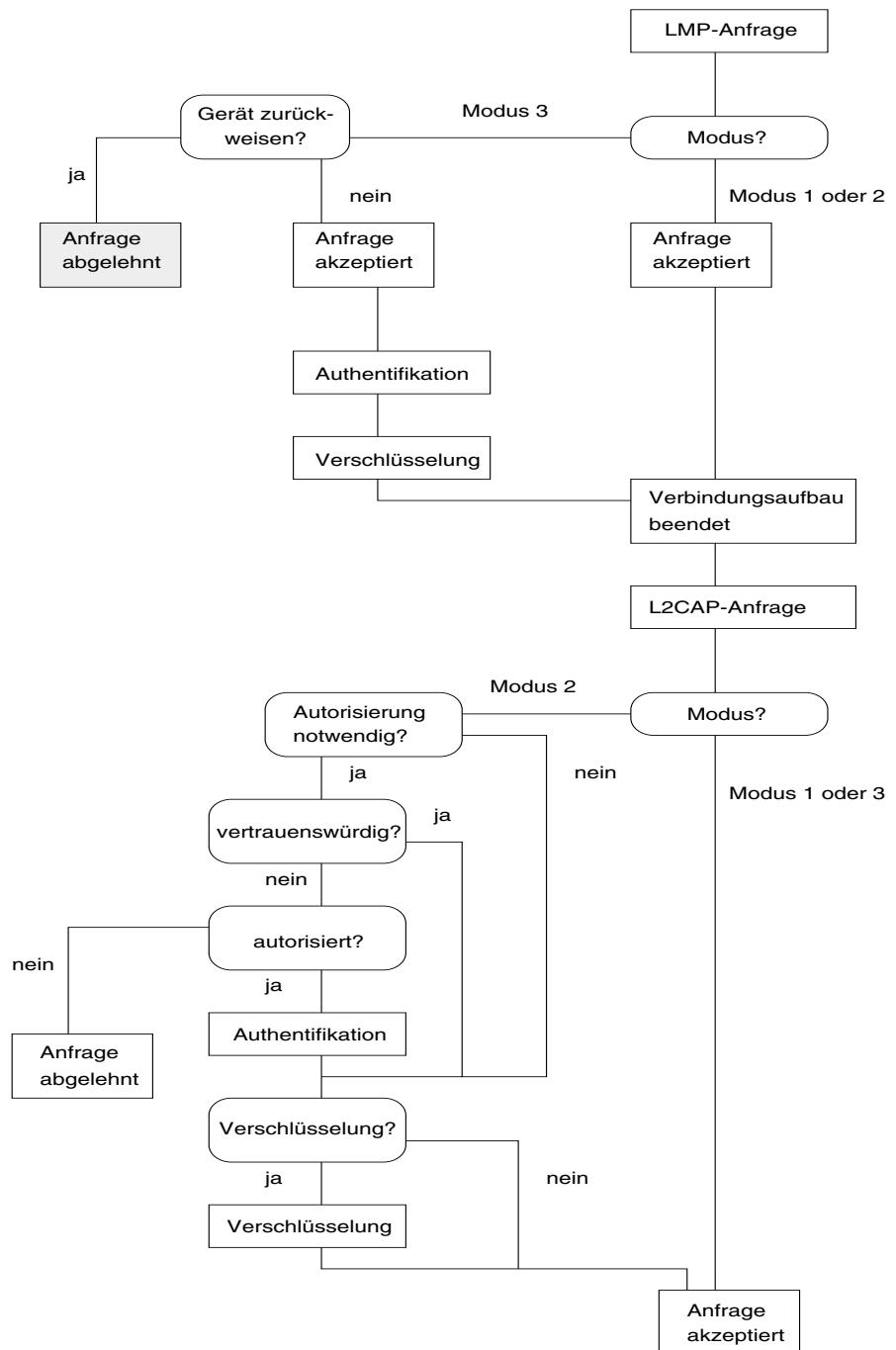


Abbildung 15.29: Sicherheitsmodi

trauenswürdig eingestuften Gerät keine Authentifikation mehr erforderlich und eine Verbindungsanfrage wird akzeptiert. Aufbauend auf den Sicherheitsdiensten des Modus 2 können unterschiedliche Sicherheitsprotokolle und Sicherheitspolicies in die Architektur integriert werden. So beinhaltet ORBIX beispielsweise eine eigene Sicherheitspolicy sowie Sicherheitsprotokolle für den Datei-Transfer oder zur Synchronisation.

Während Modus 2 eine sichere Dienstenutzung unterstützt, setzt der Modus 3 tiefer, nämlich auf der Link-Ebene, an. Irritierender Weise bedeutet also eine höhere Modusnummer (hier Modus 3) nicht, dass damit ein höheres Sicherheitsniveau etabliert wird. Der Modus 3 liefert eine Art Grundschutz, der für alle Anwendungen gleich ist, während der Modus 2 eine anwendungsspezifische Differenzierung ermöglicht. Der Sicherheitsmodus 3 erfordert, dass vor einer Dienstenutzung eine Verbindung mittels des Link-Managers etabliert ist. Der Modus 3 führt eine Authentifizierung durch und verschlüsselt die nachfolgende Kommunikation. Kommunizieren zwei Geräte erstmalig miteinander, so wird in diesem Modus ein gemeinsames Geheimnis, der Verbindungsschlüssel, vereinbart. Zur Schlüsselerzeugung wird eine PIN benötigt, die entweder in-band (z.B. fest im Gerät gespeichert) oder out-of-band (z.B. durch Eingabe durch den Benutzer) an die Geräte übertragen wird. In der Praxis wird in der Regel der Modus 3 verwendet. In Bluetooth 4.2 und 5 sollte der AES-128 zur Verschlüsselung der Verbindung eingesetzt werden.

Der Modus 4 entspricht dem Modus 2 und wurde mit der Bluetooth Version 2.0 eingeführt. Ergänzend zum Modus 2 wird im Modus 4 festgelegt, mit welcher Variante der Secure Simple Pairing Authentisierung (vgl. Abschnitt 15.5.7) der gemeinsame Link-Key vereinbart werden soll. Im Modus 4 wird in den neueren Bluetooth-Versionen der AES-128 sowie EDDH unterstützt.

Modus 3

Modus 4

Vertrauenslevel

Beim Sicherheitsmodus 2 bzw. 4 besitzen Geräte und Dienste eines Bluetooth-Netzes verschiedene Vertrauenslevels (trust-level). Geräte klassifiziert man in *vertrauenswürdige* (trusted) und *nicht vertrauenswürdige*. Vertrauenswürdige Geräte sind authentifiziert und stehen in einer festen Verbindung (pairing) zueinander und erhalten jeweils unbeschränkten Zugriff auf alle von ihnen angebotenen Dienste. Nicht vertrauenswürdige Geräte stehen in keiner permanenten festen Beziehung und der Zugriff auf Dienste wird beschränkt. Ferner gibt es noch die unbekannten Geräte, die jedoch automatisch als nicht vertrauenswürdig eingestuft werden. Bei einer entsprechenden Zuordnung von Vertrauenslevels ist es möglich festzulegen, dass ein als nicht vertrauenswürdig attributiertes Gerät nur Zugriff auf einen speziellen Dienst erhält. Die Einstufung eines Geräts ist in der

Vertrauenslevel
für Geräte

Gerätedatenbank hinterlegt. Für jedes Gerät existiert ein Eintrag, der die Bluetooth-Geräteadresse BD_ADDR, den Gerätename, den Trust-Level sowie den mit dem Gerät verabredeten Verbindungsschlüssel (link key) (s.u.) enthält.

Vertrauenslevel für Dienste

Ein Dienst kann seinerseits einen von drei Sicherheitslevels besitzen, abhängig davon, ob der Dienst eine Autorisierung, nur Authentifikation oder gar nichts benötigt, also offen für alle Geräte ist. Zusätzlich kann für einen Dienst auch festgelegt sein, ob eine Verschlüsselung erforderlich ist. In diesem Fall muss in den Verschlüsselungsmodus umgeschaltet werden, bevor der Zugriff auf den Dienst gewährt wird. Erfordert ein Dienst eine Autorisierung, so ist damit auch immer eine Anforderung zur Authentifikation des entfernt zugreifenden Gerätes verknüpft. Bei gesetztem Autorisierungsattribut wird nur vertrauenswürdigen Geräten automatisch der Zugriff gewährt. Für alle anderen Geräte muss eine manuelle Autorisierung erfolgen. Die Namen der vertrauenswürdigen Geräte werden in einer Gerätedatenbank verwaltet. Das Sicherheitsniveau eines Dienstes wird somit durch drei Attribute bestimmt und diese Attributinformation wird in einer Datenbank des Sicherheitsmanagers verwaltet.

Default- Attributierung

Die Default-Attributierung von Diensten fordert die Autorisierung und damit Authentifikation ankommender Verbindungen, während für ausgehende Verbindungen nur eine Authentifikation gefordert ist. Damit ist in Bluetooth erfreulicherweise das Erlaubnisprinzip, jedenfalls soweit dies mit dem Rechtemodell von Bluetooth möglich ist, umgesetzt worden.

Registrierung

Damit der Sicherheitsmanager die Sicherheitsattribute der Dienste verwalten kann, muss sich ein Dienst beim Manager registrieren lassen, bevor er nutzbar ist. Ohne eine solche Vorab-Registrierung setzt der Manager die Default-Attributierung voraus.

15.5.4 Schlüsselmanagement

Alle Sicherheitsmaßnahmen auf der Verbindungsebene basieren auf 128 Bit Verbindungsschlüsseln (engl. *link key*). Ein Verbindungsschlüssel wird zwischen zwei oder mehreren Partnern vereinbart. Er ist die Basis zur Durchführung der Geräteauthentifikation und aus ihm werden die geheimen Kommunikationsschlüssel abgeleitet, die eine Länge zwischen 8 und 128 Bit besitzen können. Durch die Trennung zwischen Authentifikations- und Verschlüsselungsschlüssel ist es möglich, für die Datenverschlüsselung auch bei Bedarf kürzere Schlüssel zu verwenden (z.B. um gesetzliche Beschränkungen einhalten zu können), ohne die Authentifikationsprozedur zu schwächen. Die Länge der Kommunikationsschlüssel wird bei der Geräteherstellung festgelegt und kann nicht vom Benutzer eines Geräts oder von höheren Software-Schichten verändert werden. Abbildung 15.30 gibt einen

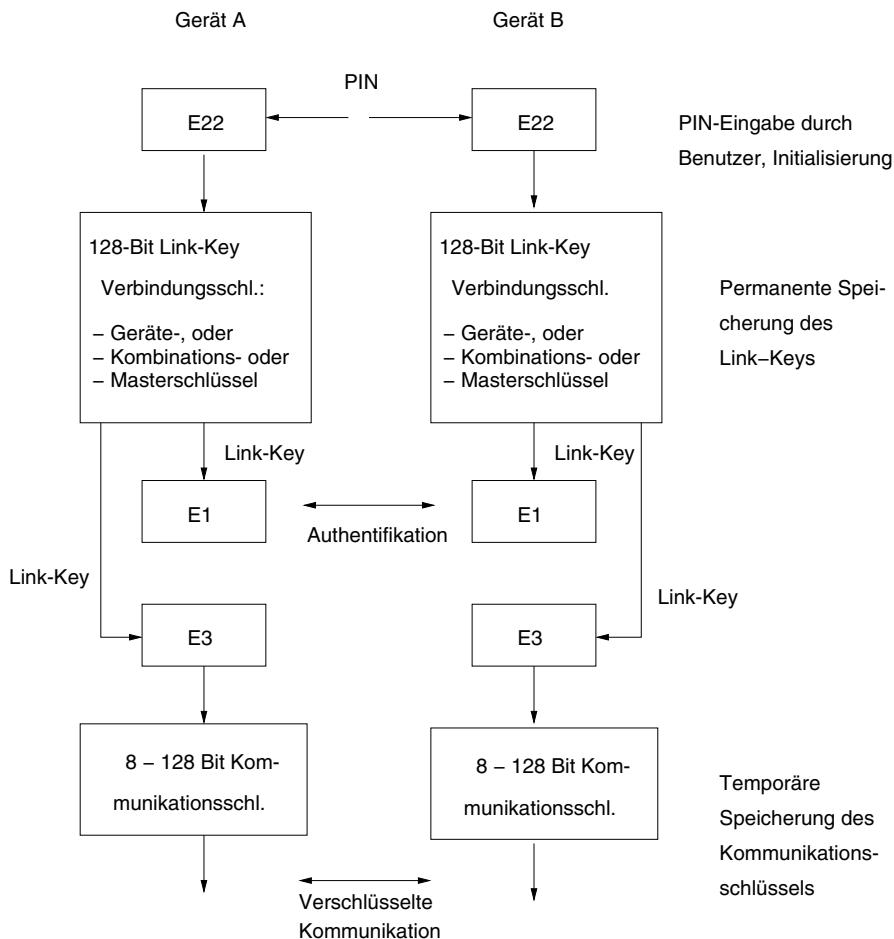


Abbildung 15.30: Schlüsseltypen unter Bluetooth

ersten Überblick über die verschiedenen Schlüsseltypen und deren Abhängigkeiten. Im Folgenden wird deren Erzeugung und Verwaltung genauer diskutiert.

Verbindungsschlüssele können temporäre Schlüsse sein, die nicht für eine weitere Sitzung verwendet werden, oder sie können semi-permanent sein, d.h. sie werden im nicht-flüchtigen Speicher abgelegt und können nach dem Terminieren einer Sitzung für weitere Sitzungen mit denjenigen Geräten, die ebenfalls diesen Schlüssel kennen, verwendet werden. Eine Sitzung ist definiert als die Zeitspanne, in der das Gerät Mitglied eines Piconetzes ist. Mit dem Entkoppeln vom Piconetz terminiert die Sitzung und der temporäre Verbindungsschlüssel kann nicht mehr verwendet werden.

Schlüsseltypen

Da ein Verbindungsschlüssel für verschiedene Aufgaben eingesetzt wird, unterscheidet Bluetooth vier Typen von Verbindungsschlüssel, nämlich den Geräteschlüssel (engl. *unit key*), den Initialisierungs-, den Kombinations- und den Masterschlüssel. Zur Verschlüsselung wird daneben noch ein weiterer Schlüssel, der Kommunikationsschlüssel, verwendet.

Geräteschlüssel

Der 128 Bit Geräteschlüssel und ein Kombinationsschlüssel sind funktional äquivalent. Sie unterscheiden sich jedoch darin, wie sie erzeugt werden. Während ein Kombinationsschlüssel abhängig ist von Informationen, die von beiden Geräten generiert werden, zwischen denen eine Verbindung etabliert wird, ist der Geräteschlüssel gerätespezifisch. Das heißt, im Gegensatz zu den Kombinationsschlüsseln, die für jede Verbindung zwischen zwei Geräten individuell konstruiert und im Speicher abgelegt werden müssen, wird bei der Wahl des Geräteschlüssels als Verbindungsschlüssel kaum Speicherplatz benötigt, da das Gerät nur diesen einen Schlüssel zur Geräteauthentifikation und als Basis für alle Schlüssel, die für seine Verbindungen zu erzeugen sind, verwendet. Ein solcher Geräteschlüssel wird unter Nutzung des Schlüsselerzeugungsverfahrens E21 bei der erstmaligen Benutzung des Geräts erzeugt. In die Schlüsselerzeugung fließt eine 128 Bit Zufallszahl sowie die 48 Bit Geräteadresse BD_ADDR mit ein. Der Schlüssel wird im nicht-flüchtigen Speicher des Geräts abgelegt und in der Regel nicht mehr geändert. Dies ist insofern sehr wichtig, als der gerätespezifische Schlüssel bei der Geräteauthentifikation auch anderen Geräten bekannt gegeben wird. Dies ist immer dann der Fall, wenn das betreffende Gerät über zu wenig Speicherplatz verfügt, um die für die Absicherung von Verbindungen benötigten Kombinationsschlüssel zu speichern. In diesem Fall wird der Geräteschlüssel desressourcenbeschränkten Geräts als Verbindungsschlüssel an den Partner bekannt gegeben und als Basis zur Ableitung von Kommunikationsschlüsseln eingesetzt. Damit ist unmittelbar klar, dass der Geräteschlüssel nur eine äußerst schwache Basis ist, die wann immer möglich vermieden werden sollte.

Pairing

Haben zwei Geräte noch keinen gemeinsamen Schlüssel ausgetauscht, so erfolgt bei ihrem erstmaligen Kontakt das Pairing. Das Pairing erfordert, dass beide Geräte über ein gemeinsames Geheimnis, nämlich eine gemeinsame PIN verfügen, um daraus jeweils die benötigten Schlüssel zu berechnen. Das heißt, dass beim Pairing die PIN auf beiden Geräten eingegeben werden, bzw. auf beiden Geräten bekannt sein muss. Falls sie nur auf einem Gerät eingegeben wird (z.B. falls das Gerät gar nicht über eine Mensch-Maschine Schnittstelle zur Entgegennahme von Eingaben verfügt), muss sie auf dem Partnergerät bereits fest gespeichert sein. Die Eingabe der PIN auf beiden Geräten bietet sich beispielsweise in Szenarien an, in denen ein Laptop und

ein Mobiltelefon des gleichen mobilen Teilnehmers verbunden werden sollen. PINs können von Benutzern verändert werden. Die Länge einer solchen PIN kann zwischen 1 und 16 Byte variieren, so dass neben den herkömmlichen 4-stelligen PINs für sicherheitskritische Anwendungen auch längere PINs einsetzbar sind. Da lange PINs kaum noch von Benutzern korrekt handhabbar sind, können derartige PINs auch über die Anwendungssoftware auf einer höheren Protokollebene, z.B. über das Diffie-Hellman Verfahren, ausgetauscht werden.

Mit dem Schlüsselgenerierungsverfahren E22 (vgl. Abbildung 15.31) erzeugt jedes Gerät nach der erfolgten PIN-Eingabe einen 128 Bit Initialisierungsschlüssel IK. Dazu verwendet E22 als Eingabe den PIN-Code, die

Initialisierungsschlüssel

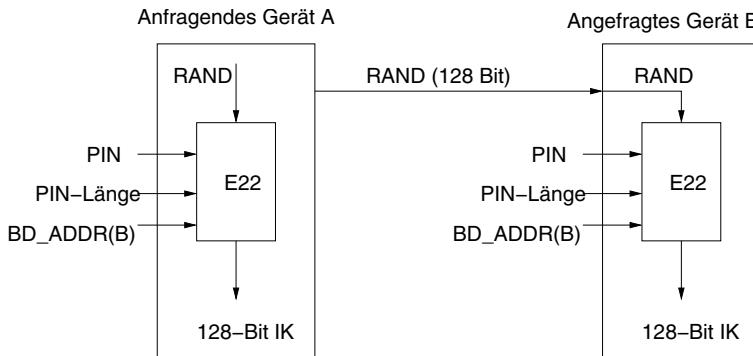


Abbildung 15.31: Erzeugung des Initialisierungsschlüssels

Bluetooth-Gerätedresse des angefragten Geräts sowie die vorab vom Anfrager gesendete Zufallszahl RAND:

$$IK = E22(PIN, BD_ADDR, RAND).$$

Der Initialisierungsschlüssel IK wird zum sicheren Austausch von Informationen in der Phase der Vereinbarung eines Verbindungsschlüssels verwendet und danach vernichtet. D.h. der Initialisierungsschlüssel dient nur während der Initialisierungsphase als Verbindungsschlüssel. Abschließend vergewissern sich die Partner, dass sie beide über den gleichen Initialisierungsschlüssel verfügen. Dazu wählt eines der Geräte eine Zufallszahl und sendet diese an den Partner. Dieser berechnet damit und unter Einbeziehung seiner Gerätedresse sowie seinem Initialisierungsschlüssel die Antwort. Der Anfrager berechnet seinerseits mit der Zufallszahl, der Gerätedresse des angefragten Partners und mit dem Initialisierungsschlüssel die erwartete Antwort. Bei Übereinstimmung ist der Partner authentifiziert.

Bei wiederholter Verbindungsaufnahme zwischen den beiden Geräten muss stets eine erneute Authentifikation durchgeführt werden, wozu dann ein wei-

terer Verbindungsschlüssel, der Kombinationsschlüssel (s.u.), der permanent gespeichert sein kann, verwendet wird. Ein Pairing muss dann nicht mehr erneut stattfinden. Falls als Verbindungsschlüssel der Geräteschlüssel eines der Partner gewählt werden muss, so verknüpft derjenige, der die Speicherbeschränkung hat, seinen Geräteschlüssel mit dem Initialisierungsschlüssel mittels XOR und sendet das Ergebnis an das Partnergerät. Der Partner hat seinerseits nach dem Pairing den Initialisierungsschlüssel berechnet und kann damit den empfangenen Geräteschlüssel extrahieren (XOR-Operation) und als neuen Verbindungsschlüssel verwenden.

Kombinations-schlüssel

Ein Kombinationsschlüssel wird im Verlauf des Initialisierungsprozesses von Informationen abgeleitet, die von zwei Geräten A und B stammen (vgl. Abbildung 15.32). Dazu generieren beide Geräte zunächst jeweils

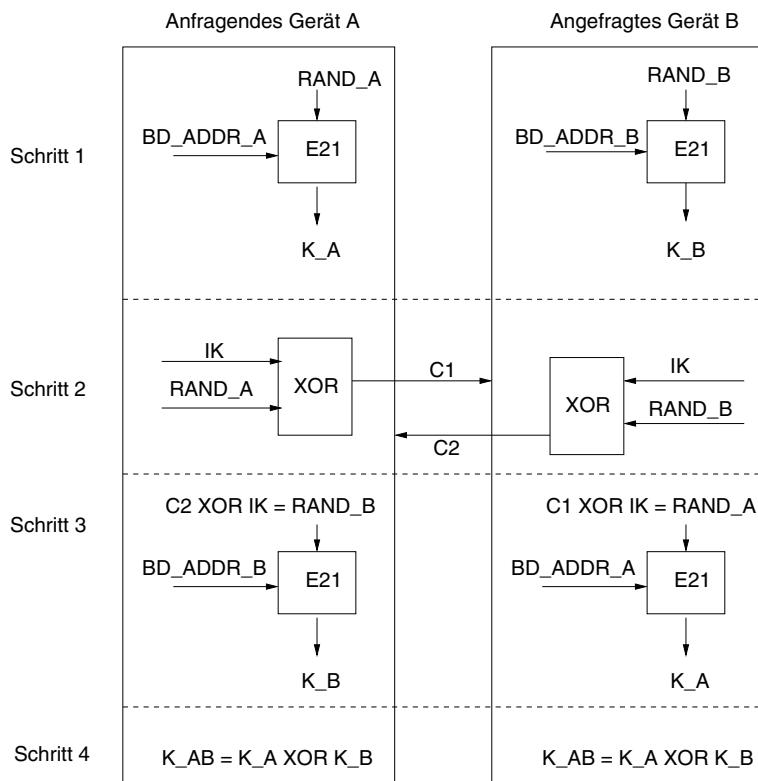


Abbildung 15.32: Erzeugung des Kombinationsschlüssels

eine Zufallszahl RAND_A bzw. RAND_B und generieren, ebenfalls unter Nutzung des E21 Verfahrens und ihrer Geräteadresse, einen Schlüssel K_A bzw. K_B. Im nächsten Schritt tauschen die Geräte auf sicherem Weg ihre Zufallszahlen aus, indem sie diese jeweils mit dem im Pairing erzeugten

Initialisierungsschlüssel IK mittels XOR verknüpfen. Durch eine erneute XOR-Verknüpfung mit dem Initialisierungsschlüssel können die Partner im nächsten Schritt die Zufallszahl ihres jeweiligen Partners wieder herstellen. Mit dieser Information berechnet A den Schlüssel K_B und B berechnet K_A. Der Kombinationsschlüssel wird im abschließenden Schritt durch die XOR-Kombination beider Teilschlüssel K_A und K_B von jedem der Geräte berechnet. Mittels eines wechselseitig durchgeführten Challenge-Response-Protokolls vergewissern sich beide Geräte, dass sie den gleichen Schlüssel verwenden.

