Zusammenfassung

Informationssicherheit 2

Michael Wieland

Hochschule für Technik Rapperswil

17. August 2017

Mitmachen
Falls du an diesem Dokument mitarbeiten möchtest, kannst du es auf GitHub unter `https://github.com/michiwieland/hsr-zusammenfassungen` forken.

Inhaltsverzeichnis

1. Grundlagen

1.1. OSI Stack

Communication layers	Security protocols

Application layer	Platform Security, Web Application Security, VoIP Security, SW Security
Transport layer	TLS
Network layer	IPsec
Data Link layer	L2TP, IEEE 802.1X, IEEE 802.1AE, IEEE 802.11i (WPA2)
Physical layer	Quantum Cryptography

Abbildung 1: OSI Stack

1.2. Kryptographiesche Stärken

Kryptographiesche Stärken

	Recommended Algorithms	Key Size	True Strength
Symmetric Encryption	AES (CBC or Counter-Mode) ChaCha20 (Stream Cipher)	128 bits 256 bits	128 bits 256 bits
Data Integrity / Hash Function	SHA-256 (SHA-2 or SHA-3)	256 bits	128 bits
Key Exchange between Peers	Diffie Hellman with Prime Modulus (MODP)	3072 bits	128 bits
Digital Signature	RSA	3072 bits	128 bits
Public Key Encryption	RSA / El Gamal	3072 bits	128 bits
User Password	Abbreviated Passphrase	14* chars	≈80 bits

Abbildung 2: Kryptograpische Schlüssel Stärken

Schlüssellängen

Symmetric Keys	RSA/DH	ECDSA/ECDH/ Hash	Validity
80	1024	160	Disallowed since 2014
112	2048	224	Acceptable until 2030
128	3072	256	Acceptable beyond 2030
192	7680	384	Acceptable beyond 2030
256	15360	512	Acceptable beyond 2030

Abbildung 3: NIST 2012 Comparative Security Strength

Hinweis 1.1: NIST 2012 Compartive Security Strenght

Diese Grafik muss auswendig gelernt werden. Kommt garantiert an der Prüfung

1.3. NSA Suite B

Die Suite B ist eine Sammlung an kryptographischer Algorithmen, die 2005 von der NSA veröffentlicht wurde. Die Suite A wurde aktuell (2016) noch nicht veröffentlicht.

2. Verschlüsselungsverfahren

2.1. Speed

Wird für die Verschlüsselung die Hardwarebeschleunigung AES-NI verwendet ist ein Performancegewinn von ca. Faktor 2-3 zu erkennen.

Ranking	Verfahren	Bemerkung
1	AES-GCM	
2	AES	Je kleiner die Schlüssellänge desto schneller. Je kleiner die Schlüssellänge desto weniger Runden (10, 12, 14)
3	Camellia	Konservativer gewählte Rundenanzahl (18, 24)
4	DES	
5	3DES	12-13 mal langsamer als AES-128

Tabelle 1: Speed von Verschlüsselungsverfahren

2.2. Symmetrisch

2.2.1. AES: Advanced Encryption Standard

- Block Cipher mit einer Blockgrösse von 128 Bit
- Es gibt drei Varianten von Schlüssellängen, welche sich nur in den Anzahl Runden unterscheiden
  1. 128Bit: 10Runden
  2. 192Bit: 12Runden
  3. 256Bit: 14Runden
- Neuere Intel CPUs unterstützen das Advanced Encryption Standard New Instructions (AES-NI) (Hardwarebeschleunigung) Instruktionsset, welches den Algorithmus um Faktor zwei bis drei schneller macht. Dieser Modus ist standardmässig aktiviert.
- AES GCM ermöglicht eine Verschlüsselung des Plaintextes und garantiert die Integrität des generierten Ciphertextes, sowie zusätzlicher unverschlüsselter Daten durch eine kryptographische Checksumme. AES GCM hat einen hohen Datendurchsatz, weshalb dieser Modus gerne bei TLS Zertifikaten an den Anfang der Liste gesetzt wird. Einerseits können alle Counter-Blöcke vorberechnet werden, sobald der IV und die Plaintext-Länge bekannt sind. Andererseits kann die Authentisierung von Ciphertext-Blöcken und die Verschlüsselung von weiteren Plaintext-Blöcken parallel ablaufen.

AES CTR: Counter Mode Der Counter Mode ist eine Betriebsart, in der Blockchiffren wie AES betrieben werden können, um daraus eine Stromchiffren zu erzeugen. Zur Verschlüsselung wird ein Initialisierungsvektor mit dem Schlüssel verschlüsselt und so ein Zwischenschlüssel produziert. Dieser wird im Anschluss mittels einer XOR-Operation mit dem Klartext kombiniert. Daraus entsteht der Geheimtext.

2.2.2. Camellia

Camellia verwendet die gleichen Parameter wie AES: eine Blockgröße von 128 Bit und Schlüssellängen von 128, 192 oder 256 Bit. Camellia wurde in Japan entwickelt und kann mit AES verglichen werden. Im Zweifelsfall sollte aber AES verwendet werden, da dieses Verfahren weiter verbreitet ist, und die Hardware eher für AES optimiert ist.

- Block Cipher mit einer Blockgrösse von 128 Bit
- Es gibt drei Varianten von Schlüssellängen, welche sich nur in den Anzahl Runden unterscheiden
  1. 128Bit: 18Runden
  2. 192Bit: 24Runden
  3. 256Bit: 24Runden

2.3. Asymmetrisch

2.3.1. RSA

Rivest Shamir Adleman (RSA) ist ein asymmetrisches kryptographisches Verfahren, das sowohl zum Verschlüsseln als auch zum digitalen Signieren verwendet werden kann.

- RSA Signatur Schlüssel haben üblicherweise einen sehr kleinen öffentlichen Exponenten, bei dem nur zwei Bits auf 1 gesetzt sind. Im Gegensatz dazu ist das Modul und der private Exponent sehr gross, bei welchen etwa die Hälfte der Bits auf 1 gesetzt sind. Daraus resultiert, dass die Verifikation der Signatur sehr viel einfacher und schneller geht, wie die Generierung.

Vorgehen

1. Zwei Primzahlen wählen und Produkt bilden: $n = p \cdot q$
2. $\varphi(n) = (p-1) \cdot (q-1)$
3. Beliebige Zahl a wählen, wobei a teilerfremd zu $\varphi(n)$ sein muss. $(ggt(\varphi(n), a) = 1)$
   $\Rightarrow$ am besten eignen sich Primzahlen für a. (privater Schlüssel)
4. Multiplikative Inverse b berechnen (öffentlicher Schlüssel) $a \cdot b = 1 \bmod \varphi(n) \Rightarrow$ erweiterter Euklidischer Algorithmus
   a) Modul: n (öffentlich)
   b) Öffentlicher Schlüssel: b
   c) Privater Schlüssel: a

Verschlüsseln

1. $\text{Wert}_{\text{verschlüsselt}} = (\text{Wert}_{\text{unverschlüsselt}})^b \cdot mod(n)$

Entschlüsseln

1. $\text{Wert}_{\text{unverschlüsselt}} = (\text{Wert}_{\text{verschlüsselt}})^a \cdot mod(n)$

2.3.2. DH: Diffie-Hellman

DH ist ein asymmetrisches Verfahren, welches dazu verwendet wird einen symmetrischen Key zu generieren. Die Idee hinter diesem Verfahren ist, dass die beiden Partner gemeinsam den geheimen Schlüssel generieren (Shared Master Secret), ohne ihn ganz übermitteln zu müssen.

Vorgehen

1. One Way Function: $f(x) = g^x$ mod p
2. Die Zahlen g und p sind öffentlich und können ungeschützt übertragen werden
3. Alice und Bob einigen sich auf die Zahlen p und g
4. Alice wählt eine zufällige Zahl a und sendet Bob den berechneten Wert gross A. $A = g^a mod(p)$
5. Bob wählt eine zufällige Zahl b und sendet Alice den berechneten Wert gross B. $B = g^b mod(p)$
6. Beide können nun den geheimen symmetrischen Schlüssel K berechnen: $K = A^b mod(p)$ rsp. $K = B^a mod(p)$

3. Hashverfahren

3.1. Speed

Auf 64Bit Plattformen ist SHA-3 512 schneller wie die SHA-2 256 Variante, da sie intern 64 Bit Wörter verwendet und die 256 nur 32Bit Wörter

Ranking	Verfahren
1	MD5
2	SHA-1
3	SHA-2
4	SHA-3

Tabelle 2: Speed von Hashverfahren

3.2. MD5

MD5 (Message Digest #5) berechnet einen 128Bit langen Hash Wert. Er darf heutzutage nicht mehr verwendet werden, da er offiziell als gebrochen gilt. (seit 2013)

3.3. SHA-1

SHA-1 (Secure Hash Algorithm) berechnet einen 160Bits langen Hash Wert. NIST empfiehlt seit 2005 SHA-1 nicht mehr zu verwenden.

3.4. SHA-2

Es existieren 4 SHA-2 Implementierungen (Secure Hash Algorithm Family) mit unterschiedlichen Hash Längen (SHA-224, SHA-256, SHA-384 und SHA-512). Die SHA-2 Variationen haben immer genau die halbe Schlüsselstärke (SHA-384 $\Rightarrow$ Schlüsselstärke = 192Bits). Obschon SHA-2 noch weit verbreitet ist, sollte man nicht mehr auf SHA-2 setzen.

3.5. SHA-3

Es existieren 4 SHA-3 (Keccak) Implementierungen mit unterschiedlichen Hash Längen (SHA3-224, SHA3-256, SHA3-384 und SHA3-512). SHA-3 ist der Gewinner eines NIST Wettbewerbs und wurde 2015 standardisiert. Keccak basiert nicht mehr auf dem ursprünglichen SHA.

4. Elliptic Curves

- Die Schlüsselstärke ist immer die Hälfte der angegeben Schlüssellänge. (z.B ECDH256 ⇒ 128bit)
- ECRSA ist wesentlich schneller wie RSA
- Mit dem D-Wave sind 1000Qbits möglich, jedoch wird der Shor's Quantum Algorithm nicht erfüllt, da nicht richtige Quantenzustände verwendet werden.

4.1. Kryptografische Stärke

Die kryptografische Stärke des ECDH Algorithmus beruht darauf, dass es sehr aufwändig ist, die Anzahl Additionen eines Kurvenpunktes P mit sich selber Modulo einer Primzahl p (d.h., das n in $nP \mod p = (P_1 + P_2 + ... + P_n) \mod p$) herauszufinden.

Grundsätzlich baut ECDH damit auf das gleiche bzw. ein sehr ähnliches Problem wie DH und RSA auf, allerdings ist ECDH gegen einige Angriffe resistent, welche die Faktorisierung von RSA/DH einfacher machen. Dadurch sind deutlich kürzere Schlüssel möglich.

4.2. Gruppe

Eine Gruppe ist ein algebraisches System bestehend aus G und einer Operation *, sodass für alle Elemente a,b,c in G folgende Bedingungen gelten:

1. a * b muss in G sein
2. a * (b * c) = (a * b) * c
3. Es gibt ein Neutrales Element ⇒ a * e = e * a = a (z.B 0 bei einer Addition)
4. Es gibt ein Inverses Element ⇒ a * a' = a' * a = e (z.B a'=-a bei einer Addition)
5. Man spricht von einer Abelschen Gruppe wenn gilt: a * b = b * a (Kommutativgesetz)

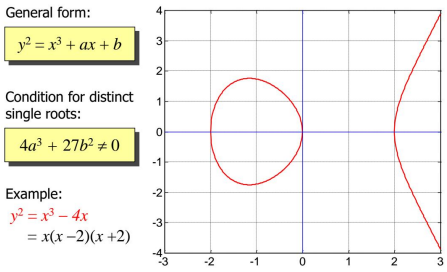

Abbildung 4: Elliptische Kurven

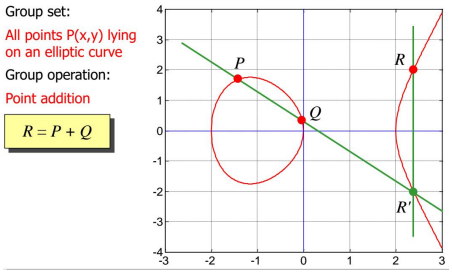

Abbildung 5: Berechnung der Gruppe

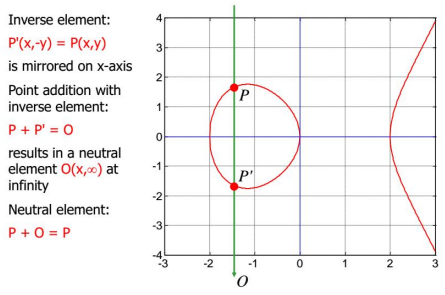

Abbildung 6: Inverses und Neutrales Element

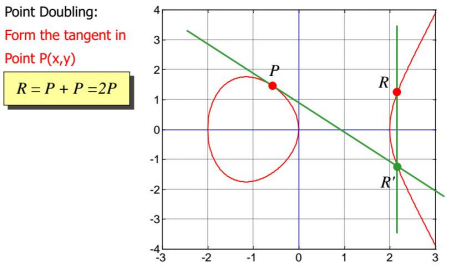

Abbildung 7: Q und P werden so verschoben, dass sie eine Tangente bildern

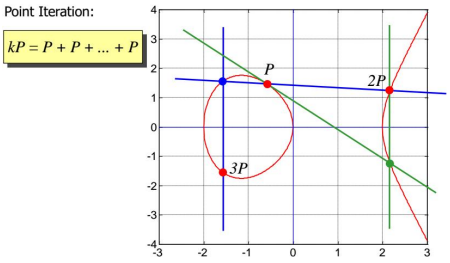

Abbildung 8: Point Iteration: Einen Punkt k-1 mal zu sich selbst hinzufügen

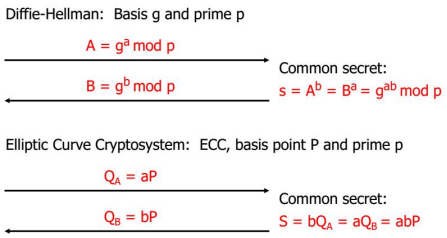

Abbildung 9: Schlüsselaustausch

4.3. ECDH

Die Schwierigkeit bei ECDH ist es, die Anzahl Punktadditionen herauszufinden (privater Teil)

1. Alice und Bob einigen sich auf eine Elliptische Kurve, einen Punkt auf der Kurve P aus der Zahlengruppe G (siehe Sektion 4.2). Diese werden ungeschützt übertragen.
2. Alice wählt eine zufällige geheime Zahl a, Bob wählt eine zufällige geheime Zahl b
3. Alice übermittelt Bob den "Public Key" $A = aP$, Bob übermittelt Alice seinen "Public Key" $B = bP$ über einen "öffentlichen" Kanal.
4. Alice und Bob können nun mithilfe der vorhandenen Zahlen den gemeinsamen geheimen Schlüssel ausrechnen: (Alice:) $aB =$ (Bob:) $bA = abP$.

