



Technische Hochschule
Ingolstadt

Fakultät Informatik

Kapitel 4: Ende-zu-Ende Sicherheit

CASE_SMN WS 2023/2024

Vorlesung „Sicherheit moderner Netze“

21.11.2023

■ Anforderung an eine „sichere Verbindung“

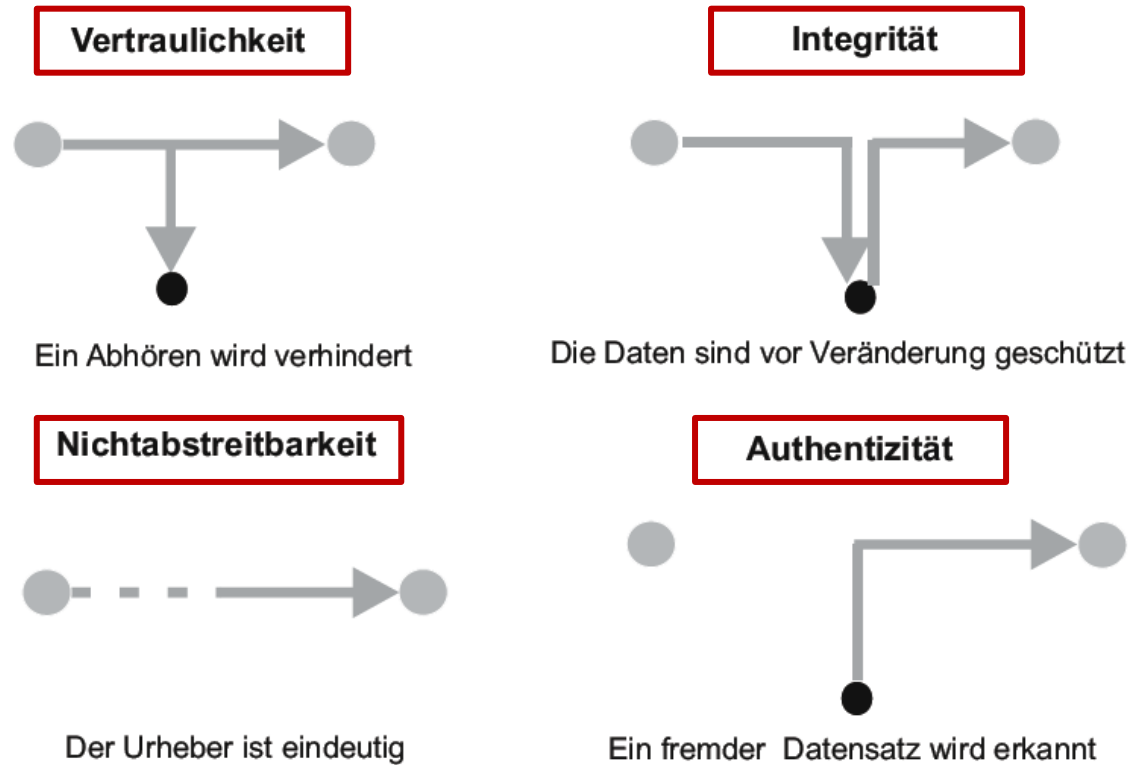


Abb. 6-1: Sicherheitsdienste in Kommunikationsnetzen nach ISO 7498-2

[Quelle: Spitz, S. et al: Kryptographie und IT-Sicherheit]

- **Transportschichtssicherheit für beliebige TCP-basierte Anwendungen über TLS- / SSL-Dienste.**

- z.B. zwischen Webbrowser und –Server für E-Commerce (https)

- **Transparent** für das IP-Netz, **nur für TCP**

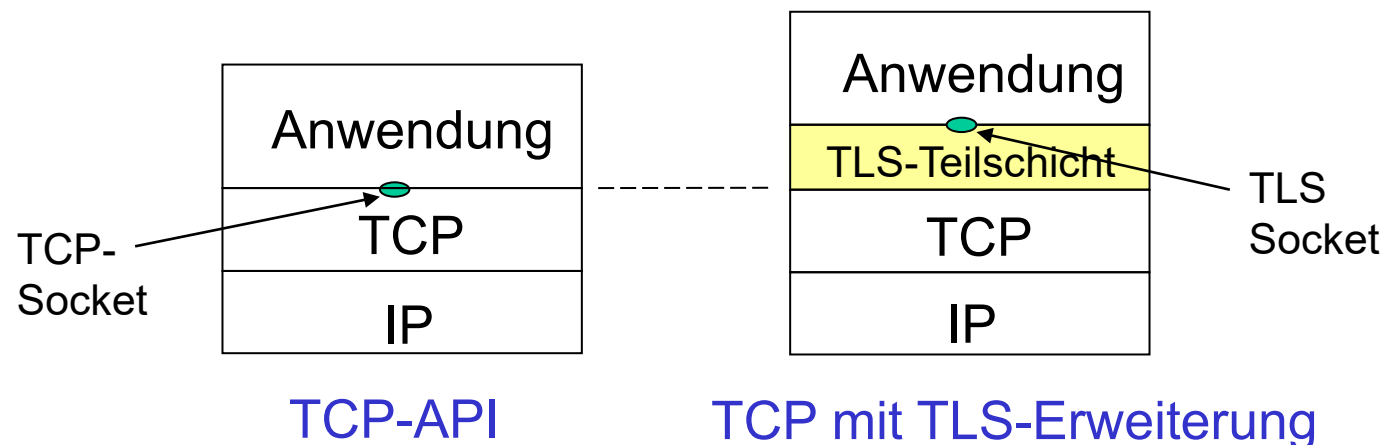
- **Sicherheitsdienste:**

- **Authentisierung** des HTTPS Secure-Servers im SSL-Handshake

- Optionale Authentisierung des Clients (Web Browsers) im SSL-Handshake

- **Verschlüsselung** der Datenübertragung über SSL-Records

- Sicherstellung der **Integrität** der übertragenen Daten über SSL-Records.



SSL bietet folgende Sicherheitsfunktionalitäten an:

- **Authentisierung** des HTTPS Secure-Servers im SSL-Handshake und
- Optionale Authentisierung des Clients (Web Browsers) im SSL-Handshake
 - ➔ asymmetrisches Krypto-Verfahren (z.B. RSA, ...)
 - ➔ Public-Key-Verfahren mit Austausch von Zertifikaten
- **Verschlüsselung** der Datenübertragung über SSL-Records
 - ➔ symmetrisches Krypto-Verfahren (z.B. RC4, DES im CBC- oder GCM-Modus,...)
- Sicherstellung der **Integrität** der übertragenen Daten über SSL-Records (MAC).
 - ➔ Hashfunktion zur Integritätsprüfung (z.B. SHA-1, ..)

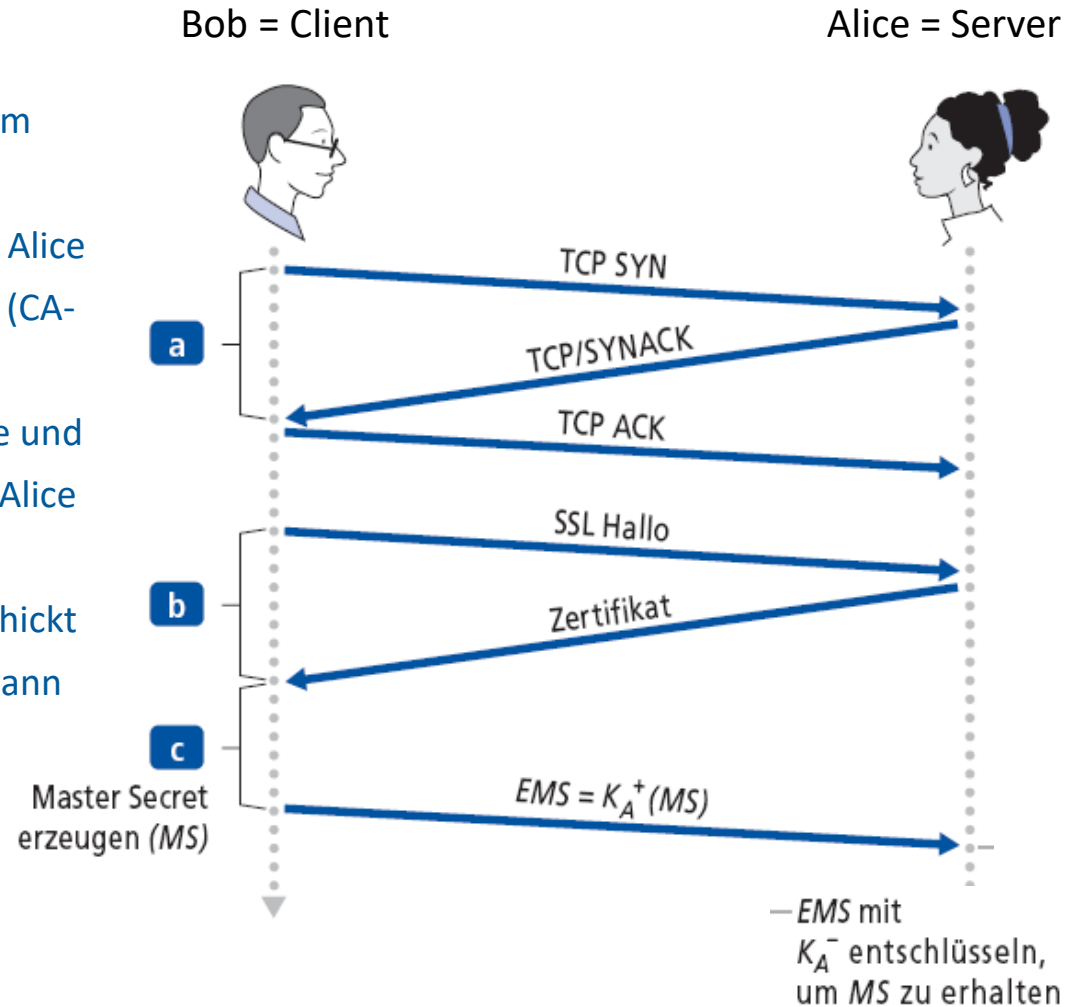
1. Handshake:

- a) Client Bob baut eine TCP-Verbindung zum Server Alice auf
- b) Bob baut dann eine SSL-Session auf und Alice authentifiziert über sich mit einem Zertifikat (CA-zertifiziert)

Das Zertifikat bestätigt die Identität von Alice und enthält auch den **öffentlichen Schlüssel** von Alice

- c) Bob erzeugt, verschlüsselt (mit Alices öffentlichem Schlüssel) und verschickt ein **Master Secret** an Alice. Daraus werden dann die Schlüssel für die Kommunikation abgeleitet

(Vereinfacht dargestellt)



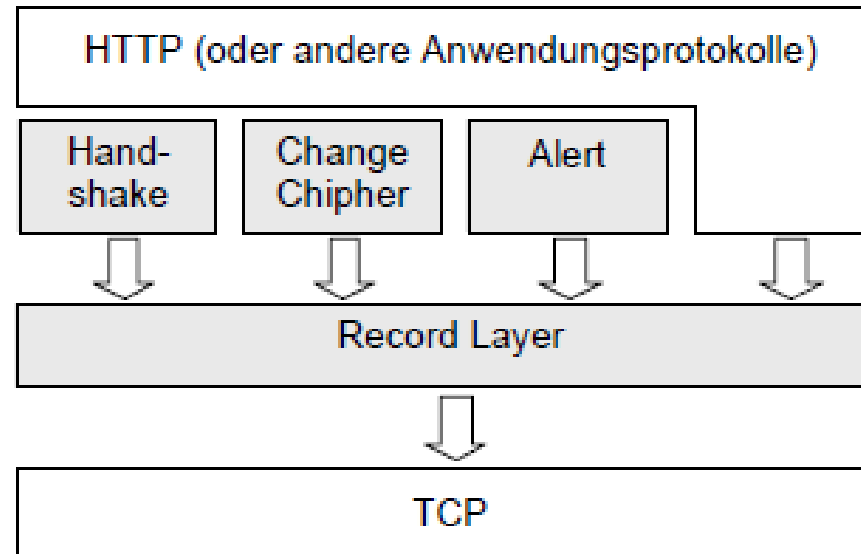


- **Spezielle Portnummern für die Verwendung von SSL definiert:**
 - https: 443
 - ssmtp: 465
 - Telnets: 563
 - Ftps: 990 SSL-basierte FTP-Kontrollnachrichten
 - ftp-data: 889 SSL-basierte FTP-Daten
- Damit wird die SSL-Verbindung für den jeweiligen Dienst initiiert

Aufbau einer SSL-Verbindung – Phase 1

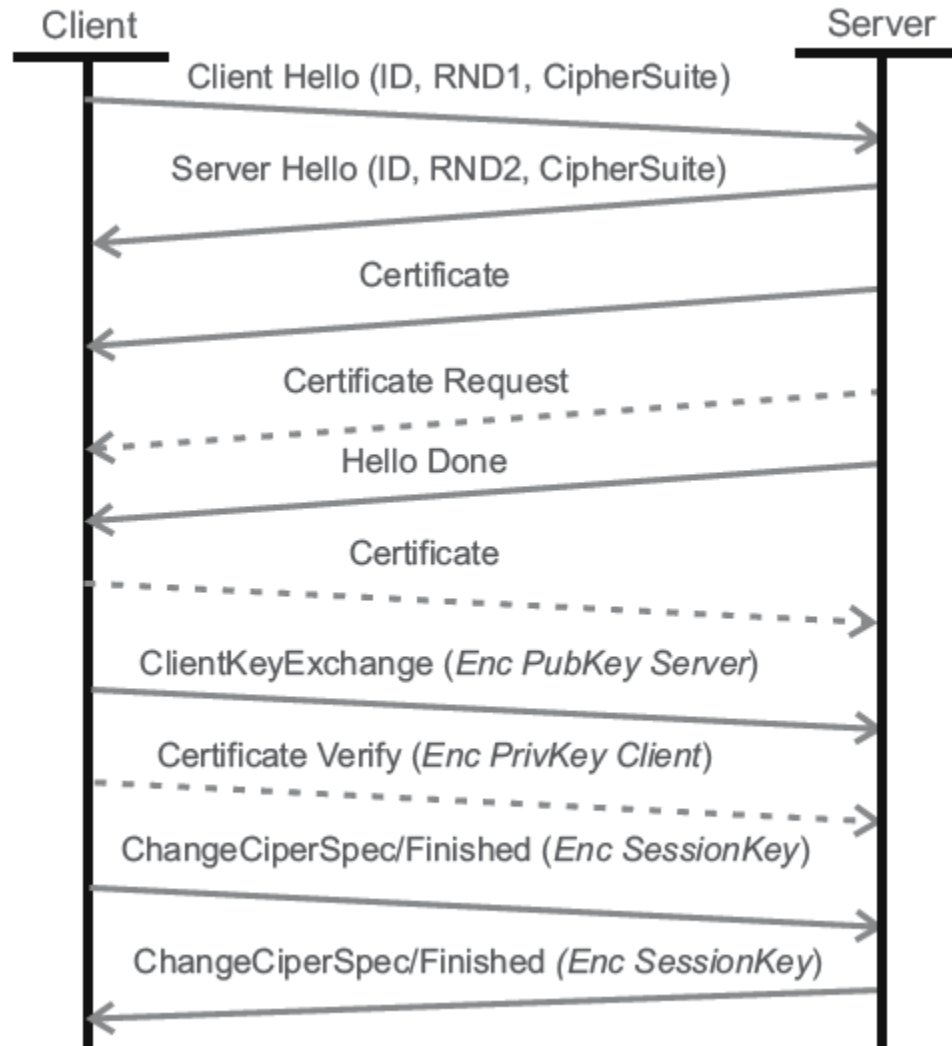
No.	Time	Source	Destination	Protocol	Length	Info
18	5.248885	Client	10.50.1.2	DNS	79	Standard query 0xe25a A banking.postbank.de
19	5.279596	10.50.1.2	Client	DNS	95	Standard query response 0xe25a A banking.postbank.de A 195.50.155.66
20	5.279937	Client	banking.postbank...	TCP	66	49339 → 80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
21	5.298294	banking.postbank.de	Client	TCP	62	80 → 49339 [SYN, ACK] Seq=0 Ack=1 Win=4140 Len=0 MSS=1380 SACK_PERM=1
22	5.298338	Client	banking.postbank...	TCP	54	49339 → 80 [ACK] Seq=1 Ack=1 Win=64860 Len=0
23	5.298425	Client	banking.postbank...	HTTP	356	GET / HTTP/1.1
24	5.317557	banking.postbank.de	Client	HTTP	174	HTTP/1.0 302 Found
25	5.320985	Client	banking.postbank...	TCP	66	49340 → 443 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
26	5.355398	banking.postbank.de	Client	TCP	62	443 → 49340 [SYN, ACK] Seq=0 Ack=1 Win=4140 Len=0 MSS=1380 SACK_PERM=1
27	5.355448	Client	banking.postbank...	TCP	54	49340 → 443 [ACK] Seq=1 Ack=1 Win=64860 Len=0
28	5.355558	Client	banking.postbank...	TLSv1.2	265	Client Hello
29	5.374299	banking.postbank.de	Client	TLSv1.2	1434	Server Hello
30	5.374313	banking.postbank.de	Client	TCP	1434	[TCP segment of a reassembled PDU]
31	5.374336	Client	banking.postbank...	TCP	54	49340 → 443 [ACK] Seq=212 Ack=2761 Win=64860 Len=0
32	5.374375	banking.postbank.de	Client	TCP	1434	[TCP segment of a reassembled PDU]
33	5.392592	banking.postbank.de	Client	TLSv1.2	424	Certificate, Server Hello Done
34	5.392606	Client	banking.postbank...	TCP	54	49340 → 443 [ACK] Seq=212 Ack=4511 Win=64860 Len=0
35	5.393540	Client	banking.postbank...	TLSv1.2	396	Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
36	5.394159	Client	10.50.1.2	DNS	72	Standard query 0xe1b6 A s2.symcb.com
37	5.412561	banking.postbank.de	Client	TCP	60	443 → 49340 [ACK] Seq=4511 Ack=554 Win=4693 Len=0
38	5.413520	banking.postbank.de	Client	TLSv1.2	60	Change Cipher Spec
39	5.413723	banking.postbank.de	Client	TLSv1.2	123	Encrypted Handshake Message
40	5.413729	Client	banking.postbank...	TCP	54	49340 → 443 [ACK] Seq=554 Ack=4586 Win=64785 Len=0
41	5.422490	10.50.1.2	Client	DNS	174	Standard query response 0xe1b6 A s2.symcb.com CNAME ocsp-ds.ws.symantec.com..