Der Masterschlüssel ist ein temporärer Schlüssel, der den aktuell verwendeten Verbindungsschlüssel ersetzt. Ein solcher Schlüssel wird benötigt, wenn Informationen (z.B. als Broadcast-Nachricht) zu mehreren Empfängern übertragen werden sollen. Das Gerät, das als Master fungiert, erzeugt dazu mittels des E22 Schlüsselerzeugungsverfahrens und zweier Zufallszahlen einen 128 Bit Masterschlüssel, der temporär die Aufgaben des Verbindungsschlüssels übernimmt. Das bedeutet, dass alle Partner dieser Punkt-zu-Multipunktverbindung den gleichen Kommunikationsschlüssel auf der Grundlage des Masterschlüssels berechnen und verwenden. Das hat zur Konsequenz, dass jeder Slave in dieser Multipunktverbindung alle Nachrichten des Masters, auch diejenigen, die gar nicht für ihn selber bestimmt sind, entschlüsseln kann. Der Master kann über einen entsprechenden Befehl die Slaves dazu auffordern, ihre alten Verbindungsschlüssel wieder einzusetzen.

Masterschlüssel

15.5.5 Authentifikation

Die Authentifikation basiert auf einem Challenge-Response Protokoll mit dem SAFER+-Algorithmus²¹, das sowohl vom Master als auch von einem der Slaves initiiert werden kann. Es sind zurzeit keine relevanten Angriffe auf diesen Algorithmus bekannt.

Challenge-Response

Abbildung 15.33 gibt einen Überblick über den Ablauf des Authentifizierungsprotokolls.

Zur Authentifikation wird, wie üblich, eine 128 Bit Zufallszahl als Challenge gesendet (LMP_au_rand), die vom Partner zu beantworten ist. Der Partner berechnet eine Antwort, in die zum einen seine eigene Bluetooth-Gerätedresse und zum anderen der geheime 128-Bit Verbindungsschlüssel K (s.o.) eingeht. Aus diesem symmetrischen Schlüssel berechnen beide Partner den ACO-Wert (Authentication Ciphering Offset), der später bei der Erzeugung des Kommunikationsschlüssels verwendet wird. Fehlt ein solcher im Vorfeld (beim Pairing) vereinbarter Verbindungsschlüssel, so sendet der angefragte Partner eine Nachricht LMP_not_accepted an den Anfrager

²¹ Siehe Bluetooth-Informationen unter <http://www.mobileinfo.com/bluetooth>.

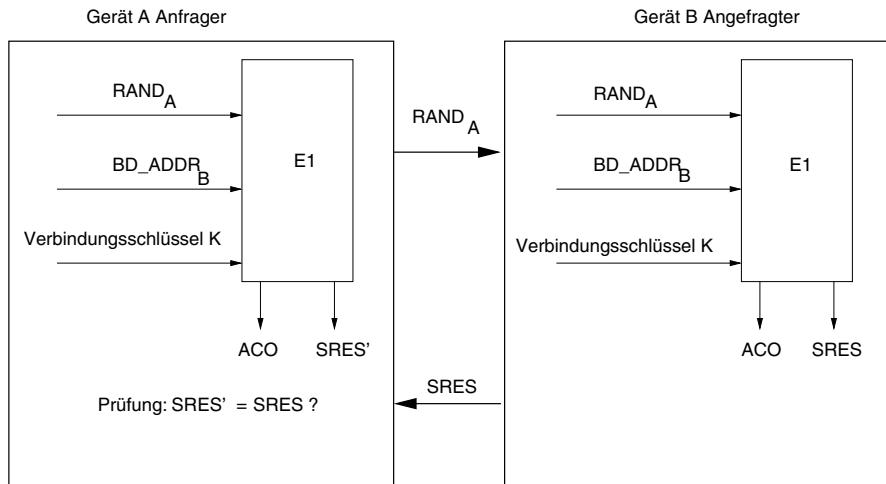


Abbildung 15.33: Authentifikationsprotokoll unter Bluetooth

zurück und gibt als Grund *key missing* an. In einem solchen Fall kann der Sicherheitsmanager des anfragenden Geräts ein Pairing (s.o.) initiieren, um einen Verbindungsschlüssel zu vereinbaren.

Policy

Viele Anwendungen bzw. Dienste sind mit einer einseitigen Authentifikation zufrieden. Es gehört zu den Aufgaben des Link-Managers, die Authentifikationsanforderungen einzuhalten, so dass, abhängig von der jeweiligen Policy, auch der Partner eine Challenge versendet, um eine Authentifikation zu starten.

Fehlschlag

Ist eine Authentifikation fehlgeschlagen, so muss zunächst ein Zeitintervall abgewartet werden, bevor eine erneute Authentifikationsanfrage gestellt werden kann. Das Warteintervall verlängert sich exponentiell mit jedem Fehlversuch. Wird in einem Intervall kein Fehlversuch mehr registriert, so wird das Warteintervall wieder zurückgesetzt. Auf diese Weise sollen Brute-Force-Angriffe durch Ausprobieren aller SchlüsseL behindert werden.

Problem

Auch wenn die Authentifikation erfolgreich abgelaufen ist, muss man sich klar darüber sein, dass sich unter Bluetooth zwar Geräte, nicht jedoch die jeweiligen Benutzer authentifizieren. Das heißt, dass ein Angreifer, der im Besitz eines Bluetooth-Geräts ist, dies meist direkt nutzen kann, da eine PIN-basierte Authentifikation ja nur beim Erstkontakt zwischen Geräten erforderlich ist. Darüber hinaus ist es auch nicht möglich, dass sich Dienste/Prozesse gegenüber den Partner-Geräten authentifizieren, so dass keine spezifische sichere Kommunikation zur Erledigung einer dedizierten Aufgabe wie beispielsweise die Datensynchronisation zwischen zwei Bluetooth-Geräten ohne weitere Zusatzmaßnahmen auf höheren Protokollschichten (also außerhalb von Bluetooth) etabliert werden kann.

Vertraulichkeit

Der zu verwendende Kommunikationsschlüssel wird mit dem Schlüsselerzeugungsverfahren E3 unter Verwendung des aktuellen 128-Bit Verbindungsschlüssels, einer 128 Bit Zufallszahl sowie der 96 Bit COF (Ciphering Offset Number) erzeugt (vgl. Abbildung 15.34). Die COF basiert auf dem ACO, der während der Authentifizierungsphase berechnet wird. Der Link-Manager erzeugt automatisch den für den sicheren Datentransfer erforderlichen Kommunikationsschlüssel, sobald das Gerät in den Verschlüsselungsmodus wechselt. Unter Verwendung dieses Schlüssels wird jedes Paket separat verschlüsselt. Gemäß der Bluetooth-Spezifikation kann die Schlüssellänge zwischen 8 und 128 Bit variieren. Dadurch soll es möglich sein, Bluetooth auch in solchen Ländern einzusetzen, in denen die Verwendung starker Kryptografie mit langen Schlüsseln untersagt ist.

Schlüssel

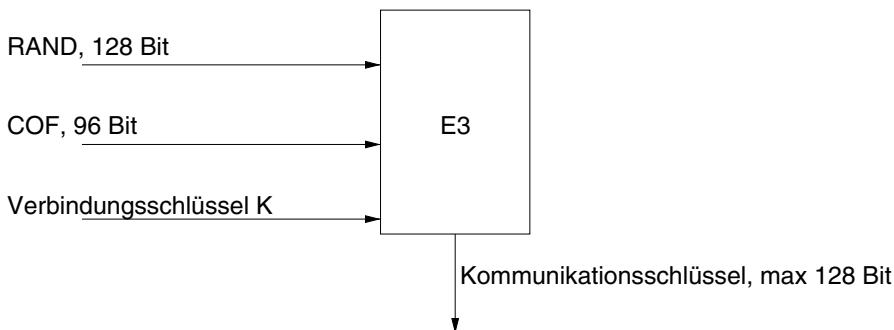


Abbildung 15.34: Erzeugung eines Kommunikationsschlüssels

Da bei dem unter Bluetooth genutzten Verschlüsselungsverfahren E0 keine feste Schlüssellänge vorgegeben ist, müssen sich die beteiligten Geräte über die aktuelle Schlüssellänge verständigen. Jedes Gerät definiert dazu einen eigenen zulässigen Minimal- und Maximalwert für eine Schlüssellänge. Diese Werte sind fest codiert und nachträglich nicht mehr änderbar. Während der Schlüsselvereinbarung sendet der Master seine Maximallänge als Vorschlag an alle Slaves. Diese antworten entweder mit einer Bestätigungsnachricht oder senden ihrerseits einen Vorschlag zurück. Dies wird so lange durchgeführt, bis ein Konsens erzielt wurde oder eines der Geräte das Protokoll abbricht. Das bedeutet, dass sich im schlechtesten Fall die Geräte auf die gemeinsame Mindestlänge verständigen, die in der Regel mit 8 Bit festgelegt ist! Dies ist auch deshalb problematisch, da ein mobiler Teilnehmer keinen Einblick in den Status dieser Schlüssellängenaushandlung erhält, also nicht weiß, welche Länge die schließlich vereinbarten Schlüssel tatsächlich haben. Zur Schlüssellängenreduktion werden übrigens einfach eine feste Anzahl von Bits des Schlüssels mit Null vorbelegt.

Schlüssellänge

Mehr Transparenz erhalten während des Aushandlungsprozesses die Anwendungen, da sie ihrerseits einen Schwellenwert festlegen können. Eine Anwendung bricht die Aushandlung ab, wenn eine Schlüssellänge angeboten wird, die geringer ist als die, die die Applikation (der Dienst) für sich als minimale Länge festgelegt hat. Damit soll verhindert werden, dass ein böswilliges Gerät einem Partner einen sehr schwachen Schlüssel aufzwingt.

Verschlüsselung

Bei Bluetooth wird die Nutzlast in den Datenpaketen unter Verwendung der Stromchiffre E0, einer symmetrischen Chiffre mit vier linearen rückgekoppelten Schieberegistern (vgl. [74]), verschlüsselt. Abhängig davon, ob ein Gerät einen semi-permanenten (Geräte- oder Kombinationsschlüssel) oder einen Masterschlüssel verwendet, sind unterschiedliche Schlüsselmodelle nutzbar. Mit einem semi-permanenten Schlüssel ist es nicht möglich, Broadcastverkehr zu verschlüsseln, sondern allein Individualverkehr. Beim Einsatz eines Masterschlüssels kann zwischen drei Modi²² gewählt werden. Im Modus 1 erfolgt keine Verschlüsselung, im Modus 2 bzw. 4 wird zwar der Individual- aber nicht der Broadcastverkehr verschlüsselt und nur im Modus 3 wird auch der Broadcastverkehr verschlüsselt. Die Stromchiffre E0 ist sehr schnell und effizient in Hardware implementierbar. Eine Kryptoanalyse von E0 hat gezeigt, dass Angriffe mit einem Aufwand von $\mathcal{O}(2^{100})$ bzw. $\mathcal{O}(2^{66})$ möglich sind. Derartige Angriffe haben jedoch keine praktische Relevanz.

Ad-hoc-Aspekte

Bei einer ad-hoc-Vernetzung können die beteiligten Bluetooth-Geräte Kombinationsschlüssel verwenden. Das bedeutet, dass der Master einen Kombinationsschlüssel mit jedem seiner Slaves vereinbart. Die Informationen eines Slaves müssen dann vom Master zu allen anderen Slaves des Piconetzes gesendet werden. Wird stattdessen ein Masterschlüssel verwendet, so können alle Slaves des Piconetzes miteinander kommunizieren und eine Weiterleitung des Datenverkehrs wird vermieden. Weitergehende Sicherheitskonzepte, wie die Verteilung von geheimen Schlüsseln durch ein Key Distribution Center, wie es beispielsweise aus Kerberos bekannt ist, werden jedoch von Bluetooth nicht direkt unterstützt. Alle weiterführenden Maßnahmen müssten somit auf die Anwendungsebene verlagert werden.

15.5.6 Bluetooth-Sicherheitsprobleme

Obwohl die Sicherheitsarchitektur recht gut durchdacht ist, weist auch Bluetooth einige Schwächen und Angriffspunkte auf. Abschließend werden nun die wichtigsten Schwachstellen diskutiert.

PIN

Als eine Hauptschwachstelle ist der Einsatz von PINs zu betrachten. Abbildung 15.30 verdeutlicht, dass die PIN die Basis aller Sicherheitsmaßnahmen

²² Vorsicht, es handelt sich hier um andere Modi, als die, die für das ganze Gerät gelten (siehe Seite 929).

ist. So hängt die Vertraulichkeit des beim Pairing generierten Initialisierungsschlüssels allein von der PIN ab und der Initialisierungsschlüssel ist wiederum die Basis zur Erzeugung des Verbindungsschlüssels, von dem letztlich alle Kommunikationsschlüssel abgeleitet werden. Die manuelle Eingabe von PINs in beide Geräte bei deren erstmaligem Kontakt ist höchstens für Personal Area Networks (PAN) praktisch umsetzbar, während das Vorgehen für ad-hoc-Vernetzungen ungeeignet ist. Aber auch bei PANs wird es einem mobilen Teilnehmer sicherlich schnell lästig, einen mehrstelligen PIN-Code in die Geräte einzugeben, so dass zu befürchten ist, dass entweder nur vierstellige, womöglich leicht zu merkende PINs, wie 0000, verwendet werden, oder dass die PIN fest in den Geräten gespeichert wird. Ein vierstelliger PIN-Code erlaubt nur 10.000 unterschiedliche PINs. Da auf den Geräten darüber hinaus auch noch häufig die PIN 0000 als Voreinstellung festgelegt ist, wird diese immer dann verwendet, wenn keine PIN-Eingabe verfügbar ist. Gemäß der Bluetooth-Spezifikation ist es ferner auch möglich, die PIN in-band, zum Beispiel über die Luftschnittstelle, zum Gerät zu übertragen, wodurch PINs natürlich abgehört werden können. Insgesamt ist klar, dass die PIN-Basis sehr schwach ist, wodurch auch die gesamte darauf aufsetzende Sicherheitsarchitektur schwach wird. Mit der Kenntnis von PINs lassen sich Initialisierungs-, Kombinations-, Masterschlüssel und natürlich auch die Verschlüsselungsschlüssel auf einfachste Weise brechen, da alle anderen Informationen entweder bekannt sind oder in den Protokollschriften im Klartext übertragen werden. Benutzer sollten deshalb für Anwendungen mit sensiblen Daten eine mindestens 64 Bit PIN wählen. Mit dem BTCrack Tool²³ steht ein frei verfügbares Werkzeug zur Verfügung, um aus abgefangenen Pairing-Daten die verwendete PIN zu knacken und den Verbindungsschlüssel herzuleiten.

Wie weiter oben bereits erwähnt, wurde mit der Bluetooth Version v2.1 das Secure Simple Pairing eingeführt, wodurch die skizzierte Problematik der PIN-basierten Authentifizierung entschärft wird. Wir gehen weiter unten auf dieses Protokoll ein.

Da ressourcenschwache Geräte nicht in der Lage sind, die benötigten Schlüssel zu berechnen und zu speichern, verwenden derartige Geräte ihren eigenen Geräteschlüssel zur Kommunikation. Ein Angreifer muss also nur Kontakt zu einem solchen Gerät A aufnehmen, um dessen Schlüssel zu erhalten und kann nachfolgend jegliche Kommunikation zwischen anderen Geräten und A entschlüsseln.

Hat ein Angreifer einen Verbindungsschlüssel (z.B. den Geräteschlüssel), der von zwei Geräten A und B verwendet wurde, kompromittiert, so ist er in der Lage, eine Man-in-the-Middle-Attacke durchzuführen. Dazu kontaktiert er beide Geräte und gibt sich A gegenüber als B und umgekehrt gegenüber B

Geräteschlüssel

Man-in-the-Middle

²³ Siehe <http://code.google.com/p/btcrack-open-source/>

als A aus und vereinbart jeweils einen neuen Verbindungsschlüssel mit A und B. Beide Geräte glauben, dass sie nach wie vor miteinander kommunizieren. Damit sie den Mittelsmann nicht entdecken, muss dieser dafür sorgen, dass A und B mit unterschiedlichen Hop-Sequenzen senden und empfangen, also unsynchronisiert sind. Da jeweils der Master eines Piconetzes das zu verwendende Hop-Muster bestimmt, kann der Angreifer beispielsweise beide Geräte zu Mastern machen. In der Aushandlungsphase des Protokolls, noch vor der Schlüsselvereinbarung, erfolgt die Festlegung, wer Slave und wer Master ist. Geräte können aber die Option, Master zu werden, ausschalten, so dass damit ein solcher Angriff abgewehrt werden kann.

Bewegungsprofil

Falls ein Gerät sich im Discoverable Modus befindet, antwortet es auf jede Anfrage eines anderen Geräts in seiner Reichweite mit seiner Geräteadresse. Da diese Adresse fest und eindeutig einem Gerät zugeordnet ist, kann ein Angreifer Aufenthalts- und Bewegungsprofile von Geräten erstellen, wenn er in der Lage ist, seinerseits Geräte in den unterschiedlichen Domänen zu platzieren, in denen sich das beobachtete Gerät aufhält. Die Beobachtungsgeräte müssen dann nur in regelmäßigen Abständen Anfrage-Nachrichten senden, um in Erfahrung zu bringen, welche Geräte sich in ihrer Reichweite aufhalten. Werden zusätzlich auch zeitbezogene Informationen aufgezeichnet, so ist es einem Angreifer darüber hinaus auch möglich zu erkennen, welche Geräte sich wiederholt treffen oder zusammen unterwegs sind. Man sollte deshalb Bluetooth-Geräte soweit möglich nur im Non-Discoverable Modus verwenden bzw. sie nur dann einschalten, wenn man sie wirklich nutzen möchte.

Authentifikation

Beim Einsatz von Bluetooth für sicherheitskritische Transaktionen wie Banktransaktionen muss man sich darüber im Klaren sein, dass eine Authentifikation ausschließlich für Geräte und nicht für einzelne Benutzer oder Dienste durchgeführt wird. Das bedeutet, dass jeder Besitzer eines Bluetooth-Geräts berechtigt ist, damit zu kommunizieren. Bluetooth sieht auch keine Maßnahmen vor, um Replay-Angriffe abzuwehren, d.h. in die Authentifizierungsnachrichten müssten Nonces, Sequenznummern oder Zeitstempel eingebracht werden, um dies zu erreichen. Anwendungen, die eine differenzierte Authentifizierung erfordern, müssen die notwendigen Maßnahmen selbst implementieren.

Ferner ist zu beachten, dass eine Überprüfung der Autorisierung nicht bei jeder Dienstenutzung erneut erfolgt, sondern nur einmalig beim Verbindungsaufbau. Damit ist das Vorgehen im Prinzip vergleichbar mit der Ausstellung von File Handles (oder Tickets) bei der Zugriffskontrolle in heutigen Betriebssystemen (vgl. Kapitel 12), obwohl natürlich unter Bluetooth keine benutzer- oder rollenspezifische Rechtevergabe möglich ist und auch keine differenzierten Nutzungsrechte festlegbar sind. Auch ein uni-

direktonaler Datenfluss ist nicht möglich, da nach einem erfolgreichen Verbindungsaufbau stets eine bidirektionale Kommunikation eröffnet wird.

15.5.7 Secure Simple Pairing

Das Ziel des Secure Simple Pairing²⁴ besteht darin, Eavesdropping und Man-in-the-Middle Angriffe während des Pairings abzuwehren. Das Protokoll besteht aus 5 Phasen, nämlich (1.) dem Public-Key Exchange auf der Basis des Diffie-Hellman-Verfahrens, (2.) und (3.) der Authentifizierung mit den Stufen 1 und 2, (4.) Berechnung des Verbindungsschlüssels und (5.) Link-Key Berechnung und Berechnung des Kommunikationsschlüssels. In den Phasen werden festgelegte kryptografische Funktionen, genannt f_1 , f_2 , f_3 und g , verwendet, die im Wesentlichen auf der Verwendung des HMAC-SHA-256 Verfahrens beruhen. Die genaue Funktionsweise der Funktionen ist für das Verständnis des Ablaufs nicht erheblich. Der interessierte Leser sei auf das angegebene Dokument verwiesen.

In der Phase 1, dem Public Key Exchange, generiert jedes der beiden zu authentifizierenden Geräte sein eigenes Elliptic Curve Diffie-Hellman Schlüsselpaar. Durch das Senden des Public Keys eines der Geräte wird das Pairing initiiert. Für die nachfolgende Beschreibung gehen wir davon aus, dass das Gerät A das Pairing initiiert und seinen Public Key Pub_A an das Gerät B sendet. Das Partner-Gerät B antwortet mit seinem eigenen öffentlichen Schlüssel Pub_B und beide Geräte berechnen gemäß dem Diffie-Hellman-Schema den gemeinsamen DH-Schlüssel DH-Key, also $DH\text{-Key} = EC\text{-DH}(Priv_A, Pub_B)$ bzw. $DH\text{-Key} = EC\text{-DH}(Priv_B, Pub_A)$.

Phase 1

Protokolle der Phase 2

In der zweiten Phase erfolgt zunächst die Authentifizierung der Stufe 1. Hierfür bietet Bluetooth v2.1 verschiedene Protokolle, nämlich den numerischen Vergleich, Just Works, Out-of Band und Passkey Entry. Abhängig von den Fähigkeiten der beteiligten Geräte wird in diesem Schritt eines dieser Protokolle zur Authentifizierung angewandt.

Voraussetzung für die Verwendung des Protokolls *Numerischer Vergleich* ist, dass beide Geräte jeweils ein Display zur Anzeige einer 6-stelligen Zahl und eine ja/nein Taste besitzen (z.B. beim Pairing eines Smartphones mit einem Laptop). Beide Geräte berechnen nach einem festgelegten Verfahren (s.u.) eine 6-stellige Zahl und zeigen diese auf dem Display der Geräte an. Der Benutzer vergleicht die Zahlen und drückt die Ja-Taste bei Übereinstimmung bzw. die Nein-Taste, falls Unterschiede erkannt werden. Auf diese Weise

²⁴ siehe http://www.bluetooth.com/NR/rdonlyres/0A0B3F36-D15F-4470-85A6-F2C-CFA26F70F/0/SimplePairing_WP_V10r00.pdf

wird ein möglicher Man-in-the-Middle Angriff abgewehrt, da das Protokoll nach dem Drücken der Nein-Taste terminiert.

Das Protokoll *Just Works* ist verwendbar, wenn mindestens eines der Geräte kein Display zur Anzeige einer 6-stelligen Zahl besitzt und auch keine Tastatur zu deren Eingabe (z.B. Pairing eines Headsets mit einem Mobiltelefon). Es wird hierbei wiederum das Verfahren des numerischen Vergleichs verwendet, so dass beide Geräte die 6-stellige Zahl berechnen. Diese Zahl wird jedoch nicht angezeigt, so dass der Benutzer nicht prüfen kann, ob ein Man-in-the-Middle-Angriff stattgefunden hat.

Wie der Name schon verrät, verwendet das *Out-of Band*-Protokoll einen externen Kanal, z.B. Near Field Communication (NFC), zum Austausch von Bluetooth-Adressen und kryptografischen Schlüsseln. Die Authentisierung findet extern statt.