4.4. Curve25519

$y^2 = x^3 + 486662x^2 + x$ in einem endlichen Körper modulo der Primzahl $2^{255} - 19$

5. Physical Layer Security

Auf dem Physical Layer gibt es nur Quanten Kryptographie das kryptographisch sicher ist. Zusätzlich gibt es noch Frequency Hopping, welches aber nicht sicher ist, wenn man das ganze Frequenzspektrum überwachen kann. Trotzdem wird es noch in der Schweizer Armee für die Funkgeräte eingesetzt.

5.1. Quanten Kryptographie

Bei der Quanten Kryptographie nutzt man die Eigenschaften der Quantenmechanik, damit zwei Parteien eine gemeinsame Zufallszahl generieren können, die als geheimer Schlüssel verwendet werden kann. Quanten können nicht geklont werden. Das bedeutet, dass es erkannt werden kann, wenn die Leitung abgehört wird. Ein Qubit kann nur zwei durch Messung sicher unterscheidbare Zustände haben. (Polarisation)

5.1.1. Ekert E91 Protokoll

Laser erzeugt von Zeit zu Zeit ein verstränktes (entangled) Photonenpaar. Wenn man den Zustand des einen misst, kennt man den Zustand des anderen. Beide Entstationen müssen mit dem gleichen Filterset messen (vertikal, horizontal). Wenn mehrere Photonen nicht ankommen (einzelne können von der Glasfaser geschluckt werden), kann der Empfänger davon ausgehen, dass die Leitung abgehört wird. Das Protokoll funktioniert aber nur auf Distanzen bis ca. 10km.

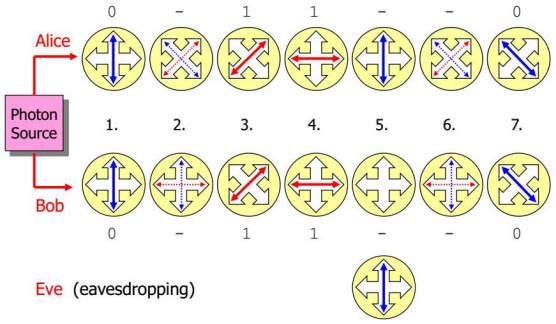

Abbildung 10: Entangled Photons

Decoy States Um zu verschleiern, dass Photonen gestolen wurden, könnte ein Angreifer bewusst Decoy States einfügen. Als Gegenmassnahme fügt der Sender zufällige Decoy States ein und sagt dem Empfänger welches die Decoy States waren. Die Decoy States werden auf einem niedrigeren Leitungspegel ausgesendet. Wenn der Angreifer auf der Leitung sitzt, resultiert eine andere statistische Verteilung (von Daten und Decoy Pulsen), was wiederum erkannt werden kann.

5.1.2. BB84 Protokoll

Beim BB84 braucht man keine verschränkte Protonen Paare. Hier wird nur ein einzelnes Photon versendet, welches 4 Zustände annehmen kann. (gleichwahrscheinlich: 45 Grad Polarisationen $(0°, 45°, 90°, 135°)$). Mit diesem Verfahren können Distanzen von 50-150km überwunden werden.

Ablauf Die Polarisation jedes empfangnen Photons wird von Bob mit einem zufällig gewählten Filterset gemessen (Rectilinear oder Diagonal). Nach den erfolgten Messungen gibt Bob über einen unsicheren Kanal bekannt, welches Filterset er jeweils für die Polarisationsbestimmung jedes Photons verwendet hat. Alice gibt anschliessend für jedes gesendete Photon bekannt, ob es rectilinear oder diagonal polarisiert wurde und ob es sich um einen Decoy-Puls oder einen normalen Puls gehandelt hat.

Verlust Je grösser die Übertragungsdistanz ist, desto mehr nehmen die Bit Slots ohne Information zu. Bei einer 1550nm breiten Fiber hat man Dämpfung von 0.2dB/km. Daraus resultieren folgende Verluste:

- 50km: 10dB $\rightarrow$ 1 von 10 Photonen überlebt
- 100km: 20dB $\rightarrow$ 1 von 100 Photonen überlebt
- 150km: 30dB $\rightarrow$ 1 von 1000 Photonen überlebt

5.1.3. Shor's Quantum Algorithm

Für eine Zahl n benötigt man einen Quantencomputer mit $\log(n)$ Qubits.

5.1.4. Grovers's Quantum Algorithm

Der Grover-Algorithmus beweist, dass Quantencomputer prinzipiell schneller als klassische Computer sind.

5.1.5. Quantenresistente Algorithmen

Für **AES** wurde nachgewiesen, dass der Einsatz von Quantencomputer die **Schlüssellänge halbiert**. Eine einfache Lösung für das Problem ist deshalb, dass man einfach doppelt so grosse Schlüssellängen verwendet.

New Hope Gitterbasierter Post-Quanten Algorithmus (Wird von Google z.Z. mit TLS in Kombination mit Curve25519 getestet.)

6. Zufallszahlen

Je länger eine Zufallszahl verwendet wird (z.B mehrere Jahre) , desto besser sollte die verwendete Zufallsquelle sein. Eine gute Zufallsquelle liefert stetig Zufallszahlen, die einer guten statistischen Verteilung unterliegen.

6.1. Eigenschaften

Gute PRF haben folgende Eigenschaften

- Liefert ständig und zuverlässig (ohne Unterbrüche) Zufallsinformationen
- Hat keine Tendez (Bias) zu 0 oder 1, d.h. hat eine gute statistische Verteilung.

6.2. PRF: Pseudo Random Functions

Ein PRF nutzt eine echte Zufallsquelle als Seed und füttert damit einen Algorithmus.

6.2.1. HMAC Functions

Der Schlüssel und der Seed wird aus einer echten zufälligen Quelle bezogen (z.B Quantenquelle) Der Seed wird mit dem Key der HMAC-Funktion übergeben. Dies wird dann solange wiederholt, bis man genügend Bytes Entropie hat. TLS nutzt HMAC PRF für Key Derivation.

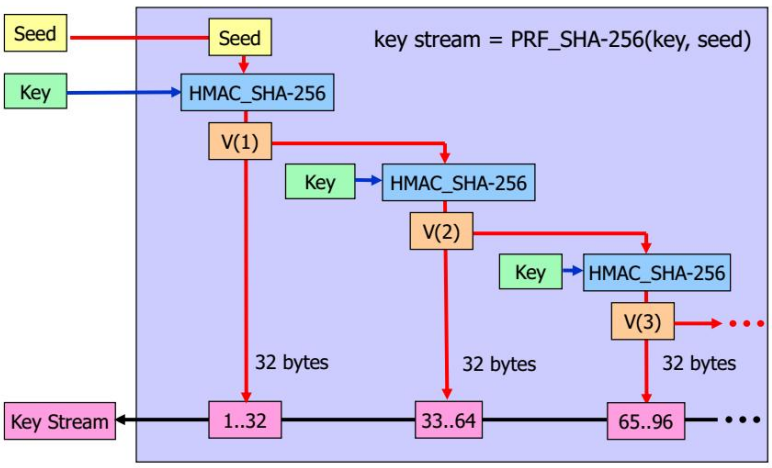

Abbildung 11: HMAC PRF

6.2.2. DRBG: Deterministic Random Bit Generator

Möchte man einen Key mit einer Schlüsselstärke "X" generieren, muss ein DRBG mindestens "X" Bits + $\frac{X}{2}$ Bits für die Random Nonce liefern. Ein DRBG hat mehrere Inputs:

Seed Das "Kaffeepulver" aus eineren wahren Zufallsquelle

Instantiate Wie Stark soll die Sicherheit sein: (112Bit), (128Bit), 192Bit, 256Bit

Entropy Input Entropy with size at least equal to security strength

Nonce

- Entropy with size at least equal to 0.5 * security strength
- Counter with repetition rate at least equal to 0.5 * security strength

Personalization String
Application Identifiers, Device Serial Numbers, User IDs, etc. (Optional, can be an empty string)

Additional Input Any other private or public input (Optional, can be empty)

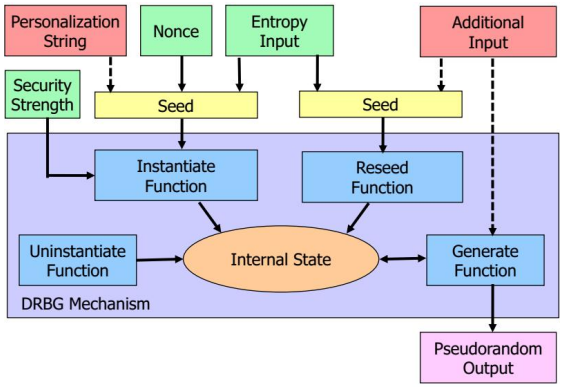

Abbildung 12: DRBG

6.3. Echte Random Generatoren

Bei echten Random Generatoren zieht man die zufälligen Zahlen meist aus physikalischen Prozessen. Am besten ist, wenn man mehrere Quellen miteinander kombiniert.

- Quantenquellen (die beste Quelle)
- Thermisches Rauschen
- Radioaktiver Zerfall
- Intel RDRAND (gut und sehr schnell, sollte mit anderen Quellen vermischt werden, da evtl. die NSA ihre Finger im Spiel hat)
- Key Stroke Timing (kleine Entropie)
- Mouse Movements Coordinates (grosse Entropie)
- Soundcard Input Noise (Gefahr von Gesamtausfall)
- Air Turbulence in Disk Drives (Veraltet, da SSD)
- Network Packet Arrival Times (eher verpönt, da manipulierbar)

7. Link Layer Security

7.1. WPA: Wi-Fi Protected Access (802.11i)

Der Client (Supplicant) stellt eine Anfrage über EAP over LAN (EAPOL) an eine Access Point oder LAN Switch (Authenticator). Dieser überprüft beim einem Radius Server (Authentication Server) über EAP RADIUS, ob die Anfrage valide ist.

7.2. Secure Device Identity (IEEE 802.1AR - DevID)

Während der Herstellung wird eine IDevID (Initial Device Identifier) sowie ein X509 Zertifikat direkt auf die Hardware (DevID Module) geschrieben. Das DevID Modul verfügt zusätzlich über einen starken Random Number Generator (RNG) und implementiert starke asymetrische Algorithmen (2048 RSA oder 256 ECDSA) sowie eine Hash Funktion (SHA-256). Auf dem DevID Module kann eine LDevID (Local Significant Device Identifier) hinterlegt werden.

7.3. MACsec: Media Access Layer Security (IEEE 802.1AE)

MACsec erlaubt **sichere E2E Verbindungen** über Ethernet zwischen direkt verbunden Clients. MACsec nutzt **AES-GCM-128** für die Verschlüsselung. Seit 2011 kann auch AES-GCM-256 verwendet werden. Eine neuer PAE muss den pre-shared CAK des CA-Verbund kennen

PAE: Port Access Entity Ein Client (SecY). Jede PAE hat einen Secure Channel zu allen anderen Mitglieder in der CA. Jeder PAE muss den CAK der CA besitzen.

CA: Connectivity Association Verbund von mehrere Clients auf L2 mit einem gemeinsamen Netzwerkschlüssel

CAK: Connectivity Association Key Statischer Schlüssel die jede PAE/SecY in der CA kennen muss. Diese müssen manuell verteilt werden. (Im Heim-WLAN auch als Pre Shared Key (PSK) bezeichnet.) Der CAK verschlüsselt die Control Plane. Der CAK ist somit das Shared Secred eines CA Verbund.