1. DNS für URL (banking.postbank.de)
2. Aufbau der TCP-Verbindung an Port 80 => http GET
3. TCP-Verbindung auf Port 443 für SSL/TLS aufbauen
4. TLS-Verbindung



- **Handshake:** Schlüsselaustausch => siehe nächste Folie
- **Change Cipher:** Ende des Handshake, Bestätigung der ausgehandelten Verfahren. Beginn des Nutzdaten Austauschs im Record Layer (1 Byte)
- **Alert: Fehlermeldungen.** warning oder fatal (beendet die Verbindung)
- **Record Layer:**
 - Byte Stream in Records-Blöcken (max 2^{14} /16 Kbyte)
 - optionale Kompression, MAC, Verschlüsseln

Aufbau einer TLS-Verbindung



- **Client Hello:** Info über Client, Version des TLS-Protokolls, Mögliche Cipher Suites, Session ID, Zufallszahl RND1
- **Server Hello:** Auswahl Cipher Suite, Zufallszahl RND2, ID
- **Certificate:** Öffentlicher Schlüssel für Pre-Master-Secret (im X.509-Format)
- Evtl. noch Anfrage für Certificate vom Client
- **Hello Server Done:** schließt Nachrichten des Servers ab
- **ClientKeyExchange:** Verschlüsseltes Premaster-Secret oder andere Parameter für Verschlüsselung (Client verschlüsselt PreMaster Secret mit öffentlichem Schlüssel des Servers.)
- **ChangeCipherSpec:** Berechnung der Keys ist abgeschlossen



- Authentisierung des Servers beim Client über ein Challenge-Response-Verfahren
- Optionale Authentisierung des Clients beim Server, ebenfalls Challenge-Response basierend
- Verhinderung einer Replay-Attacke auf die Authentisierung durch client- und serverseitige Zufallszahlen (RND1, RND2)
- Aushandlung der eingesetzten Krypto-Algorithmen, d.h. Vereinbarung der eingesetzten Cipher-Suite
- Berechnung der Schlüssel für die Sicherung der Datenübertragung in den nach dem SSL-Handshake übermittelten SSL-Records.



- Als „SSL Cipher Suite“ wird eine Kombination aus
 - einem asymmetrischen Algorithmus,
 - einem symmetrischen Algorithmus und
 - einer Hashfunktion bezeichnet,

die beim Verbindungsaufbau zwischen Client und Server vereinbart wird.

Eine mögliche Kombination von Kryptoalgorithmen in einer Cipher-Suite findet sich in [RFC2246], Bei Einsatz dieser Cipher-Suite wird z.B.:

- RSA als asymmetrisches Verfahren im SSL-Handshake genutzt,
- SHA-1 als Hashfunktion zur Integritätsprüfung verwendet,
- RC4 zur Verschlüsselung der SSL-Records eingesetzt.

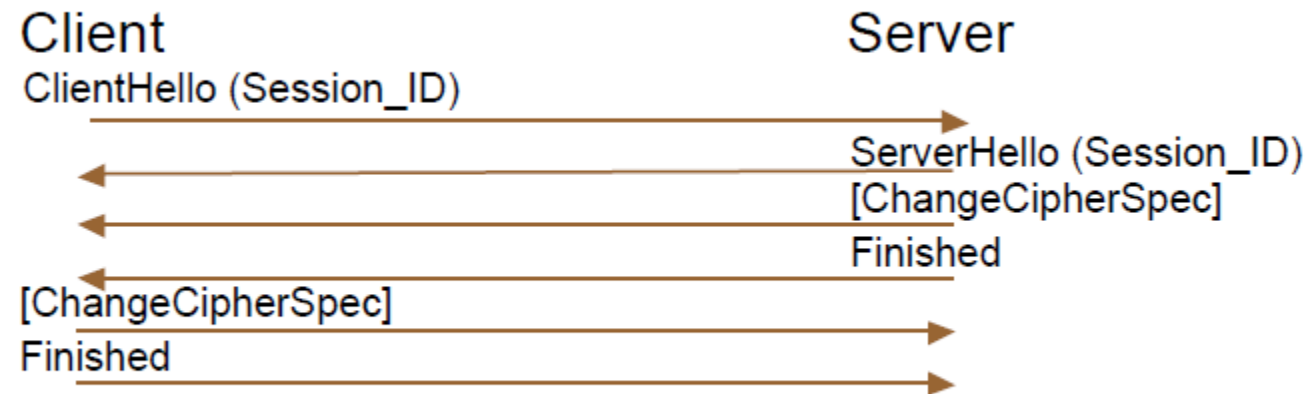


Abb. 7.11 Verkürzter SSL-Handshake

- **Verkürzter Handshake:**
 - Wiederaufnahme einer Session, Zertifikate sind bereits ausgetauscht
 - Anhand der Session ID wird versucht direkt mit dem „alten“ Master Secret weiterzumachen (Server entscheidet)
 - Es werden aber neue Schlüssel errechnet (neue Nounce-Werte)
- **Client Hello und Server Hello:** Abgleichen der Fähigkeiten (verwendete Cipher Suite)

Aufbau einer SSL-Verbindung – 1. Client Hello



25	5.320985	Client	banking.postbank...	TCP	66	49340 → 443	[SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
26	5.355398	banking.postbank.de	Client	TCP	62	443 → 49340	[SYN, ACK] Seq=0 Ack=1 Win=4140 Len=0 MSS=1380 SACK_PERM=1
27	5.355448	Client	banking.postbank...	TCP	54	49340 → 443	[ACK] Seq=1 Ack=1 Win=64860 Len=0
28	5.355558	Client	banking.postbank...	TLSv1.2	265		Client Hello
29	5.374299	banking.postbank.de	Client	TLSv1.2	1434		Server Hello
30	5.374313	banking.postbank.de	Client	TCP	1434		[TCP segment of a reassembled PDU]
31	5.374336	Client	banking.postbank...	TCP	54	49340 → 443	[ACK] Seq=212 Ack=2761 Win=64860 Len=0
32	5.374375	banking.postbank.de	Client	TCP	1434		[TCP segment of a reassembled PDU]

▶ Frame 28: 265 bytes on wire (2120 bits), 265 bytes captured (2120 bits) on interface 0
▶ Ethernet II, Src: Client (34:17:eb:e1:04:a0), Dst: Procurve_09:8b:00 (f0:62:81:09:8b:00)
▶ Internet Protocol Version 4, Src: Client (10.84.56.111), Dst: banking.postbank.de (195.50.155.66)
▶ Transmission Control Protocol, Src Port: 49340, Dst Port: 443, Seq: 1, Ack: 1, Len: 211
▲ Secure Sockets Layer
 ▲ TLSv1.2 Record Layer: Handshake Protocol: Client Hello
 Content Type: Handshake (22)
 Version: TLS 1.0 (0x0301)
 Length: 206
 ▲ Handshake Protocol: Client Hello
 Handshake Type: Client Hello (1)
 Length: 202
 Version: TLS 1.2 (0x0303)
 ▲ Random
 GMT Unix Time: Apr 23, 1981 21:58:51.000000000 Mitteleuropäische Sommerzeit
 Random Bytes: 6787083866415558b5633b07e039d205b5d4afe4a2f5b07e...
 Session ID Length: 0
 Cipher Suites Length: 22
 ▶ Cipher Suites (11 suites)
 Compression Methods Length: 1
 ▶ Compression Methods (1 method)
 Extensions Length: 139
 ▶ Extension: server_name
 ▶ Extension: renegotiation_info
 ▶ Extension: elliptic_curves
 ▶ Extension: ec_point_formats
 ▶ Extension: SessionTicket TLS
 ▶ Extension: next_protocol_negotiation
 ▶ Extension: Application Layer Protocol Negotiation
 ▶ Extension: status_request
 ▶ Extension: signature_algorithms

Zeitstempel + Random (4 + 28 byte)

Session-ID: falls Wiederaufnahme

Cipher Suites Length: 22

▲ Cipher Suites (11 suites)

- Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 (0xc02b)
- Cipher Suite: TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (0xc02f)
- Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA (0xc00a)
- Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA (0xc009)
- Cipher Suite: TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (0xc013)
- Cipher Suite: TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xc014)
- Cipher Suite: TLS_DHE_RSA_WITH_AES_128_CBC_SHA (0x0033)
- Cipher Suite: TLS_DHE_RSA_WITH_AES_256_CBC_SHA (0x0039)
- Cipher Suite: TLS_RSA_WITH_AES_128_CBC_SHA (0x002f)
- Cipher Suite: TLS_RSA_WITH_AES_256_CBC_SHA (0x0035)
- Cipher Suite: TLS_RSA_WITH_3DES_EDE_CBC_SHA (0x000a)

Compression Methods Length: 1

28	5.355558	Client	banking.postbank...	TLSv1.2	265 Client Hello
29	5.374299	banking.postbank.de	Client	TLSv1.2	1434 Server Hello
30	5.374313	banking.postbank.de	Client	TCP	1434 [TCP segment of a reassembled PDU]
31	5.374336	Client	banking.postbank...	TCP	54 49340 → 443 [ACK] Seq=212 Ack=2761 Win=64860 Len=0
32	5.374375	banking.postbank.de	Client	TCP	1434 [TCP segment of a reassembled PDU]

▶ Frame 29: 1434 bytes on wire (11472 bits), 1434 bytes captured (11472 bits) on interface 0

▶ Ethernet II, Src: Procurve_09:8b:00 (f0:62:81:09:8b:00), Dst: Client (34:17:eb:e1:04:a0)

▶ Internet Protocol Version 4, Src: banking.postbank.de (195.50.155.66), Dst: Client (10.84.56.111)

▶ Transmission Control Protocol, Src Port: 443, Dst Port: 49340, Seq: 1, Ack: 212, Len: 1380

▲ Secure Sockets Layer

- ▲ TLSv1.2 Record Layer: Handshake Protocol: Server Hello
 - Content Type: Handshake (22)
 - Version: TLS 1.2 (0x0303)
 - Length: 85
- ▲ Handshake Protocol: Server Hello
 - Handshake Type: Server Hello (2)
 - Length: 81
 - Version: TLS 1.2 (0x0303)
 - ▲ Random
 - GMT Unix Time: Apr 29, 2085 08:45:17.000000000 Mitteleuropäische Sommerzeit
 - Random Bytes: 4a97cf6a58cf25e87c4ac7ed721a7fa528ba6b054f4ecddb...
 - Session ID Length: 32
 - Session ID: 6d054e2da6db7b75278cf0aaf61d61c8c0f71e8852c92e75...
 - Cipher Suite: TLS_RSA_WITH_AES_256_CBC_SHA (0x0035)
 - Compression Method: null (0)
 - Extensions Length: 9
 - ▶ Extension: renegotiation_info
 - ▶ Extension: server_name

Zeitstempel + Random (4 + 28 byte)

Session-ID: zur Wiederaufnahme

Ausgewählte Cipher-Suite

Aufbau einer SSL-Verbindung – 3. Certificate



28	5.355558	Client	banking.postbank...	TLSv1.2	265	Client Hello
29	5.374299	banking.postbank.de	Client	TLSv1.2	1434	Server Hello
30	5.374313	banking.postbank.de	Client	TCP	1434	[TCP segment of a reassembled PDU]
31	5.374336	Client	banking.postbank...	TCP	54	49340 → 443 [ACK] Seq=212 Ack=2761 Win=64860 Len=0
32	5.374375	banking.postbank.de	Client	TCP	1434	[TCP segment of a reassembled PDU]
33	5.392592	banking.postbank.de	Client	TLSv1.2	424	Certificate, Server Hello Done

▶	Frame 33: 424 bytes on wire (3392 bits), 424 bytes captured (3392 bits) on interface 0
▶	Ethernet II, Src: Procurve_09:8b:00 (f0:62:81:09:8b:00), Dst: Client (34:17:eb:e1:04:a0)
▶	Internet Protocol Version 4, Src: banking.postbank.de (195.50.155.66), Dst: Client (10.84.56.111)
▶	Transmission Control Protocol, Src Port: 443, Dst Port: 49340, Seq: 4141, Ack: 212, Len: 370
▶	[4 Reassembled TCP Segments (4411 bytes): #29(1290), #30(1380), #32(1380), #33(361)]
▲	Secure Sockets Layer
▲	TLSv1.2 Record Layer: Handshake Protocol: Certificate
	Content Type: Handshake (22)
	Version: TLS 1.2 (0x0303)
	Length: 4406
▲	Handshake Protocol: Certificate
	Handshake Type: Certificate (11)
	Length: 4402
	Certificates Length: 4399
▲	Certificates (4399 bytes)
	Certificate Length: 1827
▶	Certificate: 3082071f30820607a00302010202104219149bbda1430c9d... (id-at-commonName=banking.postbank.de,id-at-organizationalUnitNa
	Certificate Length: 1327
▶	Certificate: 3082052b30820413a00302010202107ee14a6f6feff2d37f... (id-at-commonName=Symantec Class 3 EV SSL CA - G3,id-at-organiza
	Certificate Length: 1236
▶	Certificate: 308204d030820439a0030201020210250ce8e030612e9f2b... (id-at-commonName=VeriSign Class 3 Public Primary Certification
▲	Secure Sockets Layer
▶	TLSv1.2 Record Layer: Handshake Protocol: Server Hello Done

Abschluss der Hello-Phase vom Server

33	5.392592	banking.postbank.de	10.84.56.111	TLSv1.2	424	Certificate, Server Hello Done
34	5.392606	10.84.56.111	banking.postbank.de	TCP	54	49340 → 443 [ACK] Seq=212 Ack=4511 Win=64860 Len=0
35	5.393540	10.84.56.111	banking.postbank.de	TLSv1.2	396	Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
36	5.394159	10.84.56.111	10.50.1.2	DNS	72	Standard query 0xe1b6 A s2.symcb.com
37	5.412561	banking.postbank.de	10.84.56.111	TCP	60	443 → 49340 [ACK] Seq=4511 Ack=554 Win=4693 Len=0
38	5.413520	banking.postbank.de	10.84.56.111	TLSv1.2	60	Change Cipher Spec
39	5.413723	banking.postbank.de	10.84.56.111	TLSv1.2	123	Encrypted Handshake Message
40	5.413729	10.84.56.111	banking.postbank.de	TCP	54	49340 → 443 [ACK] Seq=554 Ack=4586 Win=64785 Len=0

▷ Frame 35: 396 bytes on wire (3168 bits), 396 bytes captured (3168 bits) on interface 0
▷ Ethernet II, Src: Dell_e1:04:a0 (34:17:eb:e1:04:a0), Dst: Procurve_09:8b:00 (f0:62:81:09:8b:00)
▷ Internet Protocol Version 4, Src: 10.84.56.111 (10.84.56.111), Dst: banking.postbank.de (195.50.155.66)
▷ Transmission Control Protocol, Src Port: 49340, Dst Port: 443, Seq: 212, Ack: 4511, Len: 342
▲ Secure Sockets Layer
 ▲ TLSv1.2 Record Layer: Handshake Protocol: Client Key Exchange
 Content Type: Handshake (22)
 Version: TLS 1.2 (0x0303)
 Length: 262
 ▲ Handshake Protocol: Client Key Exchange
 Handshake Type: Client Key Exchange (16)
 Length: 258
 ▲ RSA Encrypted PreMaster Secret
 Encrypted PreMaster length: 256
 Encrypted PreMaster: 0efbc258fd23e5b4f46df70b11558ca1fd5a34001ed15a58...
 ▷ TLSv1.2 Record Layer: Change Cipher Spec Protocol: Change Cipher Spec
 ▷ TLSv1.2 Record Layer: Handshake Protocol: Encrypted Handshake Message

- Übertragung des zufällig gewählte Pre-Master-Secrets (256 bit) im Client Key Exchange Record zum Server
- Der Wert ist mit den öffentlichen Schlüssel des Servers verschlüsselt.
- Change Cipher Spec schaltet die Nutzung frei, der Handshake bestätigt das.



Der Client übermittelt mit der Client Key Exchange-Nachricht dem Server eine geheime Basisinformation, das 48-Byte (384 Bits) Pre-Master Secret. 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, PublicS).

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.

Quelle: Claudia Eckert: IT-Sicherheit

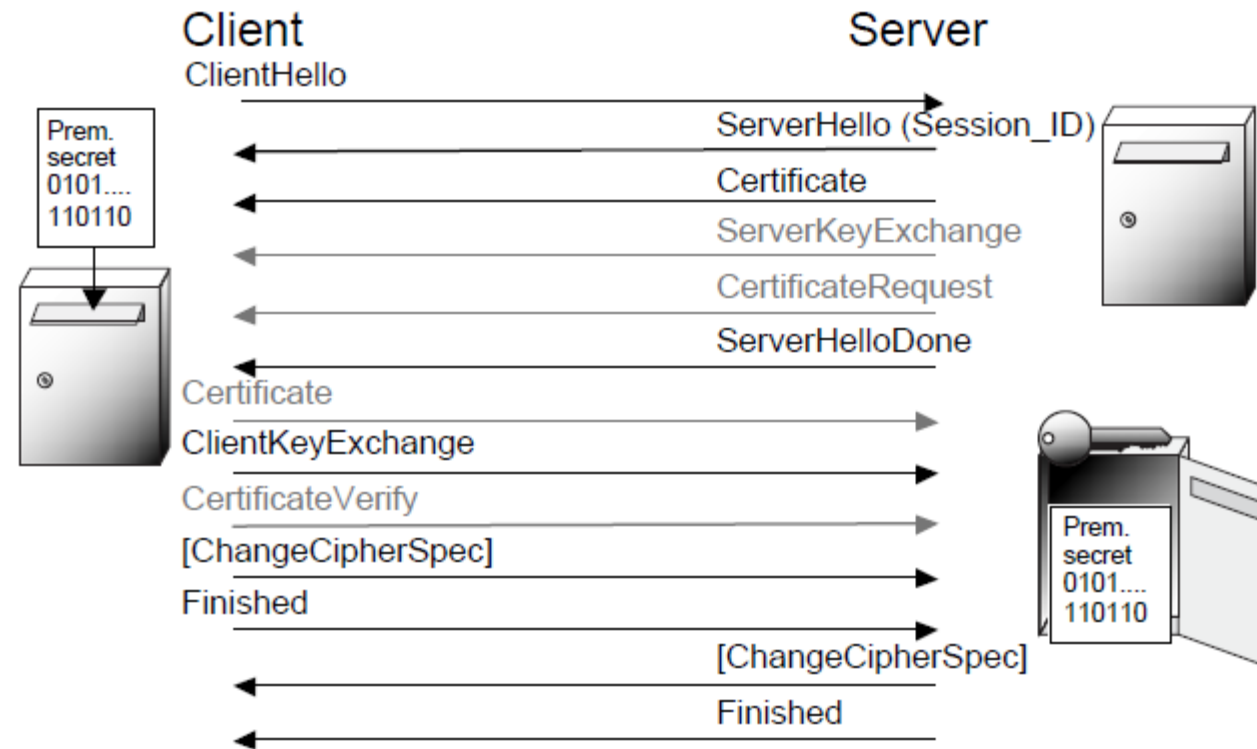


Abb. 7.10 SSL Handshake mit RSA-basiertem Schlüsseltransport. (Die grau dargestellten Nachrichten ServerKeyExchange, CertificateRequest, Certificate und CertificateVerify sind optional.)