Das *Passkey EntryProtokoll* ist einsetzbar, wenn eines der Geräte eine Eingabemöglichkeit, aber kein Display zur Anzeige einer 6-stelligen Zahl besitzt (z.B. eine BT-fähige Tastatur), das zweite Gerät jedoch über ein Display verfügt (z.B. ein Mobiltelefon). In diesem Fall generiert das Gerät mit der Anzeige-Funktion eine 6-stellige Zahl, zeigt sie dem Benutzer an und dieser gibt die Zahl über die Tastatur in das zweite Gerät ein.

Authentifikation Stufe 1 mit dem Protokoll numerischer Vergleich

Betrachten wir den Ablauf des Protokolls *Numerischer Vergleich* zur Berechnung der 6-stelligen Zahlen. Hierzu generiert zunächst jedes der Geräte A und B eine 128-Bit Zufallszahl N_a bzw. N_b .

1. Das Gerät, das das Pairing nicht initiiert hat (in unserem Beispiel also Gerät B), berechnet mit einer festgelegten Hashfunktion $f1$ einen Commitment-Wert C_b über die ausgetauschten DH-Public-Keys Pub_B und Pub_A , sowie seine Nonce N_b :

$$C_b = f1(Pub_B, Pub_A, N_b, 0)$$

2. Der Commitment-Wert C_b wird an das Gerät A übermittelt.
3. Beide Geräte tauschen im Klartext ihre jeweiligen Zufallszahlen N_a und N_b aus.
4. Das initiierende Gerät A prüft nun, ob der Commitment-Wert C_b des Partners B korrekt ist, indem es selber mit seinen Daten den Wert erneut berechnet und das Ergebnis mit dem in Schritt 2 erhaltenen Wert vergleicht. Bei Ungleichheit bricht das Gerät A das Pairing ab. Ist der Wert jedoch korrekt, so berechnen beide Geräte jeweils eine 6-stellige Zahl V_a bzw. V_b .

5. Beide Geräte berechnen die Zahlen V_a bzw. V_b nach dem gleichen Schema unter Verwendung einer festgelegten Funktion g :

$$V_a = g(Pub_A, Pub_B, N_a, N_b) \text{ bzw. } V_b = g(Pub_A, Pub_B, N_a, N_b)$$

6. Die Werte V_a bzw. V_b werden auf den jeweiligen Geräten auf dem Display angezeigt.
7. Durch Drücken einer Bestätigungstaste signalisiert der Benutzer, dass beide angezeigten Zahlen gleich oder ungleich sind. Bei Ungleichheit wird das Pairing abgebrochen.

Der Ablauf beim numerischen Vergleich ist in Abbildung 15.35 noch einmal zusammengefasst. Mit dieser Vorgehensweise ist die Gefahr von Man-in-the-Middle Angriffen deutlich abgeschwächt.

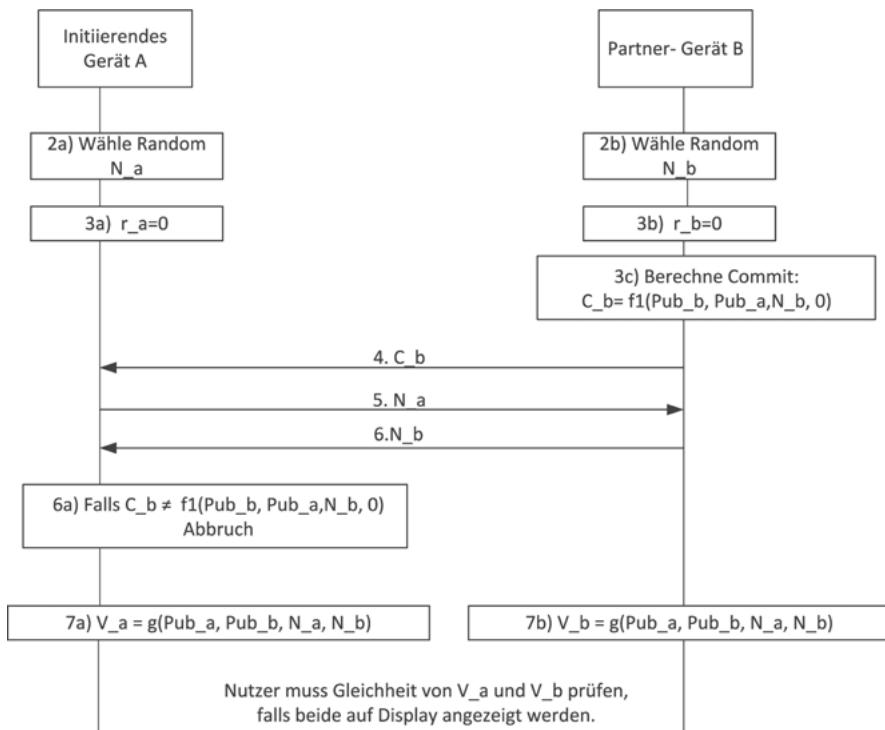


Abbildung 15.35: Simple-Pairing Ablauf mit numerischem Vergleich

Authentifikation Stufe 1 mit dem Protokoll Passkey Entry

Das *Passkey Entry* Protokoll agiert in einer etwas abgewandelten Weise. Ausgangspunkt sind hier PINs (oder Passkeys, wie sie hier genannt werden),

die entweder, wie beim ursprünglichen Standard vom Benutzer auf beiden Geräten einzugeben sind. Oder der PIN wird auf einem der beiden Geräte generiert und auf dem Display angezeigt. Der Benutzer muss die PIN dann auf dem zweiten Gerät eingegeben. Anders als beim ursprünglichen Standard bildet dieses Geheimnis jedoch nur einen Startpunkt; d.h. die Authentifikation basiert nicht allein auf einer beliebig wählbaren PIN. In k iterierenden Schritten, wobei k die Länge der PIN in Bits darstellt, führen die beiden Geräte eine Commitment-Berechnung analog zu der aus dem numerischen Verfahren durch. Das heißt, dass sie sich über jedes Bit der PIN verständigen. Für jedes der k -Commitments werden zwei neue Zufallszahlen N_A und N_B generiert. Das Pairing wird abgebrochen, sobald ein Vergleich der Commitment-Werte fehlschlägt. Abbildung 15.36 fasst den Ablauf zusammen.

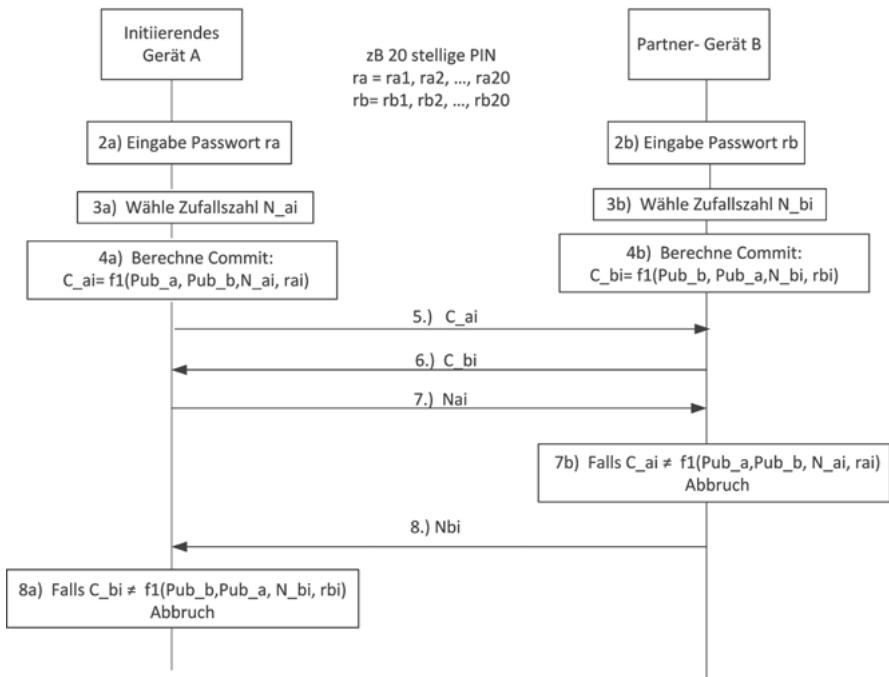


Abbildung 15.36: Simple-Pairing mit Passkey Entry

Mit diesem Bit-weisen Abgleich der PIN ist sicher gestellt, dass bei einem Man-in-the-Middle Angriff nicht mehr als ein Bit an Information über den Passkey an den Angreifer durchsickert.

In der Phase 3 des Protokolls erfolgt der Schritt 2 der Authentifizierung. In diesem Schritt bestätigen sich die Geräte wechselseitig, dass der Austausch erfolgreich war. Beim Auftreten einer Fehlermeldung wird das Protokoll abgebrochen. Für die wechselseitige Bestätigung werden die in dem Passkey-Protokoll verwendeten Zufallszahlen zu einer Zufallszahl r_a bzw. r_b konkateniert (falls ein anderes Protokoll verwendet wurde, so sind diese Werte auf 0 gesetzt) und die Bestätigungs Nachrichten werden ausgetauscht, wobei der gemeinsame Schlüssel DH-Key in die Erstellung der Bestätigung einfließt. Mit IOCaps werden die Fähigkeiten der BT-Geräte charakterisiert.

Phase 3

1. Beide Geräte berechnen einen Wert E_a bzw. E_b unter Nutzung des Verfahrens f3 wie folgt: (hier nur für B angegeben)

$$E_b = f3(\text{DH-Key}, N_b, N_a, r_b, \text{IOCaps}_B, \text{BT_Addr}_B, \text{BT_Addr}_A)$$

2. Das initiiierende Gerät A sendet seinen Wert E_a an das Gerät B.
3. Das Gerät berechnet seinerseits den Wert E_a mit den ihm vorliegenden Daten und prüft auf Gleichheit mit dem übermittelten Wert. Werden abweichende Werte festgestellt, so wird das Protokoll abgebrochen.
4. Das Gerät B sendet seinerseits seinen Wert E_b an A.
5. Das Gerät A prüft in analoger Weise.

Abbildung 15.37 fasst die Schritte bei der Authentifikation zusammen.

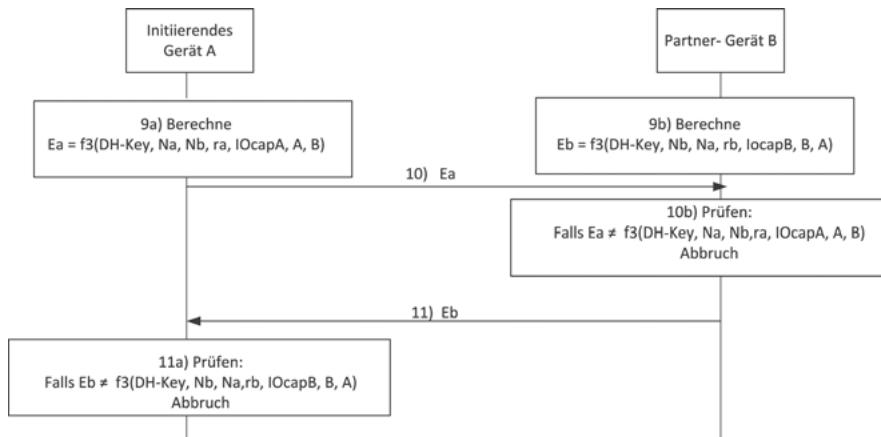


Abbildung 15.37: Simple-Pairing mit Passkey Entry

Nachdem beide Geräte bestätigt haben, dass das Pairing erfolgreich war, wird Phase 4 durchgeführt. Beide Parteien berechnen auf der Basis der

Phase 4

ihnen vorliegenden Werte, wie den Bluetooth-Adressen, der Zufallszahlen N_a und N_b , sowie dem gemeinsamen DH-Key, den gemeinsamen, neuen Verbindungsschlüssel (Link Key).

In der abschließenden Phase wird der Kommunikationsschlüssel erzeugt; diese Phase entspricht dem ursprünglichen Protokollschrift im Standard.

erhöhte Sicherheit

Mit der Verwendung des Secure Simple Pairings kann der Ablauf eines Pairings stärker abgesichert werden. Die Gefahren, die durch das Abhören der Pairingdaten im ursprünglichen Standard bestanden, werden verringert, und die Gefahr von Man-in-the-Middle Angriffe können zumindest mit einigen Ausprägungen des Protokolls eingedämmt werden.

Fazit

Bei allen genannten Schwächen und Problembereichen darf man aber nicht vergessen, dass das Bluetooth-Protokoll ursprünglich nicht dafür gedacht war, sensitive Daten über größere Entfernung zu übertragen. Vielmehr sollte Bluetooth auf einfachste Weise den Aufbau von kabellosen Personal Area Networks unterstützen. In solchen PANs werden in der Regel Daten eines Benutzers zwischen seinen eigenen Geräten innerhalb einer Reichweite von ca. 5 bis 10 Metern transferiert. Die Sicherheitsanforderungen in solchen Szenarien sind deshalb meist relativ niedrig. Mit den Weiterentwicklungen zu Bluetooth 4.2 und Bluetooth 5, sowie den Low Energy Varianten wurde das Einsatzspektrum von Bluetooth stark erweitert. die Sicherheitsarchitektur wurde jedoch kaum angepasst, lediglich Standardverschlüsselungsverfahren wie der AES und ECDH wurden in den Spezifikationen aufgenommen.

Pluspunkte

Pluspunkte des Bluetooth-Designs sind die Verwendung unterschiedlicher Schlüssel für unterschiedliche Aufgaben, eine Verankerung eines einfachen, nicht manuell durchzuführenden Schlüsselwechsels, nämlich des Kombinationsschlüssel sowie die Etablierung eines neuen Kommunikationsschlüssels für eine neue Verbindung. Weiterhin ist eine wechselseitige Authentifikation möglich und Known-Plaintext Angriffe werden durch das Design erschwert. Der Protokoll-Stack erlaubt ferner die Integration von Sicherheitsprotokollen auf höherer Ebene, um ein höheres Sicherheitsniveau zu erzielen.

Problembereiche

Diesen Pluspunkte stehen die geschilderten Problembereiche gegenüber. Dazu zählen das PIN-basierte Schutzkonzept, das natürlich ungeeignet ist für den Aufbau von ad-hoc Netzen zwischen a priori unbekannten Partnern, die keine Möglichkeit haben, eine PIN vorab zu verabreden. Standardmäßig sind die Sicherheitsdienste der Modi 2 und 3 deaktiviert. Aber auch bei ihrer Aktivierung wird lediglich eine Punkt-zu-Punkt- und keine Ende-zu-Ende Sicherheit angeboten, diese muss durch zusätzliche Protokolle wie SSL/TLS gewährleistet werden. Schließlich ist auch die Klassifikation nach Trusted

und not Trusted nur sehr grob und erlaubt kaum Differenzierung im Sinne einer benutzerbestimmbaren Zugriffskontrolle.

Mit dem verstärkten Einsatz von Bluetooth und der in Zukunft anstehenden bzw. durch spezifische Verstärker bereits erzielbaren Erweiterung der Sendeleistung auf 100 oder mehr Meter wird sich aber auch das Einsatzspektrum ändern. Da bei Bluetooth die Sicherheitsdienste ja nicht standardmäßig aktiviert sind, wird Angreifern, zumal wenn die Reichweiten der Geräte so deutlich steigen, Tür und Tor geöffnet, um Daten abzuhören, einzuschleusen etc. Werden sicherheitskritische Anwendungen ausgeführt und sensible Daten zwischen Geräten transferiert, so sollten die Basismechanismen von Bluetooth um Sicherheitsdienste auf höheren Schichten, wie IPSec oder SSL/TLS ergänzt werden.

15.6 ZigBee

15.6.1 Überblick

Bluetooth und WLAN waren für viele Anwendungsbereiche im IoT (Internet of Things) Kontext zu komplex, jedoch ändert sich das mit der Einführung von Bluetooth 5. Ältere Bluetoothversionen und WLAN benötigen zu viel Energie und sind zu teuer. ZigBee ist ein Industriestandard für eine Funkübertragung im Nahbereich von 10 bis 100 Metern, der seit 2002 von der ZigBee-Allianz, einer Herstellervereinigung von über 200 Unternehmen, entwickelt wird mit dem Ziel, eine preiswerte, energieeffiziente Technologie für drahtlose persönliche Netze (WPAN) bereitzustellen. Dafür setzt ZigBee auf einem bestehenden Standard, dem IEEE Standard 802.15.4 auf. ZigBee bietet eine lange Batterielebenszeit (über Jahre), unterstützt aber nur eine geringe Bandbreite und besitzt im Vergleich zu Bluetooth und WLAN eine deutlich geringere Reichweite. Die erste ZigBee-Spezifikation wurde 2004 veröffentlicht, 2006 vollständig überarbeitet und 2007 in einer überarbeiteten Version veröffentlicht²⁵. 2016 wurde mit ZigBee 3.0 eine weitere Überarbeitung und Aktualisierung der ZigBee-Spezifikation veröffentlicht.

Aufgrund seiner Eigenschaften wird ZigBee häufig in Anwendungsbereichen eingesetzt, in denen wartungsfreie Sensorik verwendet wird und in denen nur geringe Datenvolumina kommuniziert werden müssen, wie beispielsweise einfache Messwerte aus Temperatursensoren. Wichtige Einsatzfelder sind deshalb die Industrieautomation, die Heiamtomaticsierung, die Gebäudetechnik oder aber auch das medizinische Umfeld. Beispiele für ZigBee-basierte, vernetzte Geräte im Heiumfeld sind Türöffner, Thermostate, oder auch Lichtsteuerungen, wie Dimmer. Für häufig

Anwendungsbereiche

²⁵ vgl. : <http://www.zigbee.org/download/standards-zigbee-specification/>

Profil

genutzte Anwendungsfälle definiert ZigBee so genannte Profile. Ein Profil spezifiziert die Nachrichtenformate und Protokollschrifte für die Kommunikation zwischen ZigBee-Geräten. Zur vereinfachten und vereinheitlichten Unterstützung unterschiedlicher ZigBee-Profile wurde ZigBee 3.0 entwickelt, das seit 2016 veröffentlicht ist, und auf das wir in Abschnitt 15.6.5 noch etwas genauer eingehen.

Protokollstack

Der ZigBee-Protokollstack umfasst vier Schichten (vgl. Abbildung 15.38). Der IEEE 802.15.4-2003 Standard definiert die unteren beiden Schichten mit den Protokollen PHY für die physikalische Datenübertragung und dem Sub-Protokoll MAC (Medium Access Control) der Datenübertragungsschicht. IEEE 802.15.4-2003 spezifiziert zwei PHY Protokolle, die in zwei unterschiedlichen Frequenzbändern arbeiten. Das Protokoll im niedrigen Frequenzbereich arbeitet im europäischen 868-MHz und im US amerikanischen und australischen 915-MHZ Bereich. Der 2.4-GHz Frequenzbereich wird weltweit genutzt.

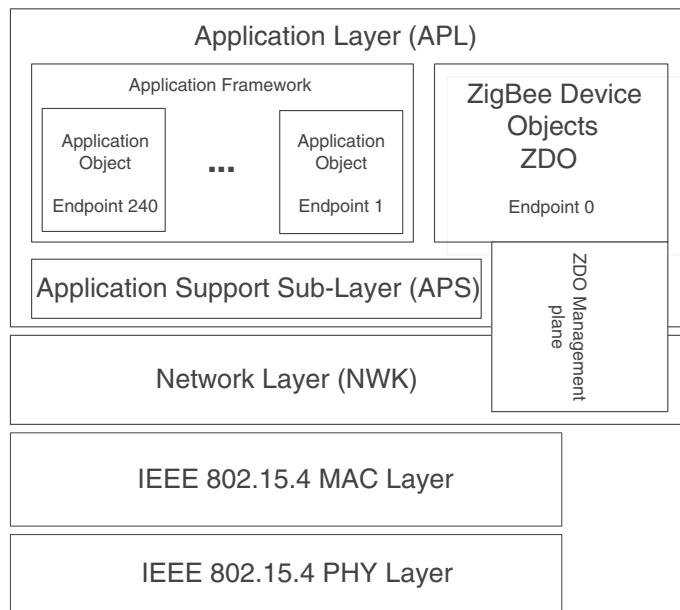


Abbildung 15.38: ZigBee-Protokollstack

Der IEEE 802.15.4-2003 MAC-Layer kontrolliert den Zugang zum Funkmedium mittels eines CSMA-CA-Verfahrens. Zusätzlich werden Mechanismen spezifiziert, um eine zuverlässige Übertragung zu ermöglichen, oder auch zur Synchronisation der Partner.

ZigBee-Protokolle

Aufbauend auf diesen beiden Protokollen hat die ZigBee-Allianz die Protokolle NWK für die Netzwerkschicht und APL für die Anwendungsschicht spezifiziert. APL besteht wiederum aus zwei Protokollen, dem APS (Ap-

plication Support Sublayer) und dem ZDO (ZigBee Device Object). Das APS definiert Dienste, die sowohl von herstellerdefinierten Anwendungsobjekten als auch von den ZigBee Device Objekten (ZDO) genutzt werden. Der Datentransfer zwischen Applikationsobjekten in einem ZigBe-Netz erfolgt mittels APS-Daten Entitäten (APSDE). Dazu unterstützen die Entitäten drei Dienste: (1) Request, Anfragen von Daten, (2) Confirmation, Bestätigung des Erhalts einer Anfrage, (3) Indication, Weitergabe der Daten einer Anfrage an ein Applikationsobjekt. Ein zu übertragendes Datenframe spezifiziert ein Kontrollfeld, in dem unter anderem eingetragen ist, ob das Frame sicher zu übertragen ist. Ist dies erforderlich, so werden im Rahmen der Datenübertragung die mit dem Kontrollfeld spezifizierten Sicherheitsdienste angewandt. Das APS hat zudem die Aufgabe, eine authentifizierte Verbindung zwischen zwei Geräten aufzubauen, falls diese Sicherheitsfunktion gewünscht wird, was durch entsprechende Flags in den Datenframes angezeigt wird. Der Application Layer definiert die Umgebung, in der die Applikationsobjekte, die von Herstellern definiert werden, ausgeführt werden. Applikationsobjekte können über das ZDO-Protokoll auf ZigBee-Geräte und auf das Netz zugreifen und erhalten auch den Zugriff auf die Sicherheitsdienste von ZigBee.

Jedes ZigBee-Netz besitzt eine 64-Bit PAN-ID, die vom Koordinator des Netzes festgelegt wird. Daneben besitzt jedes ZigBee Gerät (ZigBee-Funkmodul) eine eindeutige 64-Bit-IEEE-Adresse und erhält beim Zugang zu einem ZigBee-Netz zusätzlich eine 16-Bit-Kurzadresse zugewiesen, unter der das Gerät im Netz angesprochen werden kann. Zusätzlich ermöglicht es ZigBee, dass die herstellerspezifizierten Anwendungsobjekte über Endpunkte, die vergleichbar mit TCP-Ports sind, adressiert werden. Auf jedem ZigBee-Gerät stehen 255 Endpunkte zur Verfügung, wobei der Endpunkt 255 für den Broadcast an alle Endpunkte reserviert ist. Die an den Endpunkt 0 gesandten Daten werden vom ZDO (ZigBee Device Object) bearbeitet, das die Aufgabe hat, das Netz zu steuern.