CKN: CAK Name Netzwerk Identifier (am besten Human-Readable; WLAN: SSID)

SAK: SA Key Secure Association Key (Session Key). Es wird für jeden Client ein SAK benötigt. Jeder SAK wird vom CAK abgeleitet. (Hash über den CAK OR-verknüpft mit einem Random Wert des Key Servers). Der Keyserver ist entweder ein PAE oder der Access Point. Der SAK verschlüsselt die Data Plane.

SC: Secure Channel Jede SC umfasst eine Reihe von SA mit unterschiedlichen SAK's. Jeder SC ist unidirectional.

SA: Secure Association Eine Verbindung zwischen zwei Clients. Es gibt mehrere SA's innerhalb einer SC. Jede SA hat seinen eigenen SAK. Es dürfen mehrere Associations innerhalb von einem SA in eine Richtung koexistieren, damit die SAK's (der SA) unterbruchsfrei ausgetauscht werden können. Die SA's haben dann nur eine begrenzte Lebensdauer, während die SC's als Grundelemente einer CA immer existieren.

7.3.1. MKA: MACsec Key Agreement Protocol

Dank dem 802.1X MACsec Key Agreement (MKA) Protokoll können die SAK's mittels eines Key Servers dynamisch generiert und an alle Teilnehmer verteilt werden. Damit ist ein periodisches Rekeying möglich.

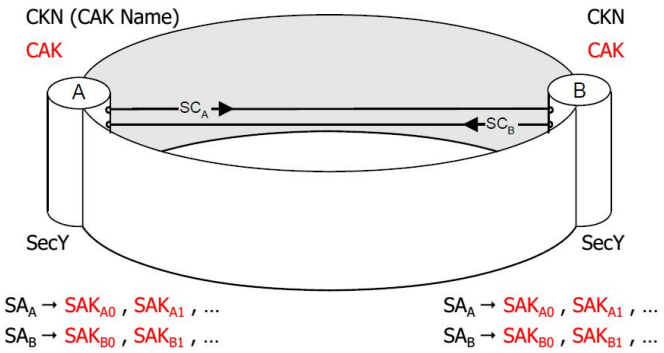

Abbildung 13: MACsec Secure Channel

7.3.2. SAI: Secure Association Identifier

Der Secure Association Identifier dient als ID für eine SA. Er besteht aus einem System Identifier, einem Port Identifier und einer Association Number. Die 2 Bit lange Association Number erlaubt es, dass währende dem Rekeying zwei verschiedenen SAK's koexistieren können.

7.3.3. MACsec Frames

MPDU: MACsec Protocol Data Unit Beinhaltet SecTag, Payload und ICV. Die PDU werden mit EAPOL verteilt.

MSDU: MAC Service Data Unit Payload

ICV: Integrity Check Value Ist der Authentication Tag der aus AES GCM/AES GMAC resultiert.

SA, DA Source und Destination MAC Adresse im Klartext

SecTag Security Tag vor dem Payload im Klatext (Ethertype 0x88E5)

- Das E Bit definiert ob verschlüsselt wird

FCS: Frame Check Sequence

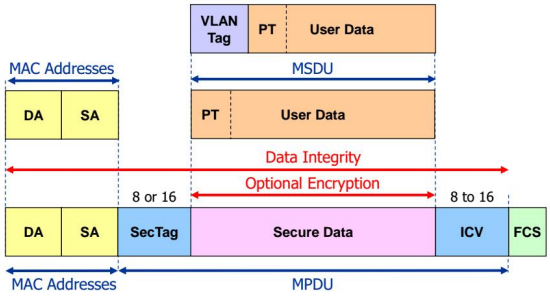

Abbildung 14: MACSec Frame

8. VPN: Virtual Private Network

Mittels VPN kann ein sogenannter Road Warrior (Client) von einem beliebigen Punkt auf der Erde via Internet auf sein heimatliches Firmennetz zugreifen. Bei IPsec geschieht das über einen verschlüsselten Tunnel, der zwischen Remote Access Client und dem Security Gateway des Heimatnetzes aufgebaut wird. Die Authentisierung der beiden Tunnelpunkte wird über ein X.509 Zertifikate bzw. Benutzername/Passwort bewerkstelligt. Es gibt verschiedene Varianten von VPNs wobei wir den Schwerpunkt auf IPsec legen. Aktuell wird IPsec mit IKEv2 empfohlen.

8.1. Terminologie

VPN Das konventionelle VPN bezeichnet einzig eine logische Trennung innerhalb einer physischen Leitung.

SSL VPN Erlaubt verschlüsselten Fernzugriff auf Unternehmensanwendungen um auf entfernte Ressourcen gesichert zuzugreifen.

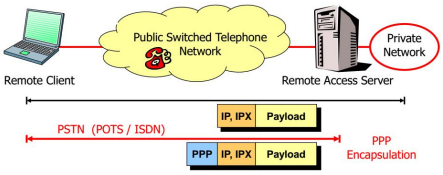

Abbildung 15: PPP Remote Access über Einwählleitung

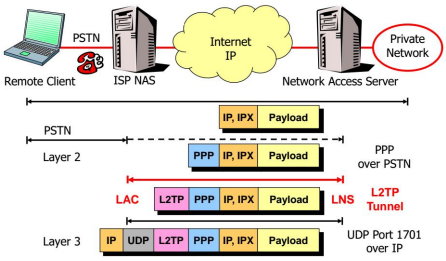

Abbildung 16: L2TP Remote Access

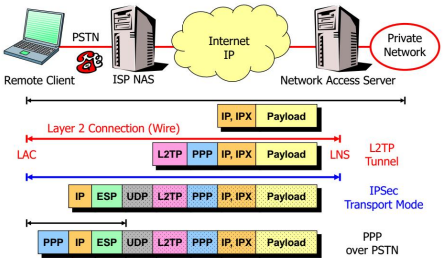

Abbildung 17: L2TP über IPsec

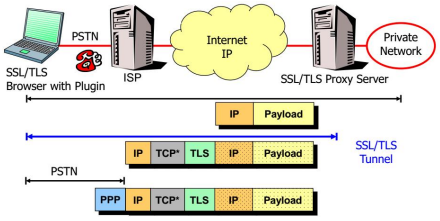

Abbildung 18: VPN als TLS Layer 4 Tunnel

8.2. MPLS-VPN: Multi-Protocol Label Switching

Bei MPLS muss beachtet werden, dass nichts verschlüsselt wird. Ein ISP möchte mit MPLS nur eine logische Trennung der Kunden durchführen.

8.3. IPsec: Internet Protocol Security

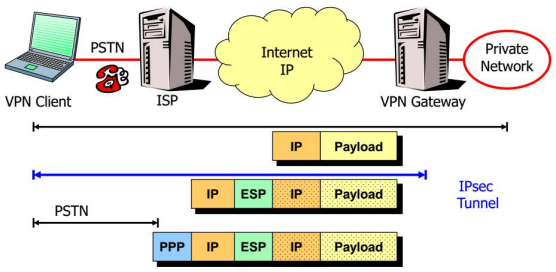

Abbildung 19: IPsec Remote Access

- IPsec ist eine Protokoll Suite (Bausatz). Dass heisst, man kann die einzelnen Teile miteinander kombinieren.
- IPsec definiert die Vorgehensweise für die Datenintegrität, die Vertraulichkeit der Inhalte sowie die Verwaltung der kryptografischen Schlüssel. Es besteht aus folgenden Bausteinen

  AH Authentication Header

  ESP Encapsulating Security Payload

  SA Security Association

  SPI Security Parameter Index (32Bit): Identifier der SA

  IKE Internet Key Exchange (Siehe Sektion 9)
- Zu Beginn der Kommunikation wird zwischen den Clients das benutzte Verfahren ausgehandelt. IPsec unterscheidet je nach Art der Verschlüsselung zwischen dem AH basierten Transportmodus und dem ESP basierten Tunnelmodus.
- Path MTU Discovery muss funktionieren. Somit kann IPsec den korrekten Overhead von der MTU abziehen und sendet so nie eine zu grosse MTU, die dann auf dem Weg gedropt wird. (Bsp. Ping funktioniert, Grösserer Download nicht → MTU Einstellungen prüfen, oder MTU manuell setzen.)
- IPsec hat seine Stärke besonders bei der Performance.
- Der äussere IP Header dient dem Routing zwischen den beiden VPN Gateways, Der innere IP Header beinhaltet die IP Adressen der beiden Clients.

8.3.1. AH: Authentication Header

- Meist als zusätzlicher IP-Header verpackt: IP Protokoll Nr. 51
- Es ist für die Authentizität und Integrität der übertragenen Pakete zuständig authentifiziert den Server
- Es schützt gegen Replay Attacken
- Der AH schützt das komplette Paket. (Ausgenommen sind Felder die auf dem Weg verändert werden. z.B TTL, ToS, Fragment Offset, IP Header Checksum)
- Ein Nachteil von AH ist, dass es inkompatibel mit NAT ist, da NAT Teile des IP Headers ändert, die gemäss AH nicht verändert werden dürfen.

Transport Mode

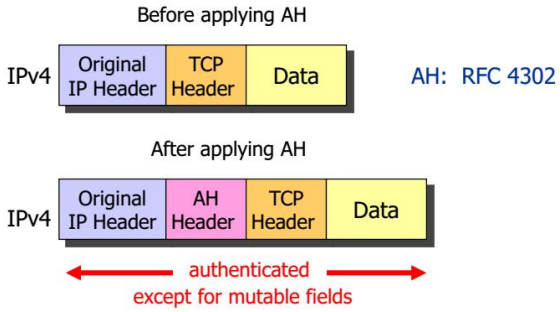

Abbildung 20: Transport Mode

Tunnel Mode

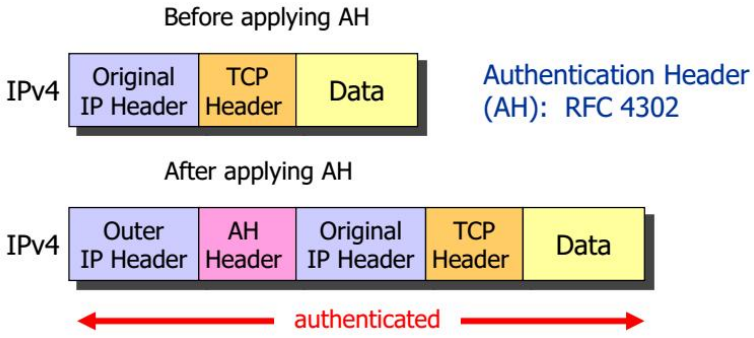

Abbildung 21: Tunnel Mode

8.3.2. ESP: Encapsulating Security Payload

- Ein zusätzlicher Header und Trailer um den Layer 4 Payload.
- IP Protokoll Nr 50 (Kein Port → Kein NAT, ausser ESP in UDP gepackt)
- Ist für die Sicherstellung der Authentizität, Integrität und Vertraulichkeit von IP Paketen zuständig
- Die Nutzdaten, Padding, Pad Lenght und Next Header werden verschlüsselt
- Im Unterschied zum AH wird der Kopf des IP-Paketes vom Integrity Check Value (ICV) nicht authentifiziert.

Transport Mode

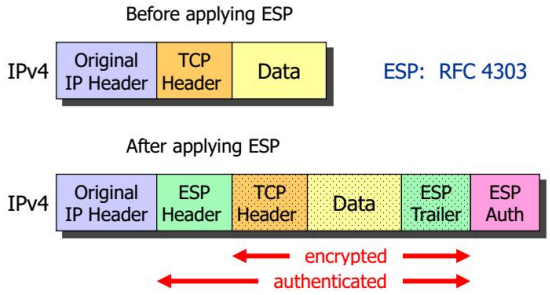

Abbildung 22: Transport Mode

Tunnel Mode

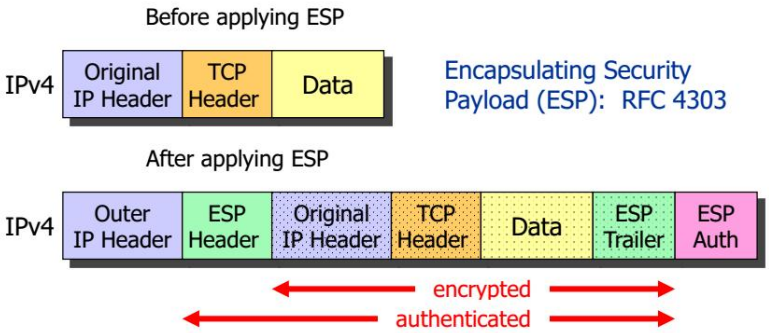

Abbildung 23: Tunnel Mode

NAT

Mit der NAT Traversal Extension werden ESP Pakete in UDP verpackt, welche dann problemlos von NAT gemappt werden könnnen. Zusätzlich werden die Daten neu an Port 4500 gesendet, damit alte NAT-Router kein IP-Basiertes Traversal machen. Der Grund dafür ist, dass ESP ein eigenständiges L3 IP Protokoll (50) ist und somit keine Ports besitzt.