Nach diesem Handshake sind folgende Informationen sowohl dem Client als auch dem Server bekannt:

- Eine **Session ID** als Name der aktuellen SSL-Verbindung.
- Eine **CipherSuite** mit jeweils einem
 - **Public Key-Algorithmus** zur Übertragung/ Aushandlung des Premaster Secret,
 - einem **symmetrischen Verschlüsselungsalgorithmus**
 - und einem **Hashalgorithmus**
 - Ein gemeinsames **Master Secret**.
- Je ein Verschlüsselungsschlüssel für den Datenverkehr von Client zum Server und umgekehrt.
- Falls erforderlich zwei Initialisierungsvektoren für die Verschlüsselungsfunktion, einen für jede Richtung.
- Für jede Übertragungsrichtung ein Schlüssel zur Bildung des MAC von Nachrichten.
- Optional eine Kompressionsfunktion.

SSL: drei Phasen

2. Schlüsselableitung:

- Alice und Bob verwenden das Master Secret,

um vier Schlüssel zu erzeugen:

- E_B : Schlüssel für Verschlüsselung Bob->Alice
- E_A : Schlüssel für Verschlüsselung Alice->Bob
- M_B : MAC-Schlüssel Bob->Alice
- M_A : MAC-Schlüssel Alice->Bob

- Verschlüsselungs- und MAC-Algorithmen (Ciphersuite)

können zwischen Bob und Alice ausgehandelt werden

- Mehr zur Sicherheit von SSL / TLS:

- Jörg Schwenk: **Sicherheit und Kryptographie im Internet**

Theorie und Praxis, 4., überarbeitete und erweiterte Auflage, Online an der thi!

- Spitz, Stephan ; Pramateftakis, Michael ; Swoboda, Joachim:

Kryptographie und IT-Sicherheit : Grundlagen und Anwendungen

2., überarb. Aufl., Online an der thi!

RSA: Erzeugen der Schlüssel aus dem Pre-Master Secret

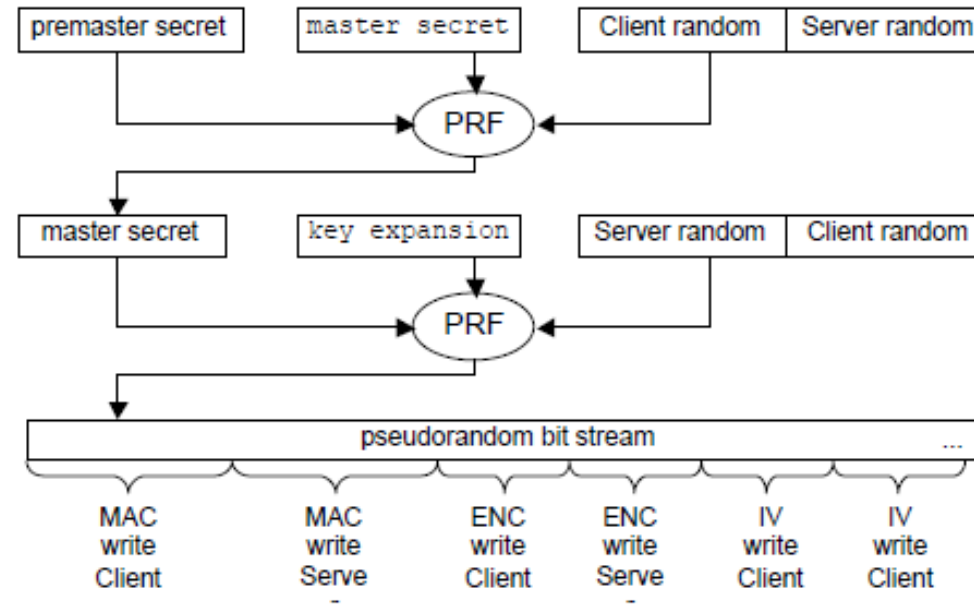
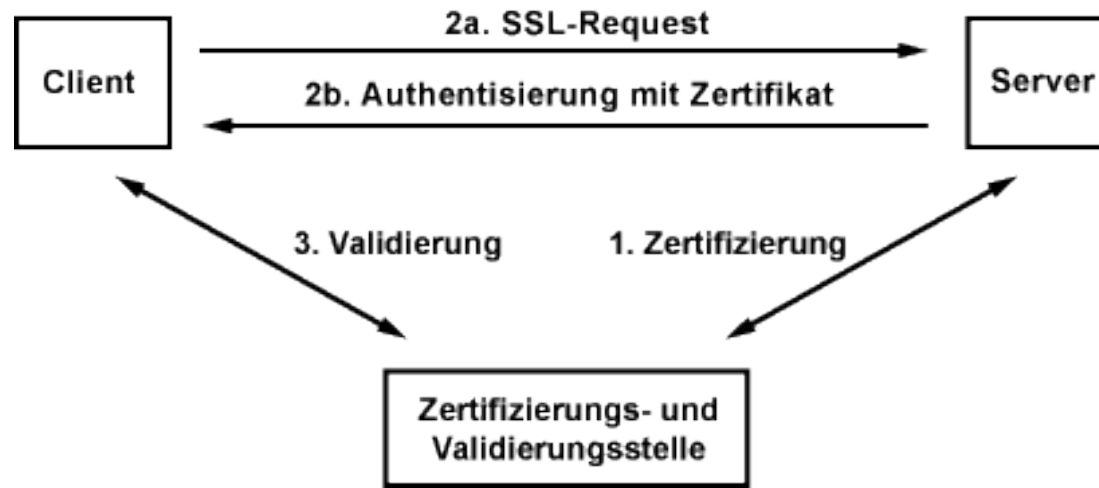


Abb. 7.23 Zweistufige Ableitung des Schlüsselmaterials aus dem Premaster Secret mit Hilfe der Pseudozufallsfunktion PRF.

- Zunächst wird aus dem Premaster Secret, der ASCII-Zeichenfolge "master secret" und den Client Random- und Server Random-Werten (in dieser Reihenfolge) mit der PRF eine Bitfolge von 48 Byte Länge erzeugt: Diese Bitfolge ist das Master Secret.
- Dann dienen das Master Secret, die ASCII-Zeichenfolge "key expansion" sowie die Server Random- und Client-Random-Werte als Eingabe für den zweiten Durchlauf der PRF, bei dem genug Bits erzeugt werden müssen, um zwei MAC-Schlüssel, zwei Verschlüsselungsschlüssel und ggf. zwei Initialisierungsvektoren daraus bilden zu können. Ergebnis:
 - Verschlüsselungsschlüssel *SessionKey*
 - Schlüssel zu Generierung von MACs (Message Authentication Codes) zur Integritätssicherung
 - Optional Initialisierungsvektoren (IV) für die Initialisierung der symmetrischen Verschlüsselung

- **SSL / TLS benutzt Schlüsselpaar aus Public Key und Private Key (asymmetrisches Verschlüsselungsverfahren)**
- **Authentisierung des öffentlichen Schlüssels durch ein Zertifikat.**
- **Zertifikat des Server-Betreiber und Domain-Inhaber enthält:**
 - Domainname
 - Gültigkeit / Ablaufdatum des öffentlichen Schlüssels
 - Instanz welche die Vertrauenswürdigkeit bestätigt hat
- **Durch Zertifikat authentisiert sich der Empfänger gegenüber dem Sender, bzw. der Server gegenüber dem Client.**
- **Client kann**
 - das Zertifikat überprüfen (Validierung)
 - und somit die Vertrauenswürdigkeit feststellen (Authentizität).



Bestandteil von SSL:

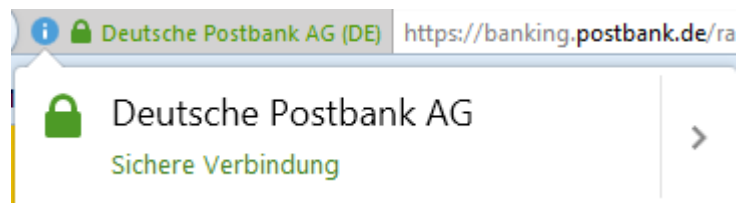
1. Zertifizierung des öffentlichen Schlüssels
2. Authentifizierung des Servers
3. Validierung des übermittelten Zertifikats

anschließende verschlüsselte Übertragung von Daten zwischen Sender und Empfänger

Authentifizierung mit Zertifikaten um Identitäten zu authentifizieren

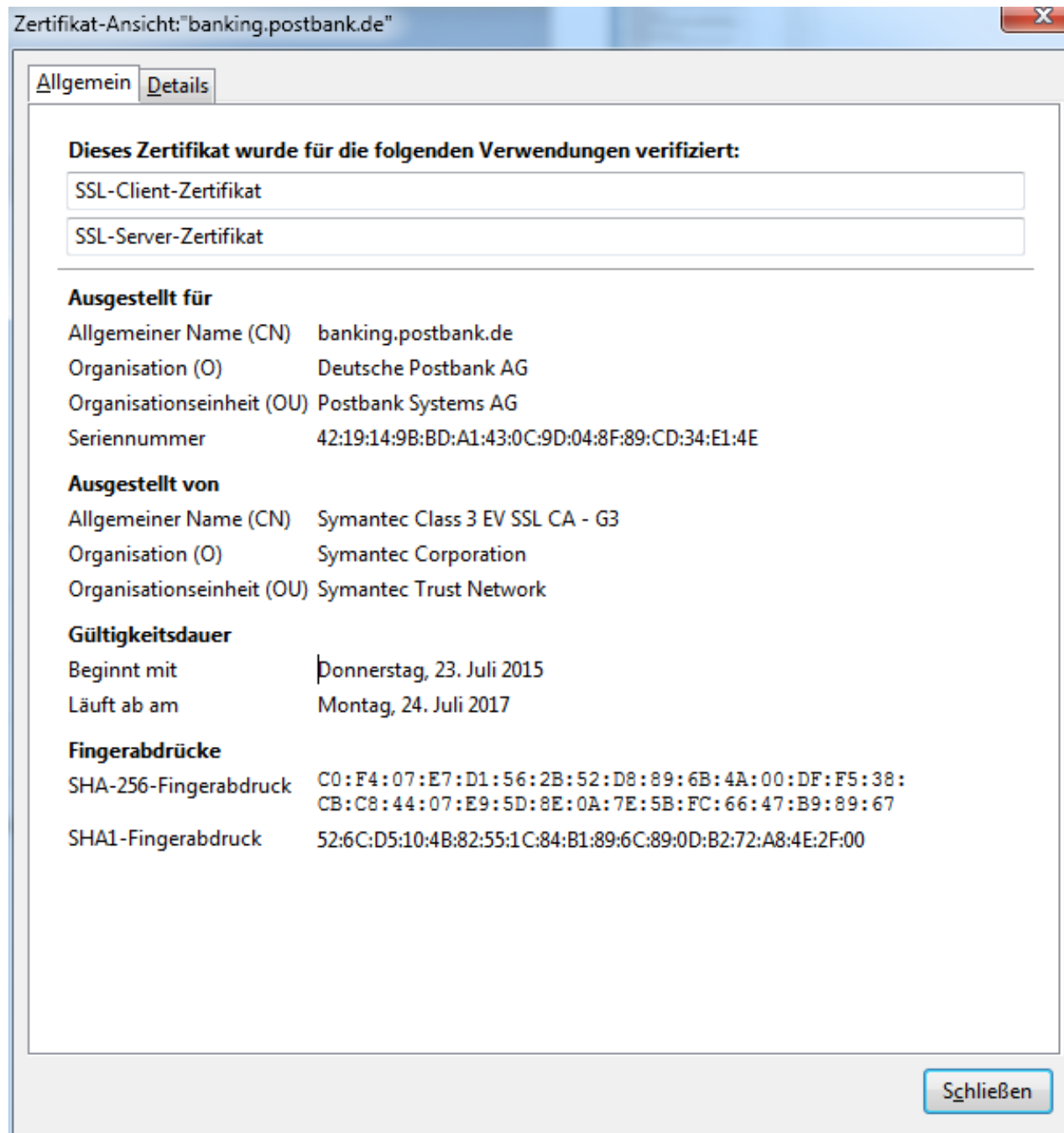
- **SSL bzw. TLS arbeitet mit PKIX-Zertifikaten bzw. mit einer Public Key Infrastructure nach X.509v3**
- **Zertifikate koppeln eine Identität an einen öffentlichen Schlüssel, der zur Authentisierung und Verschlüsselung verwendet wird.**
- **3 Zertifikatstypen:**
 - Domain-Validated-Zertifikat (DV-SSL) (preiswert, Vertrauenswürdigkeit nicht garantiert)
 - Organisation-Validation-Zertifikat (OV-SSL)
 - Extended-Validation-Zertifikat (EV-SSL) (erheblichen Prüfaufwand (teuer), höhere Vertrauenswürdigkeit)
- **Qualität der Zertifikate ist dem Nutzer nicht leicht ersichtlich**

CA für Postbank: Symantec



CA für Moodle: THI-Rechenzentrum





- **CA: Certificate Authority oder Certification Authority: ca. 700 weltweit.**
- **Ablauf:**
 - Unternehmen / Organisation lässt sich von einer Certificate Authority nach einer Überprüfung ein digitales Zertifikat ausstellen.
 - Zertifikat wird z.B. auf einem Webserver hinterlegt.
 - Mit diesem Zertifikat weist sich die Webseite gegenüber den zugreifenden Browsern als Eigentümer aus.
 - Der Browser des Besuchers überprüft die Angaben im Zertifikat und fragt bei Bedarf bei der ausstellenden Zertifizierungsstelle nach, ob das Zertifikat gültig ist.
- **Root- / Stamm-Zertifikat**
 - Auch die Zertifizierungsstelle besitzt ein Zertifikat, in dem sich deren öffentlicher Schlüssel befindet.
 - Dabei handelt es sich um ein Wurzel- bzw. Stammzertifikat, das in Browsern und Betriebssystemen hinterlegt ist.
 - Diesen Stammzertifikaten wird in der Regel bedingungslos vertraut.
 - Anhand der Signatur der Zertifizierungsstelle und dem Stammzertifikat kann ein Browser feststellen, ob das Zertifikat einer Domain wirklich von der angegebenen Zertifizierungsstelle ausgestellt wurde.

[Quelle: elektronik-kompodium.de]



- Zertifikat von einem Server: Überprüfen der Echtheit (Stammt das Zertifikat vom diesem Server?)
- Mit der Validierung eines Zertifikats wird die Identität bestätigt, ohne dass die beteiligten Kommunikationspartner vorab Authentifizierungsinformationen, wie zum Beispiel Schlüssel, austauschen müssen
- Protokoll zur Validierung:

OCSP: Online Certificate Status Protocol

- Benutzt HTTP

Validierung der Zertifikate: Verbindung zum OCSP-Server aufbauen



36	5.394159	Client	10.50.1.2	DNS	72 Standard query 0xe1b6 A s2.symcb.com
37	5.412561	banking.postbank.de	Client	TCP	60 443 → 49340 [ACK] Seq=4511 Ack=554 Win=4693 Len=0
38	5.413520	banking.postbank.de	Client	TLSv1.2	60 Change Cipher Spec
39	5.413723	banking.postbank.de	Client	TLSv1.2	123 Encrypted Handshake Message
40	5.413729	Client	banking.postbank...	TCP	54 49340 → 443 [ACK] Seq=554 Ack=4586 Win=64785 Len=0
41	5.422490	10.50.1.2	Client	DNS	174 Standard query response 0xe1b6 A s2.symcb.com CNAME ocsdp-ds.ws.symantec.com
42	5.422789	Client	e8218.dscb1.akamaie...	TCP	66 49341 → 80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
43	5.439613	e8218.dscb1.akamaie...	Client	TCP	66 80 → 49341 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1380 SACK_PERM=1
44	5.439668	Client	e8218.dscb1.akamaie...	TCP	54 49341 → 80 [ACK] Seq=1 Ack=1 Win=66048 Len=0
45	5.439765	Client	e8218.dscb1.akamaie...	OCSP	493 Request
46	5.456421	e8218.dscb1.akamaie...	Client	TCP	60 80 → 49341 [ACK] Seq=1 Ack=440 Win=30272 Len=0
47	5.456838	e8218.dscb1.akamaie...	Client	TCP	1434 [TCP segment of a reassembled PDU]
48	5.456867	e8218.dscb1.akamaie...	Client	OCSP	790 Response
49	5.456876	Client	e8218.dscb1.akamaie...	TCP	54 49341 → 80 [ACK] Seq=440 Ack=2117 Win=66048 Len=0

▶ Frame 41: 174 bytes on wire (1392 bits), 174 bytes captured (1392 bits) on interface 0

▶ Ethernet II, Src: Procurve_09:8b:00 (f0:62:81:09:8b:00), Dst: Client (34:17:eb:e1:04:a0)

▶ Internet Protocol Version 4, Src: 10.50.1.2 (10.50.1.2), Dst: Client (10.84.56.111)

▶ User Datagram Protocol, Src Port: 53, Dst Port: 53834

▲ Domain Name System (response)

 [Request In: 36]

 [Time: 0.028331000 seconds]

 Transaction ID: 0xe1b6

 ▶ Flags: 0x8180 Standard query response, No error

 Questions: 1

 Answer RRs: 3

 Authority RRs: 0

 Additional RRs: 0

 ▲ Queries

 ▶ s2.symcb.com: type A, class IN

 ▲ Answers

 ▶ s2.symcb.com: type CNAME, class IN, cname ocsdp-ds.ws.symantec.com.edgekey.net

 ▶ ocsdp-ds.ws.symantec.com.edgekey.net: type CNAME, class IN, cname e8218.dscb1.akamaiedge.net