Geräte-
identifikation

ZigBee unterscheidet drei Rollen, die Geräte im Netz einnehmen können:

Rollen

1. Jedes Netz besitzt ein *Koordinator*-Gerät, das über ausreichend Speicherkapazität und Kommunikationsfähigkeit verfügt und Routing sowie auch Kontrollfunktionen wahrt. Der Koordinator übernimmt zudem die Aufgaben eines *Trust Centers*. Diesem Gerät vertrauen alle anderen Geräte im Netzwerk. Es ist für die Verteilung von Schlüsseln verantwortlich. Darauf gehen wir weiter unten noch ein. In jedem ZigBee-Netz gibt es genau ein solches Trust Center²⁶.

Trust Center

²⁶ Unter ZigBee Pro/3.0 gilt dies nur für die Architekturvariante der zentralen Sicherheitsarchitektur.

2. Router-Komponenten sind zuständig für die Weiterleitung von Datenpaketen im Netzwerk.
3. Endgeräte, die nur mit einem Eltern-Knoten, das kann ein Router oder der Koordinator sein, kommunizieren und keine Managementaufgabe für andere Geräte im Netz wahrnehmen.

Topologie

ZigBee-Netze können in einer Stern-, Baum- oder Meshnetz-Topologie konfiguriert werden. Die Topologien werden vom ZiBee-Netzwerk Layer NWK unterstützt.

In einer Stern topologie wird das Netz vom ZigBee-Koordinator kontrolliert, alle anderen Geräte kommunizieren mit diesem Koordinator. In Mesh- und Baumtopologien ist der Koordinator dafür verantwortlich, netzwerkweite Parameter, wie beispielsweise den netzwerkweit gültigen Netzwerkschlüssel, festzulegen. ZigBee-Router haben die Aufgabe, die Daten mittels Routing-Protokollen hierarchisch gesteuert, wie bei der Baumtopologie, oder Peer-to-Peer, wie bei der Meshtopologie, weiterzuleiten. ZigBee Router in Meshnetzen unterstützen laut Spezifikation nur eine intra-PAN-Kommunikation, also eine Kommunikation, die im selben Netz startet und endet.

15.6.2 Sicherheitsarchitektur

IEEE 802.15.4

Im IEEE 802.15.4 Standard werden Basis-Sicherheitsfunktionen auf der Data-Link-Layer, also der Schicht 2, spezifiziert. Der Zugriff auf das Netz kann durch Access-Control-Listen (ACL) reglementiert werden, die auf der Basis der Adressen der Geräte einen Netzzugang erlauben oder verbieten. Zur Integritätssicherung von Datenpaketen legt der Standard die Verwendung von AES-CBC-MAC mit Längen von 32, 64 und 128-Bit fest, wobei sowohl Header als auch Payload geschützt werden. Zur Nachrichtenvertraulichkeit wird eine symmetrische Verschlüsselung mit AES-CTR und mit einem 128-Bit Kommunikationsschlüssel festgelegt. Für den Schutz vor Replays werden Sequenznummern und Acknowledgements eingesetzt.

Die Sicherheitsfunktionen werden in drei sogenannten Sicherheits-Suiten gebündelt:

1. No security (default)
2. Encryption only: AES-CTR
3. Encryption und Authenticity: AES CCM: AES-CCM-32, AES-CCM-64, AES-CCM-128

Die ZigBee-Protokolle der höheren Schichten nutzen diese Basis-Sicherheitsdienste. ZigBee definiert Sicherheitsdienste, die eine Schlüsselvereinbarung, eine Schlüsselerneuerung, eine sichere und unsichere Schlüsselübertragung und den Schutz von Datenpaketen spezifizieren.

Analog zu Bluetooth kennt auch ZigBee unterschiedliche Schlüsseltypen. Der Netzwerkschlüssel (Network Key) ist allen Geräten im Netz bekannt und wird zur Verschlüsselung der Nachrichten auf der Netzwerkschicht NWK im ZigBee-Netz genutzt, wenn für das ZigBee-Netz eine sichere Kommunikation gefordert wird. Der Trust Center des Netzes, der in der Regel das erste Gerät ist, das ein neues ZigBee-Netz initiiert und den Netzwerknamen (PAN-ID) dafür festlegt, bestimmt, ob eine Netzwerkverschlüsselung durchzuführen ist. Da der Netzwerkschlüssel allen autorisierten Teilnehmern des Netzes bekannt gemacht wird, wird mit diesem Sicherheitskonzept kein hoher Sicherheitsstandard festgelegt. Ein Angreifer, der in den Besitz des Schlüssels gelangt, hat Zugriff auf alle verschlüsselten Daten und kann seinerseits verschlüsselte Daten einspeisen.

Schlüsseltypen

Erfordert eine Applikation ein höheres Sicherheitsniveau, so kann ein bilateraler, symmetrischer Schlüssel, der Link-Key, zwischen zwei ZigBee-Geräten etabliert, und Anwendungsdaten können zusätzlich damit verschlüsselt zwischen den beiden Geräten übertragen werden. Solche Link-Schlüssel sind nur der Anwendungsschicht bekannt. Im Folgenden werden die Schlüsseltypen etwas genauer erläutert.

15.6.3 Schlüsseltypen

1. Ein ZigBee-Gerät kann einen *Master-Schlüssel* besitzen, der entweder bereits vorinstalliert ist oder aus einer Nutzereingabe, wie einer PIN oder einem Passwort, abgeleitet wird, oder aber mittels des Key-Transport-Kommandos vom Trust Center an das Gerät übertragen wird. Dieser Schlüsseltransport kann gesichert unter Nutzung eines Key Transport-Schlüssels oder auch ungesichert erfolgen. Ein ungesicherter Transfer stellt natürlich eine Verwundbarkeit dar, wenn es Angreifern gelingt, diesen Transfer abzuhören. Ein Master Key wird benötigt, wenn ein symmetrischer, bilateraler Link Key auf zwei Geräten mittels des SKKE-Protokolls (siehe unten) etabliert werden soll.

Master Key

In der Version ZigBee 3.0 wird anstelle des Master Keys ein so genannter Install Code als eine gemeinsame, sichere Basis für die Etablierung von Link-Schlüsseln verwendet, darauf wird auf Seite 961 noch einmal eingegangen.

2. Die vertrauliche Unicast-Kommunikation zwischen zwei ZigBee-Geräten basiert auf einem symmetrischen 128-Bit *Link Key*, der zwischen den

Link Key

kommunizierenden Geräten etabliert wird. Es werden zwei Klassen von Link Keys unterschieden: global und unique.

Von besonderer Bedeutung sind die Link Keys des Trust Centers. Ein *Global Trust Center Link Key* wird von allen Geräten des Netzes genutzt, während ein Unique Trust Center Link Key nur für die sichere Kommunikation zwischen dem Trust Center und einem direkten Peer-Gerät genutzt wird. Ein Unique Link Key, der zwischen zwei Geräten etabliert ist, von denen keines der Trust Center ist, wird *Application Link Key* genannt.

Gemäß Spezifikation muss jedes ZigBee-Gerät einen in der Spezifikation festgeschriebenen *Default Global Trust Center Link Key* kennen. Damit ist es möglich, dass auch unbekannte ZigBee-Geräte einem ZigBee-Netz beitreten können (join), wenn zwischen dem beitrittswilligen Gerät und dem Trust Center noch keine bilateralen Link oder Master Keys vereinbart worden sind. Der TC nutzt dann den Default Schlüssel, um den Network Key des Netzes dem neuen Gerät bekannt zu geben. Diese Fallback-Lösung stellt eine unmittelbar ersichtliche Schwachstelle im Sicherheitskonzept dar. Gelingt es einem Angreifer, einen mit diesem Default Schlüssel verschlüsselten Transfer des Network Keys vom Trust Center zum neu hinzutretenden Gerät abzuhören, so gelangt er in den Besitz des aktiven Network Keys, da auch der Angreifer den Default Key des Trust Centers kennt und den verschlüsselten Network Key entschlüsseln kann. Mit diesem ist der Angreifer in der Lage, alle im Netzwerk übertragenen Daten zu beobachten und gegebenenfalls auch eigene Daten einzuspielen.

Ein Link Key kann auf drei unterschiedlichen Wegen etabliert werden.

- (a) *Pre-shared Key* durch eine Konfigurierung beispielsweise bei Fertigung des Geräts.
 - (b) Durch den gesicherten oder ungesicherten *Austausch* (key transport), wobei der gesicherte Austausch unter Nutzung eines Key Load Keys erfolgt, der aus einem Link Key abgeleitet wird, oder
 - (c) durch eine *Schlüsselvereinbarung* SKKE (Symmetric Key Key Establishment) basierend auf einem Master Key. Die Schlüsselvereinbarung erfolgt in drei Schritten: (1) Austausch von flüchtigen Daten, (2) Nutzen der Daten, um einen Link Key daraus abzuleiten, (3) Bestätigungen, dass der Link Key korrekt berechnet worden ist.
- Network Key
3. Der 128-Bit symmetrische *Network Key* wird für das Versenden von Broadcast-Nachrichten und für den verschlüsselten Transfer von Daten auf der Netzwerkebene eingesetzt. Das Trust Center speichert in der Regel mehrere Netzwerkschlüssel, jedoch ist zu jedem Zeitpunkt stets

nur ein Netzwerkschlüssel in einem Netzwerk aktiv. Netzwerkschlüssel werden von den NWK- und APL-Schichten der Geräte genutzt.

Der Netzwerkschlüssel ist allen Geräten im ZigBee-Netz bekannt. Eine Bekanntgabe erfolgt entweder durch eine Vorkonfigurierung (Pre-configured), dies kann beispielsweise bei Fertigung des Geräts durch den Hersteller erfolgen, oder der Schlüssel wird vom Trust Center zum Zeitpunkt des Netzzugangs zum Gerät transferiert (key transport). Eine solche Verteilung des Network Keys kann auf unsichere oder auf sichere Weise erfolgen, abhängig davon, welche gemeinsamen Schlüssel zwischen dem Trust Center und dem beitretenden Gerät bereits etabliert sind. Bei einem unsicheren Transfer wird der Network Key unverschlüsselt übertragen. Der sichere Key Transport erfolgt unter Nutzung eines Key Transport Keys, der aus einem bereits etablierten Link Keys abgeleitet wird.

Schlüsselverteilung

4. Von den Link Keys werden weitere Schlüssel, Kommunikations-, Schlüsseltransport- und Schlüssel-Lade-Schlüssel, abgeleitet. Dazu wird eine festgelegte Hashfunktion auf den Link Key konateniert mit einem festgelegten Eingabestring angewandt. Zur Generierung des Transportschlüssels wird der Eingabestring 0x00 verwendet. Der Kommunikationsschlüssel entspricht dem Link Key.
5. Der Ladeschlüssel (key-load-key) wird zum sicheren Transport von Master oder Link Keys verwendet, während der Transportschlüssel (key transport key) zum verschlüsselten Transport von Network Keys genutzt wird.

ZigBee unterscheidet zwischen zwei unterschiedlichen Sicherheitsmodi: Standard²⁷ und hochsicher²⁸. Beim hochsicheren Modus verwaltet der Trust Center eine Liste von Geräten und Master, Link sowie Network Keys, die der Trust Center benötigt, um seine Regeln für einen kontrollierten und abgesicherten Netzzugang, falls dies die Regeln fordern, oder auch zur Schlüsselerneuerung durchzusetzen.

Modi

Das ZigBee-Protokoll basiert auf dem Open-Trust-Modell, das besagt, dass sich alle Protokolle auf den verschiedenen Schichten des Protokoll-Stacks vertrauen, so dass eine Verschlüsselung nur zwischen Geräten stattfindet, und Schlüssel auf den verschiedenen Schichten des Stacks wiederverwendet werden.

Trust-Management

²⁷ In älteren Spezifikationen wird dies auch als residential mode bezeichnet.

²⁸ Der Begriff wurde erst 2010 mit der Spezifikation von ZigBee Pro eingeführt, in älteren Spezifikationen wird ein Netz, für das hohe Sicherheitsanforderungen gelten, als commercial bezeichnet.

Der ZigBee-Netzwerk-Layer stellt die vertrauliche und integritätsgesicherte Übertragung von Datenframes mittels 128-Bit Netzwerkschlüsseln und AES-CCM* sicher. AES-CCM* ist eine geringfügig geänderte Variante des AES-CCM, bei der Verschlüsselung und Authentisierung kombiniert sind. Link- und Masterschlüssel sind nur auf der Anwendungsschicht nutzbar. Die APS-Schicht ist zudem für die Schlüsselvereinbarung sowie den Schlüsseltransport verantwortlich. Wie bereits kurz ausgeführt, stellt das Koordinatorgerät als Trust Center die Basis für die Schlüsselverteilung und Etablierung von Vertrauensstrukturen dar. Ausgangspunkt für den Aufbau von Trust-Beziehungen sind initiale Master Keys oder der aktive Netzwerkschlüssel, die vom Trust-Center an die jeweiligen Geräte mittels Key Transport übertragen werden. In Anwendungsbereichen mit hohen Sicherheitsanforderungen (high security) empfiehlt der Standard, dass die Adresse des Trust Centers und ein initialer Master Key vorinstalliert werden. Andernfalls muss in Kauf genommen werden, dass trotz der geltenden hohen Sicherheitsanforderungen der Master-Schlüssel ungeschützt in-band übertragen wird. Im Standard Security Mode (standard mode) wird in der Regel ein Netzwerkschlüssel ungeschützt übertragen, da man davon ausgeht, dass die kurze Zeitspanne der Unsicherheit, in der ein Angreifer diesen Nachrichten-transport abfangen kann, ein tolerierbares Sicherheitsrisiko darstellt. Weitere Schlüssel, wie Link-Schlüssel oder weitere Master-Schlüssel akzeptiert ein Gerät nur, wenn sie vom Trust Center über einen gesicherten Transportkanal übertragen werden. In Anwendungen mit geringen Sicherheitsanforderungen (residential mode) kommuniziert ein Gerät mit seinem Koordinator mittels des 128-Bit Netzwerkschlüssels; dieser Schlüssel ist entweder im Gerät vorkonfiguriert oder wird an das Gerät übertragen.

15.6.4 Netzzutritt und Schlüsselmanagement

Wie bereits ausgeführt, ist es die Aufgabe des Trust Centers, die Regeln für den Zugang zum ZigBee-Netz festzulegen und den Zutritt von Geräten zu kontrollieren sowie die angeforderten Schlüssel auszutauschen.

Soll das ZigBee-Netz eine verschlüsselte Netzwerkkommunikation umsetzen (das bestimmt der Trust Center), so benötigt das Gerät, das dem Netzwerk beitreten möchte, den aktiven Netzwerkschlüssel NK. Entweder ist dieser vorinstalliert oder muss ausgetauscht werden. Folgende Fälle sind zu unterscheiden:

Unsicherer Transfer Es ist kein dedizierter Link Key oder Master Key zwischen dem Trust Center und dem beitretenden Gerät A etabliert. Es erfolgt ein ungesicherter Transfer des Network Key NK unter Nutzung des Kommandos *key transport*:

Trust Center → Gerät A: *key transport NK*.

Default Global Trust Center Link Key Es ist kein dedizierter Link Key oder Master Key zwischen dem Trust Center und dem beitretenden Gerät A etabliert, es soll aber ein gesicherter Transfer des Network Key NK erfolgen. Der Default Global Trust Center Link Key (DGTCLK) wird genutzt:

Trust Center → Gerät A: key transport AES-CCM*(NK, DGTCLK).

Bem.: Dieses Vorgehen bietet nur einen schwachen Schutz vor Ausspähen des Network Keys.

Etablierter Unique Link Key Es ist bereits ein dedizierter Link Key LK_A zwischen dem Trust Center und dem beitretenden Gerät A etabliert (entweder vorkonfiguriert oder in einem früheren Join etabliert):

1. Sicherer Transfer des Network Keys NK unter Nutzung des Key Transfer Keys KFK_A , der aus dem Link-Schlüssel LK_A abgeleitet ist:

Trust Center → Gerät A: key transport AES-CCM*(NK, KFK_A).

Link Key Etablierung Es ist noch kein dedizierter Link Key zwischen dem Trust Center und dem beitretenden Gerät A etabliert.

1. Es ist ein gemeinsamer Link-Key LK_A zu etablieren:

Bekannter Master Key Es gibt einen vorkonfigurierten Master Key MS_A , der auch dem Trust Center bekannt ist.

- (a) Mit dem MS_A wird mittels SKKE ein bilateraler Link Key LK_A zwischen dem Trust Center und dem Gerät A vereinbart.
- (b) Ableiten des Key Transfer Keys KFK_A aus LK_A .

Initialer Master Key des TC Das Gerät besitzt keinen dedizierten Master Key.

- (a) Ungesicherter Transfer des initialen Master Keys IMS des Trust Centers zwischen dem Trust Center und dem Gerät A: IMS :

Trust Center → Gerät A: key transport IMS

Bem.: Hierbei besteht die Gefahr, dass dieser ungesicherte Transfer abgehört wird.

- (b) SKKE-basierte Vereinbarung eines Link Keys LK_A zwischen dem Trust Center und Gerät A auf der Basis des IMS .
- (c) Ableiten des Key Transfer Keys KFK_A aus LK_A .

2. Sicherer Transfer des Network Keys NK unter Nutzung des Key Transfer Keys KFK_A :

Trust Center → Gerät A: key transport AES-CCM*(NK, KFK_A).

Nach einem erfolgreichen Join muss im hochsicheren Modus noch eine Authentisierung mittels Challenge-Response durchgeführt werden, damit das

Gerät akzeptiert wird. Bei einem Re-Join kann ein Gerät den bereits etablierten Netzwerkschlüssel nutzen. Der Trust Center kann den Netzwerkschlüssel erneuern. Dazu verteilt er den neuen Schlüssel an die Netzteilnehmer mittels eines verschlüsselten Key Transports unter Nutzung des alten Netzwerkschlüssels.

Application Key

Soll zwischen zwei Geräten A und B ein gemeinsamer Application Link Key vereinbart werden, so sendet eines der beiden Geräte einen entsprechenden Request an den Trust Center. Dieser sendet an beide Geräte entweder einen Link oder einen Master Key. Wird ein Master Key gesandt, wickeln die Geräte A und B miteinander das SKKE-Protokoll ab, um einen gemeinsamen Application Link zu etablieren.

15.6.5 ZigBee 3.0

Im Jahr 2016 hat die ZigBee-Allianz den erweiterten Standard ZigBee 3.0 veröffentlicht. Ziel des überarbeiteten Standards war es, Profile zu homogenisieren und die Interoperabilität zwischen Geräten unterschiedlicher Hersteller deutlich zu vereinfachen. ZigBee 3.0 unterstützt eine Anwendungsbibliothek, die unter anderem Anwendungen enthält für Licht-, Temperatur-, Druck- oder Feuchtigkeitssensoren. ZigBee 3.0 kann auch als Ersatz für Infrarot-basierte Fernbedienungen beispielsweise im Entertainmentbereich (smart Home) eingesetzt werden. ZigBee 3.0 umfasst drei unabhängige Standards: ZigBee Pro, ZigBee RF4CE und ZigBee IP.

ZigBee 3.0

ZigBee Pro ermöglicht den Aufbau von größeren drahtlosen Sensornetzen (mesh networks), die bis zu 64.000 Geräte (genannt Knoten) enthalten können. ZigBee-Pro unterstützt verschiedene Profile, wie ZigBee Light Link (ZLL), ZigBee Home Automation (ZHA), ZigBee Building Automation (ZBA), ZigBee Telecom Service (ZTS), ZigBee Retail Services (ZRS), ZigBee Health Care (ZHC) und ZigBee Smart Energy (ZSE). Zusammen mit der Option ZigBee Green Power ermöglicht ZigBee Pro zudem den Einsatz von ZigBee bei batterielosen Sensoren, wie beispielsweise Lichtschaltern. Diese Sensoren verwenden Konzepte des Energy Harvesting, wie beispielsweise die Energiegewinnung aus einem Tastendruck, die von der Option ZSE unterstützt werden.

ZigBee Pro

Der Standard ZigBee RF4CE (radio frequency for consumer electronics) spezifiziert den Aufbau und Betrieb von einfachen und kostengünstigen drahtlosen Netzen zur Fernsteuerung von Geräten, insbesondere im Bereich des Entertainments oder der Heimautomatisierung. Eine Datenweiterleitung mittels Routing-Protokollen wird in derartigen Netzen nicht unterstützt. Das zugrundeliegende Konzept sieht vor, dass jedes fernzusteurende Gerät ein eigens PAN definiert und die Fernsteuerungskomponente diesem Netz beitritt (join).

ZigBee RF4CE

Mit dem in 2012 veröffentlichten ZigBee IP-Standard wird die Anbindung von ZigBee-Netzen an das Internet spezifiziert. Der Standard setzt dafür auf vorhandene Standards, wie dem von der IETF entwickelten Protokoll 6LoWPAN auf, das die direkte Anbindung von Geräten oder Sensornetzwerken an das Internet über IPv6 ermöglicht.

ZigBee IP

Wesentliche Weiterentwicklungen in Bezug auf die Sicherheit umfassen die Geräte-individuelle Authentisierung beim Netzzugang (join), Schlüssel-Updates während der Laufzeit, Over-the-Air Firmware-Update, virtuell-private Verbindungsverschlüsselung.

Um unterschiedliche Anwendungsbereiche zu unterstützen, definiert ZigBee3.0 zwei Architekturen mit zugehörigem Sicherheitsmodell, die zentrale und die dezentrale Sicherheits-Architektur.

Das Gerät, das ein neues ZigBee Netz aufbaut, legt den Sicherheitslevel des Netzes fest, indem ein entsprechendes Attribut gesetzt wird: *nwkSecurityLevel = 0*, dann ist das Netz unsicher, anderenfalls gesichert. Bei einem Netz mit einer zentralen Sicherheits-Architektur wird die Adresse des Trust Centers als Attribut in das Datenfeld *apsTrustCenterAddress* eingetragen. Ein Eintrag *0xFFFFFFFFFFFFFF* impliziert, dass ein Netz mit einer dezentralen Sicherheits-Architektur etabliert wird.

Initialisierung

Dezentrale Sicherheits-Architektur

In einem dezentral gesicherten Netzwerk (Distributed Security Network) gibt es keinen dedizierten Trust Center. Jeder Router kann als Trust Center agieren und Netzwerkschlüssel verteilen, jedoch dürfen sie keine Link-Keys verteilen. Falls ein Gerät bereits einen Netzwerk-Key besitzt, beispielsweise durch eine Vorkonfiguration, oder durch eine Out-of-Band Installation, beispielsweise mittels eines Install-Codes (s.u.) oder aus einem vorherigen Join, so wird das Gerät ohne weitere Authentisierung im Netz akzeptiert. Ein Update der Netzwerkschlüssel findet nicht statt. Bei einem Install-Code-basierten Vergehen wird ein Gerät vom Hersteller mit einem Schlüssel vorkonfiguriert ausgeliefert und es ist dann erforderlich, dass auch der Router diesen Schlüssel kennt.

dezentral

Wenn ein Gerät dem Netzwerk beitreten darf und ein vorkonfigurierter Netzwerkschlüssel NK im Gerät existiert oder wenn auf der Basis eines Install-Codes der Netzwerkschlüssel NK im Gerät gespeichert ist, der auch dem Router bekannt ist (z.B. durch manuelles Out-of-Band konfigurieren beim Router), dann kann das Gerät unmittelbar verschlüsselte Daten unter Nutzung des NK im Netz versenden. Andernfalls wird mittels eines Secure Key Transport der Netzwerkschlüssel NK dem neuen Gerät mitgeteilt. Hierfür ist der *Distributed Security Global Link-Key* zu nutzen, der allgemein bekannt und in den Produkten vorinstalliert ist. Der vergröberte Ablauf bei einem Join in eine dezentrale Sicherheitsarchitektur, bei dem ein Netzwerk-

Join

schlüssel verteilt werden muss, ohne dass vorkonfigurierte Informationen als Vertrauensbasis zur Verfügung steht, ist in Abbildung 15.39 zusammengefasst. In dem dargestellten Szenario möchte Gerät A dem Netz beitreten und benötigt den Netzwerkschlüssel NK. Es wendet sich dazu an den nächstgelegenen Router mit einem *associate_request*. Der Router antwortet mit einem *associate_response* Paket und signalisiert damit, dass das Gerät A an das Netz gebunden ist (join), aber noch keine Autorisierung besitzt. Dazu benötigt das Gerät A den Netzwerkschlüssel, den der Router mittels eines APS-Pakets und mit dem Key-Transport-Kommando verschlüsselt unter Nutzung des allgemein bekannten, vorinstallierten *Distributed Security Global Trust Center Link Keys* zur Verfügung stellt.