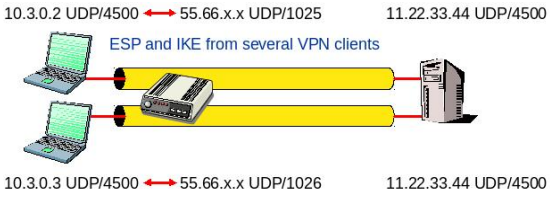

Abbildung 24: IPsec NAT

8.4. StrongSwan

Bei der Konfiguration von StrongSwan muss beachtet werden dass **left** die **lokale Domäne** und **right** für den **Remote** steht.

/etc/ipsec.d/cacerts CA Zertifikate

/etc/ipsec.d/certs User Zertifikate

/etc/ipsec.d/private User Keys

/etc/ipsec.secrets : ECDSA userKey.pem "your_secrete_pw"

Listing 1: /etc/ipsec.conf

```
conn %default
   leftcert=loginCert.pem // Cert das an die Gegenstelle gesendet wird
   // leftid = Per default distinguished name aus Cert

conn gateway-net
   also=gateway // reuse content of "gateway" connection
   rightsubnet= 10.5.0.0/16

conn gateway
   right= 152.96.31.50 // ip adresse der Gegenstelle
   rightid=intsec.hsr.ch
   auto=add

ca hsr
   cacert=hsrCert.pem
   ocspuri=http://intsec.hsr.ch:8880
   auto=add
```

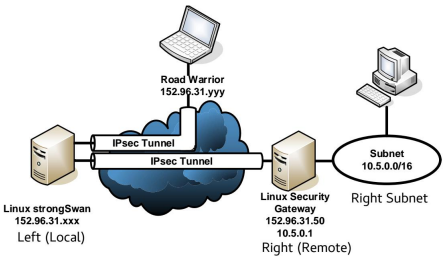

Abbildung 25: strongSwan Üubungs Aufbau

8.4.1. Beispiel

Für einen fixen Tunnel zwischen zwei Netzwerken (auf den Routern `deviceA` und `deviceB`)

deviceA

```
conn %default
   ikelifetime=60m
   keylife=20m
   rekeymargin=3m
   keyingtries=1
   authby=secret
   keyexchange=ikev2
   mobike=no

conn net-net
   left=deviceA.hsr.ch
   leftsubnet=192.168.1.0/24
   leftid=@deviceA.hsr.ch
   leftfirewall=yes
   lefthostaccess=yes
   right=deviceB.hsr.ch
   rightsubnet=192.168.2.0/24
   rightid=@deviceB.hsr.ch
   auto=route
```

deviceB

```
conn %default
   ikelifetime=60m
   keylife=20m
   rekeymargin=3m
   keyingtries=1
   authby=secret
   keyexchange=ikev2
   mobike=no

conn net-net
   left=deviceB.hsr.ch
   leftsubnet=192.168.2.0/24
   leftid=@deviceB.hsr.ch
   leftfirewall=yes
   lefthostaccess=yes
   right=deviceA.hsr.ch
   rightsubnet=192.168.1.0/24
   rightid=@deviceA.hsr.ch
   auto=route
```

9. IKE: Internet Key Exchange

Für den Verbindungsaufbau einer IPsec Verbindung ist das sogenannte Internet Key Exchange Protokoll zuständig. Seit 2005 kann das verbesserte IKEv2 verwendet werden. 4 Jahre später war Windows 7 die erste kommerziell verfügbare Implementierung des Standards.

- Das Internet Key Exchange (IKE) Protokoll ist für den Aufbau einer Security Association (SA) zuständig. Eine SA muss **einmal für jede Richtung (unidirektionl)** ausgehandelt werden.
- Das Protokoll dient der Verwaltung und zum Austausch der Schlüssel in IPsec. Zusätzlich bietet es ein standardisiertes Verfahren für die Authentifizierung von Kommunikationsparter sowie zur Erzeugung von gemeinsam genutzten Schlüsseln.
- UDP Port 500 (Socketverbindung) / 4500 (Übertragung)
- Aktuell gibt es zwei Versionen von IKE: Version 1 arbeitet in zwei Phasen und Version 2 braucht viel weniger Meldungen für den Verbindungsaufbau.

9.1. Version 1

- ISAKMP_SA handelt die Paramter für die IKE Übertragung selbst aus.
- Verwendet Pre-Shared Secrets oder Public Key
- Verwendet XAUTH zur erweiterten Authentisierung (Überträgt Passwort im Klartext)
- Ein Verbindungsaufbau zwischen zwei Endstations benötigt 15 Meldungen.

$$12 = 6(Phase1) + 3(Phase2 : RichtungA) + 3(Phase2 : RichtungB)$$

9.1.1. Ablauf

In der Version 1 wird der IPsec Verbindungsaufbau in zwei Phasen eingeteilt:

1. In einer ersten Phase baut IKE eine Verbindung mit relativ schwachen Sicherheitsmechanismen auf, die der Absicherung und Authentifizierung der weiteren Verwaltungsvorgänge dient. Nach der ersten Phase steht eine ISAKMP_SA, welche bidirektional ist. Dieser Schritt benötigt 6 Pakete (ausser im Agressive Mode)
2. In einer zweiten Phase wird das zu verwendende Sicherheitsprotokoll ausgehandelt und umgesetzt. Dazu wird der in der Phase 1 erstellte Tunnel verwendet. Für diese Phase werden für **jede Richtung** je 3 Pakete verwendet. Es resultiert eine ESP SA über welche dann die wirklichen Daten übertragen werden.
   - Zuerst schlägt der Client seine unterstützten Cipher Suites vor
   - Das Gateway antwortet dann mit seiner Auswahl
   - In dieser Phase werden auch die DH Schlüssel generiert

9.1.2. Phase 1: Modes

In einer ersten Phase handelt IKE die Parameter für die ISAKMP_SA aus. Dabei gibt es verschiedene Modes:

Main Mode Das Problem beim Main Mode ist die Performance. Zusätzlich besteht das Problem, dass ein MitM Angreifer den Session Key des DH Exchange abfangen kann, da die ersten 4 Pakete im **Klartext** übertragen werden. Das Zertifikat wird in den letzten beiden Paketen übertragen.

Main Mode mit Pre Shared Keys Die ersten vier Pakete sind gleich wie im Main Mode. Zusätzlich wird das Password (PSK) in den Hash eingearbeitet. Dies verhindert einen MitM Angriff. Ebenfalls ist somit die Indentität verschlüsselt.

Agressive Mode mit Pre Shared Keys Beim Aggressive Mode ist das Ziel möglichst schnell die Phase 1 abzuarbeiten. Man schickt deshalb mehr Informationen pro Paket. Das grosse Problem ist, dass der Hash unverschlüsselt übertragen werden. (beim 2. Paket)

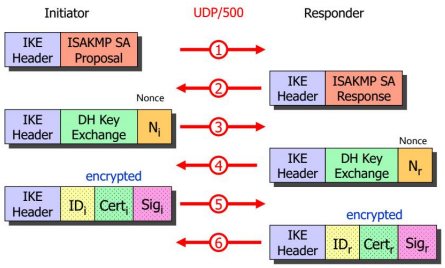

Abbildung 26: IKE Main Mode

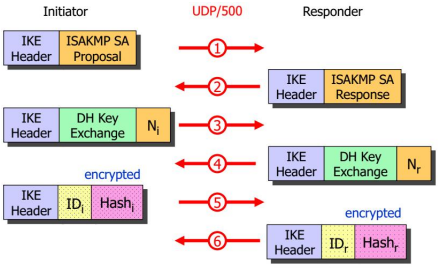

Abbildung 27: Main Mode mit Pre-Shared Keys

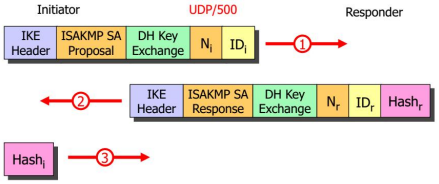

Abbildung 28: Agressive Mode mit Pre-Shared Keys

9.1.3. Phase2: Quick Mode

In der Phase zwei wird mit 3 weiteren Paketen die SA über den sicheren Kanal der in Phase 1 ISAKMP_SA aufgebaut wurde, erstellt. Die ISAKMP ist **bidirektonal**. Die IPsec SA besteht aus zwei Traffic Selectors. Es gibt einen Traffic Selector für den Initiator und den Responder. Der Traffic Selektor definiert also eine Endstation.

9.2. Version 2

- IKEv2 verwendet standardmässig nur 1024Bit DH für den Schlüsselaustausch.
- IKE SA ist das v2 Pendant zum ISAKMP_SA unter v1
- IKE v2 ist komplett neu, bis auf den Port 500
- IKE v2 verfügt über die Möglichkeit Child SA zu erstellen. Dies wird besonders beim Re-Keying verwendet.
- Verwendet Extensible Authentication Protocol (EAP) zur erweiterten Authentisierung
- IKE v2 unterstütz Cookie Mechanismus gegen DoS Attacken (nicht aber gegen DDoS, da die Zombies die Cookies zurücksenden)
- Bei IKEv2 wurde auf einen präventiven Cookie-Austausch verzichtet (Es gibt selten Probleme mit DOS). Der Cookie Mechanismus kann jedoch manuell aktiviert werden.
- Nur noch 4 anstatt den 9 Meldungen in V1
- Ein Verbindungsaufbau zwischen zwei Endstations benötigt 6 Meldungen:

$6 = 4$ (je 2 für IKE_SA und IKE_AUTH: beinhaltet bereits eine Child SA!) + 2 (Child SA)

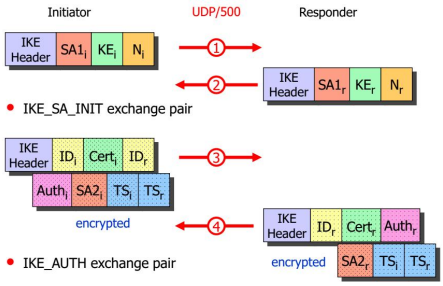

Abbildung 29: IKE V2: First Child SA

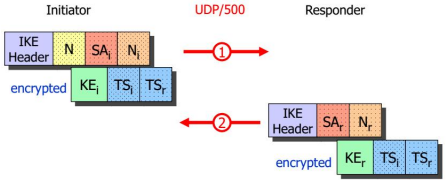

Abbildung 30: IKE V2: Additional Child SA

9.2.1. Ablauf

- IKE_SA_INIT Request und Response im Klartext
  - Im SA Payload sind die ausgehandelten Cipher Suites sichtbar
- IKE_AUTH Request und Response (Erstellt SA zum Gateway)
- CREATE_CHILD_SA Request und Response (Erstellt SA vom Gateway ins Netz hinter dem Gateway)

9.2.2. Narrowing

Narrowing vereinfacht die Konfiguration von IKEv2 um ein Vielfaches. Beim Narrowing wird automatisch das kleinste Subnetz genommen.

10. DNSSEC: DNS Security Extensions

10.1. DNS

- DNS basiert auf UDP und deshalb ohne erweiterten Sicherheitsmassnahmen (Source Port Randomization und Query ID Randomzation) einfach zu spoofen.
- Rekursive Auflösung: Bei der rekursiven DNS Server fragt der lokale Namensserver rekursiv alle benötigten externen Nameserver an, um die Adresse aufzulösen.
- Der Source Port ist im Standard nicht definiert. Der Destination Port von DNS ist immer 53
- Die QID (16 Bit Query ID) identifiziert einen DNS Request. Sie muss gleich wie die DNS Response Transaction ID sein.
- Query ID und Source Port müssen passen, damit ein DNS Server einen Request beantwortet.

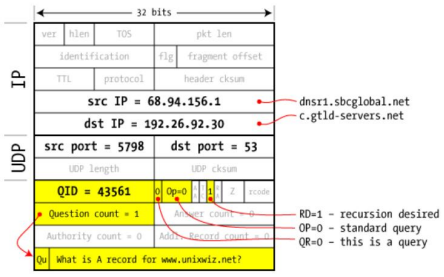

Abbildung 31: DNS Request

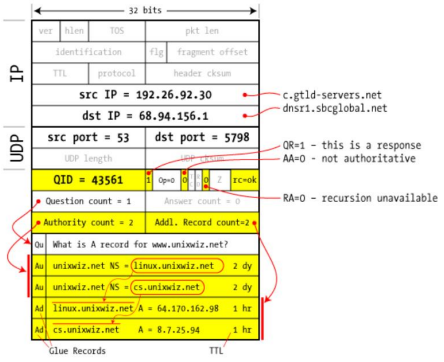

Abbildung 32: DNS Response

10.2. DNS Cache Poisoning

Beim DNS Cache Poisoning versucht ein Angreifer die Query ID und den Source Port zu erraten und dabei eine schnellere Antwort zu liefer, als das dies der korrekte NS macht. Der erste DNS Response der beim Client NS ankommt, wird in den Cache geschrieben und alle weitere Anfragen laufen dann, im schlimmsten Fall über den NS des Angreifers. Hat er sein Ziel erreicht, kann er alle DNS Requests und MX Records für eine Zone manipulieren.

10.3. DNSSEC Resource Records

Mit DNSsec kann man sicher gehen, dass die zurückgelieferten DNS Records vertrauenswürdig sind. Dazu muss man einzig dem KSK der Root Zone vertrauen. Dieser muss im DNS Resolver hinterlegt werden.

KSK: Key Signing Key (Flag=257) langfristiger Schlüssel (2048Bit RSA). Mit dem KSK wird wird der ZSK signiert.