 ▶ e8218.dscb1.akamaiedge.net: type A, class IN, addr 23.37.43.27

41	5.422490	10.50.1.2	Client	DNS	174 Standard query response 0xe1b6 A s2.symcb.com CNAME ocsp-ds.ws.symantec.com.e
42	5.422789	Client	e8218.dscb1.akam...	TCP	66 49341 → 80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
43	5.439613	e8218.dscb1.akamaie...	Client	TCP	66 80 → 49341 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1380 SACK_PERM=1 WS=32
44	5.439668	Client	e8218.dscb1.akam...	TCP	54 49341 → 80 [ACK] Seq=1 Ack=1 Win=66048 Len=0
45	5.439765	Client	e8218.dscb1.akam...	OCSP	493 Request
46	5.456421	e8218.dscb1.akamaie...	Client	TCP	60 80 → 49341 [ACK] Seq=1 Ack=440 Win=30272 Len=0
47	5.456838	e8218.dscb1.akamaie...	Client	TCP	1434 [TCP segment of a reassembled PDU]
48	5.456867	e8218.dscb1.akamaie...	Client	OCSP	790 Response
49	5.456876	Client	e8218.dscb1.akam...	TCP	54 49341 → 80 [ACK] Seq=440 Ack=2117 Win=66048 Len=0

▷	Frame 45: 493 bytes on wire (3944 bits), 493 bytes captured (3944 bits) on interface 0
▷	Ethernet II, Src: Client (34:17:eb:e1:04:a0), Dst: Procurve_09:8b:00 (f0:62:81:09:8b:00)
▷	Internet Protocol Version 4, Src: Client (10.84.56.111), Dst: e8218.dscb1.akamaiedge.net (23.37.43.27)
▷	Transmission Control Protocol, Src Port: 49341, Dst Port: 80, Seq: 1, Ack: 1, Len: 439
▷	Hypertext Transfer Protocol
▲	Online Certificate Status Protocol
▲	tbsRequest
▲	requestList: 1 item
▲	Request
▲	reqCert
▲	hashAlgorithm (SHA-1)
	Algorithm Id: 1.3.14.3.2.26 (SHA-1)
	issuerNameHash: b9e9b287028503f8eca5fb42e13e0f49c72426e2
	issuerKeyHash: 7fd365a7c2dddecbbf03009f34339fa02af333133
	serialNumber: 0x7ee14a6f6feff2d37f3fad654d3adab4

45	5.439765	Client	e8218.dscb1.akam...	OCSP	493 Request
46	5.456421	e8218.dscb1.akamaie...	Client	TCP	60 80 → 49341 [ACK] Seq=1 Ack=440 Win=30272 Len=0
47	5.456838	e8218.dscb1.akamaie...	Client	TCP	1434 [TCP segment of a reassembled PDU]
48	5.456867	e8218.dscb1.akamaie...	Client	OCSP	790 Response
49	5.456876	Client	e8218.dscb1.akam...	TCP	54 49341 → 80 [ACK] Seq=440 Ack=2117 Win=66048 Len=0

▶ Frame 48: 790 bytes on wire (6320 bits), 790 bytes captured (6320 bits) on interface 0

▶ Ethernet II, Src: Procurve_09:8b:00 (f0:62:81:09:8b:00), Dst: Client (34:17:eb:e1:04:a0)

▶ Internet Protocol Version 4, Src: e8218.dscb1.akamaiedge.net (23.37.43.27), Dst: Client (10.84.56.111)

▶ Transmission Control Protocol, Src Port: 80, Dst Port: 49341, Seq: 1381, Ack: 440, Len: 736

▶ [2 Reassembled TCP Segments (2116 bytes): #47(1380), #48(736)]

▶ Hypertext Transfer Protocol

▲ Online Certificate Status Protocol

responseStatus: successful (0)

▲ responseBytes

ResponseType Id: 1.3.6.1.5.5.7.48.1.1 (id-pkix-ocsp-basic)

▲ BasicOCSPResponse

▶ tbsResponseData

▶ signatureAlgorithm (sha1WithRSAEncryption)

Padding: 0

signature: 33a6d87f59e3d3094a3b3431f9874437400b8c237ba0a015...

▲ certs: 1 item

▲ Certificate (id-at-commonName=Symantec Class 3 PCA - G5 OCSP Responder Certi,id-at-organizationalUnitName=Symantec)

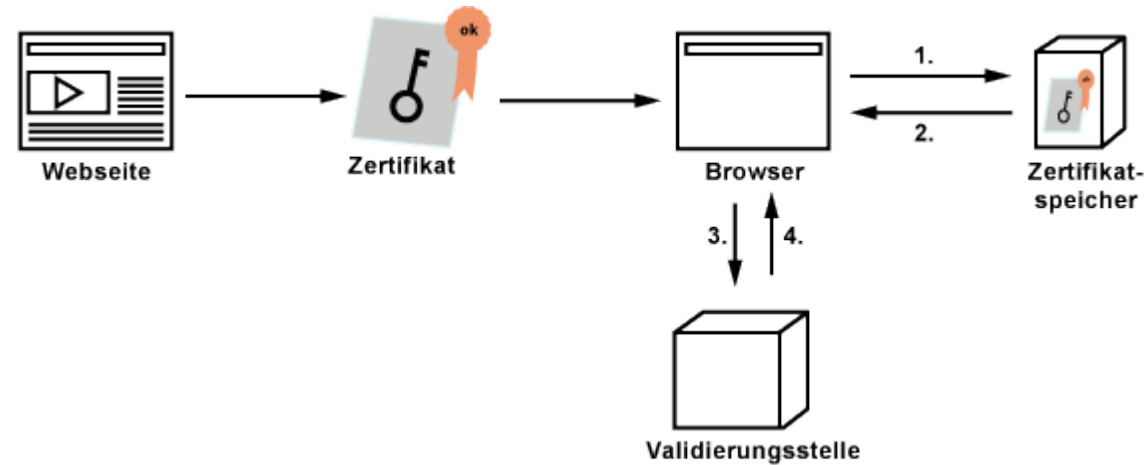
▶ signedCertificate

▶ algorithmIdentifier (sha1WithRSAEncryption)

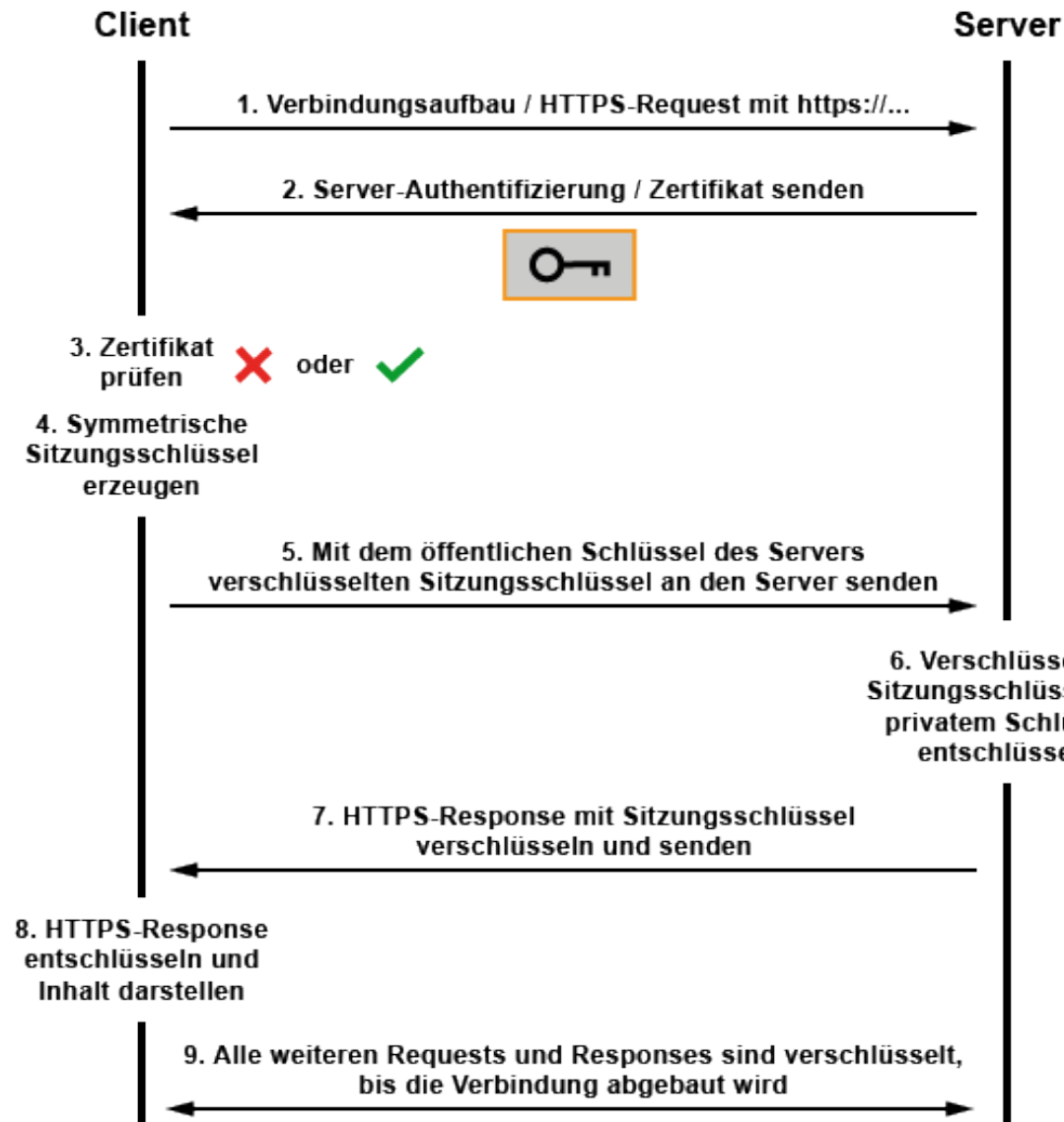
Padding: 0

encrypted: 9bf889e6714c1c8faa9bb05206c4b5890cedd175a1254deb...

Validierung eines Zertifikats am Bsp. HTTPS



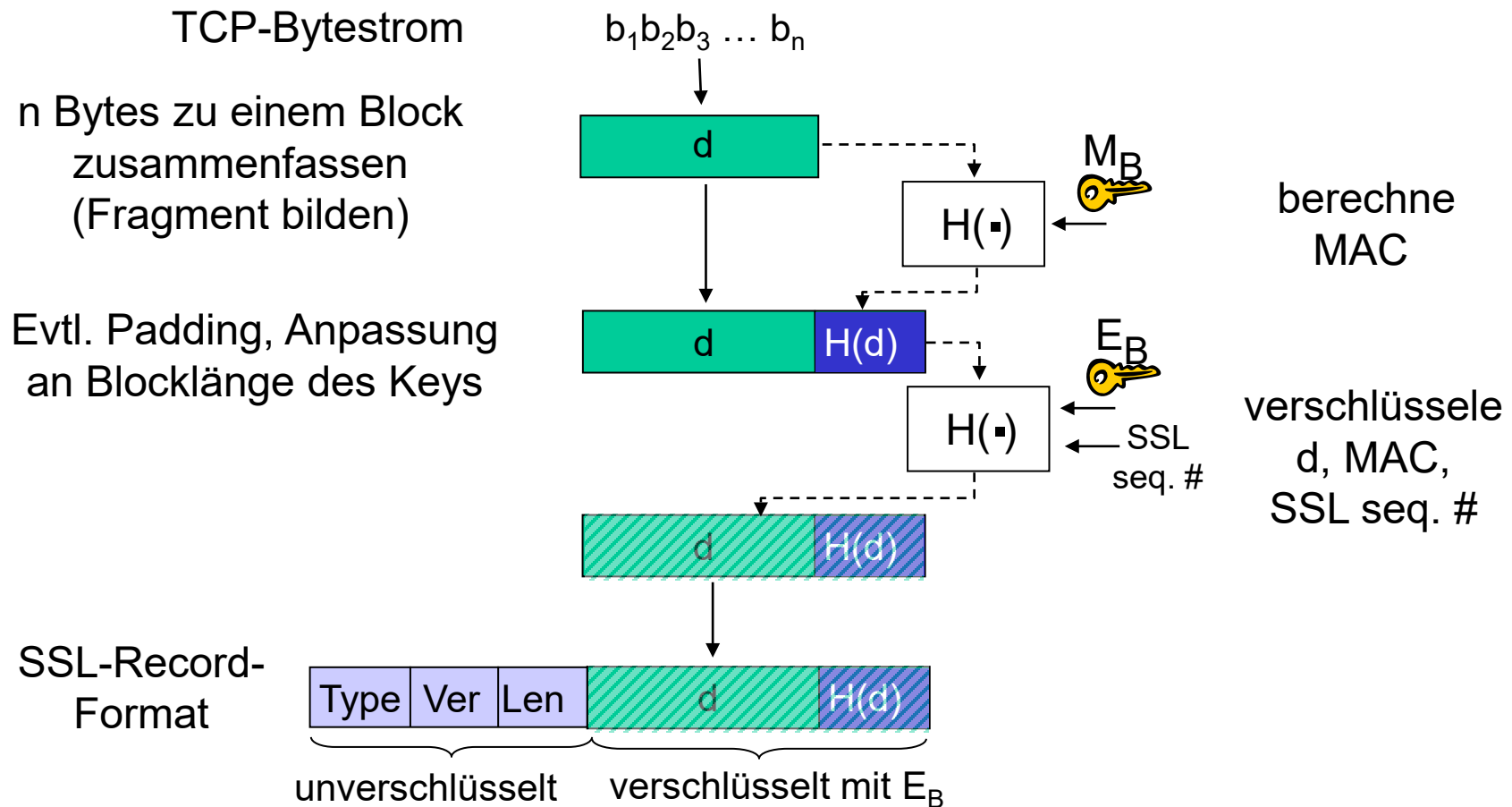
- Browser erhält vom Webserver ein Zertifikat ➔ Validierung.
- Browser prüft zuerst, ob er dem Aussteller des Zertifikats, der Zertifizierungsstelle oder Certificate Authority (CA), vertraut. Dazu muss das entsprechende Wurzelzertifikat der Zertifizierungsstelle im Browser hinterlegt sein (1. und 2.). Der Browser hat dazu eine Liste mit Zertifizierungsstellen, denen er per Default vertraut.
- Im zweiten Schritt kontaktiert der Browser die angegebene Validierungsstelle bzw. Zertifizierungsstelle (3. und 4.). Diese prüft, ob das Zertifikat gültig ist und meldet das Ergebnis an den Browser zurück.
- Sofern das Zertifikat bereits bekannt ist, ist eine Validierung über die Zertifizierungsstelle nicht mehr zwingend erforderlich.



4. Erkennt der Client das Zertifikat als gültig erzeugt der Client sym-metrische Sitzungsschlüssel.
5. Mit dem öffentlichen Schlüssel des Servers verschlüsselt der Client den Sitzungsschlüssel und schickt ihn an den Server. Vorher haben sich Client und Server auf ein gemeinsames Verschlüsselungsverfahren geeinigt.
6. Mit seinem privaten Schlüssel kann der Server den verschlüsselten Sitzungsschlüssel entschlüsseln.
7. Anschließend verschlüsselt der Server den HTTPS-Response mit dem Sitzungsschlüssel.
8. Der Client entschlüsselt den HTTPS-Response und stellt den Inhalt dar.
9. Danach werden alle Requests und Responses verschlüsselt, bis die Verbindung abgebaut wird.

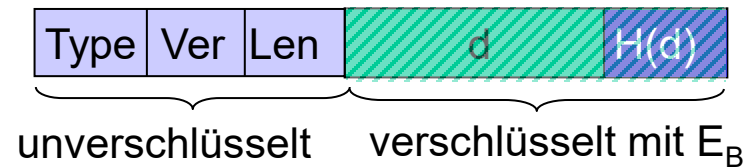
3. Datentransfer

„MAC-then-PAD-then-Encrypt“-Verfahren;



3. Datentransfer

SSL-Record-
Format



MAC: Message Authentication Code

Kompression optional möglich, wird oft nicht eingesetzt

Type (1 Byte): Hier wird der Typ der übertragenen Nachricht beschrieben, wobei nur zwischen den SSL-Typen ChangeCipherSpec (Type=20), Alert (Type=21) und Handshake (Type=22) auf der einen, und allen anderen Anwendungsdaten wie z.B. HTTP oder FTP (Type=23) auf der andern Seite unterschieden wird.

Version (2 Byte): Das erste Byte gibt die Hauptversion, das zweite die Unterversion an. In Gebrauch sind die Werte 3.0 (SSL 3.0), 3.1 (TLS 1.0) und 3.2 (TLS 1.1).

Länge (2 Byte): Hier wird die Länge des nachfolgenden Datenblocks in Bytes angegeben.

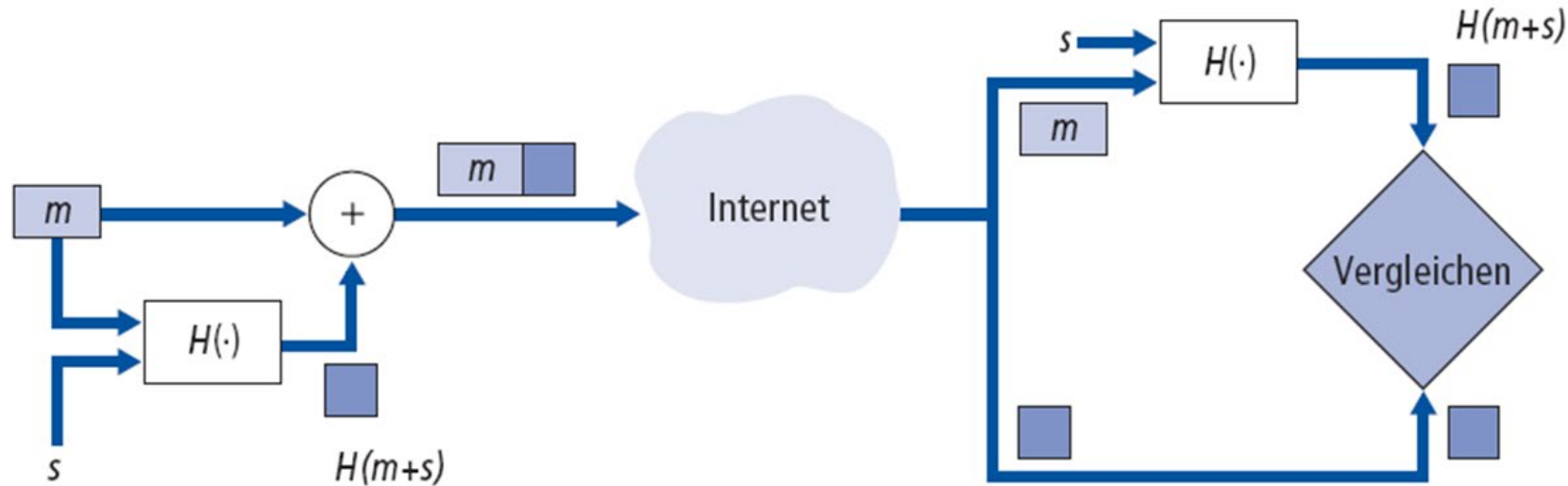
Bob empfängt eine Nachricht von Alice und möchte sicherstellen, dass

- die Nachricht tatsächlich von Alice stammt
- die Nachricht seit dem Versand durch Alice nicht verändert wurde

Kryptographische Hashfunktion:

- nimmt Eingabe m , erstellt Hash $H(m)$ fester Länge
 - wie z.B. die Internet-Prüfsumme
- rechnerisch nicht möglich, zwei unterschiedliche Nachrichten x, y zu finden, für die $H(x) = H(y)$
 - äquivalent: gegeben $m = H(x)$, (x unbekannt), ist es nicht möglich, x zu bestimmen.
 - Anmerkung: Die Internet-Prüfsumme erfüllt diese Bedingung *nicht*!