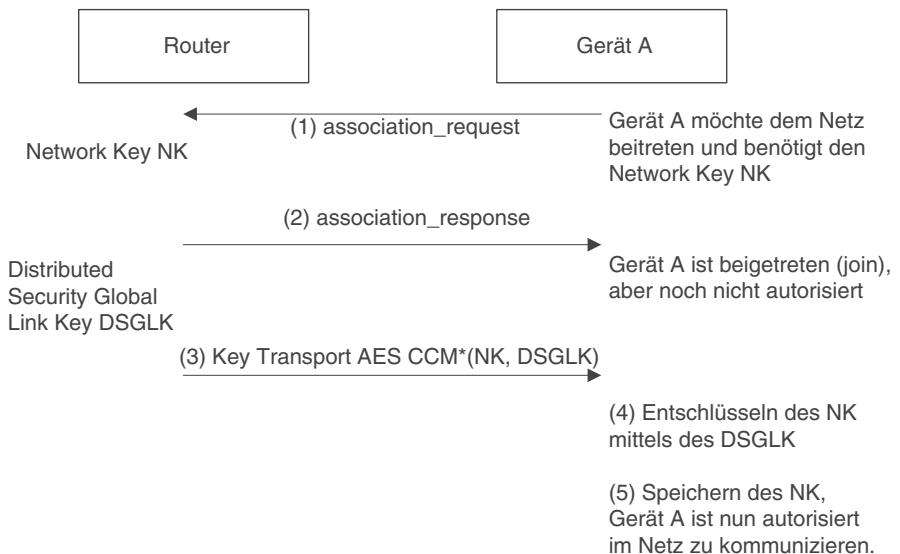


Abbildung 15.39: Join einer dezentralen Sicherheitsarchitektur

Der Key Transfer mittels des vorinstallierten, bekannten Link Keys ist offensichtlich keine starke Sicherheitsmaßnahme. Weitere, anwendungspezifische Schlüssel müssen durch Protokolle auf höheren Schichten ausgetauscht werden.

Zentrale Sicherheitsarchitektur

Zentrale Architekturen besitzen die Rolle des Trust Centers, wie oben beschrieben. Der Trust Center muss einen Netzwerkschlüssel und ein Regelwerk festlegen, das die Zutrittsberechtigungen zum Netzwerk beschreibt. Der Trust Center gewährt nur solchen Geräten Zutritt zum Netz, die die erforderlichen Credentials gemäß der festgelegten Policy vorweisen können. Ein Trust Center kann zudem eine Liste von Link- und Netzwerkschlüsseln verwalten für Geräte, die zum Netzwerk gehören.

Der Trust Center legt für den Netzzugang fest, auf welche Weise der aktive Netzwerkschlüssel NK an das neue Gerät übermittelt wird. Es werden folgende Link-Schlüssel als Basis für den Netzzugang unterschieden:

- Der *Default Global Trust Center Link Key*, der in der Spezifikation festgelegt und damit allgemein bekannt ist,
- ein *Global Trust Center Link Key*,
- ein *Unique Trust Center Link Key LK_A*, der zwischen dem beitretenden Gerät und dem Trust Center bilateral etabliert ist. Ein solcher bilateraler Schlüssel kann initial auf drei Wegen auf den beteiligten Geräten zur Verfügung gestellt werden:
 - als vorkonfigurierter Link Key oder
 - als Link Key, der aus einem Install Code abgeleitet wird, oder
 - als vorkonfigurierter *Touchlink Link Key*, falls das Gerät die Touchlink Anbindung unterstützt.

Das Konzept des Install Codes ist eine Erweiterung des ursprünglichen Master Key-Konzepts. Bei der Produktion eines Geräts wird ein solcher Code als 128-Bit Zufallszahl vom Hersteller generiert, in dem Gerät vorkonfiguriert und dem Trust Center zusammen mit der IEEE-Adresse des Geräts auf manuellem Weg bekannt gemacht (Out-of-Band). Der Install Code wird auf eine herstellerspezifische Weise mit dem Gerät ausgeliefert. Beispielsweise kann der Install Code als QR-Code mit dem Gerät mitgeliefert werden, und der Nutzer scannt ihn via Smartphone ein und übermittelt ihn an den Trust Center.

Ein Install Code ist eine 128-Bit Zufallszahl, die mit einem 16-Bit CRC geschützt wird. Ausgehend von dieser 128-Bit Zufallszahl berechnen sowohl das Gerät als auch der Trust Center den gemeinsamen Link Key LK_A unter Anwendung der Matyas-Meyer-Oseas-Hashfunktion. Der Trust Center nutzt den aus dem Install Code abgeleiteten Link Key LK_A, um den Netzwerkschlüssel damit abgesichert zum Gerät zu transportieren.

Abbildung 15.40 fasst vergröbert die Abläufe bei einem Join in ein Netzwerk mit einer zentralen Sicherheitsarchitektur zusammen. Falls das beitretende Gerät A nicht direkt mit dem Trust Center kommuniziert, muss der kontaktierte Router die Anfragen an das Trust Center weiterleiten und erhält von diesem den verschlüsselten Netzwerkschlüssel. Dies erfolgt mittels eines APS-Layer-Pakets unter Nutzung des Key-Transport-Kommandos, wobei der Trust Center den Netzwerkschlüssel verschlüsselt überträgt. Diese Verschlüsselung kann stark sein, wenn ein dedizierter Link Key LK_A, der nur dem Gerät A und dem Trust Center bekannt ist, für den Netzzugang genutzt wird, oder aber auch schwach, wenn der Default Global Trust Center Link

Trust Center

Install Code

Install Code

Key genutzt wird. Der Router agiert als Gateway, das getunnelt wird und reicht das verschlüsselte Key Transport-Paket an das Gerät A zur weiteren Bearbeitung weiter. Nachdem das Gerät auf diese Weise autorisiert wurde, erfolgt ein Link Key-Update, durch den zwischen dem Trust Center und dem Gerät A ein neuer bilateraler Link Key LK_{Aneu} etabliert wird. Dazu transferiert der Trust Center den neuen Schlüssel LK_{Aneu} unter Nutzung des etablierten Schlüssels LK_A mittels eines sicheren Key-Transports zum Gerät A. In einem nachfolgenden Handshake überzeugen sich der Trust Center und das Gerät A davon, dass sie jeweils diesen neuen Schlüssel besitzen, indem sie in *verify-* und *conform-Nachrichten* den neuen Schlüssel einsetzen.

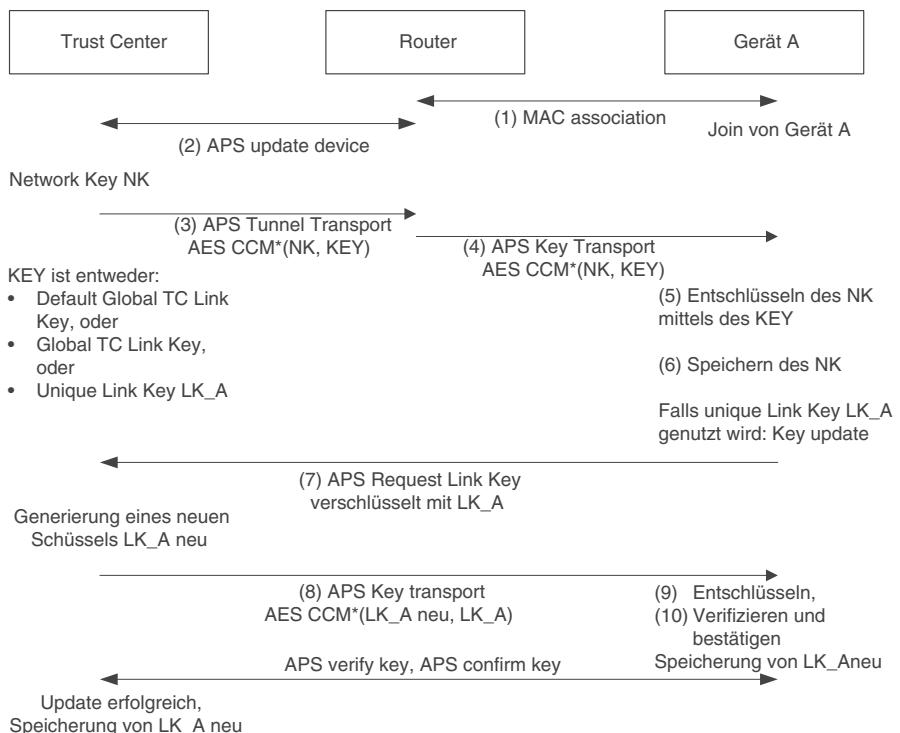


Abbildung 15.40: Join einer zentralen Sicherheitsarchitektur

Schlüssel- erneuerung

In zentralen Sicherheitsarchitekturen generiert der Trust Center in periodischen Abständen neue Netzwerkschlüssel und verteilt diese verschlüsselt an die angeschlossenen Geräte, wozu der gemeinsam mit dem Trust Center berechnete Link Key genutzt wird. Angreifer, die in den Besitz des Netzwerkschlüssels gelangen, sollen auf diese Weise nur für eine kurze Zeit widerrechtlich Zugriff auf verschlüsselte Daten erhalten können. Application Link Keys werden nicht periodisch erneuert.

In der zentralen Sicherheitsarchitektur darf nur der Trust Center Schlüssel verteilen. Gerätepaare erhalten auf Anfrage einen paarweisen Link Key. Mit diesen paarweisen Application Link Keys werden virtuell private Verbindungen (virtual private links) zwischen Gerätepaaren etabliert²⁹. Ein mögliches Einsatzszenario aus der Heimautomatisierung ist die Vernetzung einfacher Sensoren, wie Thermostate oder Leuchten, unter Nutzung des gemeinsamen Netzwerkschlüssels, und die Nutzung von Link-Schlüsseln als zusätzliche Credentials zur Absicherung sensibler Sensoren, wie Türöffnern.

Over-the-Air-Update

Mit dem Konzept des Over-the-Air-Update (OTA) ist es Geräteherstellern möglich, neue Features, aber auch Firmwareupdates auf die ausgerollten Geräte einzuspielen. Zur Absicherung dieses Prozesses werden die transferierten Update-Daten mit einem unique Link Key verschlüsselt und mit einem anderen unique Link Key symmetrisch signiert. Die Update-Daten (image) können im on-chip-Speicher abgelegt werden, für den das Feature des Auslesens deaktiviert wird, so dass ein Reverse Engineering unterbunden wird. Das verschlüsselte Update wird vom Bootloader entschlüsselt, die Signatur wird validiert und das Update wird ausgeführt. Der Bootloader prüft auch bevor er bootet, ob das aktive Image unmodifiziert ist.

Zur Abwehr von Replay-Attacken wird eine Art Sequenznummernkonzept verwendet. Jedes ZigBee Kommando verwendet einen Frame Counter, der vom Empfänger geprüft wird; Duplikate werden ignoriert.

Frame-Counter

Trust Management

Ein Gerät akzeptiert einen Link- oder Netzwerkschlüssel nur, wenn dieser von seinem Trust Center stammt und via Key Transport (dies kann auch ein unsicherer Transport sein) zugestellt wurde. Ein initialer Netzwerkschlüssel wird in einer zentralen Sicherheitsarchitektur nur dann von einem Gerät akzeptiert, wenn dieser mit dem Trust Center Link Key verschlüsselt wurde. Weitere Link- oder Netzwerkschlüssel akzeptiert ein Gerät nur, wenn sie ursprünglich von seinem Trust Center stammen und via einem sicheren Key Transport übertragen wurden, oder wenn sie durch Protokolle auf einer höheren Schicht etabliert wurden. Abbildung 15.40 zeigt ein Szenario, in dem ein Router als Gateway zwischen dem Trust Center und dem Gerät A agiert und die verschlüsselten Key Transport-Nachrichten des Trust Centers im Tunnelmodus an das Empfängergerät A weiterleiten.

15.6.6 Sicherheitsbetrachtungen

Die Sicherheit der ZigBee-Sicherheitsarchitektur basiert darauf, dass die genutzten symmetrischen Schlüssel geheim gehalten werden, dass also die

²⁹ Dies entspricht dem Application Link Key in den Vorgänger-Spezifikationen von ZigBee.

Schlüssel sicher auf dem Gerät gespeichert werden und die Initialisierung und auch der Schlüsselaustausch bzw. Schlüsseltransfer vertraulich ablaufen. Der Standard sieht jedoch keine Anforderung in Bezug auf die Integration von zum Beispiel Hardware-Sicherheitskomponenten wie HSMs zur vertrauenswürdigen Speicherung von Schlüsseln vor, da ZigBee auf preiswerte Geräte ausgerichtet ist. Angreifer, die physischen Zugriff auf ein ZigBee-Gerät haben, können damit Zugriff auf Verschlüsselungsmaterial oder auch auf die Sicherheitsfunktionen erlangen. Ebenso werden keine Anforderungen hinsichtlich der Isolierung von Anwendungen auf dem Gerät gestellt, so dass alle Anwendungen als vertrauenswürdig untereinander eingestuft werden.

Hard-coded Schlüssel

In der Praxis nutzen viele ZigBee-Geräte Schlüssel, die fest einprogrammiert sind und bei Inbetriebnahme des Geräts in den RAM geladen werden. Ein Angreifer kann versuchen, beim Startvorgang über eine spezielle serielle Schnittstelle den Transfer des Schlüssels vom Flash-Speicher in den RAM abzufangen. Für solche Angriffe stehen frei verfügbare Tools, wie der Open-source JTAG Adapter GoodFET (<http://goodfet.sourceforge.net/>) oder Bus Pirate (http://dangerousprototypes.com/docs/Bus_Pirate) zur Verfügung.

OTA

Werden Schlüssel mittels des Over-the-Air-Protokolls an ZigBee-Geräte verteilt, so kann ein Angreifer die übertragenen Daten sniffen und versuchen, sie zu analysieren. Frei verfügbare Tools, wie KillerBee (<https://tools.kali.org/wireless-attacks/killerbee>), ermöglichen das Abfangen und die Analyse von 802.15.4-Paketen. Darauf hinaus sind ZigBee-Geräte auch anfällig gegen Replay-Angriffe.

Fallback-Problem

Wie bereits ausgeführt, erlaubt der ZigBee-Standard auch den ungesicherten Transfer von Netzwerkschlüsseln oder initialen Master Keys. Dies stellt natürlich eine beträchtliche Sicherheitslücke dar und stellt das gesamte Konzept in Frage, falls der entsprechende Schlüsseltransport abgehört wird. Jedoch enthält auch der gesicherte Key Transport Schwachstellen, die auf die Fallback-Protokollschriften, die der Standard vorschreibt, zurückzuführen sind. Diese Fallback-Varianten sehen vor, dass der *Default Global Trust Center Link Key* zu nutzen ist, wenn kein weiteres Schlüsselmaterial ausgetauscht wurde. Verschiedene ZigBee-Anwendungs-Profile nutzen diesen Fallback, wie beispielsweise die Profile HAPAP und ZLL.

HAPA

Verwundbarkeit der ZigBee-Profile HAPAP und ZLL

Das HAPA bzw. HA (Home Automation Public Application Profile) ist ein weit verbreitetes Profil, das für den Einsatz in Geräten der Heimautomatisierung genutzt wird. Damit Geräte unterschiedlicher Hersteller interoperabel sind, müssen sie das so genannte SAS, das Startup Attribute Set, implementieren. Dieses Set enthält den festgelegte *Default Global Trust Center*

Link Key. Wie weiter oben bereits erläutert, fordert die ZigBee-Spezifikation, dass der aktuelle Netzwerkschlüssel unter Verwendung dieses *Default Global Trust Center Link-Keys* zu übertragen ist, wenn ein neues, unbekanntes Gerät dem Netzwerk beitreten möchte. Diese Fallback-Lösung eröffnet einem Angreifer die Möglichkeit, diese Join-Nachrichten abzufangen und in den Besitz des transportierten Netzwerkschlüssel zu gelangen, da der *Default Global Trust Center Link Key* ja allgemein, also auch dem Angreifer, bekannt ist. Die einzigen Hürden für den Angreifer sind somit die räumliche Nähe zum Gerät, das das Join ausführt, um die transferierten Daten auszuspähen, und die Kenntnis des Zeitpunkts, zu dem ein Gerät dem Netz beitreten möchte. Da aber ZigBee auch Nachrichtenpakete vorsieht, durch die dem Trust Center der Rejoin-Wunsch eines Geräts signalisiert werden kann, ist es einem Angreifer möglich, gezielt den Austausch eines solchen unsicheren Schlüsseltransfers zu initiieren. Die Rejoin-Anfrage erfordert lediglich die Kenntnis der Adresse des Geräts und die eines Routers im ZigBee-Netz, dies ist aber keine geheime Information. Alternativ kann der Angreifer mittels Jamming-Nachrichten den Funkverkehr zwischen Geräten so lange stören, bis die Geräte sich selbst neu anmelden. Um den Kommunikationswünschen des rechtmäßig agierenden Geräts zuvorzukommen, kann ein Angreifer die niedrigen Frequenzbänder zur Datenübertragung nutzen, da diese von den Routern zuerst bedient werden und ZigBee-Geräte in der Regel auf den höheren Frequenzbändern agieren.

Die Reichweite von ZigBee-Modulen ist meist gering, so dass sich ein Angreifer in der räumlichen Nähe der ZigBee-Geräte aufhalten muss. Durch die Nutzung spezieller Antennen kann ein Angreifer aber auch diese Hürde beträchtlich verringern, so dass insgesamt durch das Fallback-Konzept eine ernste Schwachstelle besteht.

Obwohl diese Fallback-Option von Anfang an im Design von ZigBee vorgesehen war und auch entsprechend deutlich dort dargestellt ist, wurde die Bedeutung der Schwachstelle in der breiteren Öffentlichkeit erst nach der Präsentation von Angriffen auf ZigBee-Geräte durch T. Zillner auf der Black Hat-Konferenz 2015 in Las Vegas bekannt³⁰. Da beispielsweise auch programmierbare Türschlösser dieses Profil häufig unterstützen, ist damit eine ernst zu nehmende Sicherheitslücke verbunden. Hinzu kommt, dass Heimautomatisierungsgeräte in der Regel keine Möglichkeit vorsehen, den Netzwerkschlüssel zu erneuern, so dass ein Nutzer auch beim Wahrnehmen von ungewöhnlichem Verhalten den Angreifer nicht durch ein Reset wieder aus dem Netz entfernen kann.

³⁰ T. Zillner: ZigBee Exploited: The good, the bad and the ugly.

<https://www.blackhat.com/docs/us-15/materials/us-15-Zillner-ZigBee-Exploited-The-Good-The-Bad-And-The-Ugly-wp.pdf>

ZLL

Das Profil ZigBee Light Link (ZLL) ermöglicht es, dass Leuchtmittel, wie Dimmer und Lichtschalter, einfach kommunizieren können. Das Profil sieht einen vorinstallierten Master Key vor, den nur die zertifizierten Hersteller der Leuchtprodukte kennen und von diesen in die ZigBee-Geräte eingebracht werden. Bei einem Netzwerkzugang eines solchen Geräts transferiert ein Router oder der Trust Center den aktiven Netzwerkschlüssel mittels eines gesicherten Key Transports unter Nutzung dieses Master Keys zum beitretenden Gerät. Da ZigBee-Geräte nicht gegen physische Angriffe gesichert sind, kann ein Angreifer natürlich zunächst versuchen, den Master Key aus einem eigenen ZLL-unterstützenden Gerät herauszulesen, um damit an den Netzwerkschlüssel zu gelangen. Durch die Spezifikation werden aber noch weitere Angriffe ermöglicht. So legt die Spezifikation fest, dass die Geräte auch die oben beschriebene Fallback-Lösung mit dem *Default Global Trust Center Link Key* unterstützen müssen. Dieses Fallback wird aktiviert, wenn bei einem beitrittswilligen Gerät beim Entschlüsseln des Netzwerkschlüssels ein Fehler auftritt; das Gerät führt dann den unsicheren Join durch. Ein Angreifer kann nun gezielt durch Jamming eine solche Fehlersituation für ein Gerät bei dessen Join-Versuch herbeiführen und dafür sorgen, dass das Gerät die Fallback-Option nutzt. ZLL unterstützt zudem das Feature Touchlink Commissioning, womit ein Gerät mit einem Controller gepaartet werden kann. Hierbei wird dann auch der Default Global Trust Center Key genutzt, wodurch das bereits beschriebene Problem auftritt.

Ein Angreifer kann sich Zugriff auf den Netzwerkschlüssel verschaffen und die Geräte des Netzwerks vollständig kontrollieren. Werden größere öffentliche oder nicht öffentliche Bereiche mit ZigBee-vernetzten Leuchtmitteln beleuchtet und kann ein Angreifer ein Gerät attackieren, so ist er in der Lage, das gesamte Netz und damit die Beleuchtung beispielsweise in einem Einkaufspark oder in öffentlichen Bereichen zu kontrollieren.

ZigBee-Wurm

Ein mögliches Angriffsszenario wird in dem Papier von A. Shamir (vgl. <http://iotworm.eyalro.net/>) dargestellt. Das Papier beschreibt einen durchgeführten Angriff, bei dem es gelang, einen ZigBee-Wurm in ein Netz zu platzieren, der sich selbstständig ausbreitet hat. Dazu gelang es den Sicherheitsforschern mittels einer speziellen Seitenkanalanalyse die symmetrischen Schlüssel zu extrahieren, mit denen Over-the-Air-Updates vom Hersteller der analysierten Produktfamilie signiert und verschlüsselt werden. Mit diesem Schlüssel wurde ein modifiziertes Firmware-Update verschlüsselt und auf Geräte eingespielt, indem es gelang, die Geräte zu einem Reset zu veranlassen und die modifizierte Firmware zu nutzen. Dazu nutzten sie eine Festlegung des ZLL-Profil aus, die es ermöglicht, dass auch nicht-ZLL-Geräte an ein ZLL-Netz angebunden werden können. Hierfür ist die erwähnte SAS-Konfiguration erforderlich, und für den Austausch des

Netzwerkschlüssels wird der allgemein bekannte Schlüssel verwendet. Die Forscher haben diese Option ausgenutzt und ein präpariertes Gerät als nicht-ZLL-Gerät in das zu attackierende Netz eingebunden. Da es den Forschern zudem gelang, einen Fehler im genutzten ATMEL-Chip zu finden, konnten sie dessen integrierten Proximity-Test umgehen, der normalerweise prüft, dass ein Gerät, das ein Reset veranlasst, in unmittelbarer Nähe des zurückzusetzenden Geräts sein muss. Durch das Umgehen dieses Tests war es möglich, ein Reset auch aus großer Distanz von mehreren hundert Metern zu veranlassen und auf diese Weise Geräte im Netz zu infiltrieren. Auch wenn der exemplarisch durchgeführte Angriff spezielle Gegenbenheiten eines Herstellers ausgenutzt hat, zeigt er auf, dass ZigBee-Netze verwundbar sind, und dass die Ursachen dafür in der ZigBee-Spezifikation bzw. in der ZLL-Profil-Spezifikation zu finden sind. Vor dem Hintergrund der wachsenden IoT-Vernetzung besteht hier also deutlicher Handlungsbedarf für eine Verbesserung des Standards.