ZSK: Zone Signing Key (Flag=256) Mit diesem werden alle Zone Records signiert. Dies wird bei jeder Generierung des Zonen-Files gemacht. (1024Bit RSA). Mit dem ZSK werden die DS Records der darunterliegenden Zonen signiert. (Root → Top Level → ..)

DS: Delegation Signer Der DS Record ist ein Hash über den Public Key (KSK) der darunter liegenden Zone. Erstellt man mit dem definierten Hash Algorithmus einen Hash über den Public Key kann dieser mit dem DS Record verglichen werden.

Listing 2: DNSKEY

```
1  domain.ch. <time to live> IN DNSKEY <flag> 3 <algorithm>
2  <base 64 public_key>; {id=<key tag id> (zks|ksk), size = <size>}
```

RRset Ein DNS Record Eintrag

DNSKEY Ist ein Base64 codierter Public Key. Mit diesem wird die RRSIG Signatur wieder entschlüsselt. Daraus resultiert ein Hash. Mit dem gegebenen Hash Algorithmus muss der RRset (z.B A Record) gehashed werden. Sind die beiden Hashes gleich, kann dem DNS Server vertraut werden.

RRSIG: Resource Record Signature Enthält die Signatur über ein RRset (z.B A Record, IPSECKEY Record oder DS Record)

Listing 3: RRSIG

```
1  sub.domain.ch. <time to live> IN RRSIG <record_type> <algorithm> 3 <time to
       live>
2  <valid from>
3  <valid to>
4  <key tag id> <parent domain>
5  <signature>
```

NSEC: Next Owner Name Mit NSEC Records ist es möglich, alle Namen einer Zone aufzulisten. Man beginnt mit einer Anfrage nach einem NSEC Record bei einem bliebigen Namen und erhält damit den nächsten gültigen Namen. Seit Version 3 werden die Records mit gehashed. Man erhält also einen Hash zurück. Dieser kann nur noch mit einer Dictionary Attack geknackt werden.

10.3.1. Ablauf: Chain of Trust

1. Die Zone `switch.ch.` hat einen KSK (DNSKEY Eintrag) und einen ZSK (DNSKEY Eintrag). Der KSK signiert den ZSK. Der ZSK signiert alle weiteren DNS Records. Daraus resultieren RRSIG Einträge (Hashes)
2. Die Zone `.ch.`, hat ebenfalls einen KSK (DNSKEY Eintrag) und einen ZSK (DNSKEY Eintrag). Auch hier signiert der KSK den ZSK. Der ZSK signiert alle weiteren DNS Records. Auch hier resultieren RRSIG Einträge (Hashes)
3. Damit dem KSK in der Zone `switch.ch.` vertraut wird, muss diese manuell einen DS Eintrag in der darüberliegenden Zone (.ch.) erstellen. (Aufgabe des Sysadmin: Anfrage bei Top Level Zone)
4. Dieser DS Eintrag wird vom ZSK der Zone ".ch." signiert. Es resultiert wieder ein RRSIG Eintrag.
5. Dies wird wiederholt bis am in der Root Zone angelangt ist.

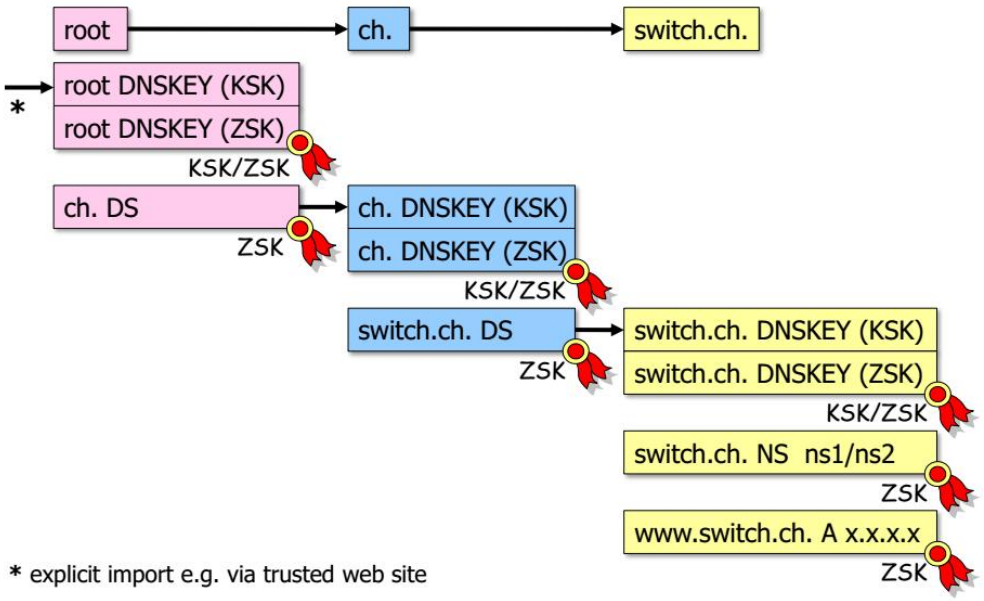

Abbildung 33: DNSsec Chain of Trust

10.3.2. Verifikation

Für die Verifikation muss der KSK der Root Zone, beim Client hinterlegt sein.

1. Zuerst stellt der Client eine Anfrage für einen beliebigen Record an `www.potaroo.net.`
2. Die dafür zuständige Zone antwortet mit den geforderten Record und den RRSIG zu diesem Record. Um diesen RRSIG zu überprüfen wird der ZSK der Zone benötigt.
3. Der ZSK an sich muss aber auch überprüft werden. (Der könnte ja gefakt sein) Man überprüft deshalb den RRSIG des ZSK mit dem KSK der Zone
4. Dem KSK kann aber auch nicht vertraut werden, solange er nicht verifiziert ist. Für den KSK existiert deshalb einen DS Eintrag in der darüberliegenden Zone.
5. Der DS Eintrag hat wiederum einen RRSIG Eintrag der mit dem ZSK der `.net` Zone erstellt wurde.
6. Die vorgehenden Schritte werden wiederholt, bis man bei der Root Zone "." angekommen ist. Die Root Zone an sich wird mit dem beim Client hinterlegten KSK der Root Zone validiert.

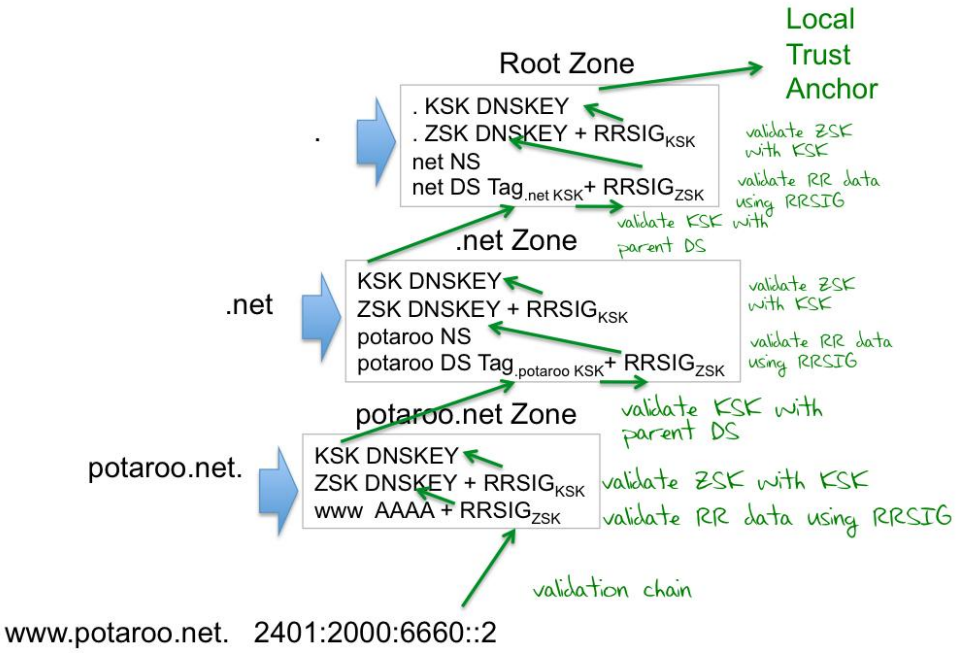

Abbildung 34: DNSSec Verifikation

10.4. DANE: DNS-based Authentication of Named Entities

Mit DANE können Clients den DNS Server anfragen, welche TLS Zertifikate als vertrauenenswürdig eingestuft werden können. DANE erweitert somit TLS, dass die verwendeten Zertifikate **nicht unbemerkt ausgewechselt** werden können. Dazu werden X.509 Zertifikate mit DNS Einträgen verknüpft und per DNSSec als TLSA Resource Record gesichert. Ausserdem können Domaininhaber eigene Zertifikate ausstellen und so die Dienste der CA's umgehen.

10.4.1. Funktionsweise

1. Ruft ein Client eine Webseite auf, möchte er sich sicher sein, dass der korrekte Server die Webinhalte ausliefert. Dazu prüft er das Zertifikat des Servers
2. Anschliessend muss er sicherstellen, dass das Zertifikat von einer vertrauenswürdigen CA ausgestellt wurde. Da es sehr viele CA's gibt, kamen Zweifel an der Verlässlichkeit der CA's auf. Hier setzt DANE an
3. Clients können mit können mit DANE bei den DNS Server nachfragen, welche Zertifikate sie als vertrauenswürdig einstufen können. Dies wird über einen neuen TLSA Resource Record gemacht. Dieser enthält das Zertifikat und dessen Finger oder public Key.

10.4.2. TLSA Resource Record

Abbildung 35: TLSA Resource Record

Certificate Usage

- 0 – CA Certificate Constraint
- 1 – Server Certificate Constraint
- 2 – Trust Anchor Assertion for Private CA
- 3 – Domain Issued Certificate

Selector

- 0 – Full Certificate
- 1 – Public Key Info (Public Key plus Key Type Information)

Matching Type

- 0 – Exact Match on Selected Content
- 1 – SHA-256 Hash of Selected Content
- 2 – SHA-512 Hash of Selected Content

11. VoIP Security

Die SIP (Session Initiation Protocol) Kommunikation ist standardmässig unverschlüsselt. Ein SIP Client registriert sich normalerweise bei einem SIP Proxy. Dabei muss sich der Client beim Proxy authentifizieren. Der Proxy verbindet dann zwei VoIP Client miteinander. Hat der Proxy die Verbindung mit dem Endknoten hergestellt (Callee) wird ein RTP Kanal zwischen den beiden SIP Clients hergestellt. Bei der RTP muss die Confidientiality und Data Integrity sichergestellt werden. Um eine ersten Schutz zu implementieren, stellt man die VoIP Phones in ein eigenes VLAN. Dies bietet aber nur einen minimalen Schutz, da es immer noch möglich ist, einen VLAN Tag zu faken.

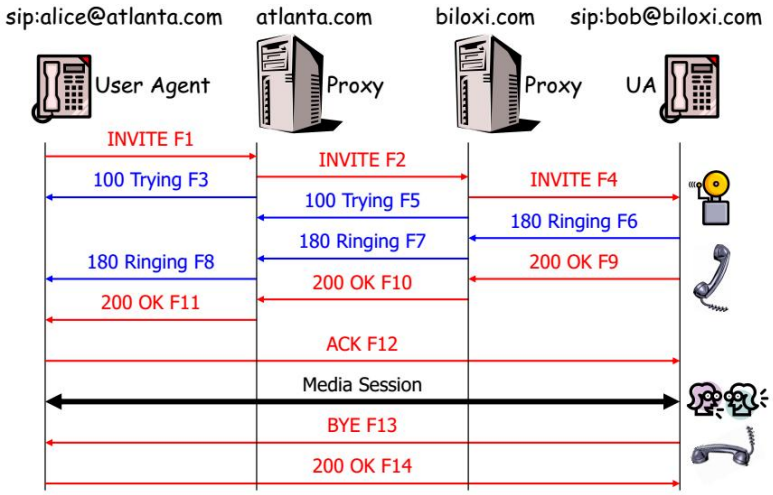

Abbildung 36: SIP Verbindung

11.1. SDES: Session Description Protocol Security Descriptions

SDES ist ein Verfahren um Schlüssel via SIP auszutauschen. SDES überträgt die Schlüssel im Klartext. Aus diesem Grund sollte die Verbindung zum Proxy mit SIPS verschlüsselt sein und man sollte dem Proxy trauen können.

11.2. ZRTP: Phil Zimmermann RTP

ZRTP wurde von Phil Zimmermann (PGP) entwickelt. Sobald die Verbindung über den Proxy aufgebaut wurde, handeln die beiden Clients über RTP einen Diffie Hellman Schlüssel aus. Die ausgehandelten Keys werden nach dem Verbindungsaufbau mit einer Prüfsumme (4 Zeichen = Base32 Encoding der 20 MSB des 32 Bit langen **Short Authentication String (SAS)**) verifiziert. Somit kann sichergestellt werden, dass kein MitM auf der Leitung sitzt, da dann verschiedene SAS angezeigt werden.