Message Authentication Code (MAC)



Legende:

m = Nachricht

s = gemeinsames Geheimnis

S = Authentifizierungsschlüssel

- wird mit der Nachricht verkettet $\Rightarrow H(m+s) = \text{MAC}$
- Stellt sicher, dass die Nachricht vom richtigen Absender kommt

Ein Message Authentication Code (MAC) ist eine kryptographische Prüfsumme, in die neben dem Datensatz auch noch der geheime (symmetrische) Schlüssel von Sender und Empfänger einfließt. Ein MAC kann daher im Gegensatz zu einem Hashwert nur von Sender oder Empfänger berechnet und auch nur von diesen beiden verifiziert werden.

■ weit verbreitet ist MD5 (RFC 1321)

- berechnet einen 128-Bit-MAC in einem 4-stufigen Prozess
- gegeben einen 128-Bit-String x, erscheint es schwierig, eine Nachricht m zu konstruieren, deren MD5-Hash x ist
 - in der Vergangenheit (2005) gab es Ansätze für Angriffe auf MD5

■ SHA-1 wird auch oft verwendet

- US-Standard [NIST, FIPS PUB 180-1]
- 160-Bit-MAC
- Seit 2005 sind auch Angriffe auf SHA-1 bekannt

Bsp.: SSL Application Data



68	5.553948	banking.postbank.de	10.84.56.111	TCP	706 [TCP segment of a reassembled PDU]
69	5.554108	banking.postbank.de	10.84.56.111	TCP	1434 [TCP segment of a reassembled PDU]
70	5.554134	10.84.56.111	banking.postbank.de	TCP	54 49340 → 443 [ACK] Seq=1268 Ack=7204 Win=64860 Len=0
71	5.554211	banking.postbank.de	10.84.56.111	TCP	1434 [TCP segment of a reassembled PDU]
72	5.554219	banking.postbank.de	10.84.56.111	TCP	1434 [TCP segment of a reassembled PDU]
73	5.554225	10.84.56.111	banking.postbank.de	TCP	54 49340 → 443 [ACK] Seq=1268 Ack=9964 Win=64860 Len=0
74	5.554301	banking.postbank.de	10.84.56.111	TCP	1434 [TCP segment of a reassembled PDU]
75	5.554378	banking.postbank.de	10.84.56.111	TCP	1434 [TCP segment of a reassembled PDU]
76	5.554385	10.84.56.111	banking.postbank.de	TCP	54 49340 → 443 [ACK] Seq=1268 Ack=12724 Win=64860 Len=0
77	5.572510	banking.postbank.de	10.84.56.111	TCP	1434 [TCP segment of a reassembled PDU]
78	5.572516	banking.postbank.de	10.84.56.111	TCP	1434 [TCP segment of a reassembled PDU]
79	5.572522	10.84.56.111	banking.postbank.de	TCP	54 49340 → 443 [ACK] Seq=1268 Ack=15484 Win=64860 Len=0
80	5.572620	banking.postbank.de	10.84.56.111	TCP	1434 [TCP segment of a reassembled PDU]
81	5.572626	banking.postbank.de	10.84.56.111	TLSv1.2	1348 Application Data, Application Data
82	5.572631	10.84.56.111	banking.postbank.de	TCP	54 49340 → 443 [ACK] Seq=1268 Ack=18158 Win=64860 Len=0
83	5.590536	10.84.56.111	banking.postbank.de	TLSv1.2	651 Application Data

▶	Frame 81: 1348 bytes on wire (10784 bits), 1348 bytes captured (10784 bits) on interface 0
▶	Ethernet II, Src: Procurve_09:8b:00 (f0:62:81:09:8b:00), Dst: Dell_e1:04:a0 (34:17:eb:e1:04:a0)
▶	Internet Protocol Version 4, Src: banking.postbank.de (195.50.155.66), Dst: 10.84.56.111 (10.84.56.111)
▶	Transmission Control Protocol, Src Port: 443, Dst Port: 49340, Seq: 16864, Ack: 1268, Len: 1294
▶	[10 Reassembled TCP Segments (12933 bytes): #68(652), #69(1380), #71(1380), #72(1380), #74(1380), #75(1380), #77(1380), #78(1380), #80(1380), #81(1241)]
▲	Secure Sockets Layer
▲	TLSv1.2 Record Layer: Application Data Protocol: http-over-tls
Content Type:	Application Data (23)
Version:	TLS 1.2 (0x0303)
Length:	12928
Encrypted Application Data:	314a03a52ca6dc5d32dcfce8227bc985b6ad57ee6ed0847f...
▲	Secure Sockets Layer
▲	TLSv1.2 Record Layer: Application Data Protocol: http-over-tls
Content Type:	Application Data (23)
Version:	TLS 1.2 (0x0303)
Length:	48
Encrypted Application Data:	cfa06f4412756e4aff528da45a0c647af9edf57bc05e4f3e...

- Auslösen der TCP-Verbindung mit FIN => möglicher Angriff (Truncation-Angriff)
- Daher wird SSL selbst ausgelöst => im Type Feld wird das Ende einer SSL-Session angegeben

Übersicht: Schwachstellen von SSL/TLS

- Certification Authority (CA) / Zertifizierungsstelle
 - Zertifizierung / Zertifikatsausstellung
 - Herausgeber-Zertifikate
 - Schlüsselerzeugung
 - Kompromittierte Zertifikate
 - Validierung / Prüfung der Zertifikate
 - Optionale Verschlüsselung
 - Schwache Verschlüsselung
 - Verschlüsselung und Integrität
 - Software-Implementierung
 - Heimlich nachinstallierte CA-Root-Zertifikate
 - Zu hohe Komplexität für den Normalnutzer
- ➔ oft sind dies Schwachstellen, die Möglichkeiten zur Umgehung der Verschlüsselung eröffnen



- Die Zertifikate werden von weltweit über 700 Zertifizierungsstellen bzw. Certification Authoritys (CA) ausgestellt. Die Aussteller sind Dienstleister im Internet, die Zertifikate ausstellen und auch überprüfen. **Die größte Schwachstelle von SSL bzw. TLS liegt bei den Zertifizierungsstellen.**
- Die Vertrauenswürdigkeit einer Certification Authority liegt im Ermessen der Software-Hersteller, die diese in ihre Listen mit aufnehmen. Als Nutzer hat man darauf meist gar keinen Einfluss, weil zum Beispiel ein typischer Browser per Default eine lange Liste "scheinbar" vertrauenswürdiger Root-CAs mitbringt.
- Doch warum vertrauen Software-Hersteller CAs? Die CAs durchlaufen einen Audit-Prozess, bei dem sie auf Sicherheit und der Registrierprozess geprüft werden. Die CA muss den Hersteller davon überzeugen, dass sie vertrauenswürdig ist. Aber im Prinzip gibt es keine echte Kontrolle. Erst wenn etwas bekannt wird, verlieren die CAs das Vertrauen.
- Umgekehrt kaufen viele Webhosting-Kunden ihre Zertifikate nicht direkt bei einer Zertifizierungsstelle, sondern bedienen sich bei Resellern oder den Angeboten von Hosting-Providern, die SSL-Verschlüsselung als Zusatzdienstleistung verkaufen. Das bedeutet, man hat nur bedingt Einfluss auf die Wahl der Zertifizierungsstelle.
- Zertifizierungsstellen sind für Angreifer oft lohnende Angriffsziele und deshalb grundsätzlich einem nicht unerheblichen Risiko ausgesetzt.

Schwachstelle: Zertifizierung / Zertifikatsausstellung

- Grundsätzlich kann jede Zertifizierungsstelle Zertifikate für "jeden beliebigen Hostnamen" ausstellen. Deshalb kann es für jeden Hostnamen von unterschiedlichen Zertifizierungsstellen mehrere Zertifikate geben.
- Problem: Wie seriös / zuverlässig sind CA?
(➔ Korruption, Einflussnahme, Unfähigkeit,...)
- Nutzer kann die Vertrauenswürdigkeit der Zertifikate nicht nachprüfen oder ob die Zertifikate richtig ausgestellt wurden.
- Manipulation und Missbrauch sind möglich
- **Schwachstelle: Herausgeber-Zertifikate / Fälschbarkeit von Zertifikatsketten:**
- Systeme zur Gefahrenabwehr in Unternehmen, beispielsweise Viren-Scanner, Einbruchserkennung und Data Loss Prevention (DLP), untersuchen den Datenverkehr auf den Inhalt. Dafür erhalten diese Anbieter von den Zertifizierungsstellen Intermediate-CA-Zertifikate (Herausgeber-Zertifikate) ➔ **Man-in-the-Middle-Aktion**



- Bevor ein Server-Betreiber ein Zertifikat erhalten kann, muss er ein Schlüsselpaar, bestehend aus einem privaten und einem öffentlichen Schlüssel, erzeugen. Das Zertifikat erhält der Server-Betreiber dann, wenn er einen Antrag mit dem öffentlichen Schlüssel an eine Zertifizierungsstelle gestellt hat.
- Einige CA erstellen neben dem Zertifikat auch das Schlüsselpaar für den Kunden. Damit ist die Sicherheit des privaten Schlüssels gefährdet / nicht mehr gewährleistet.
(Versand per Mail, Angriffe auf den Rechner,...)
- ➔ Schlüsselpaar sollte nur auf dem eigenen Rechner erzeugt werden.



- Die Verschlüsselung mit Zertifikat beinhaltet zwei Schlüssel. Der öffentliche Schlüssel, der mit dem Zertifikat verbunden ist und ein dazu passender privater Schlüssel, der niemals in fremde Hände gelangen darf.
- Wenn dem Besitzer eines Zertifikats der **private Schlüssel gestohlen** wird, dann muss das Zertifikat in eine Blacklist (Sperrliste) aufgenommen werden, damit die Validierungsstelle diesen Schlüssel bei einer Abfrage als ungültig erklären kann.
- Dazu muss der Diebstahl vom Besitzer des Zertifikats erkannt und gemeldet werden. Wenn nicht kann ein Angreifer, der den privaten Schlüssel besitzt, die verschlüsselten Daten entschlüsseln.

Schwachstelle: Validierung / Prüfung der Zertifikate

- Folgende Punkte sollten dazu führen, dass ein Zertifikat als ungültig erkannt wird und die Verbindung abgebrochen wird:
 - **abgelaufener Gültigkeitszeitraum**
 - **unterschiedlicher Hostname im Zertifikat**
 - **unterbrochene Vertrauenskette zum CA-Root-Zertifikat**
- Browser akzeptieren ein ungültiges Zertifikat auch dann, wenn sie keine Antwort von der Validierungsstelle/Zertifizierungsstelle bekommen (OCSP, Online Certificate Status Protocol).



- Generell startet jeder Verbindungsaufbau unverschlüsselt. Über diese unsichere Verbindung vereinbaren die Kommunikationspartner eine verschlüsselte Verbindung. Wenn jetzt einer der beiden Kommunikationspartner kein SSL bzw. TLS beherrscht, dann wird die Verbindung unverschlüsselt hergestellt.

- ➔ Verschlüsselung ist nur Option!

Angreifer kann dies Unterdrücken

- Abhilfe:

Für HTTP gibt es HTTP Strict Transport Security, kurz HSTS. Darüber teilt der Webserver dem Browser mit, dass HTTP-Requests über eine verschlüsselte Verbindung erfolgen sollen. (Üblich bei Banking)



■ Schwache Verschlüsselung

- Schwache Verschlüsselungen (z.B. RC4) sind möglich und können gewählt werden (geringere Rechenleistung erforderlich als bei AES mit 128 bit)

■ Verschlüsselung und Integrität

- Erst MAC, dann Verschlüsselung → hat Schwachstellen

■ Software-Implementierung

- Entwickler nutze meist vorhandene Software-Bibliotheken (JSSE, OpenSSL oder GnuTLS)
- → viele Optionen und Einstellungen, die Entwickler überfordern
- Schnittstellen der Bibliotheken sind meist schlecht implementiert
- Testmöglichkeiten nicht ausreichend

■ Heimlich nachinstallierte CA-Root-Zertifikate

- Z.B. Microsoft lädt neue Root-Zertifikate nach

■ Hohe Komplexität für den Normalnutzer



aktuell | OpenSSL-GAU

c't 10/14, S. 16 (erschieden am 19.4.2014)

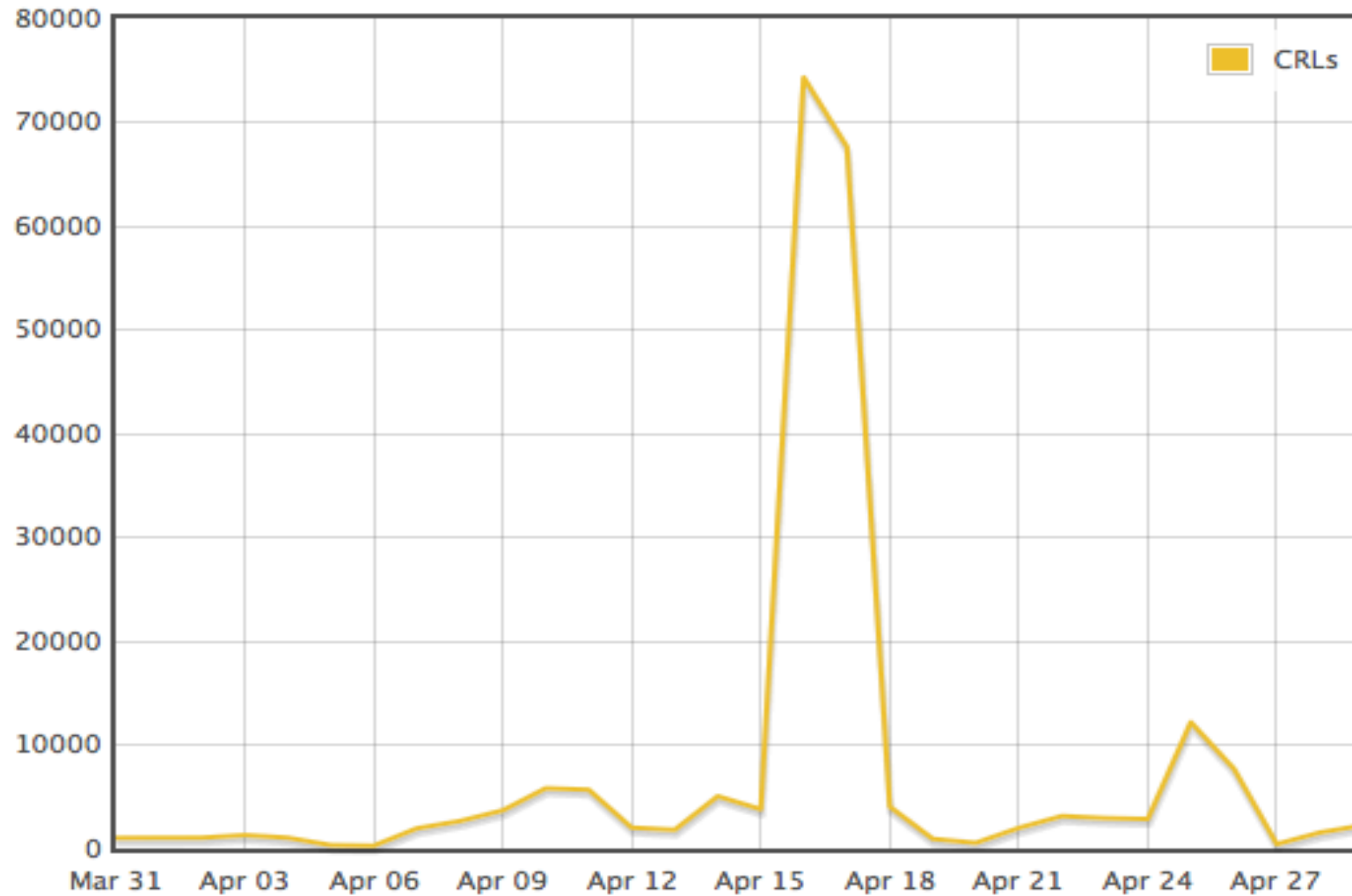
Uli Ries

SSL-GAU lässt Server bluten

Kritischer Fehler bei der Verschlüsselung

Ein Fehler in der Krypto-Bibliothek OpenSSL betrifft Millionen von Server und gefährdet unter anderem Passwörter und geheime Schlüssel. Krypto-Guru Bruce Schneier spricht von einem „katastrophalen“ Bug: „Auf einer Skala von 1 bis 10 ist das eine 11“.