Problematisch an Angriffen, die sich über das ZigBee-Netz verbreiten, ist zudem, dass hierbei die ZigBee-Protokolle und nicht die klassischen Internet-Protokolle TCP/IP genutzt werden. Standardsoftware zur Netzüberwachung ist auf die Analyse von TCP/IP-Netzverkehr fokussiert, so dass die ZigBee-basierten Angriffe von den existierenden Monitoring-Tools nicht erkannt werden können.

Fazit

Der ZigBee-Standard nutzt den etablierten und kryptografisch starken AES CCM* mit 128 Bit Schlüsseln zur Verschlüsselung und Integritätssicherung. Damit orientiert sich der Standard an dem State-of-the-Art bei der symmetrischen Kryptografie. Die Sicherheit eines ZigBee-Netzes hängt deshalb von der sicheren Speicherung der genutzten Schlüssel auf den Geräten, von der sicheren Initialisierung und dem sicheren Transport von Schlüsseln ab. Der Standard unterstützt Maßnahmen wie die Out-of-Band-Schlüsselübertragung, das Vorinstallieren von Schlüsseln oder das Nutzen von Install Codes sowie Protokolle zur regelmäßigen Schlüsselerneuerung, so dass zusammen mit dem Trust-Center-Konzept bei zentralen Sicherheits-Architekturen ZigBee-Netze mit einem angemessenen Sicherheitslevel aufgebaut und betrieben werden können.

Die Möglichkeiten, ein hohes Sicherheitsniveau in ZigBee-Netzen zu implementieren, stößt jedoch häufig an Grenzen, da ZigBee darauf ausgelegt ist, eine lange Batterielebenszeit zu ermöglichen (zertifizierte Geräte müssen mindestens zwei Jahre Laufzeit ermöglichen). Zudem sind die vernetzten Geräte häufig sehr ressourcenschwach. So besitzen sie nur wenig Speicher zur Aufnahme von Schlüsselmaterial, verfügen nur über eine geringe Rechenfähigkeit und besitzen auch meist keine Eingabemöglich-

AES

Ressourcen-
Beschränkung

keit. Aufwändige Sicherheitsmanagementprotokolle zur Authentisierung von Netzteilnehmern, zur Prüfung von Zertifikaten gemäß einer PKI oder zur Durchführung asymmetrischer Verschlüsselungen sind deshalb nicht praktikabel. Zudem sollen mit ZigBee preiswerte Geräte vernetzt werden, so dass spezielle Tamper-resistente Hardware in der Regel bei den einfachen Endgeräten nicht zum Einsatz kommt. Jedoch könnten ressourcenstarke Trust Center manipulationsgeschützt sein und dafür sorgen, dass im Fall einer erkannten Kompromittierung alle Schlüssel im Netz gelöscht und neue Schlüssel Out-of-Band verteilt werden, falls dies das Einsatzumfeld zulässt.

Default-Schlüssel

Wie die dargestellten Sicherheitsprobleme verdeutlicht haben, sollte in ZigBee-Implementierungen darauf verzichtet werden, den Default Global Trust Center Link Key für den verschlüsselten Transport des Netzwerkschlüssels beim Join eines Geräts zu verwenden. Darüber hinaus sollte der Netzwerkschlüssel in regelmäßigen Abständen erneuert werden.

IoT-Sicherheit

Insgesamt zeigt der Standard und dessen Einsatz sehr gut die Schwierigkeiten, die sich bei der Vernetzung von einfachen, preiswerten und ressourcenschwachen Geräten im Internet of Things ergeben. Hier sind noch weitere Forschungsarbeiten erforderlich, um auch unter den genannten Randbedingungen einen hohen Sicherheitslevel garantieren zu können. Vor dem Hintergrund des rasanten Vordringens der IoT-Geräte in zunehmend sicherheitskritische Bereiche, wie die Gesundheits- und Energieversorgung, oder auch die Industrie-Automatisierung kommt der Verbesserung der Sicherheit von IoT-Geräten eine gesellschaftlich sehr wichtige Rolle zu.

Literaturverzeichnis

- [1] 3GPP. Design of the UMTS Cipher and Integrity Functions.
- [2] 3GPP. UMTS Security Architecture.
- [3] Martín Abadi and Roger M. Needham. Prudent Engineering Practice for Cryptographic Protocols. *IEEE Transactions on Software Engineering*, 22(1):6–15, jan 1996.
- [4] R. Anderson. Stretching the limits of steganography. *Lecture Notes in Computer Science*, 1174:39–48, 1996.
- [5] Ross Anderson. *Security Engineering*. John Wiley, 2001.
- [6] Ross Anderson and Markus Kuhn. Tamper Resistance – a Cautionary Note. In *Second USENIX Workshop on Electronic Commerce*, pages 1–11, Oakland, California, 1996.
- [7] Michael Arnold, Martin Schmucker, and Stephen D. Wolthusen. *Techniques and Applications of Digital Watermarking and Content Protection*. Artech-House, 2003.
- [8] Will Arthur, David Challener, and Kenneth Goldman. *A Practical Guide to TPM 2.0: Using the New Trusted Platform Module in the New Age of Security*. Apress, 2015.
- [9] M.J. Bach. *The Design of the UNIX Operating System*. Prentice–Hall International, 1986.
- [10] D. Balenson. RFC 1423: Privacy enhancement for Internet electronic mail: Part III: Algorithms, modes, and identifiers, February 1993.
- [11] D.W. Bauder. An Anti-Counterfeiting Concept for Currency Systems, 1983.
- [12] F. L. Bauer. *Entzifferte Geheimnisse*. Springer, Berlin, 1997.
- [13] D. Bell and L. LaPadula. Secure computer systems: A mathematical model. Technical Report MTR-2547, Vol 2, MITRE Corp., Bedford, MA, November 1973.
- [14] D. E. Bell and L. J. LaPadula. Secure computer systems: Mathematical foundations and model. Technical Report M74-244, The MITRE Corp., Bedford MA, May 1973.

- [15] D.E. Bell and L. LaPadula. Secure Computer Systems: Unified Exposition and MULTICS Interpretation. Technical Report MTR - 2997, MITRE Corp, Bedford, July 1975.
- [16] M. Bellare, R. Canetti, and H. Krawczyk. The HMAC construction. *CryptoBytes*, 2(1):12–15, 1996.
- [17] S. M. Bellovin and M. Merritt. Encrypted key exchange: Password-based protocols secure against dictionary attacks. In *Proc. IEEE Symposium on Research in Security and Privacy*, pages 72–84, 1992.
- [18] S.M. Bellovin. Security Problems in the TCP/IP Protocol Suite. *Computer Communication Review*, 19(2):32 – 48, April 1989.
- [19] Jens Bender, Marc Fischlin, and Dennis Kügler. Security analysis of the pace key-agreement protocol. In *ISC*, pages 33–48, 2009.
- [20] Jens Bender, Dennis Kuegler, Marian Margraf, and Ingo Naumann. Sicherheitsmechanismen für kontaktlose Chips im deutschen elektronischen Personalausweis. *DuD, Datenschutz und Datensicherheit*, 32(3):850–864, 2008.
- [21] D. Bernstein. The salsa20 family of stream ciphers. 2007.
- [22] D. Bernstein. Chacha, a variant of salsa20. 2008.
- [23] D. Bernstein. The poly1305-aes message-authentication code. 2008.
- [24] Bastian Best, Jan Jürjens, and Bashar Nuseibeh. Model-based security engineering of distributed information systems using UMLsec. In *ICSE*, pages 581–590. IEEE Computer Society, 2007.
- [25] K.J. Biba. Integrity Considerations for Secure Computer Systems. Technical Report MTR – 3153, NTS AD A039324, MITRE Corp, Bedford, April 1977.
- [26] Eli Biham and Adi Shamir. Differential Cryptanalysis of the full 16-round DES. In *Lecture Notes in Computer Science: Advances in Cryptology – CRYPTO ’93*, pages 487–497. Springer Verlag, 1992.
- [27] Eli Biham and Adi Shamir. *Differential Cryptanalysis of the Data Encryption Standard*. Springer-Verlag, 1993.
- [28] Manuel Blum and Silvio Micali. How to generate cryptographically strong sequences of pseudo-random bits. *SIAM Journal on Computing*, 13(4):850–864, 1984.
- [29] Andrey Bogdanov, Dmitry Khovratovich, and Christian Rechberger. Biclique cryptanalysis of the full aes. In *ASIACRYPT*, pages 344–371, 2011.

- [30] D. Boneh, R. A. Demillo, and R. J. Lipton. On the importance of checking cryptographic protocols for faults. *Lecture Notes in Computer Science*, 1233:37–51, 1997.
- [31] D.F.C. Brewer and M.J. Nash. The Chinese Wall Security Policy. In *Proceedings 1989 IEEE Symposium on Security and Privacy*, pages 206 – 214, 1989.
- [32] BSI. Technische Spezifikationen des deutschen elektronischen Personalausweises, Chipkartenspezifikationen. Report, BSI.
- [33] BSI. Technische Richtlinie TR03110, Advanced Security Mechanisms for Machine Readable Travel Documents, Extended Access Control (EAC), Password Authenticated Connection Establishment (PACE), and Restricted Identification (RI), Version 2.03. Report, BSI, 2007.
- [34] BSI. Messung der Abstrahleigenschaften von RFID-Systemen (MARS). Report, BSI, 2008 veröffentlicht.
- [35] BSI – Bundesamt für Sicherheit in der Informationstechnik. *IT-Grundschutzhandbuch*. <http://www.bsi.bund.de/gshb>, Köln, 2001.
- [36] J. Buchmann. *Einführung in die Kryptographie*. Springer, Berlin, 1999.
- [37] Bundesregierung. Gesetz über Rahmenbedingungen für elektronische Signaturen und zur Änderung weiterer Vorschriften. Technical report, Bundesregierung, 2001.
- [38] Carlos E. Caicedo, James B. D. Joshi, and Summit R. Tuladhar. IPv6 security challenges. *IEEE Computer*, 42(2):36–42, 2009.
- [39] CCITT. Recommendation X.509, The Directory Authentication Framework. Technical Report, Consultation Committee, Genf, 1989.
- [40] D. Brent Chapman and Elizabeth D. Zwicky. *Building Internet firewalls*. O'Reilly & Associates, Inc., second edition, 2000.
- [41] David Chaum. Blind signature system. In David Chaum, editor, *Advances in Cryptology: Proceedings of Crypto 83*, page 153. Plenum Press, New York and London, 1984, 22–24 August 1983.
- [42] D.L. Chaum. Untraceable Electronic Mail, Return Addresses and Digital Pseudonyms. *Communications of the ACM*, 24(2):84, 1981.
- [43] F. Cohen. Computer Viruses. *Computer & Security*, 6:22–35, 1987.
- [44] Katriel Cohn-Gordon, Cas Cremers, Benjamin Dowling, Luke Garrett, and Douglas Stebila. A Formal Security Analysis of the Signal

- Messaging Protocol. In *2nd IEEE European Symposium on Security and Privacy*. IEEE, 2017.
- [45] U. Dammann and S. Simitis. *Bundesdatenschutzgesetz (BDSG) mit Materialien*. Namos-Verlag, Baden-Baden, 1977.
 - [46] Charles H. Green David H. Maister and Robert M. Galford. *The Trusted Advisor*. Free Press, 2001.
 - [47] Christian Decker and Roger Wattenhofer. Information Propagation in the Bitcoin Network. In *13th IEEE International Conference on Peer-to-Peer Computing (P2P), Trento, Italy*, September 2013.
 - [48] J.B. Dennies and E.C. Van Horn. Programming Semantics for Multiprogrammed Computations. *Communications of the ACM*, 9(3):143–155, 1966.
 - [49] D. E. Denning and D. K. Branstad. A Taxonomy for Key Escrow Encryption Systems. *Communications of the ACM*, 39(3):34–40, March 1996.
 - [50] D.E. Denning. A Lattice Model of Secure Information Flow. *Communications of the ACM*, 19(5):236–241, 1976.
 - [51] D.E. Denning. *Cryptography and Data Security*. Addison-Wesley Publishing Company, 1982.
 - [52] D.E. Denning and G.M. Sacco. Timestamps in Key Distribution Protocols. *Communications of the ACM*, 24(8):533, 1981.
 - [53] Departement of Defense. Trusted Computer System Evaluation Criteria. Technical Report CSC-STD-001-83, Departement of Defense, 1985.
 - [54] W. Diffie and M. Hellman. Privacy and authentication: An introduction to cryptography. *Proceedings of the IEEE*, 67(3):397–427, March 1979.
 - [55] W. Diffie and M.E. Hellman. New Directions in Cryptography. *IEEE Transactions on Information Theory*, IT-22(6):644–654, 1976.
 - [56] Whitfield Diffie, Paul C. van Oorschot, and Michael J. Wiener. Authentication and authenticated key exchanges. *Des. Codes Cryptography*, 2(2):107–125, 1992.
 - [57] Whitfield Diffie, Paul C. Van Oorschot, and Michael J. Wiener. Authentication and authenticated key exchanges. *Designs, Codes and Cryptography*, 2(2):107–125, Jun 1992.
 - [58] Yevgeniy Dodis, Leonid Reyzin, and Adam Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. In

- Christian Cachin and Jan Camenisch, editors, *Advances in Cryptology - EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 523–540. Springer Berlin / Heidelberg, 2004.
- [59] Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In *LNCS*, volume 196, pages 10–18. Springer-Verlag, 1985, August 1984.
 - [60] U. Emmert. Strafbare Sicherheits-Tools? *Zeitschrift für Kommunikations- und EDV-Sicherheit*, 18(2):6–9, 2002.
 - [61] Dan Farmer and Wietse Venema. *Forensic Discovery*. Addison-Wesley, 2004.
 - [62] D. Ferraiolo and R. Kuhn. Role-based access controls. In *15th NIST-NCSC National Computer Security Conference*, pages 554–563, 1992.
 - [63] David F. Ferraiolo, D. Richard Kuhn, and Ramaswamy Chandramouli. *Role-Based Access Control*. Artech House Computing Library, 2003.
 - [64] A. Fiat and A. Shamir. How To Prove Yourself: Practical Solutions to Identification and Signature Problems. In *Proceedings of CRYPTO 86, LNCS 263*, pages 186 – 194, 1986.
 - [65] Electronic Frontier Foundation. *Cracking DES: Secrets of Encryption Research, Wiretap Politics, and Chip Design*. O'Reilly and Associates, 1998.
 - [66] A. Freier, P. Karlton, and P. Kocher. The SSL Protocol Version 3.0. Netscape Communication Corporation, 1996. <http://home.netscape.com/eng/ssl3/index.html>.
 - [67] K. Fuhrberger. *Internet-Sicherheit*. Carl Hanser Verlag, 2000.
 - [68] S. Garfinkel. *PGP: Pretty Good Privacy*. O'Reilly & Associates, 1994.
 - [69] Blaise Gassend, Dwaine Clarke, Marten van Dijk, and Srinivas Devadas. Silicon physical random functions. In *Proceedings of the 9th ACM Conference on Computer and Communications Security, CCS '02*, pages 148–160, New York, NY, USA, 2002. ACM.
 - [70] Martin Georgiev, Subodh Iyengar, Suman Jana, Rishita Anubhai, Dan Boneh, and Vitaly Shmatikov. The most dangerous code in the world: validating ssl certificates in non-browser software. In *ACM Conference on Computer and Communications Security*, pages 38–49, 2012.
 - [71] J. D. Golic. Cryptanalysis of alleged A5 stream cipher. *Lecture Notes in Computer Science*, 1233:239, 1997.

- [72] L.A. Gordon, M.P. Loeb, W. Lucyshin, and R. Richardson. CSI/FBI: Computer Crime and Security Survey. Technical report, CSI, Computer Security Institute, 2006.
- [73] G.S. Graham and P.J. Denning. Protection – Principles and Practice. In *Proceedings of the Spring Joint Computer Conference*, pages 417–429, 1972.
- [74] Bluetooth Special Interest Group. Specification of the bluetooth system. Technical report, www.bluetooth.com/developer/specification/specification.asp, 1999.
- [75] Daniel Gruss, Moritz Lipp, Michael Schwarz, Richard Fellner, ClÃ©mentine Maurice, and Stefan Mangard. *KASLR is Dead: Long Live KASLR*, volume 10379 LNCS of *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, pages 161–176. Springer-Verlag Italia, Italy, 2017.
- [76] N. Haller. The S/Key one-time password system. In *Symposium on Network and Distributed Systems Security*, San Diego, California, February 1994.
- [77] M.A. Harrison, W.L. Ruzzo, and J. Ullman. Protection in Operating Systems. *Communications of the ACM*, 19(8):461–471, 1976.
- [78] Changhua He, Mukund Sundararajan, Anupam Datta, Ante Derek, and John C. Mitchell. A modular correctness proof of ieee 802.11i and tls. In *Proceedings of the 12th ACM Conference on Computer and Communications Security*, CCS ’05, pages 2–15, New York, NY, USA, 2005. ACM.
- [79] F. H. Hinsley and A. Stripp, editors. *Code Breakers – the Inside Story of Bletchley Park*. Oxford U. P., 1993.
- [80] Scott Hogg and Eric Vyncke. *IPv6Security*. Cisco Press, 2008.
- [81] J. Hopcroft and J. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley, N. Reading, MA, 1980.
- [82] Ralf Hund, Carsten Willems, and Thorsten Holz. Practical timing side channel attacks against kernel space aslr. In *Proceedings of the 2013 IEEE Symposium on Security and Privacy*, SP ’13, pages 191–205, Washington, DC, USA, 2013. IEEE Computer Society.
- [83] C. I’Anson and C. Mitchell. Security Defects in CCITT Recommendation X.509 – the Directory Authentication Framework. *Computer Communications Review*, 20(2):30 –34, April 1990.

- [84] ICAO. Doc 9303, Machine Readable Travel Documents, Part 1 – Machine Readable Passport – Volume 1 Passports with Machine Readable Data Stored in Optical Character Recognition Format. Report, International Civil Aviation Organization, 2006.
- [85] ICAO. Doc 9303, Machine Readable Travel Documents, Part 1 – Machine Readable Passport – Volume 2 Specifications for Electronically Enabled Passports with Biometric Identification Capabilities. Report, International Civil Aviation Organization, 2006.
- [86] IEEE. LAN MAN Standards of the IEEE Computer Society, wireless LAN Medium Access Control (MAC) and physical Layer specification. Technical report, IEEE Standard 802.11, 1997.
- [87] IEEE. Standards for local and metropolitan area networks: Standard for port based network access control. Technical report, IEEE Draft P802.1X/D11, 2001.
- [88] American National Standard Institute. American National Standard for Data Encryption Algorithm (DEA). Technical Report, 1981.
- [89] American National Standard Institute. American National Standard for Personal Information Number (PIN) Management and Security. Technical Report, 1983.
- [90] T. Ishiguro, S. Kiyomoto, and Y. Miyake. Modified version of 'latin dances revisited: New analytic results of salsa20 and chacha. <https://eprint.iacr.org/2012/065.pdf>, 2012.
- [91] ISO. Information processing systems – OSI reference model – part 2: Security architecture, international standards organization. Technical Report 7498-2, ISO, 1989.
- [92] IT-Sicherheitskriterien. *Kriterien für die Bewertung der Sicherheit von Systemen der Informationstechnik, 1. Fassung*. Zentralstelle für Sicherheit in der Informationstechnik, Bundesanzeiger, Jan. 1989.
- [93] ITSEC. *Information Technology Security Evaluation Criteria – Harmonised Criteria of France, Germany, the Netherlands, the United Kingdom, Version 1*. May 1990.
- [94] Neil F. Johnson and Sushil Jajodia. Computing practices: Exploring steganography: Seeing the unseen. *Computer*, 31(2):26–34, February 1998.
- [95] Ari Juels, David Molnar, and David Wagner. Security and Privacy Issues in E-passports. Report 2005/095, Cryptology ePrint Archive, March 2005.

- [96] D. Kahn. *The Codebreakers*. Macmillan, New York, 1967.
- [97] D. Kahn. The history of steganography. In Ross Anderson, editor, *Information hiding: first international workshop, Cambridge, U.K., May 30–June 1*, volume 1174 of *Lecture Notes in Computer Science*, pages 1–5. Springer-Verlag, 1996.
- [98] B. Kaliski. RFC 2315: Cryptographic message syntax version 1.5, March 1998.
- [99] Stefan Katzenbeisser, Ünal Koçabas, Vladimir Rozic, Ahmad-Reza Sadeghi, Ingrid Verbauwhede, and Christian Wachsmann. Pufs: Myth, fact or busted? a security evaluation of physically unclonable functions (pufs) cast in silicon. In *CHES*, pages 283–301, 2012.
- [100] St. T. Kent. Internet Privacy Enhanced Mail. *Communications of the ACM*, 36(8):48 –60, August 1993.
- [101] D. Knuth. *The Art of Computer Programming Vol I – III*. Addison-Wesley Pub Co, 3rd edition edition, 1997.
- [102] Neal Koblitz. Elliptic Curve Cryptosystems. *Mathematics of Computation*, 48(177):203–209, January 1987.
- [103] P. Kocher, J. Jaffe, and B. Jun. Differential power analysis. In Proc. *19th International Advances in Cryptology Conference – CRYPTO ’99*, pages 388–397, 1999.
- [104] Paul Kocher, Daniel Genkin, Daniel Gruss, Werner Haas, Mike Hamburg, Moritz Lipp, Stefan Mangard, Thomas Prescher, Michael Schwarz, and Yuval Yarom. Spectre attacks: Exploiting speculative execution. *CoRR*, abs/1801.01203, 2018.
- [105] RSA Laboratories. Public-key cryptography standards #1: Rsa cryptography standard. Technical report, RSA Laboratories, 1998.
- [106] Leslie Lamport. Password authentication with insecure communication. *CACM*, 24(11):770–772, November 1981.
- [107] Leslie Lamport, Robert Shostak, and Marshall Pease. The byzantine generals problem. *ACM Trans. Program. Lang. Syst.*, 4(3):382–401, July 1982.
- [108] B.W. Lampson. A Note on the Confinement Problem. *Communications of the ACM*, 16(10):613–615, 1973.
- [109] Moritz Lipp, Michael Schwarz, Daniel Gruss, Thomas Prescher, Werner Haas, Stefan Mangard, Paul Kocher, Daniel Genkin, Yuval Yarom, and Mike Hamburg. Meltdown. *CoRR*, abs/1801.01207, 2018.
- [110] R.J. Lipton and L. Snyder. A Linear Time Algorithm for Deciding Subject Security. *Journal of the ACM*, 24(3):455–464, 1977.