11.3. SRTP: Secure Real Time Protocol

Bei SRTP wird der RTP Payload verschlüsselt und der Header inkl. Payload gehashed. Standardmässig wird mit AES-CTR (Stream Cipher) mit der Stärke von 128Bit verschlüsselt. Der Vorteil des Stream Ciphers ist, dass die Payload nicht grösser wird (kein Padding). Die Sequenznummer wird als Counter bei der Verschlüsselung verwendet. Anhand des Tags erkennt man ob ein SRTP Paket verschlüsselt ist.

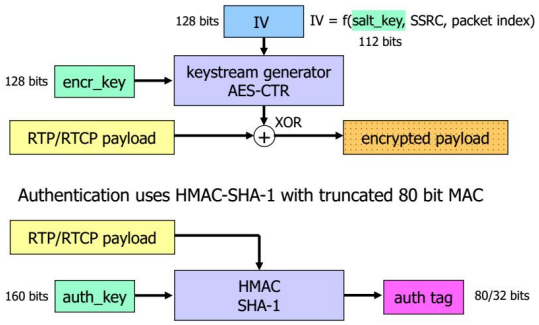

Abbildung 37: SRTP Encryption und Authentication mit AES CTR rsp. SHA-1

11.4. Session Key Derivation

Zwischen den zwei Stationen muss einzig der Master Key übermittelt werden.

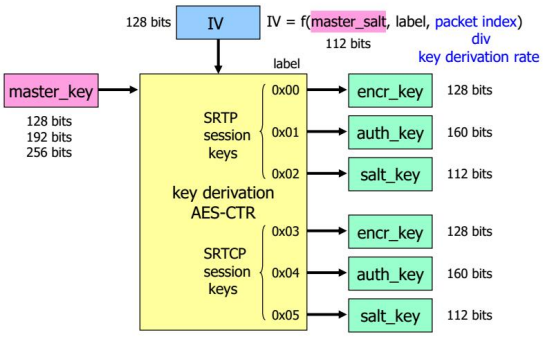

Abbildung 38: SRTP Key Derivation

11.5. SIPS: Session Initiation Protocol Secure

SDP (Session Description Protocol) ist für die Übermittlung des Master Keys zuständig. Um SDP zu schützen wird SIPS verwendet. SIPS erstellt TLS Verbindungen zwischen SIP Caller und Proxy, Proxy und Proxy und Proxy zu SIP Callee. (Hop to Hop) Das Problem ist, wenn jemand Zugriff auf den Proxy Server hat, (Staat oder Hacker) kann der Master Key gesnifft werden, da die TLS Verbindung nicht E2E ist.

Listing 4: Unverschlüsselte SDP Master Key Übertragung (crypto)

```
v=0
o=jdoe 2890844526 2890842807 IN IP4 10.47.16.5
s=SDP Seminar
i=A Seminar on the session description protocol
u=http://www.example.com/seminars/sdp.pdf
e=j.doe@example.com (Jane Doe)
c=IN IP4 161.44.17.12/127
t=2873397496 2873404696
m=video 51372 RTP/SAVP 31
a=crypto:1 AES_CM_128_HMAC_SHA1_80
inline:d0RmdmcmVCspeEc3QGZiNWpVLFJhQX1cfHAwJSoj|2^20|1:32
m=audio 49170 RTP/SAVP 0
a=crypto:1 AES_CM_128_HMAC_SHA1_32
inline:NzB4d1BINUAvLEw6UzF3WSJ+PSdFcGdUJShpX1Zj|2^20|1:32
m=application 32416 udp wb
a=orient:portrait
```

11.6. MIKEY: Multimedia Internet KEYing

MIKEY erlaubt End zu End Verschlüsselung für die Master Key Übermittelung. MIKEY wird hauptsächlich für Realtime Multimediaanwendungen im Zusammenhang mit SRTP eingesetzt wobei entweder das PSK, **Public Key oder Diffi-Hellman** Verfahren verwendet wird.

11.7. IPsec

VoIP Kommunikation kann auch über ein IPsec VPN Tunnel geschickt werden. Dies erlaubt ebenfalls End zu End Verschlüsselung, hat aber den Nachteil, dass der Overhead recht gross ist.

12. Anonymität

Beim surfen im Internet hinterlässt man immer Spuren. (Log Einträge auf Webserver, Mail Transfer Agents, Referer Einträge, ISP Monitoring, etc.). Unter Lawful Inspection versteht man, dass der Staat beim ISP diese Daten analysieren darf. Anonymität ist deshalb wichtig, um die Privatsphäre einer Person zu schützen.

12.1. Pseudo Anonymous Remailers

Eine Mail wird über einen Remailer gesendet. Dieser ändert die Absendermail und arbeitet als Proxy. Die Informationen auf dem Remailer sind dort aber nicht wirklich sicher. (Hacking, Gerichtsbeschluss)

12.2. Davis Chaum's Cascade of Mixes

Bei Mix Kaskaden arbeiten mehrere Server für die Anonymisierung. Der Benutzer bestimmt, welche Knoten für den Weg gewählt werden. Durch global stationierte Knoten, werden die Gültigkeitsbereiche des lokal geltenden Rechts überschritten. Essentiell ist, dass im Netzwerk viel Verkehr herrscht. Dabei ist sekundär, wie viele Knoten zur Verfügung stehen. Davis Chaum's Mix Kaskade sieht den Einsatz von RSA vor

1. Die zu verwenden Knoten werden bestimmt. Von jedem Knoten ist der Public Key bekannt.
2. Die Payload wird gepadded, damit alle Pakete die selbe Grösse haben
3. Danach wird das Frame mit dem Public Key des Exit Nodes verschlüsselt und die Adresse des Exit Nodes angehängt
4. Dies wird entlang dem Pfad rückwärts wiederholt, bis der Entry Point erreicht wird.
5. Das Paket hat dann die Adresse des Entry Points als Zieladresse
6. Der Entrypoint kann seinen Teil des Paket entschlüsseln und findet dann die Zieladresse des nächsten Hops. Damit das Paket immer noch gleich lang ist (ansonsten könnte man erkennen, wo in sich das Paket in der Kette befindet) wird hinten am Paket Junk eingefügt.
7. Beim Exit Knoten wird der Junk/Padding entfernt und dem Zielserver die Daten übermittelt.

12.2.1. Eigenschaften

- Eingehende Pakete müssen immer auf eine zufällige Weise umgeschlüsselt werden, damit keine Input/Output Korrelationen möglich sind.
- Duplikate müssen unterdrückt werden. Man erstellt dazu Hashes über die Pakete und merkt sich diese.
- Mix Cascaden verwenden sogenannte Resorting Buffer, damit keine Rückschlüsse auf eingehende und ausgehende Pakete getroffen werden können. Der Resorting Buffer wird bei einem Knoten solange gefüllt, bis dieser voll ist. Anschliessend werden alle Pakete in einer Aktion an den nächsten Knoten weitergeleitet. Tor verwendet im Gesensatz zu Davis Chaum's wenige bis keine Resorting Buffers, was dazu führt das die Latenz viel geringer ist. Um auf Resorting Buffer verzichten zu können, müssen die Mix Ketten ständig gewechselt werden.

12.3. Tor: The second-generation Onion Router

Tor ist ein Anonymisiserungsnetzwerk das eine bi-direktionale Kommunikation erlaubt. Es ist wichtig dass die Daten verschlüsselt übertragen werden (z.B HTTPS), da ein Exit Node ansonsten die unverschlüsselten Daten mitlesen könnte.

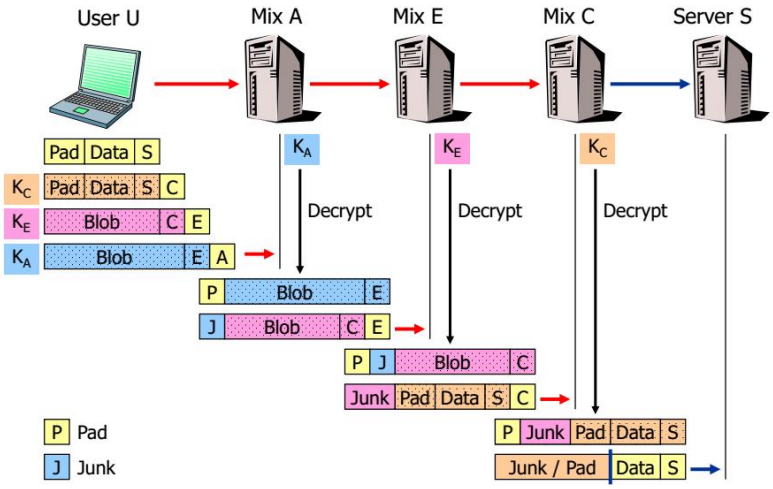

Abbildung 39: Onion Routing

12.3.1. Circuits

Circuits werden Schritt für Schritt aufgebaut, wobei Perfect Forward Secrecy garantiert wird. Ebenfalls benutzt der Client keinen Public Key, weshalb er anonym bleibt. Die Circuit ID wird immer vom Absender gesetzt.

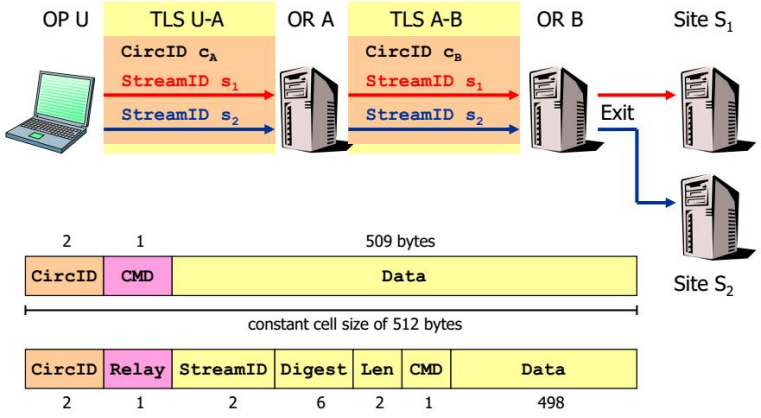

Abbildung 40: Cells, Circuits und Streams

12.3.2. Vorgehen

1. Der Client sendet die erste Hälfte des DH Handshake verschlüsselt an den Entry Point. g^X (Create Cell)
2. Der Entry Point sendet die zweite Hälfte des DH Handshake, sowie einen Hash über den ausgehandelten Key ($K = g^{XY}$) zurück an den Client. g^Y (Created Cell)
3. Nun muss der Circuit um weitere Nodes erweitert werden. Dazu sendet der Client ein Relay Extend Cell zum Entry Point, mit der Information, welcher der nächste Node im Circuit sein soll. Die Meldung enthält zusätzlich einen neuen ersten Teil des DH Handshake mit dem nächsten Node
4. Der Entry Point leitet kopiert den ersten Teil des DH in eine Create Cell und leitet sie an den nächsten Node weiter. Ebenfalls wählt der Entry Point eine neue Circuit ID zwischen ihm und dem neuen Node.
5. Der neue Node antwortet wieder mit einer Created Cell und der Entry Point mit einem Relay Extended. Nun kennt der neue Node und der Client den neuen ausgehandelten Key $K_2 = g^{X_2 Y_2}$
6. Dieser Schritt wird nun noch mindestens einmal wiederholt.

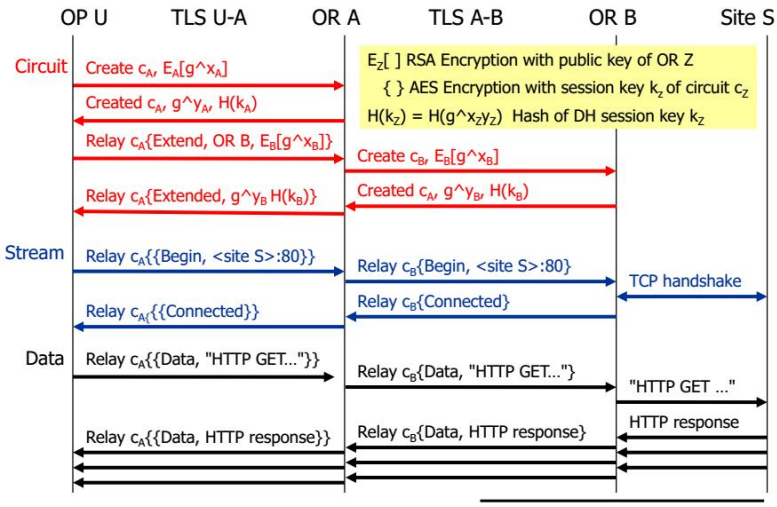

Abbildung 41: Erstellung eines Tor Circuit

12.4. Hidden Services

1. Ein beliebiger Nutzer möchte sensitive Daten über Tor verbreiten. Er setzt dafür einen Webserver auf und konfiguriert Tor so, dass die Daten auf dem Webserver gefunden werden könnnen (`etc/tor/torrc` → `HiddenServiceDir`).
2. Nach dem Neustart wird ein Schlüsselpaar erstellt, das den Hidden Service identifiziert

3. Der Hidden Service baut Tor Circuits zu zufälligen Introduction Points auf und sendet ihnen seinen Public Key. Danach erstellt er einen Hidden Service Descriptor. Dieser besteht aus genau diesen Introduction Points und seinem Public Key. Der Descriptor wird mit dem Private Key signiert und auf einen Verzeichnis Server geladen. Der Hidden Service ist nun verfügbar.