Certificates Revoked per Day



29.04.2014 11:20

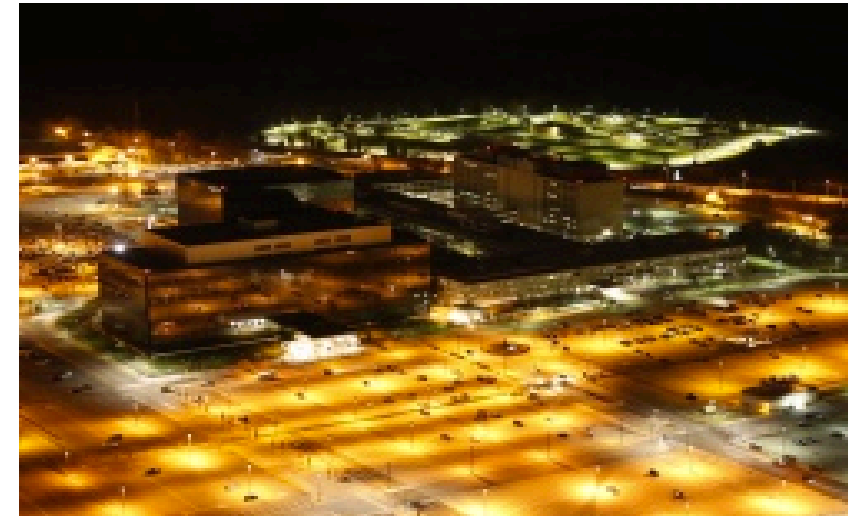
« Vorige | Nächste »

Heartbleed: US-Regierung hält IT-Schwachstellen geheim

 vorlesen / MP3-Download

Nachdem die SSL-Lücke Heartbleed bekannt geworden war, wurde berichtet, die NSA habe von der Lücke gewusst. Das bestreitet die US-Regierung weiter, gibt aber zu, gefundene Sicherheitslücken nicht immer zu veröffentlichen.

Die US-Regierung hat bestätigt, dass ihre Geheimdienste neu entdeckte Sicherheitslücken in Computersystemen mitunter für Spionage und Cyberangriffe ausnutzen. Es gebe Kriterien, nach denen entschieden werde, ob eine Sicherheitslücke öffentlich gemacht werde oder nicht, erklärte Michael Daniel, der Berater von US-Präsident Barack Obama in Fragen der Cybersicherheit. Er bestätigte damit Informationen, die der ehemalige NSA-Mitarbeiter Edward Snowden enthüllt hatte.




NSA-Hauptquartier: Welche IT-Lücken kennen sie hier? 

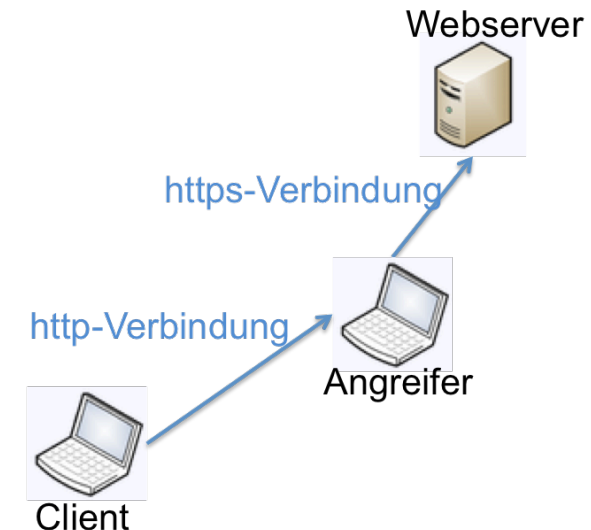
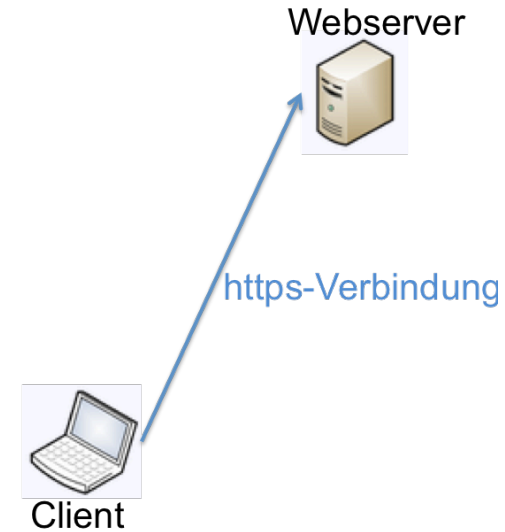
Bild: Trevor Paglen

- **Man-in-the-Middle Angriff zum Auslesen von HTTPS-Seiten**

- Angreifer erzwingt zum Client hin wird nur http, fällt von https auf http zurück.
- SSL-Symbol im Browser entfällt!
- Wird aber oft nicht beachtet.
- Vom Angreifer zum Server wird dann SSL benutzt, Angreifer kann aber Daten lesen und manipulieren.

- **Gegenmaßnahmen**

- Anzeige im Browserfenster kontrollieren!
- Webseite erzwingt den Einsatz von SSL:
 - Flag: HTTP Strict Transport Security (HSTS) wird gesetzt pro sicherer Webseite
 - Browser kann nur verschlüsselte Verbindungen mit dieser Domäne aufbauen
 - Unverschlüsselte Verbindungen werden dann abgelehnt



IP Security als Funktion im Netz: IPSec

■ IPSec (Internet Protocol Security)

ist eine **Protokoll-Suite**, für eine gesicherte Kommunikation über IP-Netze

■ IPSec arbeitet direkt auf dem IP-Layer, ist also für Transport-Protokolle / Anwendungen **transparent**

■ Einsatz-Szenarien

Virtuelle Private Netze (VPN) in unterschiedlichen Ausprägungen

- Gateway-zu-Gateway
(Vernetzung von Standorten)
- Host-zu-Host
- Host-zu-Gateway (Remote Access, erfordert weitere Funktionen,
z.B. mit L2TP oder PPTP)

Bestandteile von IPsec-VPNs

- kryptografischer Schutz der übertragenen Daten
- Zugangskontrolle
- Datenintegrität
- Authentisierung des Absenders (Benutzer-Authentisierung)
- Verschlüsselung
- Authentifizierung von Schlüsseln
- Verwaltung von Schlüsseln (Schlüsselmanagement)

IP Security als Funktion im Netz: IPSec

■ Ziele von IPSec

- Vertraulichkeit,
- Authentizität und
- Integrität

■ Methoden:

- Zusätzlicher Header mit Message Authentication Code (MAC).
- Verschlüsselung

■ IPSec arbeitet direkt auf dem

IP-Layer, ist also **transparent**

für Transport-Protokolle / Anwendungen

■ Vertraulichkeit:

- Sender verschlüsselt IP-Payload
- Für jede IP-Payload,
z.B. TCP- und UDP-Segmente;
ICMP- und SNMP-Nachrichten.

■ Authentifizierung:

- Zielhost kann Quell-IP authentifizieren

■ zwei zentrale Protokolle:

- **Authentication-Header-Protokoll (AH)**
- **Encapsulation-Security-Protokoll (ESP)**

■ Schlüsselaustausch:

- Manual Keying (fest konfiguriert)
- Internet Key Exchange (IKE / IKEv2)

■ Authentisierung:

- Pre Shared Keying (PSK) oder
- Zertifikate (CA)

IPSec basiert auf der

Security Association (SA)

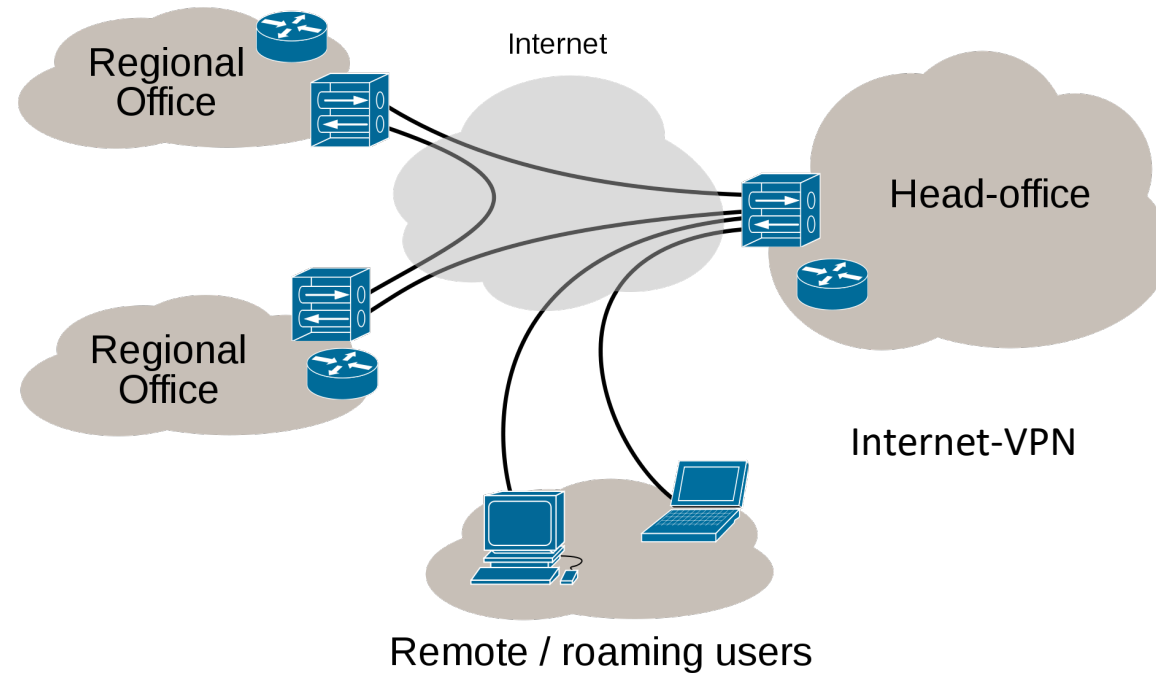
zwischen zwei Kommunikationspartnern

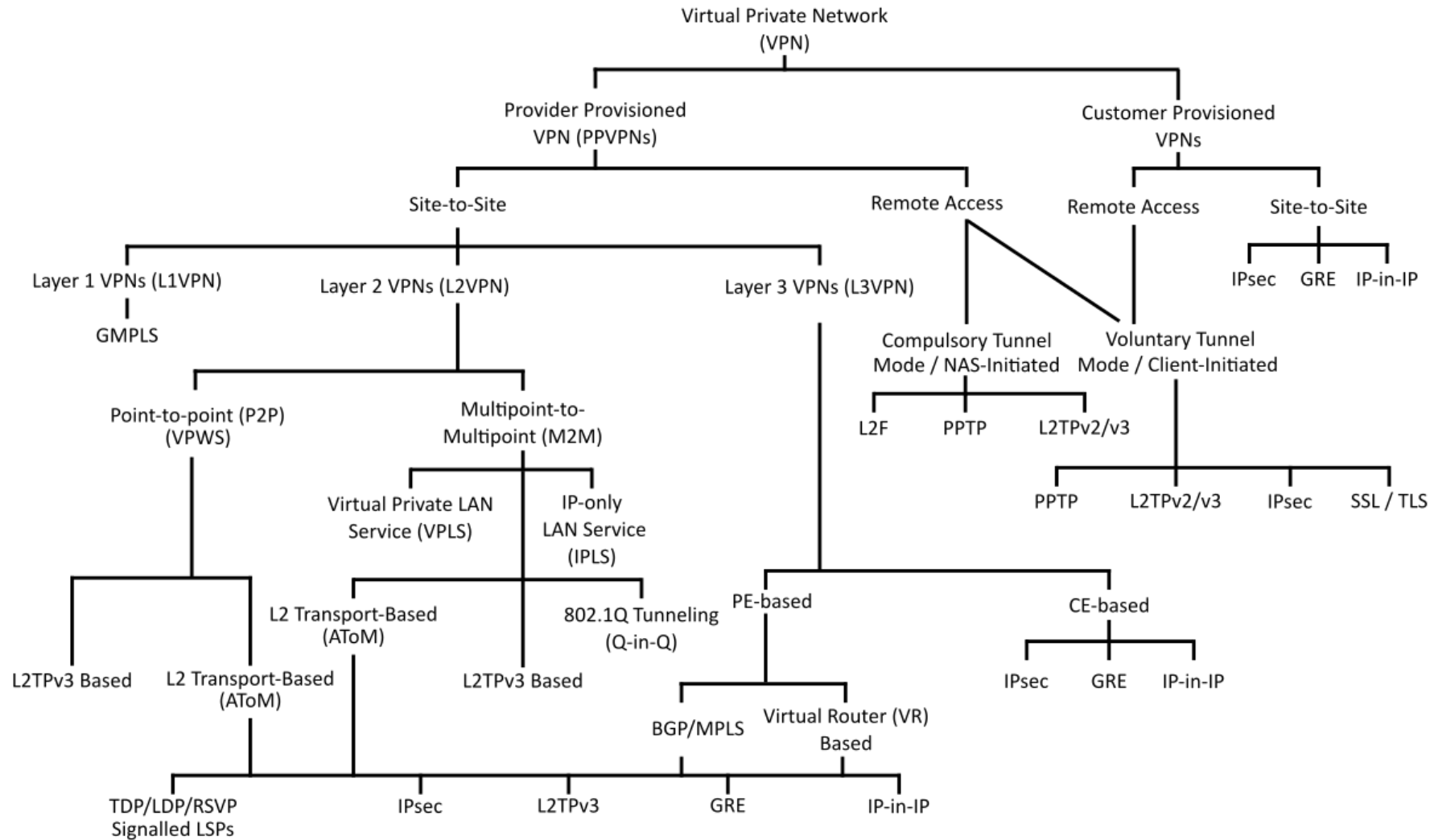
Parameter einer SA:

- Art der gesicherten Übertragung (Authentifizierung oder Verschlüsselung)
- Verschlüsselungsalgorithmus
- Schlüssel
- Dauer der Gültigkeit der Schlüssel

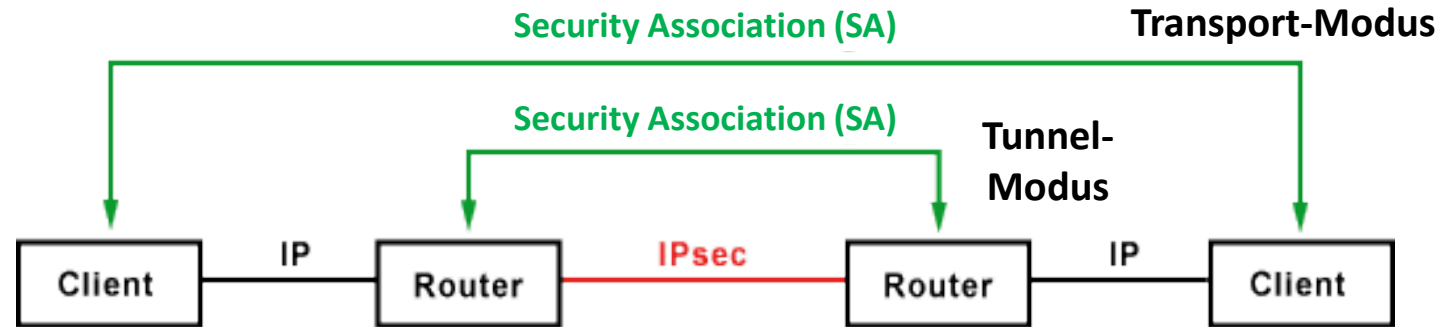
2 Betriebsarten:

- **Tunnel-Modus:** Verbindung zwischen 2 Routern
- **Transport-Modus:** Verbindung zwischen 2 Endsystemen





Quelle: https://en.wikipedia.org/wiki/File:VPN_classification-en.svg



■ Tunnelmodus:

- Gesamtes IP-Paket wird vollständig eingekapselt und geschützt
- Neuer zusätzlicher IP-Header (Altes Paket ist die Nutzlast)
- Typisch zwischen IPSec-Routern zur Verbindung von Unternehmens-Standorten
- Kommunikationsbeziehung (originale IP-Header) bleibt verborgen

■ Transport-Modus:

- Original IP-Header bleibt erhalten,
- IPSec Header werden zusätzlich eingefügt
- Kommunikationsbeziehung (originale IP-Header) ist sichtbar
- Typisch zwischen 2 Client oder Client – IPSec-Router

■ Security Association (SA) bestimmt alle Parameter für IPSec

■ Security Association (SA)

- Idee: Im Header nur minimale Info für IPSec
- Security Association (SA) ist eindeutig bestimmt durch:
 - **Security Parameters Index (SPI)** = 32-Bit-Verbindungs-ID
 - **Ziel-IP Adresse**
 - **Sicherheitsprotokoll (AH oder ESP)**
- jede SA ist unidirektional
- logische Netzwerkschicht-Verbindung
- SA führt zur SAD:

■ SA Database (SAD) enthält:

- Für AH: Authentifikationsverfahren, Modi und Schlüssel für AH-Protokoll
- Für ESP: Verschlüsselungsverfahren, Authentifikationsverfahren, Modi und Schlüssel für ESP-Protokoll

Initialisierungsvektor

- Lebensdauer der Schlüssel bzw. der SA
- IP-Adressen der SA-Teilnehmer (Client oder Subnetze)

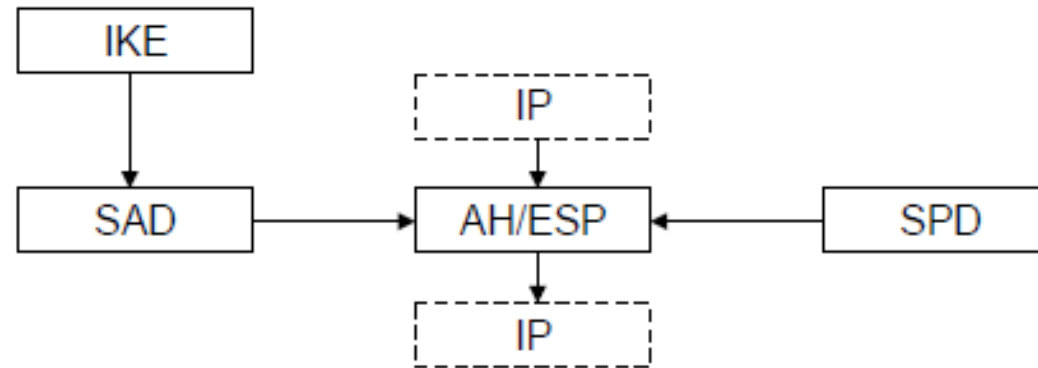


Security Association Database (SAD)