- [111] G. Lorenz, T. Moore, G. Manes, J. Hale, and S. Shenoi. Securing ss7 telecommunications networks. In *Proceedings of the 2001 IEEE Workshop on Information Assurance and Security*.
- [112] Roel Maes and Ingrid Verbauwhede. Physically unclonable functions: A study on the state of the art and future research directions. In *Towards Hardware-Intrinsic Security*, pages 3–37. Springer, 2010.
- [113] Moxie Marlinspike and Trevor Perrin. The x3dh key agreement protocol. *Open Whisper Systems Specification, The X3DH Key Agreement Protocol*, <https://signal.org/docs/specifications/x3dh/>, 2016.
- [114] Mitsuru Matsui. Linear Cryptanalysis Method for DES Cipher. In *Lecture Notes in Computer Science: Advances in Cryptology – EUROCRYPT ’93*, pages 386–397. Springer Verlag, 1993.
- [115] Stuart McClure, Joel Scambray, and George Kurtz. *Hacking Exposed: Network Security Secrets and Solutions, Fourth Edition*. McGraw-Hill Professional, 2003.
- [116] D. McCullough. A Hookup Theorem for Multilevel Security. *IEEE Transactions on Software Engineering*, 16(6):563 – 568, June 1990.
- [117] G. McGraw. *Software Security: Building Security In*. Addison-Wesley, 2006.
- [118] Alfred J. Menezes, Paul C. van Oorschot, and Scott A. Vanstone. *Handbook of Applied Cryptography*. CRC Press series on discrete mathematics and its applications. CRC Press, 1996.
- [119] Ralph C. Merkle. Secure communications over insecure channels. *Communications of the Association for Computing Machinery*, 21(4):294–299, April 1978.
- [120] Ralph C. Merkle and Martin E. Hellman. On the security of multiple encryption. *Commun. ACM*, 24(7):465–467, 1981.
- [121] Dominik Merli, Georg Sigl, and Claudia Eckert. Identities for embedded systems enabled by physical unclonable functions. In *Number Theory and Cryptography*, pages 125–138, 2013.
- [122] Victor Miller. Use of Elliptic Curves in Cryptography. In *Advances in Cryptology: Proceedings of Crypto ’85*, volume 218 of *LNCS*, pages 417–426, Berlin, 1986. Springer-Verlag.
- [123] A. Mishra and W. A. Arbaugh. An Initial Security Analysis of the IEEE 802.1X Standard. Technical report, Department of Computer Science, University of Maryland, 2002.
- [124] R. Morris and K. Thompson. Password Security: A Case History. *Communications of the ACM*, 22(11), Nov. 1979.

- [125] G. Müller and A. Pfitzmann. *Mehrseitige Sicherheit in der Kommunikationstechnik*. Addison Wesley, 1 edition, 1997.
- [126] Nathan J. Muller. *Bluetooth*. McGraw Hill, 2001.
- [127] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system, <http://bitcoin.org/bitcoin.pdf>, 2008.
- [128] National Computer Security Center. Trusted Network Interpretation of the Trusted Computer System Evaluation Criteria. Technical Report NCSC-TG-005, National Computer Security Center, July 1987.
- [129] National Institute of Standards and Technology (U. S.). Escrowed Encryption Standard (EES). Federal information processing standards publication 185, National Institute for Standards and Technology, Gaithersburg, MD, USA, 1994.
- [130] R.M. Needham and M.D. Schroeder. Using Encryption for Authentication in Large Networks of Computers. *Communications of the ACM*, 21(12):993–999, 1978.
- [131] R.M. Needham and M.D. Schroeder. Authentication Revisited. *ACM Operating Systems Review*, 21(1):7 – 8, Jan 1987.
- [132] M. Negin, T. Chmielewski, and M. Salganicoff. An Iris Biometric System for Public and Personal Use. *IEEE Computer*, 33(2):70–75, February 2000.
- [133] NIST. The Digital Signature Standard Proposed by NIST. *Communications of the ACM*, 35(7):36 – 41, July 1992.
- [134] Magnus Olsson. *SAE and the Evolved Packet Core*. Academic Press, 2009.
- [135] Dag Arne Osvik, Adi Shamir, and Eran Tromer. Cache attacks and countermeasures: The case of aes. In *Proceedings of the 2006 The Cryptographers’ Track at the RSA Conference on Topics in Cryptology*, CT-RSA’06, pages 1–20, Berlin, Heidelberg, 2006. Springer-Verlag.
- [136] C. Paar and J. Pelzl. *Understanding Cryptography: A Textbook for Students and Practitioners*. Springer, 2010.
- [137] Sharath Pankanti, Ruud M. Bolle, and Anil Jain. Biometrics: The future of identification. *IEEE Computer*, 33(2):46–49, February 2000.
- [138] Ravikanth Pappu, Ben Recht, Jason Taylor, and Neil Gershenfeld. Physical one-way functions. *Science*, 297(5589):2026–2030, September 2002.
- [139] Jaehong Park and Ravi S. Sandhu. Towards usage control models: beyond traditional access control. In *SACMAT*, pages 57–64, 2002.

- [140] Alex Pentland and Tanzeem Choudhury. Face Recognition for Smart Environments. *IEEE Computer*, 33(2), February 2000.
- [141] Trevor Perrin and Moxie Marlinspike. The double ratchet algorithm. *GitHub*, <https://signal.org/docs/specifications/doubleratchet/>, 2016.
- [142] Birgit Pfitzmann. *Digital signature schemes: general framework and fail-stop signatures*, volume 1100. Springer-Verlag, Berlin, Heidelberg, 1996.
- [143] Derrell Piper. The internet IP security domain of interpretation for ISAKMP, rfc 2407. RFC, September 1998.
- [144] Carl Pomerance. The quadratic sieve factoring algorithm. In T. Beth, N. Cot, and I. Ingemarsson, editors, *LNCS*, volume 209, pages 169–182. Springer-Verlag, 1985, April 1984.
- [145] CC Project. The Common Criteria for Information Technology Security Evaluation (CC) version 2.0, 1999.
- [146] Jean-Jacques Quisquater and Louis Guillou. A practical zero-knowledge protocol fitted to security microprocessor minimizing both transmission and memory. In *Proceedings of eurocrypt'88*, pages 123–128. Springer Verlag, 1988.
- [147] Pappu Srinivasa Ravikanth. *Physical One-way Functions*. PhD thesis, Massachusetts Institute of Technology, 2001. AAI0803255.
- [148] R. Reischuk. *Einführung in die Komplexitätstheorie*. B.G. Teubner, Stuttgart, 1990.
- [149] R. Rieke. Tool based formal modelling, analysis and visualisation of enterprise network vulnerabilities utilising attack graph exploration. In *Eicar 2004*, 2004.
- [150] C. Rigney, S. Willens, A. Rubens, and W. Simpson. Remote Authentication Dial In User Service (RADIUS). Technical report, RFC 2865, IEFT, 2000.
- [151] R.L. Rivest. Response to NIST's proposal. *Communications of the ACM*, 35(7):41–47, July 1992.
- [152] R.L. Rivest, A. Shamir, and L. Adleman. A Method for obtaining Digital Signatures and Public Key Cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.
- [153] P. Röder, O. Tafreschi, and C. Eckert. History-Based Access Control for XML Documents. In *Proceedings of the ACM Symposium on Information, Computer and Communications Security (ASIACCS'07)*, March 20-22, 2007.

- [154] Phillip Rogaway. Authenticated-encryption with associated-data. In *Proceedings of the 9th ACM Conference on Computer and Communications Security*, CCS '02, pages 98–107, New York, NY, USA, 2002. ACM.
- [155] Ulrich Rührmair and Jan Söltner. Puf modeling attacks: An introduction and overview. In *DATE*, pages 1–6, 2014.
- [156] Travis Russel. *Signaling System # 7*. Mc Graw Hill, New York, USA, 2000.
- [157] J.H. Saltzer and M.D. Schroeder. Protection of Information in Computer Systems. In *Proceedings of the IEEE*, 63(9), pages 1278 – 1308, March 1975.
- [158] R. S. Sandhu. Role Hierarchies and Constraints for Lattice-Based Access Controls. In *Proceedings of the European Symposium on Research in Security and Privacy*, 1996.
- [159] B. Schneier. *Applied Cryptography*. John Wiley & Sons, 1996.
- [160] Bruce Schneier. Inside risks: the uses and abuses of biometrics. *Communications of the ACM*, 42(8):136–136, August 1999.
- [161] P. E. Sevinç, D. Basin, and E.-R. Olderog. Controlling access to documents: A formal access control model. In Günter Müller, editor, *ETRICS 2006*, volume 3995 of *LNCS*, pages 352–367. Springer-Verlag, June 2006.
- [162] H. Shacham, M. Page, B. Pfaff, E. Goh, N. Modadugu, and D. Boneh. On the effectiveness of address-space randomization. *Proceedings of 11th ACM Conference on Computer and Communications Security*, 2004.
- [163] C. E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27:379–423, 623–656, July, October 1948.
- [164] C. E. Shannon. Communication theory of secure systems. *Bell System Technical Journal*, 28, 1949.
- [165] J.G. Steiner, C. Neuman, and J.I. Schiller. Kerberos: An Authentication Service for Open Network Systems. In *USENIX Winter Conference*, pages 191 – 202, Feb. 1988.
- [166] Douglas R. Stinson. *Cryptography Theory and Practice*. CRC Press, Boca Raton, 1995.
- [167] F. Stumpf, O. Tafreschi, P. Röder, and C. Eckert. A robust integrity reporting protocol for remote attestation. In *Second Workshop on Advances in Trusted Computing (WATC '06 Fall)*, Tokyo, Japan, November 2006.

- [168] Frederic Stumpf, Michael Benz, Martin Hermanowski, and Claudia Eckert. An approach to a trustworthy system architecture using virtualization. In *Proceedings of the 4th International Conference on Autonomic and Trusted Computing (ATC-2007)*, volume 4158 of *Lecture Notes in Computer Science*, pages 191–202, Hong Kong, China, 2007. Springer-Verlag.
- [169] Dafydd Stuttard and Marcus Pinto. *The Web Application Hacker’s Handbook: Discovering and Exploiting Security Flaws*. Wiley & Sons, 1 edition, October 2007.
- [170] D. Sutherland. A Model of Information. In *Proceedings of the 9th National Computer Security Conference*, September 1986.
- [171] Nick Szabo. Formalizing and securing relationships on public networks. *First Monday*, 2(9), 1997.
- [172] A.S. Tanenbaum. *Computer Networks*. Prentice-Hall, Inc. Englewood Cliffs, New Jersey, 1996.
- [173] A.S. Tanenbaum. *Modern Operating System*. Prentice-Hall Inc, 2001.
- [174] Mathy Vanhoef and Frank Piessens. Key reinstallation attacks: Forcing nonce reuse in wpa2. In *Proceedings of the 24th ACM Conference on Computer and Communications Security (CCS)*. ACM, 2017.
- [175] J. Viega and G. McGraw. *Building Secure Software: How to avoid Security Problems the Right Way*. Addison-Wesley Professional, 2002.
- [176] J. Vollbrecht, P. Calhoun, S. Farell, L. Gommans, G. Gross, and B. de Bruijn. AAA Authorization Framework. Technical report, RFC 2904, IETF, 2000.
- [177] H.P. Riess W. Fumy. *Kryptographie: Entwurf, Einsatz und Analyse von symmetrischen Kryptosystemen*. R. Oldenbourg Verlag München, 1994.
- [178] W.Effing W. Rankl. *Handbuch der Chipkarten*. Carl Hanser Verlag, 5-te Auflage, 2008.
- [179] Michael J. Wiener. Efficient DES Key Search. In *Lecture Notes in Computer Science: Advances in Cryptology – CRYPTO ’92*. Springer Verlag, 1993.
- [180] Wayne Wolf. Cyber-physical systems. *Computer*, 42:88–89, 2009.
- [181] Zishuang Ye and Sean Smith. Trusted Path for Browsers: An Open-Source Solution for Web-Spoofing. Technical report, Department of Computer Science, Dartmouth College, 2002.

- [182] Zishuang Ye, Yougu Yuan, and Sean Smith. Web-Spoofing Revisited: SSL and Beyond. Technical report, Department of Computer Science, Dartmouth College, 2002.
- [183] P. Yosifovich, A. Ionescu, M. E. Russinovich, and D. A. Solomon. *Windows Internals, Part 1: System architecture, processes, threads, memory management, and more*. Microsoft Press, 7th Edition, Redmond, 2017.
- [184] Wenjun Zeng, Heather Yu, and Ching-Yung Lin. *Multimedia Security Technologies for Digitale Rights Management*. Academic Press, 2006.
- [185] Aviv Zohar. Bitcoin: Under the hood. *Commun. ACM*, 58(9):104–113, August 2015.

Abkürzungsverzeichnis

A

AAA	Authentication, Authorisation, Accounting
ABC	Always best Connected
AC	Authentication Center
ACL	Access Control List
ACL	Asynchronous Connection-Less
AES	Advanced Encryption Standard
AH	Authentication Header
AKA	Authentication and Key Agreement Protocol
API	Application Programming Interface
ARP	Address Resolution Protocol
ARPA	Advanced Research Projects Agency
ASCII	American Standard Code for Information Interchange
ASLR	Address Space Layout Randomization
ASN.1	Abstract Syntax Notation Number 1
ATM	Asynchronous Transfer Modus
AV	Authentication Vector

B

BAC	Basic Access Control
BAN	Burrows Abadi Needham
BAN	Body Area Network
BDSG	Bundesdatenschutzgesetz
Blob	Binary Large Object
BLP	Bell LaPadula Sicherheitsmodell
BIOS	Basic Input/Output System
BNetzA	Bundesnetzagentur

BSI	Bundesamt für Sicherheit in der Informationstechnik
B2B	Business to Business
B3G	Beyond 3G (dritte Generation Mobilfunk)

C

CA	Certification Authority
CC	Common Criteria
CCMP	Counter Mode CBC MAC Protocol
CBC	Cipher Block Chaining
CEM	Common Evaluation Methodology
CERT	Computer Emergency Response Team
CFB	Cipher Feedback
CGI	Common Gateway Interface
CHAP	Challenge Handshake Authentication Protocol
CHV	Card Holder Verification
COPS	Computer Oracle and Password System
CPS	Cyber Physical System
CRC	Cyclic Redundancy Code
CRL	Certificate Revocation List
CSMA/CD	Carrier Sense Multiple Access/Collision Detection
CSRF	Cross Site Request Forgery

D

DAC	Discretionary Access Control
DARPA	US Defense Advanced Research Projects Agency
DCE	Distributed Computing Environment
DDoS	Distributed Denial of Service Angriff
DEP	Data Execution Prevention
DES	Data Encryption Standard
DFN	Deutsches Forschungs-Netz
DLL	Dynamic Link Library
DH	Diffie Hellman
DHCP	Dynamic Host Configuration Protocol

DLL	Dynamic Link Library
DMZ	Demilitarisierte Zone
DNS	Domain Name Service
DNSSEC	Domain Name System Security
DOI	Domain of Interpretation
DPA	Differential Power Analysis
DoS	Denial of Service Angriff
DSS	Digital Signature Standard

E

EAC	Extended Access Control
EAL	Evaluation Assurance Level
EAP	Extensible Authentication Protocol
ECB	Electronic Code Book
ECC	Elliptic Curve Cryptography
ECDLP	Elliptic Curve Discrete Logarithm Problem
EEPROM	Electrically Erasable Read Only Memory
EFS	Encrypting File System
eGK	Elektronische Gesundheitskarte
EJB	Enterprise Java Beans
ePass	Elektronischer Reisepass
ePA	Elektronischer Personalausweis
ESP	Encapsulating Payload
ETSI	European Telecommunication Standards Institute
EPC	Evolved Packet Core

F

FAR	False Acceptance Rate
FIDO	Fast IDentity Online (Allianz)
FOMA	Freedom of Multimedia Access
FRR	False Rejection Rate
FTP	File Transfer Protocol
FTAM	File Transfer, Access and Management

G

GF	Galois Feld
GPRS	General Packet Radio Service
GPS	Global Positioning System
GSM	Global System for Mobile Communication
GSS	Generic Security Services
GUI	Graphical User Interface

H

HAL	Hardware Abstraction Layer
HBCI	Homebanking Computer Interface Standard
HLR	Home Location Register
HTML	Hypertext Markup Language
HTTP	Hypertext Transport Protocol

I

IANA	Internet Assigned Numbers Authority
ICAO	International Civil Aviation Organisation
ICMP	Internet Control Message Protocol
IDEA	International Data Encryption Algorithm
IDS	Intrusion Detection System
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
IIS	Internet Information Service
IKE	Internet Key Exchange Protocol
IMSI	International Mobile Subscriber Identity
IP	Internet Protocol
IrDA	Infrared Data Association
ISAKMP	Internet Security Association Key Management Protocol
ISDN	Integrated Services Digital Network
ISO	International Organization for Standardization
ISP	Internet Service Provider
IT	Informationstechnik

ITSEC	Information Technology Security Evaluation Criteria
IuK	Informations- und Kommunikationstechnik
IuKDG	Informations- und Kommunikationsdienstegesetz
IV	Initialisierungsvektor

J

JAR	Java Archive File
JDK	Java Development Kit
JIT	Just in Time Compiler
JSP	Java Server Pages
JVM	Java Virtual Machine

K

KDC	Key Distribution Center
KonTraG	Gesetz zur Kontrolle und Transparenz im Unternehmensbereich

L

LAI	Location Area Identifier
LAN	Local Area Network
LCP	Link Control Protocol
LDAP	Lightweight Directory Access Protocol
LEAF	Law Enforcement Access Field
LFSR	Linear Feedback Shift Registers
LMP	Link Manager Protocol
LSA	Local Security Authority
LSB	Least Significant Bit
LTE	Long Term Evolution
L2CAP	Logical Link Control and Adaption Layer
L2TP	Layer 2 Tunneling Protocol

M

MAC	Message Authentication Code
MAC	Medium Access Control
MAC	Mandatory Access Control

MBR	Master Boot Record
MD	Message Digest
MIC	Message Integrity Code
MIME	Multipurpose Internet Mail Extension Protocol
MLS	Multi Level Security
MRZ	Machine Readable Zone
MSC	Mobile Services Switching Center
MS	Mobile Station

N

NAT	Network Address Translation
NAS	Network Access Server
NGN	Next Generation Network
Nonce	Number used once
NOP	No Operation Befehl
NFS	Network File System
NTFS	Windows NT File System
NIS	Network Information System
NIST	National Institute of Standard
NSA	National Security Agency
NSS	Network and Switching Subsystem
NT	New Technology

O

OBEX	Object Exchange Protocol
OCSP	Online Certificate Status Protocol
OFB	Output Feedback
OLE	Object Linking and Embedding Technologie
OSF	Open System Foundation
OSI	Open Systems Interconnection
OSPF	Open Shortest Path First
OTP	One Time Password
OWASP	Open Web Application Security Project

P

PACE	Password Authenticated Connection Establishment
PAN	Personal Area Network
PCR	Platform Conguration Register
PAT	Port Address Translation
PAP	Password Authentication Protocol
PDA	Persönlicher digitaler Assistent
PDU	Protocol Data Unit
PEAP	Protected Extensible Authentication Protocol
PEM	Privacy Enhanced Mail
PFS	Perfect Forward Secrecy
PGP	Pretty Good Privacy
PIN	Personal Identification Number
PKI	Public Key Infrastruktur
PKCS	Public Key Cryptography Standards
PP	Protection Profile
PPP	Point-to-Point Protocol
PPTP	Point-to-Point Tunneling Protocol
PSTN	Public Switched Telephone Network
PUF	Physically Unclonable Function
PUK	Pin Unblocking Key

R

RA	Registration Authority
RADIUS	Remote Authentication Dial In User Service
RAN	Radio Access Network
RFID	Radio Frequency Identification
RBAC	Role-based Access Control
RegTP	Regulierungsbehörde für Telekommunikation und Post
RFC	Request for Comment
RFI	Remote File Inclusion
RGN	Random Number Generator
RIP	Routing Information Protocol

RMI	Remote Method Invocation
ROM	Read Only Memory
RPC	Remote Procedure Call
RSA	Rivest Shamir Adleman
RSN	Robust Security Network
S	
SA	Security Association
SAE	System Architecture Evolution
SAM	Security Accounts Manager
SAML	Security Assertion and Markup Language
SAS	Secure Attention Sequence
SC	Security Class
SCO	Synchronous Connection-Oriented
SDP	Service Discovery Protocol
SDR	Software Defined Radio
SET	Secure Electronic Transaction
SGSN	Service GPRS Support Node
SHA-1	Secure Hash Algorithm
SIM	Subscriber Identity Module
SIP	Session Initiation Protocol
SMIME	Secure Multipurpose Mail Extension
SMTP	Simple Mail Transfer Protocol
SLIP	Serial Line Internet Protocol
SLA	Service-level Agreement
SOA	Service-oriented Architecture
SPI	Security Parameter Index
SSI	Server Side Include
SSID	Service Set Identifier
SSL	Secure Socket Layer
SSH	Secure Shell
SSO	Single-Sign-On
SS7	Signal System No.7

T

TCPA	Trusted Computing Platform Alliance
TCB	Trusted Computing Base
TCG	Trusted Computing Group
TCP	Transport Control Protocol
TCS BIN	Telephony Control Specification Binary
TCSEC	Trusted Computer System Evaluation Criteria
TDG	Teledienstegesetz
TDDSG	Teledienstedatenschutzgesetz
TKIP	Temporal Key Integrity Protocol
TLLI	Temporary Logical Link Identifier
TLS	Transport Layer Security
TMSI	Temporary Mobile Subscriber Identificator
TPM	Trusted Platform Module
TTP	Trusted Third Party
3GPP	3rd Generation Partnership Project

U

UAM	Universal Access Method
UDDI	Universal Description, Discovery and Integration
UDP	User Datagram Protocol
UMTS	Universal Mobile Telecommunication System
URL	Uniform Resource Locator
URI	Uniform Resource Identification
USIM	UMTS Subscriber Identity Module

V

VLR	Visitors Location Register
VoIP	Voice over IP
VPN	Virtuelles privates Netz

W

WAE	Wireless Application Environment
WAP	Wireless Application Protocol

WAN	Wide Area Network
WEP	Wired Equivalent Privacy
WLAN	Wireless Local Area Network
WML	Wireless Markup Language
WPA	Wi-Fi Protected Access
WSDL	Web Service Description Language
WWW	World Wide Web
W3C	World Wide Web Consortium

X

XML	eXtensible Markup Language
XSS	Cross-Site-Scripting Angriff

Z

ZKDSG	Zugangskontrolldiensteschutzgesetz
--------------	------------------------------------

Index

- *property 266
- /etc/passwd 461
- 2 step Verification 456
- 2DSE 325
- 3DES 521
- 3GPP 876
- §202c StGB 26
- 802.11i 902
- 802.15.4 949
- A**
 - A3 857
 - A5 859
 - A8 858
 - AAA 490
 - Abwehr 386
 - Access Point 104, 894
 - Access Token 702
 - AccessCode 457
 - Accounting 13, 491, 717, 733
 - Acknowledgement 113
 - ACL 622, 623, 896
 - Active Directory 700
 - Adressraum 618
 - AEAD 315
 - AES 319, 326, 327, 871, 902
 - Ajax 156
 - Akkreditierung 401
 - Android 82
 - Angreifer 286
 - Angreifer-Modell 192, 194
 - Angriff 18
 - aktiv 18
 - birthday 369
 - brute force 358
 - Cache-Poisoning 127
 - DDoS 68
 - Denial-of-Service 19, 121, 126
 - exhaustive search 358
 - Known Plaintext 307
 - look-alike fraud 515
 - Man-in-the-Middle 432
 - Maskierung 19
 - Meet-in-the-middle 325
 - passiv 18
 - Passwort-Cracking 447
 - Replay 135
 - Session Hijacking 133
 - Smurf 122
 - Sniffer 19, 103, 135, 150
 - Spoofing 19, 119, 136
 - SQL 168
 - SSL-Heartbleed 788
 - SYN-flood 69, 122
 - Typejacking 160
 - Wörterbuch 467
 - XSS 164
- Anonymisierung 13, 150
- Anonymität 13, 123
- App 79
- Apple Pay 686
- AppleTV 695
- Applikationsfilter 731
- ARP 127
- ASLR 53
- ASN.1 102
- Assertion 650, 655, 656
- asset 4
- Attachment 60, 68
- Attestierung 568
- Attributierung 192
- Audit 13, 717
- Ausweis App 539, 549