4. Möchte der Client auf den Hidden Service zugreifen, muss er einen Hash über den Public Key des Service bilden. (`16_char_hash.onion`). Mit diesem Hash kann der Service im Verzeichnisdienst abgefragt werden. Er erhält darauf die Liste der Introduction Points, sowie den Public Key. Der Client erhält den Public Key auf eine beliebige Art und Weise (weitersagen, über Tor, etc.)

5. Der Client baut eine Verbindung zu einem zufälligen Rendezvous Point auf und übermittelt diesem ein One Time Secret. (Cookie) Dieses Cookie wird dort hinterlegt.

6. Zusätzlich wird eine Verbindung zu einem der vorliegenden Introduction Points hergestellt. Der Client sendet dann eine mit dem Public Key verschlüsselte Introduction Meldung an einen ausgewählten Introduction Point und verlangt, dass die Meldung an den Hidden Service weitergeleitet wird. Diese Meldung enthält das One Time Secret (Cookie), der erste DH Teil, sowie der ausgewählte Rendezvous Point.

7. Der Hidden Service entschlüsselt die Meldung und weiss nun über den Rendezvous Point bescheid. Durch das enthaltene One Time Secret (Cookie) kann die Verbindung vom Rendezvous Point zwischen Client und Service verbunden werden. Dazu sendet der Hidden Service das Rendezvouz Cookie und der zweite DH Teil an den Rendezvouz Point.

8. Beide können nun Daten austauschen ohne ihre gegenseitige Identität zu kennen. Ebenfalls weiss der Rendezvouz Point weder über Alice und Bob, noch die Daten die die beiden senden, bescheid!

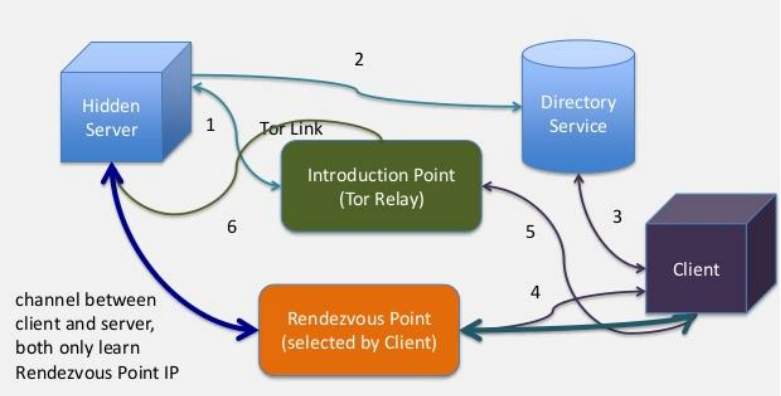

Abbildung 42: Hidden Service

12.4.1. RP: Rendezvous Point

Der Rendezvous verbindet den Client mit dem Hidden Service. Wird er mit einer DDoS Attacke in die Knie gewungen, kommt keine Kommunikation zu stande.

13. Firewalls

Heutzutage sind alle Firewalls Stateful Inspection Firewalls. Früher waren die Firewalls nur Paketfilter, was zur Folge hat, dass man extrem viele Regeln hatte.

Outer Perimeter DMZ

Inner Perimeter Intranet

Mission Cirtical Systems HR, Finance, Know How

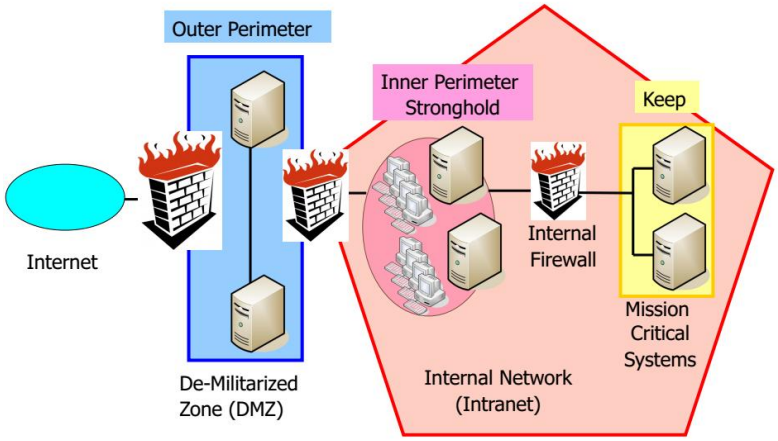

Abbildung 43: Firewalls

14. Intrusion Detection

Firewalls können nicht sämtliche Angriffe zuverlässig abwehren. Aus diesem Grund sind Intrusion Detection Systeme nötig, um festzustellen ob, jemand die Firewalls überwinden konnte. Man unterscheidet zwischen zwei Typen von IDS.

HIDS: Host-based IDS Überwachen einzelne Hosts. Man installiert ein Stück Software auf dem Host, das als Sensor agiert und z.B die Logs durchforstet.

- Vorteile: Relative wenige False Positives
- Nachteile: Sieht nur einen Hosts und kann mit einem Rootkit unterwandert werden. Verursacht zusätzliche Last auf dem Host

NIDS: Network IDS Überwachen ganze Netwerk Segmente. Man verbindet spezielle NIDS Geräte mit dem Monitorport eines Switches/Router.

- Vorteile: Sieht ein grösseres Spektrum wie HIDS
- Nachteile: Hat Probleme mit verschlüsseltem Verkehr, benötigt zusätzliche Hardware

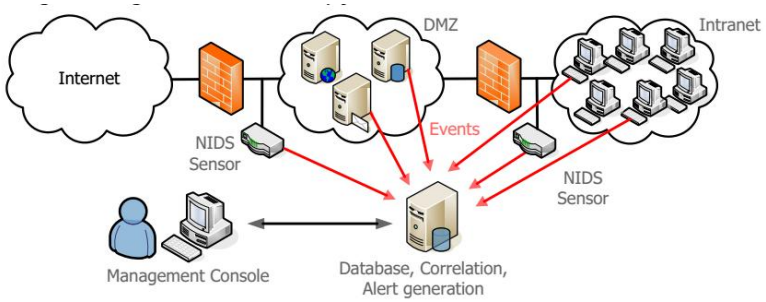

Abbildung 44: Hybrid IDS: HIDS kombiniert mit NIDS

14.1. Terminologie

Sensor Sensort der System und Netzwerkaktiviäten überwacht

Event Ein Sensor generiert einen Event, wenn er es für Nötig hält, etwas zu melden.

Alert Ein oder mehrere Events können in einem Alarm eskalieren (Abhängig von den konfigurierten Regeln). Ein Alert hat menschliche Interaktion zur Folge

Datenbank und Correlation Engine Ein zentralisierter Host sammelt Events von verschiedenen Sensoren, korreliert diese und erstellt Alert wenn nötig

Management Console Hierüber wird das IDS konfiguriert und Reports generiert

False Positive Fehlerhafter ALert den von einem IDS ausgelöst wird

False Negative Ein Angriff der von einem IDS nicht erkennt wird

14.2. Erkennung

IDS arbeiten mit zwei Techniken, um Angriffe zu erkennen. Aktuell domierniert die Signatur Variante, wobei auch Hybrid Variaten genutzt werden.

Signaturen Man erstellt Signaturen und prüft die Daten gegen diese Signaturen. z.B Suche nach `/etc/passwd` in Paket. Das Problem bei dieser Variante ist, dass sie neuartigen Attacken oder Variationen (anderes Encoding, Fragmentierung, etc.) immer hinterherhinkt.

Anomaly Detection Man vergleicht die überwachten Daten mit jeden, die man als normal betrachtet.

14.3. Reaktion

Auf einen Alert kann ebenfalls auf verschiedenne Weisen reagiert werden

Nichts Unternehmen Nichts unternehmen und einfach nur Loggen

Manuell Manuell intervenieren, was sehr langsam ist

Automatisch Die Verbindung automatisch unterbrechen und dynamisch die Firewalls Rules anpassen

14.4. Evasion

Fragmentierung IDS Systeme können auch umgangen werden, wenn man z.B die Pakete fragmentiert. (z.B `nmap -f --send-eth`) Die meisten IDS Systeme sind aber standardmässig so eingestellt, dass sie die fragmentierten Pakete reassemblieren. Ebenfalls könnte die Reihenfolge der IP Fragmente geändert werden.

Verzögerung Ein Port Scan könnte vom Angreifer verzögert werden, damit das IDS die einzelnen Zugriffsversuche nicht als einen zusammenhängenden Portscan erkennt.

Signatur "anpassen" Eine andere Variante wäre es, die Pakete so anzupassen, dass die Snort Rules (Signaturen) nicht mehr greifen. Dafür ist aber ein erweitertes Wissen über die Rules nötig.

SYN Scan Ein Angreifer könnte bei einem Portscan, nicht einen kompletten TCP Handshake durchführen (ACK dropen) und hoffen, dass der ISP nur komplette TCP Handshakes beachtet.

Distributed Der Portscan von mehreren Clients aus durchgeführt werden.

14.5. Snort

Snort ist ein freies Network Intrusion Detection System (NIDS) und ein Network Intrusion Prevention System (NIPS). Es kann zum Protokollieren von IP-Paketen genauso wie zur Analyse von Datenverkehr in IP-Netzwerken in Echtzeit eingesetzt werden.

```
1  [alert|log|pass|activate|dynamic] [tcp|udp|icmp|ip] [src ip|any] [src port|any]
       [->|<>] [dest ip|any] [dest port|any] ( [option]:""; msg:""; )
2
3  // example
4  alert icmp any any -> 192.168.0.150 any (itype:8; msg:"ICMP Echo Request detected";
       sid: 1000001)
5
6  // preprocessor (detects nmap -sS 192.168.0.0/24)
7  preprocessor sfportscan: proto { TCP } scan_type { all } sense_level { medium }
       watch_ip { 192.168.0.0/24 } logfile { /var/log/snort/portscan }
8
9  // reassembly fragmented packages
10 preprocessor frag3_global: max_frags 65536
11 preprocessor frag3_engine: policy first detect_anomalies
```

15. Network Access Controll

Die Network Access Controll (NAC) ist eine Technologie, um unauthorisierte Zugriffe auf das Firmennetzwerk zu unterbinden. Mit NAC werden Clients während der Authentisierung auf Richtlinienkonformität geprüft. Neben einer Authentisierung wie bei MACSec (Kapitel 7.3) ist es mit der NAC Policy Enforcement möglich, Client-Konfigurationen vorauszusetzen, wie z.B. bestimmte Betriebssystem- und Softwareversionen, Binary Integrity Checks oder Netzwerkkonfigurationen. Diese werden jeweils vom NAC Policy Enformcement Point vom Client abgefragt; je nach dem wird darauf ein bestimmter Zugang zum Netzwerk ermöglicht.

Virenverseuchten Computern kann so (im Prinzip) der Zugriff auf das Firmennetzwerk verboten werden; bei zu alten Software-Versionen könnte der Client in ein isolierte Umgebung (VLAN) eingebunden werden, um Updates durchzuführen.

15.1. Aufgaben

Kernaufgabe sind:

- eindeutige Identifizierung und Rollenverteilung von Nutzern und Geräten
- Wahrung von erstellten Sicherheitsrichtlinien
- Quarantäne und automatische Wiederherstellung nichtkonformer Endgeräte
- Verwaltung und Erstellung individueller Richtlinien und Rollen für verschiedene Nutzergruppen

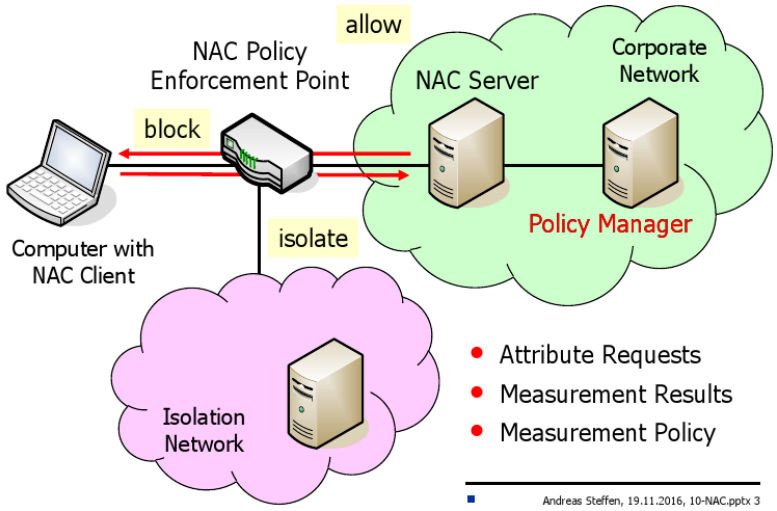

Abbildung 45: Network Access Control

15.2. TNC: Trusted Network Connect

Im Gegensatz zu Network Access Control (NAC) und Network Access Protection (NAP), handelt es sich beim TNC-Konzept um eine offene Architektur für die Netzwerkzugangskontrolle.

15.2.1. Standard

Es gibt bei der Trusted Computing Group und IETF einige Standards, welche allerdings noch in den Startlöchern sind, und einige proprietäre Lösungen (z.B. von Microsoft, Cisco und HP). Jeder Herrsteller ist befugt eigenen Attribute in seinem Namespace hinzuzufügen. Die Standardattribute des IETF können somit erweitert werden.