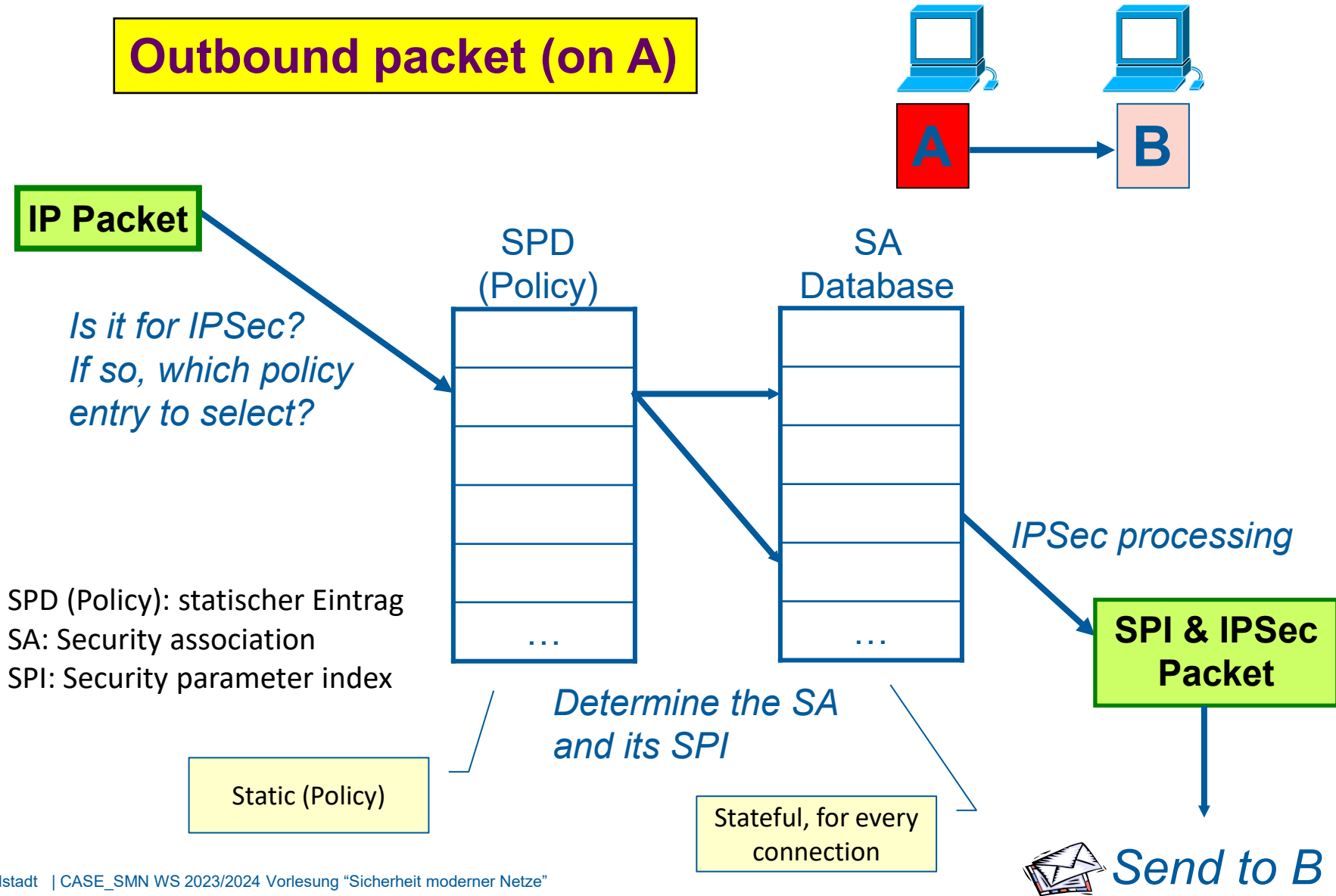
- **Holds parameters for each SA**
 - Lifetime of this SA
 - AH and ESP information
 - Tunnel or transport mode
- **Every host or gateway participating in IPsec has their own SA database**
- **Stateful Information on every active IPsec connection**

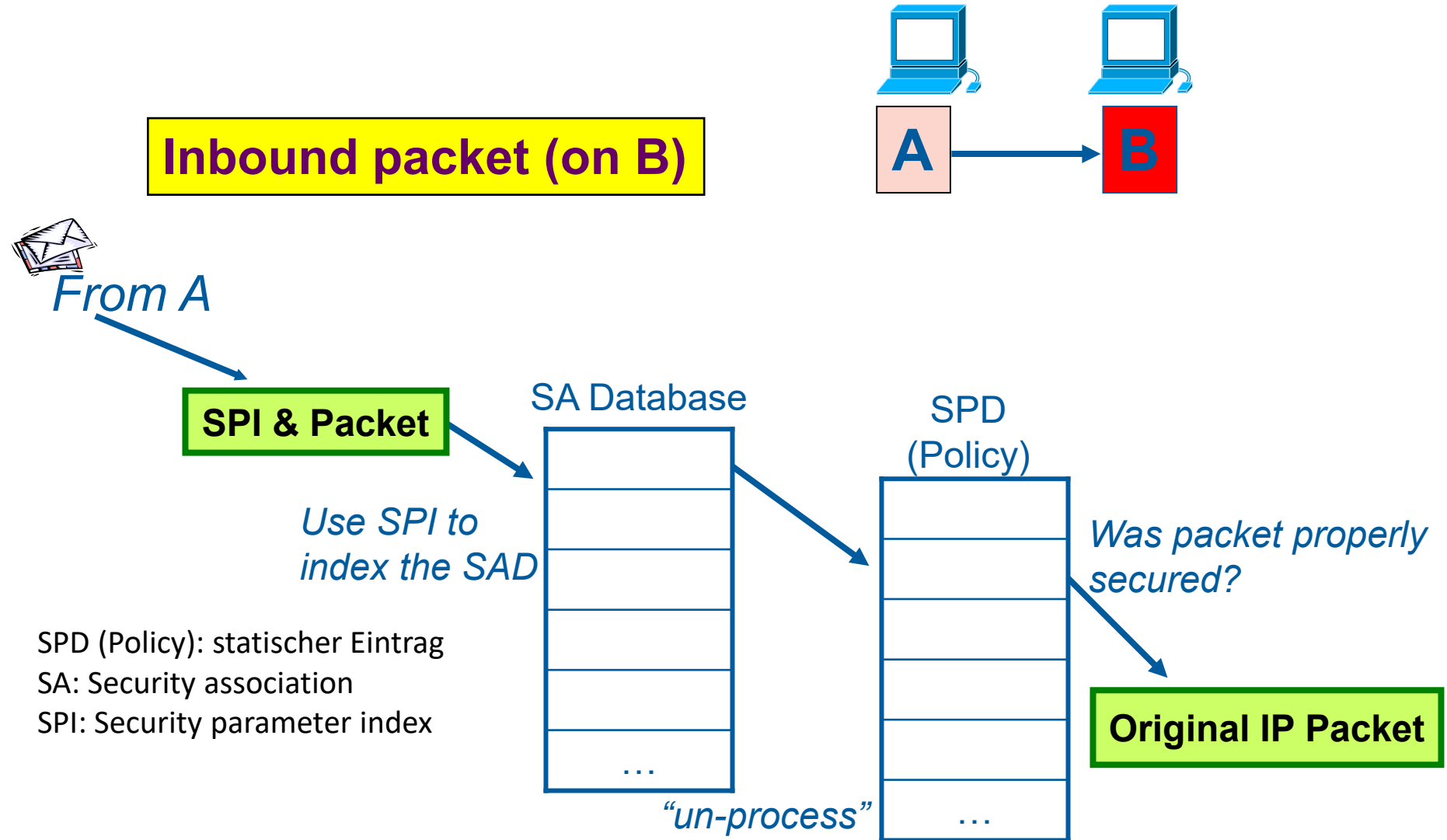
Security Policy Database – SPD

- **What traffic to protect?**
- **Policy entries define which SA or SA bundles to use on IP traffic**
- **Each host or gateway has their own SPD**
- **Index into SPD by Selector fields**
 - Dest IP, Source IP, Transport Protocol, IPsec Protocol, Source & Dest Ports, ...
- **Stateless (Static entries)**

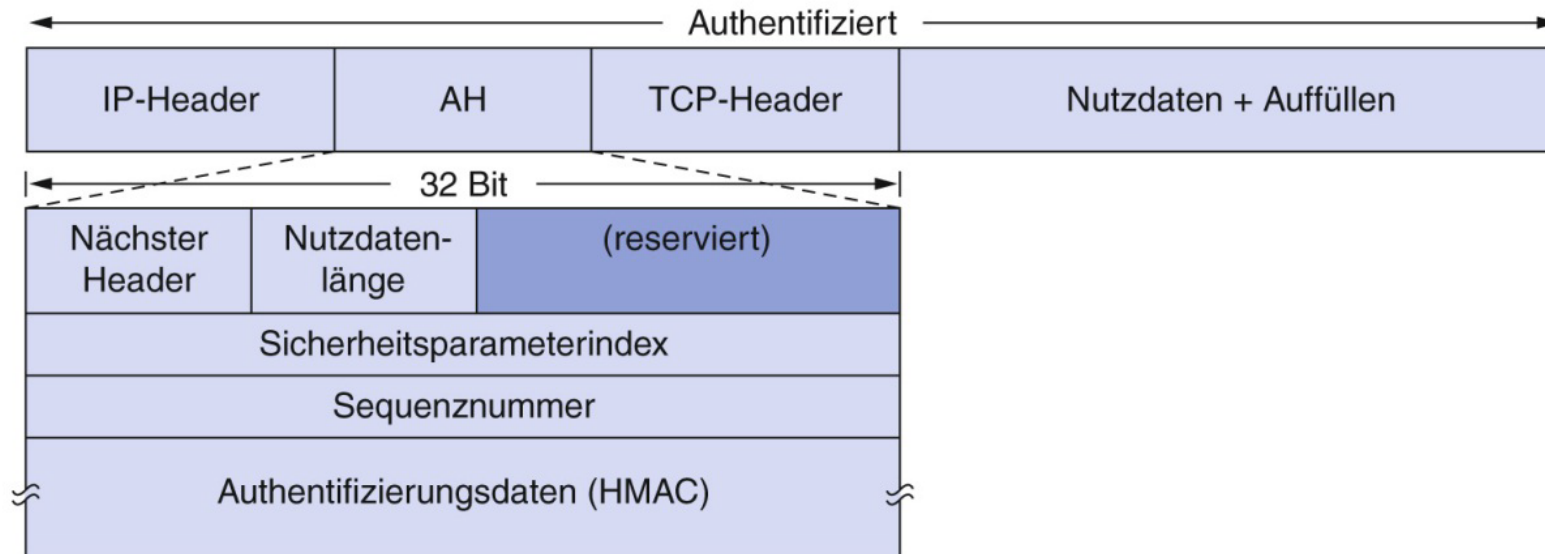


- IKE Internet Key Exchange (Pre-Shared Key oder Public Key Exchange)
- SA: Security Association
 - SPI: Security parameter index
 - IP Adressen
 - Auswahl IPSec Protokoll (AH / ESP)
- SAD Security Association Database: enthält alle Daten für eine IPSec-Verbindung
- SPD (Policy): statischer Eintrag

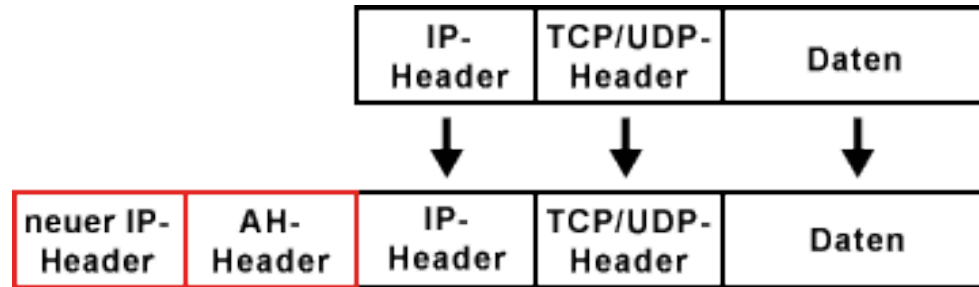




Authentifizierungs Header (Bsp. Transportmodus)

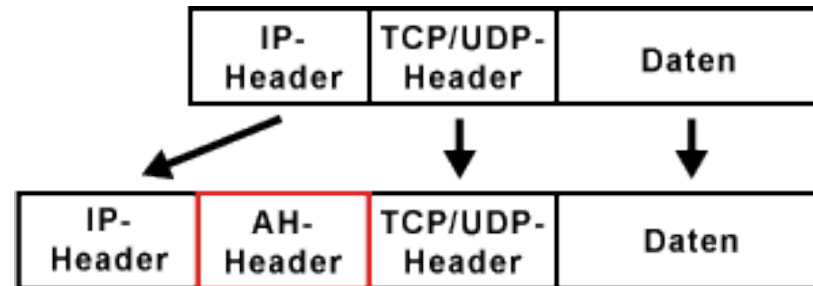


- Der IPsec AH-Header wird nach einem IP-Header eingefügt, sichert aber das ganze IP-Paket inklusive des davor liegenden IP-Headers ab.
- Im Protokoll-Feld des IP-Headers wird 51 eingetragen (AH)
- Der **Authentication Header (AH)** umfasst:
 - Verbindungs-ID (Security Parameter Index (SPI)) steht dabei für die Parameter einer SA, die durch einen vorgeschalteten IKE ausgehandelt wurden.
 - Authentifizierungsdaten: von der Quelle signierter Hashwert über das Original-Datagramm → MAC
 - Next-Header-Feld: Payload-Typ (z.B., TCP, UDP, ICMP)
 - Durch einen Sequenzzähler (Sequence Number Field) bietet AH auch Schutz vor Replay-Attacken.



Authentication Header im Tunnelmodus:

- gesamte IP-Paket wird in ein neues IP-Paket gepackt und geschützt.
- Geeignet für alle VPN-Verbindungen



Authentication Header im Transportmodus:

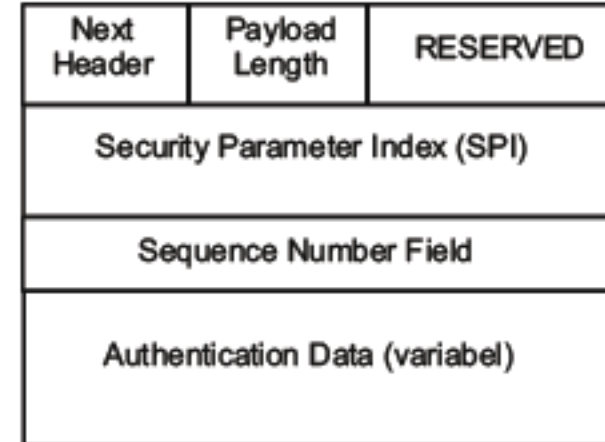
- IP-Header bleibt erhalten und wird mit dem Rest geschützt.
- Nur geeignet für Host – Host Verbindungen

Authentication Header

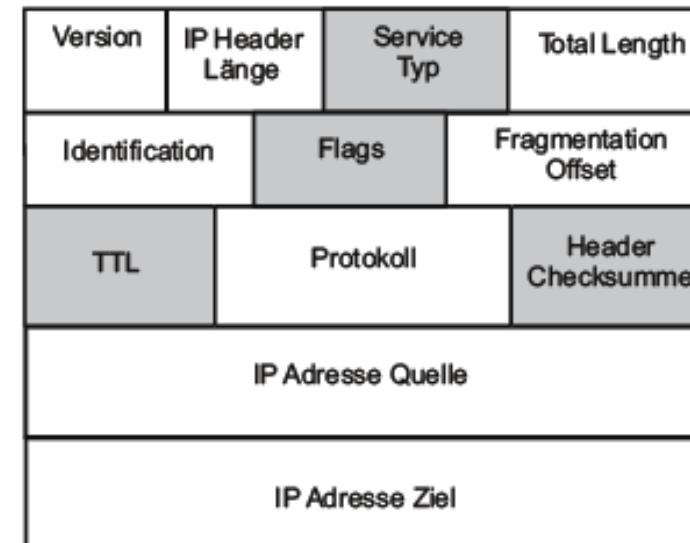


Authentication Header (AH) ist eine Transportabsicherung nach RFC 2402 die

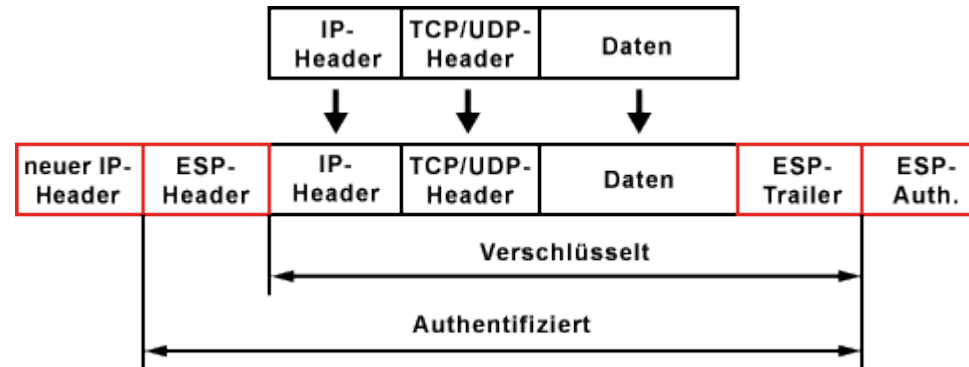
- eine Authentizität des Senders sowie die
- Integrität und die
- Nichtabstreitbarkeit von versendeten Daten gewährleistet.
- Keine Verschlüsselung in AH
- Der IPSec AH-Header wird nach einem IP-Header eingefügt, sichert aber das ganze IP-Paket inklusive des davor liegenden IP-Headers ab.
- Der Security Parameter Index (SPI) steht dabei für die Parameter einer SA, die durch einen vorgeschalteten IKE ausgehandelt wurden.
- Durch einen Sequenzzähler (Sequence Number Field) bietet AH auch Schutz vor Replay-Attacken.