- Authentication Header (AH) 753, 760
Authencode 402
Authentifikation 201, 443, 453, 477, 856
 biometrisches Merkmal 472
 Kerberos 491
 Wissen 445, 504
 zwei Faktor 445
Authentifizierungsserver 420
Authentifizierungs-Vektor 871
Authentikator 496
Authentizität 8, 119, 376, 496
Autorisierung 5
Avalanche-Effekt 324
- B**
- BAC 528
Bastion-Host 729
Bedrohung 17, 184
Bedrohungsanalyse 184
Bedrohungsbaum 186
Bedrohungsmatrix 184
Bell LaPadula 265, 642
Benutzermodus 617
Benutzerprofile 159
Berechtigungskontrolle 624
Berechtigungszertifikat 543
Beschränkung 272
Betriebsmodus 308, 616
 CBC 309
 CFB 312
 CTR 313
 ECB 309
 OFB 311, 875
Beweissicherung 203
Biometrie 472, 610
 Fehlerrate 478
 Fingerabdruck 474
 Gesichtserkennung 475
 Iris 475
BIOS 57, 594
Birthday Attack 369
Bitcoin 826, 841
BitLocker 414
Bitpermutation 301
- blindes Schreiben 270
Blockchain 826
Blockchiffre 302, 764
Bluetooth 920
 Authentifikation 937
Modi 929
Pairing 934
Schlüsseltypen 934
Sicherheitsdienste 927
Vertrauenslevel 931
Booten 594
Bot-Netz 23, 75
Bridge 104
Browser 151
BSI 24, 177, 182, 195, 198, 206
 Sicherheitsprozess 174
BSI-Lagebericht 24
BTB 86
Buffer-Overflow 45, 69
Bump in the Stack 752
Bundesnetzagentur 393, 405
- C**
- CA 399
Cache 137
Cache-Poisoning 138
Caesar-Chiffre 280, 287
Callback-Problem 724
Capability 627, 641, 707
Capabilityliste 627
CBC-Modus 310, 377, 764
CCMP 906
Certification Authority 399
CFB-Modus 312
CGI 152
ChaCha20 307
Challenge-Response 465, 857, 872
Challenge-Semantik 525
CHAP 749
Chinesischer Restsatz 334
Chipkarte 504
Chosen-ciphertext 343, 359
Chosen-plaintext 359, 362, 447, 776, 875
Ciphertext-only 358
Classification 265

Clearance 265, 642
Client-Server 444, 486, 624
Cloud 661
COMP-128 856, 859
Computer Forensik 30
Confinement-Problem 10
Cookie 154, 769
Country Signing Authority 523
covert channel 5
CPS 404
CRC 101, 123, 760
Credential 8, 443, 591
Cross-Site-Scripting 165
Cross-Zertifikat 408
CrypTool 281
CSCA 530
CSMA/CD 101
CTR-Modus 313
Cut-and-Paste-Angriff 775
Cyber Sicherheit 41
Cyber Space 41
Cyber-Kriminalität 24
Cyber-Physical System 40

D

DAO 837
Dateisystem
 verschlüsselt 709
Daten 4
Datenschutz 159
Datensicherung 6
DDoS 68, 75, 121
Delegation 629
Denial-of-Service 126
DES 317, 362, 372, 377, 462, 764
 3DES 325, 450
 Initialpermutation 320
 S-Box 320
 Schlüssel 322
 Sicherheit 323
Deskriptor 623
DHCP 116
Differenzial Power Analysis 513
Diffie-Hellman 428, 525, 540, 767
 ephemeral 525, 541
Diffusion 297

DigiNotar 793
Discretionary Access Control 214
diskreter Logarithmus 333, 386,
 428, 432
DNS 109, 135
 Poisoning-Angriff 138
DNSSEC 796
Document Security Object 519
Document Signer 522
DOI 759
Domänenname 109, 135
Doppelverschlüsselung 426
DoS 121, 125, 129, 899
DREAD 211
Drive-by-Download 157
DSA 386
DSS 386, 770
Dual_EC_DRBG 355
Dual-Home Firewall 735

E

EAC 409, 529
EAL 224, 226
EAP 750, 917
eAT 535
ECB-Modus 309
ECDH 434
EC-DH 557
ECDHP 435
ECDLP 347, 351
EEPROM 506, 511
EFS 438, 710
eID 537
eID Server 551
eIDAS 391
Einheitswurzel 429
Einweg-Eigenschaften 376
Einweg-Funktion 332, 367, 446,
 452
 Falltür 334
Elektronischer Personalausweis 515
Elevation of Privileges 50, 209
Elliptische Kurve 347
E-Mail 146, 801
Encapsulating Security Payload
 (ESP) 753, 763

eNodeB 878
 Entropie 294, 609
 Entwicklungsprozess 172
 ePass 515
 Authentifizierung 520
 EPC 877
 EPS 877
 EPS-AKA 884
 Equal Error Rate (EER) 478
 Ethereum 837, 846
 Evaluierung 213
 Evaluierungsgegenstand 224
 Evaluierungsstufe 224, 394
 EVG 224
 Exhaustive Search 358, 861
 existentielle Fälschung 344
 Exploit 21
 Extension Header 129

F

Faktorisierung 333, 342
 False Acceptance Rate (FAR) 478
 False Rejection Rate (FRR) 478
 Feistel-Chiffre 319
 Fiat-Shamir-Verfahren 470
 FIDO 555
 File Handle 142, 144
 File-Descriptor 640
 File-Handle 145
 Fingerabdruck 474, 480
 digital 63
 Firewall 716
 Anwendungs-Gateway 731
 Architektur 734
 Dual-Home 735
 Paketfilter 719
 Proxy 728
 Screened-Host 736
 Flussrelation 272
 Forensik 203
 Tools 30
 Frequency hopping 923
 Fritz Chip 581
 FTP 148
 φ -Funktion 336
 Funktionsklasse 218, 225

G
 Galois Counter 316
 Galois-Feld 326, 428
 Gateway 105
 GCM 316
 GDPR 6
 Geburtstagsangriff 369, 379
 Geburtstags-Paradoxon 368
 Gefährdungsfaktor 16
 Gleichfehlerrate 478
 Google-Authenticator 456
 GPRS 865
 Grundschutz 206
 Grundschutz-Katalog 174
 Grundschutzkatalog 177, 182, 195,
 206
 GSM 852
 GSM-R 864

H

Hacker 21
 -Paragraph 26
 Handgeometrie 474
 Handshake 115, 421, 424
 Hashfunktion 366, 452
 HMAC 378
 Keyed 377
 kollisionsresistent 369
 MAC 376
 Passwort 446
 preimage resistance 367
 schwach kollisionsresistent
 366
 second preimage 367
 SHA 373
 stark kollisionsresistent 369
 Hasse Diagramm 273
 Heap 47
 Heartbeat-Funktion 789
 Helper Data 611
 HMAC 378, 762, 778
 HomeKit 691
 Honeypot 733
 Hot-Spot 896
 HSS 880
 HTML5 162

- HTTP 151
zustandslos 151
- HTTPS 778
- Hub 103
- I**
- IANA 108
- ICMP 125
redirect 125
- ICMPv6 112, 129
- Identifikation 201, 460, 632, 854
- IDS 65, 732
- IEEE 802.11 891
- IEEE 802.15.4 952
- IKE 758, 767
- IKT 1
- Impersonation 703
- implizites Cachen 86
- IMSI 854
- IMSI-Catcher 861
- Information 4
- informationelle Selbstbestimmung 6, 14, 147, 180, 483
- Informationsfluss 267, 274, 642
- Informationsgehalt 293
- Informationstheorie 291
- Initialisierungsvektor 310, 372, 764
- inode 142
- Integrität 9, 370
- Interface Identifier 111
- Interference Control 10
- Internet 97
- Internet Protocol, IP 105
- Interworking 881
- Intrusion Detection 732
- iOS 83, 664
class-key 672
Datei-Schlüssel 672
Datei-Schutz 671
Effaceable Storage 665
ephemeral Key 667
file system key 673
keybag 680
keychain 682
Krypto-Engine 666
Passcode 671
- Root Ca Public Key 669
- Sandbox 683
- Schutz-Klassen 675
- secure boot 669
- sicheres Update 670
- Touch ID 667
- UID 666
- wipe 673
- IoT 39
- IP 107
- IP-Adresse 109, 495, 720, 742
- IP-Paket 109
- IPSec 113, 751
- IPv4 108
- IPv6 111
Sicherheit 128
- IRC 76
- Iris 475
- ISAKMP 759
- ISO 7
- ISO/OSI-Referenzmodell 99, 741
- ISP 105
- IT-System 3
- J**
- Jailbreak 685
- JavaCard 514
- K**
- KAISER 94
- Kanal 291
kryptografisch 292
stochastischer 291
- Kartenleser 550
Basis- 550
Komfort 550
Standard- 550
- Kasiski-Test 361
- KASUMI 863, 875
- KDE 886
- KDF-chain 819
- Keccak 370
- Kerberos 491, 625, 701
- Kerckhoffs-Prinzip 280, 290
- Key Escrow 436
- Key Handle 558

- Key Recovery 436
 Key-Blob 567
 Known-plaintext 359, 364, 467
 Kollision 366, 369
 Komplexität
 NP 299
 NP-hart 300
 P 298
 Komplexitätstheorie 298
 Kompression 371
 Konfusion 297
 Konsensus 832
 Konstruktion
 Entwicklungsphasen 174
 KonTraG 177
 KPTI 94
 Kryptoanalyse 279, 358
 differentielle 362
 lineare 363
 Kryptografie 279
 kryptografisches System 285
 asymmetrisch 289, 331
 hybrid 288
 Public-Key 289
 Secret-Key 288
 symmetrisch 288, 300
 Kryptographie
 elliptische Kurve 353
 Kryptologie 279
 Kryptosystem 285, 332
- L**
- L2TP 749
 labeling 214
 Landau-Symbol 298
 Langzeitarchivierung 395
 LCP 107
 ledger 828
 LFSR 305
 Lock/Key 631
 logische Bombe 72
- M**
- MAC 376, 760
 MAC-Adresse 109, 127, 896
 MAC-Geheimnis 376
 Makro 59
 malware 24
 Mandatory Access Control 214, 642
 Man-in-the-Middle 419, 918, 941
 Maskierung 19, 127, 143, 445, 454,
 498
 Maskierungsangriff 187
 Master Key 417, 494, 770
 Maximum-Likelyhood 292
 MD5 762
 Mechanismus 217
 Meltdown 84, 93
 Merkle-Tree 829
 Message Authentication Code 376
 Message Digest 366
 MFT 706
 Mikrocontroller 505
 MIME 60, 146, 148
 mining 832
 Minutie 480
 Mixe 147
 MME 879
 Modell 239
 Modulo-Rechnung 335
 Monitoring 200
 Mounten 140
 MRZ 518, 536
 Multimedia 725
 H.232 725
 T.120 725
 multiplikative Inverse 336
 Muster 360
- N**
- NAT 116, 720, 763, 767, 772
 Needham-Schroeder Protokolle 419,
 492
 Netztopologieplan 177
 NFS 140, 498
 Nonce 420, 423, 497
 no-read-up 266, 642
 no-write-down 266, 642
 nPA 535, 564, 570
 NSA 22, 788
 Nutzungsprofil 159

O

OASIS 647
Objekt 4, 240, 623
Objektschutz 622
OCSP 404
OFB-Modus 311
one-pass processing 741, 803
One-time-pad 296, 305, 306
O-Notation 298
OPIE 456
Orange Book 213
Orchestrierung 645
Out-of-Order 86
OWASP 163
Owner 242

P

P2P 277
PACE 540, 547
Padding 303, 340, 345, 764
Pairing 692, 934
Paketfilter 719
 dynamisch 723
 Filterregel 722
 stateful 723
 zustandslos 722
PAN 922
PAP 107, 749
Passphrase 809
Password-Cracking 464
Passwort 446
 benutzerwählbar 461
 einmal 451
 shadowed 464
 systemgeneriert 450
PAT 117, 720
PDN 880
PDP 624
Penetrationstest 197
PEP 624
Perfect Forward Secrecy 418, 571, 788, 791
Perimeter-Sicherheit 737, 739
Permutation 301
PFS 418

PGP 807
Mantra 809
Schlüsselring 809
Phishing 23, 26, 455
PHP 153
Physical Unclonable Function 604
Piconetz 923
PIN 450, 509, 540, 855, 934
Pixel 283
PKCS 346, 404, 804
PKI 384, 402–404, 530
 CA 404
 CRL 404
 OCSP 404
 RA 404
PMK 904
PNG 410
Policy 199
Poly1305 308
Port 113, 721
 trusted 114
PPP 107, 749
PPTP 749
prepared statement 169
Prinzip 172
 -Erlaubnis 172
 minimale Rechte 172
 need-to-know 172, 240
 -Owner 202
 Vollständigkeit 172
Privacy 6, 14, 146, 149, 592
Privacy Extension 111
Privatsphäre 14, 158
Produktchiffre 302
Proof-of-Stake 836
Proof-of-Work 831
Protection 6
Protection Profile 229, 582
Proxy 729, 731
Pseudomisierung 13, 587
Pseudonym 13, 546
Pseudozufallsfolge 305
Pseudozufallszahl 410
PTK 905
PUF 604
 Arbiter 606

-
- Controlled 609
 - Eigenschaften 608
 - optisch 605
 - Ring-Oszillator 606
 - Schlüsselerzeugung 612
 - SRAM 607
 - Strong 609
 - Weak 610
 - PUK 510, 855

 - Q**
 - Qualität 218
 - Qualitätsstufe 219

 - R**
 - RADIUS 486, 915
 - Rainbow-Table 863
 - RAM 506, 511
 - RAN 877
 - Ransomware 23
 - Ratchet 819
 - DH 821
 - RBAC 252
 - Recht 5, 202, 241, 245, 265, 620, 623, 633
 - Rechteverwaltung 202, 615
 - Recovery 401, 712
 - Redundanz 294
 - Referenzmonitor 244
 - Regelkreis 174
 - Registry 698
 - RegTP 393, 405
 - Remailer 147
 - remote Shell 69
 - Repeater 102
 - Restricted Identification 546
 - Retina 475
 - Reverse Lookup 135
 - RFID 516
 - RIP 124
 - Risiko 17, 191
 - Risikoanalyse 191
 - Risikobewertung 191
 - Risikomanagement 177
 - risk assessment 191
 - RNG 410

 - Rolle 243, 253, 626, 915
 - Hierarchie 255
 - ROM 505
 - ROP 53
 - Router 99, 105, 742
 - Routing 101, 124
 - RPC 121
 - RSA 289, 335, 770
 - Challenge 343
 - Modul 337, 341
 - multiplikativ 345
 - Sicherheit 341
 - RSA SecureID 458
 - RTM 575

 - S**
 - S/Key 451
 - S/MIME 801
 - SAE 876
 - Safety 6, 248
 - Salt 463
 - Same Origin Policy 155
 - SAML 650, 656
 - S-Box 320
 - Scatternetz 924
 - Schadenszenarien 179, 181
 - Schadsoftware 43
 - Schieberegister 305
 - Schlüssel
 - Master Key 417
 - Sitzungs- 417
 - Schlüsselaustausch 415
 - asymmetrisches Protokoll 423
 - symmetrisches Protokoll 420
 - Schlüsselerzeugung 410
 - Standard X9.17 412
 - Schlüsselhierarchie 417, 886
 - Schlüsselhinterlegung 436
 - Schlüsselraum 290
 - Schlüsselrückgewinnung 436
 - Schlüsselspeicherung 412
 - Schlüsselvernichtung 415
 - Schutzbedarf 179, 182
 - Schutzbedarfskategorie 179
 - Schutzdomäne 630
 - Schutzprofil 224

- Schutzziel 7
Schwachstelle 15
Schwachstellen 157
Screened-Host Firewall 736
Screened-Subnet 737
Screening Router 722
SDL 207
Secure Element 555
Secure Enclave 666
Security 6
Security Association 654, 755
Security Engineering 171
Security Gateway 716
Segment 620
Seite 621
SE-Linux 268
sendmail 67
Sequenznummer 115
 -angriff 131
Service 644
 Web-Service 647
Service-Gateway 880
Session Hijacking 133, 776, 918
SHA 373
SHA-1 762
SHA-2 375
SHA-3 370
Shannon 291
Sicherheit 6, 295
 absolut 295
 datensicher 6
 funktionssicher 6
 informationssicher 6
 mehrseitige 34
 praktische 296
Sicherheitsarchitektur 34
Sicherheitsdienst 460
Sicherheitseigenschaft 248, 256
Sicherheitsgrundfunktion 201
Sicherheitsinfrastruktur 34
Sicherheitskern 173
Sicherheitskette 37
Sicherheitsklasse 265, 273, 642, 756
Sicherheitskriterien 213
 Common Criteria 222
IT- 217
ITSEC- 219
TCSEC 213
Sicherheitsmarke 260, 265
Sicherheitsmodell 199, 215, 239
 Bell-LaPadula 265, 642
 Biba 271
 Chinese-Wall 260
 RBAC 252
 Verband 272
 Zugriffsmatrix 244
Sicherheitsprozess 174
Sicherheitsrichtlinie 32
 benutzerbestimmbar 32
 discretionary policy, DAC 32
 mandatory policy, MAC 32
 Multi-Level 33
 RBAC 32
 rollenbasiert 32
 systembestimmt 32
Sicherheitsstrategie 199, 718
 benutzerbestimmbar 214, 242
 Informationsfluss- 243
 mandatory policy, MAC 243
 Multi-Level 265, 268
 RBAC 243
 systembestimmt 214, 262, 266
Sicherheitsstufe 214
Sicherheitsvorgabe 224
Signal 814
Signal-Protokoll 814
Signatur 332, 380
 Angriff 385
 DSS 386
 EU-Richtlinien 390
 qualifiziert 392
Signaturgesetz 390, 405
SIM
 Clon 863
SIM-Karte 855
Simple Pairing 943
simple-security 266
Single-Sign-On 492, 500
skimming 534
Skript Kiddie 22
SLA 645
SLIP 106

- Smart Contract 836
Smartcard 504
 PIN 509
 Sicherheit 512
SMTP 146, 721, 801
Sniffer 135, 150
SOA 644
 Sicherheit 645
SOAP 647
Sobig 61
Social Engineering 25, 61
Spam 77
SPD 756
Spectre 85
Speicherschutz 616
spekulative Ausführung 85
SPI 756, 760
Spoofing 119, 136
 IP 119
 Web 160
Spread Spektrum 923
Spyware 24
SQL-Angriff 168
SRP 692
SS7 857
SSID 895
SSL 778
SSI
 Heartbleed 788
Stack 48
Stack Canary 52
Stack Cookie 52
Steganografie 279
 linguistisch 282
 technisch 283
STRIDE 209
Stromchiffre 304, 311
STS 433
Subjekt 5, 240, 444, 623
Substitutionschiffre 360
Substitution 302, 320, 360
Switch 104
System 3
Systemarchitektur 200
Systemmodus, privilegiert 616
- T**
TCG 414, 575
 Subsystem 575
TCP 113, 131
 Handshake 115
TCP/IP 720
Teledienstedatenschutzgesetz
 (TDDSG) 176
Telemediengesetz 176
Telnet 150
Terminal 550
Testen 200
TFTP 149
Thread 619
Ticket 495
Ticket-Granting 493
Timestamps 422
TKIP 902
TLS 778
 Handshake 781
 Master Secret 784
 Pre-Master Secret 784
TLS1.3 795
TPM 414, 575
 AIK 587
 Endorsement Key 587
 Ownership 588
 Storage Root Key 588
 TPM 2.0 603
Transposition 301
Trojaner 71
Trojanisches Pferd 62, 71, 638
trust 14, 120, 383, 397, 575, 579
trust anchor 406
Trust Center 393, 399, 440
Trusted Computing Base 173, 215
Trusted Host 120
Trusted Path 215, 643
TSS 576
Tunneling 738, 749, 754
Typejacking 160
- U**
U2F 457, 554
 Registrierung 556
 Sicherheit 563

- UAM 896, 898
UDDI 648
UDP 115, 130
uid 142
UMTS 867
 AKA 870
 Sicherheitsarchitektur 868
Unentscheidbarkeit 249
Unix 459, 460
 ACL 634
 crypt(3) 462
 guid 460
 inode 639
 suid 636
 uid 633
 Zugangskontrolle 460
 Zugriffskontrolle 632
 Zugriffsrechte 633
Unix MLS 268
URL 151
USIM 868
UTXO 841
- V**
Validierungspfad 408
Verband 272
Verbindlichkeit 12, 123
Verbindung 114
Verfügbarkeit 204
VeriSign 402, 806
Verkehrsflussanalyse 744, 754
Verlässlichkeit 7
Verschlüsselung
 Ende-zu-Ende 744
 Verbindungs 742
Vertrauen 14, 277
Vertrauenswürdigkeit 15
Vertraulichkeit 10, 123
Verwundbarkeit 16
Virtuelle Adresse 617
virtueller Adressraum 47
Virus 55
 ANI 61
 Boot 595
 Boot- 57
 Makro 60
- Programm- 57
Scanner 64
Sobig 61
VPN 747, 864
Vulnerability 15
- W**
W3C 647
Wasserzeichen 281
Watermarking 281
Web of Trust 383, 811
Web-Anwendung 150
Web-Service 647, 738
 Standards 649
Web-Service Security 649
WEP 899
WhatsApp 814
Wiederaufbereitung 204
Windows 204, 710
 Object Handle 707
 SAM 700
 security descriptor 704
Windows 2000
 Registry 61, 596
Windows10 695
 Hyper-V 696
 WHQL 695
Wirksamkeit 221
WLAN 417, 891
 4-Wege-Handshake 905
 Client-Isolierung 910
 Hole 196 908
 KRACK 911
 Michael 902
WPA 902
WPA2 902
WS 651
 Policy 655
 Secure Conversation 654
 Secure Exchange 653
 Security 651
 Security Context 654
 Trust 653
WSDL 648
WSS 649

- Wurm 66
Blaster 68
ILOVEYOU 68
Internet- 66
Lovesan 68
Wurzelinstanz 405
WWW 151
- X**
X3DH 815
X.509 398, 783
XACML 660
XML 647, 656
 Encryption 651
 Signature 651
XPath 649
XSS 164, 165
- Z**
Zeitstempel 381, 384, 401
Zero-Knowledge-Verfahren 469,
 594
Zertifikat 392, 783
 Berechtigungs- 543
 CV 543
 Überprüfung 406
Zertifizierung 237
 Infrastruktur 405
Zertifizierungsstelle 399
ZigBe
 HAPA 964
ZigBee 949
 Link Key 953
 Link-Key 953
 Master Key 953
 Network Key 953, 954
 OTA 963
 Profil 950
 Schlüsseltypen 953
 Topologie 952
 Trust Center 951
 ZigBee 3.0 958
 ZLL 966
 ZigBee IP 958
 ZigBee Pro 958
 ZigBee RF4CE 958
 ZigBee3.0
 dezentral 959
 Install Code 961
 zentral 960
ZKDGS 25
Zufallszahlengenerator 305, 410,
 469
Zugangskontrolle 448, 460
Zugriffsausweis 144, 203, 627
Zugriffsbeschränkung 214, 242, 266
Zugriffshistorie 261
Zugriffskontrolle 141, 202, 269,
 615, 616, 632, 643, 718
Zugriffskontrollliste 623, 634
Zugriffsmatrix 244, 622, 627, 720
Zugriffsrecht 5, 241, 260, 265
Zulässigkeit 625, 629
Zuordnbarkeit 12
zustandslos 144