Authentication Header

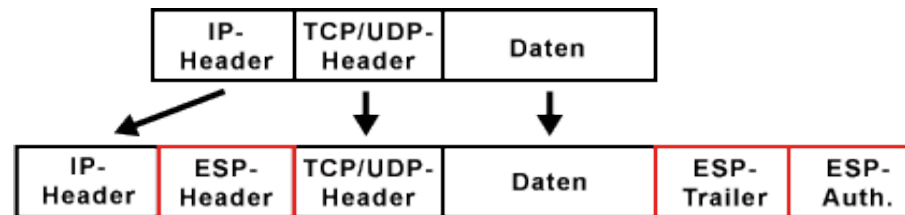


Feste und veränderbare Felder eines IPv4-Headers



ESP-Tunnelmodus:

- gesamte IP-Paket wird in ein neues IP-Paket gepackt und geschützt.
- Original-IP-Adressen sind nicht mehr sichtbar
- Geeignet für alle VPN-Verbindungen



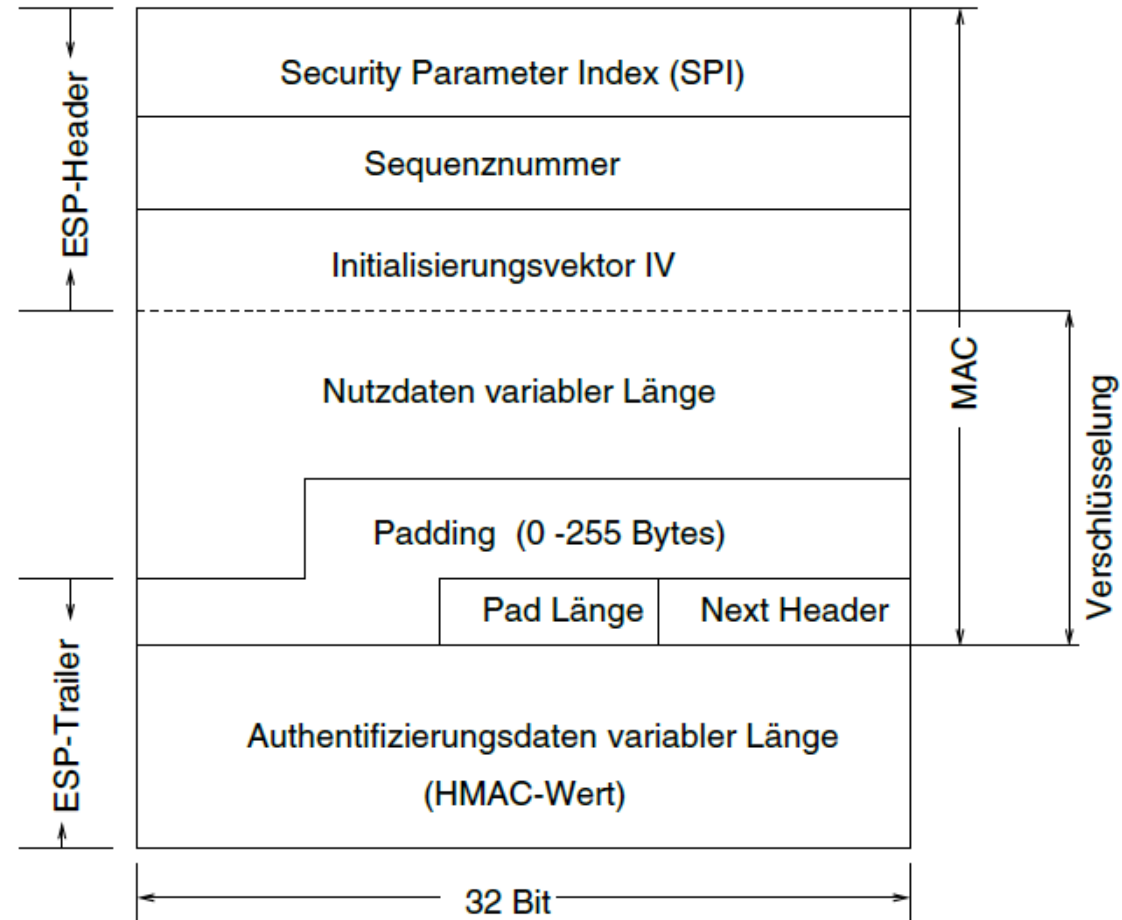
ESP-Transportmodus:

- IP-Header bleibt erhalten und wird mit dem Rest geschützt.
- Nur geeignet für Host – Host Verbindungen

Encapsulating Security Payload - Format

ESP-Header (ähnlich AH) nicht verschlüsselt um Auswertung zu ermöglichen:

- Security Parameter Index (SPI) steht dabei für die Parameter einer SA, die durch einen vorgeschalteten IKE ausgehandelt wurden.
- Durch einen Sequenzzähler (Sequence Number Field) bietet ESP auch Schutz vor Replay-Attacken.
- Initialisierungsvektor



Quelle Claudia Eckert, IT-Sicherheit, 8. Aufl.

Warum sind AH und ESP eigenständig?

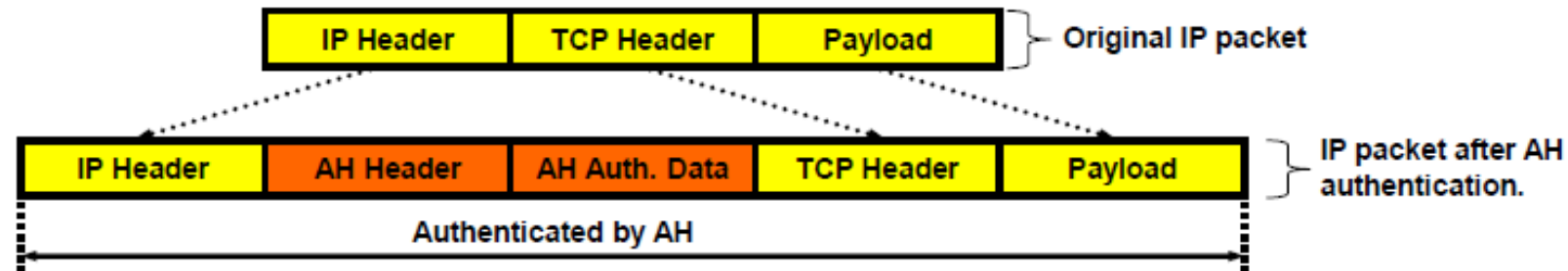
- **AH bietet Authentifikation, ESP bietet Verschlüsselung (Encryption) unabhängig voneinander.**
 - Oft ist AH (Authentifikation) ausreichend und bietet ausreichend Sicherheit
 - ESP (Verschlüsselung) ist aufwendiger (Rechenpower, Zeit)
 - Manchmal ist Verschlüsselung gesetzlich auch nicht erlaubt
- **AH ist einfacher;**
 - ESP fordert eine starke Verschlüsselung, komplexeres Format
 - IP Header muss in jedem Router bewertet werden, daher ist ein einfacheres Protokoll besser
- **Konsequenz: AH und ESP können unabhängig voneinander eingesetzt werden**

- **IPv6:**
 - AH und ESP sind Erweiterungsheader, werden bei Bedarf angefügt.
 - AH und ESP sind nur für die Endpunkte der Verbindung, daher sollen Erweiterungen für Router davor platziert werden.

Transport Mode vs. Tunnel Mode (AH only)

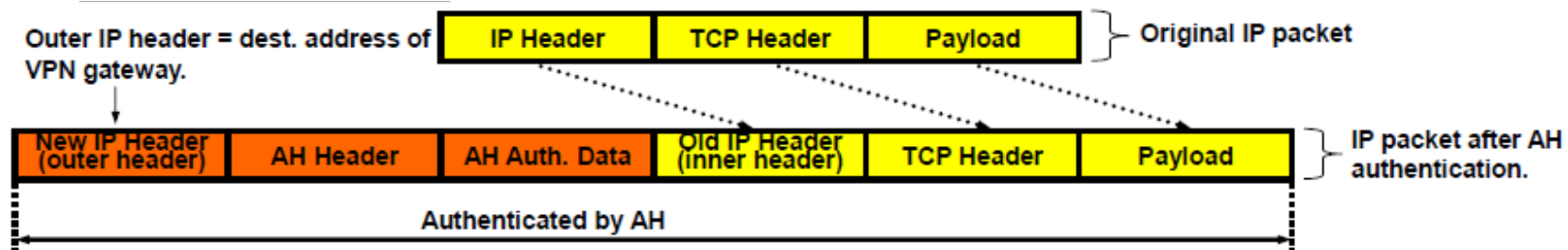
IPSec Transport Mode:

- Transport mode only protect upper layer protocols (TCP and above).
- Devices implementing only transport mode are called **IPSec hosts**.



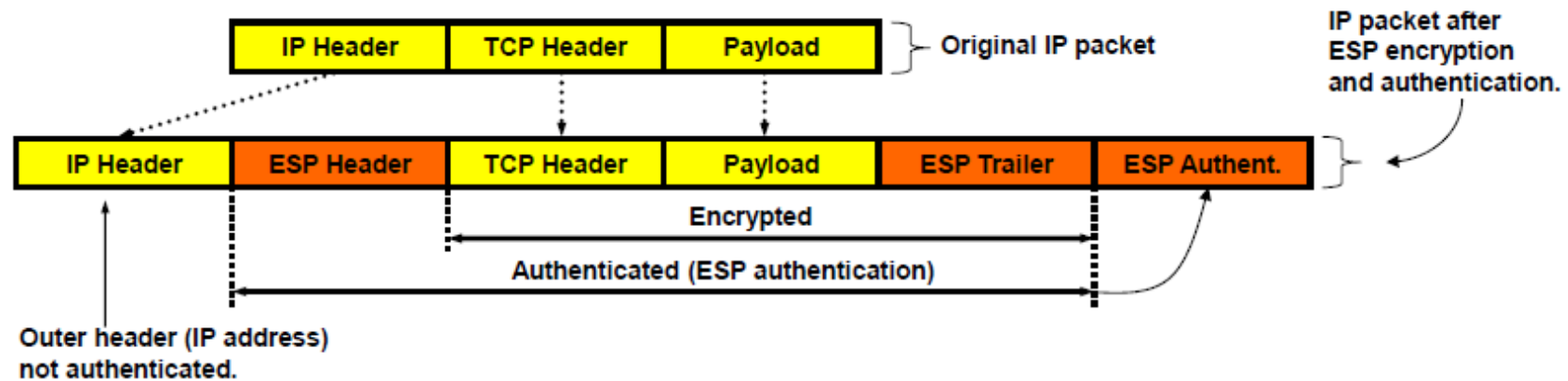
IPSec Tunnel Mode:

- Tunnel mode protects the entire IP packet and tunnel in a secured transport path.
- Devices implementing tunnel mode are called **IPSec gateways**.

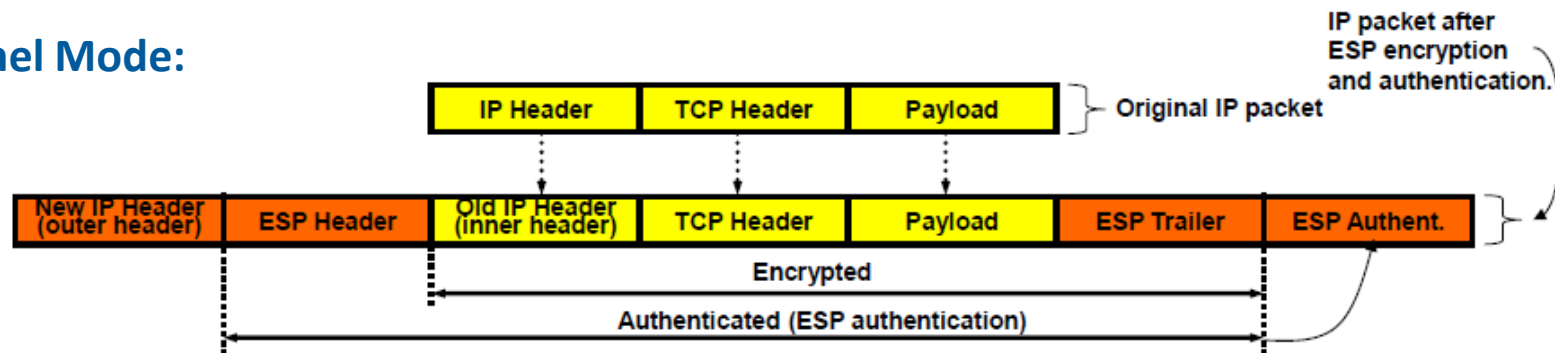


IPSec mit ESP (Verschlüsselter Inhalt)

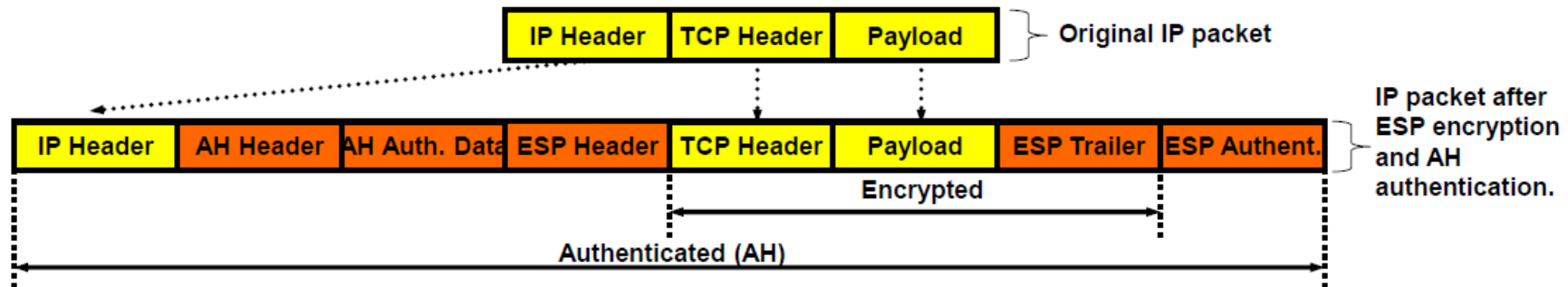
■ IPSec ESP Transport Mode:



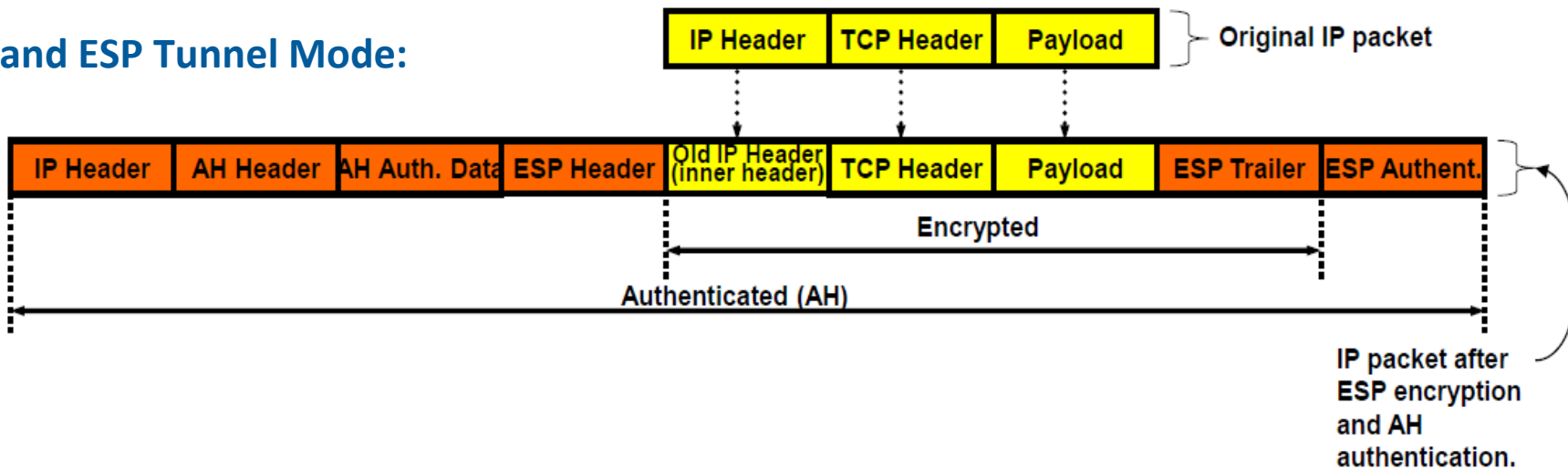
■ IPSec ESP Tunnel Mode:



■ IPSec AH and ESP Transport Mode (called „Transport Adjacency“):

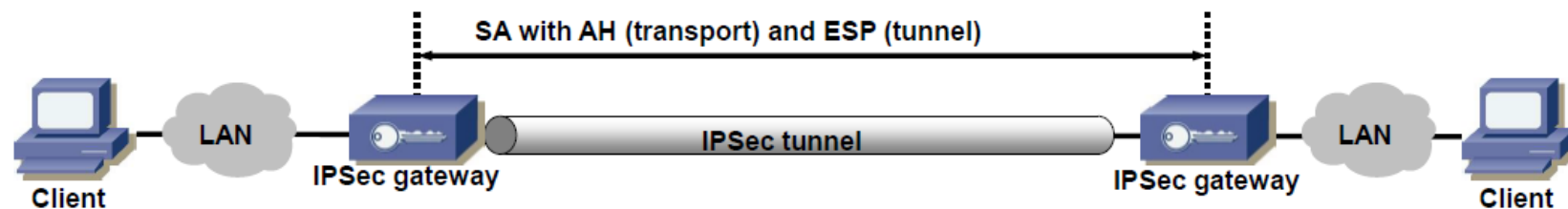


■ IPSec AH and ESP Tunnel Mode:



Einige IPSec Design Regeln

- Ist einer der IPSec Endpunkte ein Gateway → **Tunnel Mode**
 - Sind beide IPSec Endpunkte Hosts → **Transport Mode**
- der äußere IP Header entspricht dem inneren und ist sichtbar
- Wenn das gesamte IP-Paket incl. Header geschützt werden soll → **Tunnel Mode**
- Damit sind Ursprungs- und Ziel IP Adressen verschlüsselt.
- Werden AH und ESP gemeinsam eingesetzt:
 - AH ist das äußere Protokoll
 - ESP ist das innere Protokoll
 - → Erst Authentifizieren, dann entschlüsseln
 - Gemeinsamen Mode für beide nutzen (Transport Mode oder Tunnel Modus)
 - Beispiel für einfachen IPSec Tunnel



Probleme mit NAT

- Durch NAT erhält ein IPSec-Paket eine neue IP-Adresse und einen anderen Quell-Port. Wird ein IPSec-Paket verändert, wird es ungültig.
- Original-IP-Adressen und TCP-Ports sind verschlüsselt, daher hat der NAT Router keinen Zugriff. Die Information wird im SPI-Wert (Security Parameters Index) mitgegeben.

Lösung: **IPsec-Erweiterung NAT-Traversal:**

- ESP-Pakete werden in UDP-Pakete verpackt und über Port 4500 verschickt. Dann können die NAT-Router ohne Probleme IP-Adressen und Ports umschreiben.
- NAT-Traversal ist im IKE-Protokoll integriert (Negotiation of NAT-Traversal in the IKE).

Schwachstellen:

- Komplexität der Einstellungen
- Administration der SPD
- Verwalten der Parameter