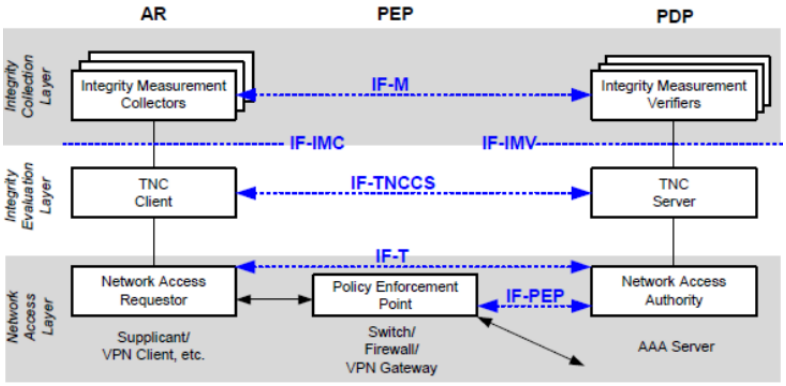

Abbildung 46: Trusted Network Connect

Die Integrity Measure Collectors sammeln auf der Clientseite alle relevanten Informationen (Antivirus-Status, Netzwerk-Status, OS-Status etc.)

Über das IF-TNCCS (Interface-TNC Client Server Protokoll) wird der aktuelle Stand des Clients an den Server gesendet. Der TNC-Server entscheidet daraufhin, ob und was für ein Zugriff erlaubt wird.

15.2.2. Verbindungsaufbau

Funktioniert über IKEv2 with EAP. Um MITM zu verhindern, kann der Client zuerst ein Zertifikat beim Authentisierungsserver anfragen (siehe Kapitel zu IPSec 8.3), daraufhin finden eine EAP-Identity & Challange-Responste mit PSK statt.

Üblicherweise findet die Kommunikation über ein TNC IF-T Protokoll via IKEv2 EAP-TTLS. Dies ist allerdings nicht der Standard (dank Intervention von Cisco).

Standardmässig findet die Kommunikation über ein spezielles Tunneled-EAP-Protokoll statt.

15.2.3. Forcierung der Polycies

Üblicherweise wird die Policy einmal beim einwählen in das Netzwerk sowie periodisch geprüft (z.B. alle 20 Sekunden).

15.2.4. Problem des "Lying Clients"

Die Authentizität der Antwort des Clients kann mit dem TPM (siehe Kapitel 18.1) verifiziert werden.

15.2.5. Terminologie

TNC: Trusted Network Connect Offene Architektur zur Netzwerkzugangskontrolle von TCG

AR: Access Requestor Ersucht Zulassung in ein Netzwerk

PEP: Policy Enforcement Point Setzt die Richtlinien des PDP bezüglich Netzwerkzugang durch. (z.B verschieben in eine isolierte Zone)

PDP: Policy Decision Point Trifft die Authorisierungsentscheidungen, bei einer Anfrage durch den AR.

TNCC: TNC Client Der Client kommuniziert über das IF-TNCCS Interface mit dem Server.

TNCS: TNC Server Der Server kommuniziert über das IF-TNCCS Interface mit dem Client.

IMC: Integrity Measurement Collectors Erlaubt TNCC und TNCS Plugin Komponenten von anderen Hersteller zu verwenden. Die gesammelten Daten werden über das IF-IMC Interface an den TNC Client gesendet.

IMV: Integrity Measurement Verifier Erlaubt TNCC und TNCS Plugin Komponenten von anderen Hersteller zu verwenden. Die Daten werden über das IF-IMV Interface an den TNC Server übertragen.

PTS: Platform Trust Service

MAP: Metadata Access Point Speichert und liefert Statusinformationen zu einem AR, die z.B im PDP/PEP benötigt werden.

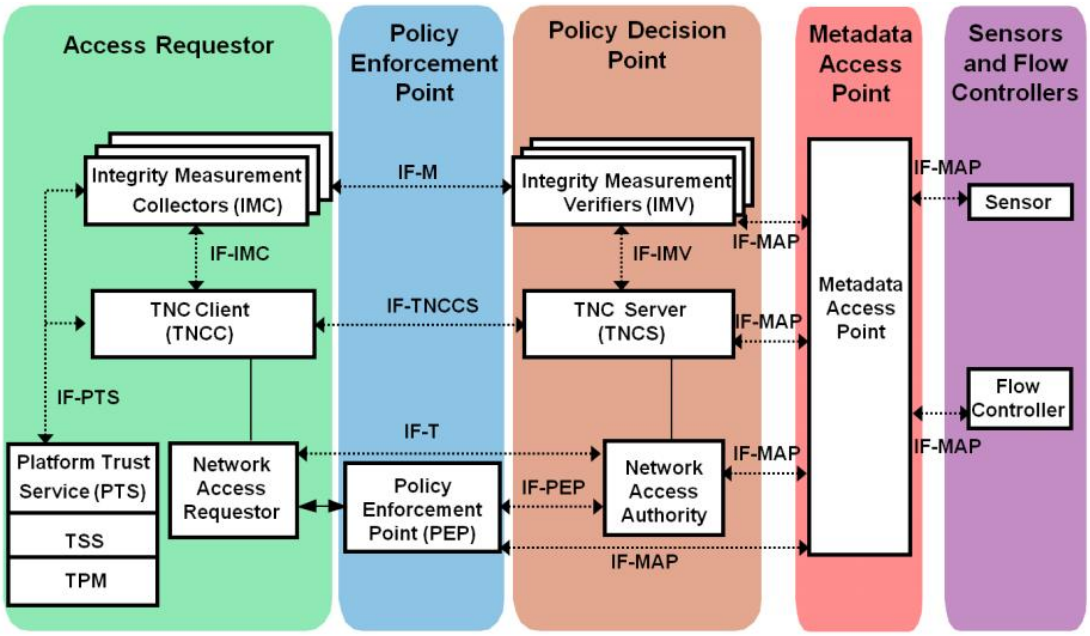

Abbildung 47: TNC Framework er Trusted Computing Group

16. Buffer Overflow

Normalerweise werden Buffer Overflow Attacken Remote ausgeführt. Damit die Attacke nicht zu viel Schaden ausrichten kann, sollten Server Daemon's nie mit root Rechten ausgeführt werden. Heutzutage sind Buffer Overflows nicht mehr ganz trivial durchzuführen. Die meisten aktuellen Angriffe gehen öfters auf den Heap.

Das Ziel des Buffer Overflows ist es, die Rücksprungadresse (Instruktionspointer) auf dem Stack zu manipulieren, dass dieser auf den Exploit zeigt. Damit die Exploits robuster werden, wird am Anfang NOP (No Operation) hinzugefügt und die neue Rücksprungadresse (EIP) mehrere Male am Schluss angehängt wird. Somit hat man mehr spielraum, um die Rücksprungadresse korrekt zu überschreiben.

- Nie `strcpy()` verwenden sondern `strncpy()`
- `execve()` übernimmt den aktuellen Prozess.

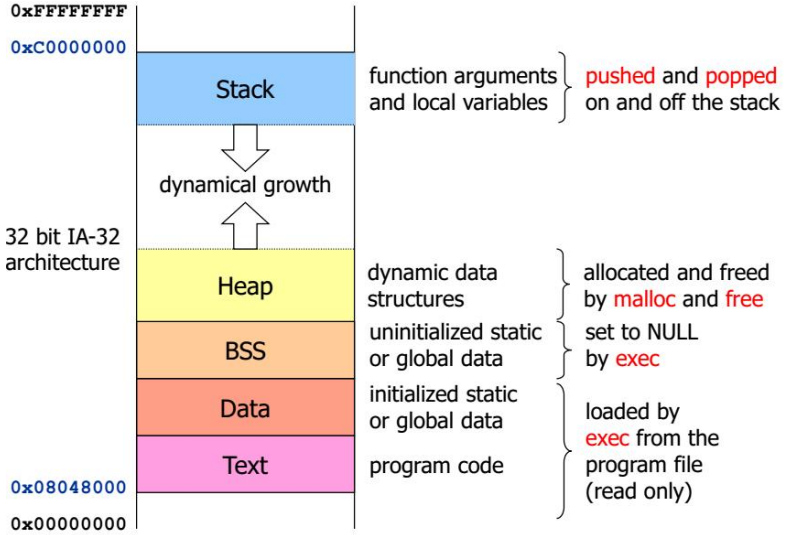

Abbildung 48: Virtual Process Memory Organization

IP: Instruction Pointer Adresse des nächsten Befehls (%eip Register)

SP: Stack Pointer Zeigt immer auf TOS (%esp Register). Je tiefer die Adresse desto grösser ist der Stack.

BP: Base or Frame Pointer Speichert den letzten Stackpointer (%ebp Register)

call call wirft die nächste Adresse auf den Stack, damit nach Ablauf des Stackframes am richtigen Ort weitergefahren wird.

16.1. Schutz

ASLR: Address Space Layout Randomization Der Anfang des Stacks wird zufällig gewählt. Dies kann jedoch umgangen werden, wenn man die Adresse einer bekannten Umgebungsvariable kennt und dann das Offset ausrechnet.

Canaries Werden zwischen Buffer und EIP zwischengeschoben. Die Canaries beginnen mit 0x00 und sind ansonsten komplett zufällig. Wird ein Buffer überschrieben, wird auch das Canarie überschrieben. Danach wird das Canarie aktuelle Canarie mit dem gemerkten XOR verknüpft und so ein allfälliger Buffer Overflow detektiert.

Exetuable Space Protection (Non-eXexute, NX bit) Markiert das physische Memory als nicht ausführbar (Stack, Heap). Funktioniert nur auf 64Bit Systemen und in 32Bit Systememn mit aktiviertem PAE Adressierungsmodus.

17. Smartcards

Da Schlüssel auf der Festplatte rsp. Memory eines Notebooks nicht sicher sind, sollten Verschlüsselungs und Signatur-Keys immer auf einer Smartcard gespeichert werden. Von diesen ist es extrem schwierig, den Key zu klauen. Ein möglicher Nachteil ist es, dass dem Chipkarten-Hersteller vertraut werden muss, dass er einen zuverlässigen Schlüssel mit viel Entropie auf der Karte generiert.

- Chipkarten haben oft eine tiefe Taktfrequenz, damit weniger Leistung benötigt wird.
- Chipkarten haben eine erhöhte physiche Sicherheit gegenüber herkömmlichen Prozessoren.
- Wegen der mechanischen Belastung, sind Chipkarten Prozessor sehr klein (2-3mm)
- Die kontaktlosen Zahlungskarten operieren auf der 13.56MHz Frequenz. Man muss deshalb so nah an den Leser gehen, damit genügend Spannung für den Kartenprozessor übertragen werden kann. (für die Spannungsübertragung ist eine magnetische Kopplung nötig)

17.1. NFC: Near Field Communication

Bei NFC gibt es das Problem, dass die Übertragung nicht sicher ist, wenn das Smartphone verseucht ist. Die einzige Lösung wäre es, dass die Mobile App über die sichere SIM Karte mit dem NFC Controller kommuniziert. Da die SIM Karte aber Eigentum der Netzbetreiber ist, wollen diese eine grosse Provision, weshalb die Kommunikation nach wie vor direkt auf den NFC Controller geht. Der NFC Controller kommuniziert mit dem Lesegerät. Es können dann beliebig viele Abbuchungen getätigt werden.

17.2. Memory Layout

ROM Betriebssystem

EEPROM Keys

RAM Zwischenspeicher für Berechnungen

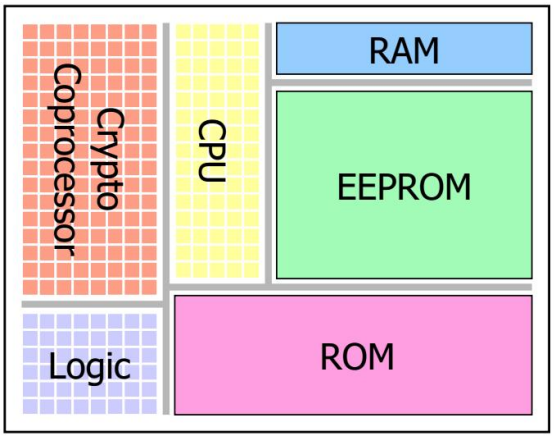

Abbildung 49: Memory Layout

17.3. PKCS #15: Cryptographic Token Information Format

Es gibt ein Master File (Root Directory) und mehrere Dedicated (Folder) und Elementry Files (Daten). Ein Filename besteht aus 2Bytes (meist HEX). Die Anzahl Schreibzugriffe ist begrenzt.

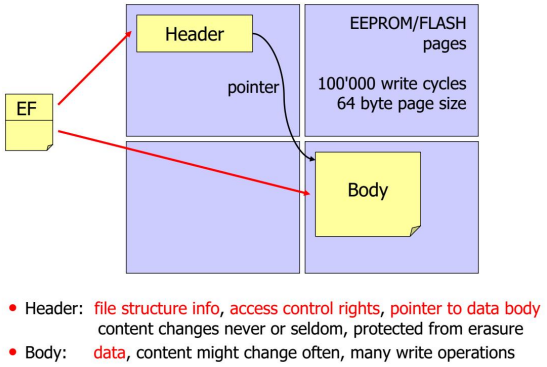

Abbildung 50: Smart Card File Struktur

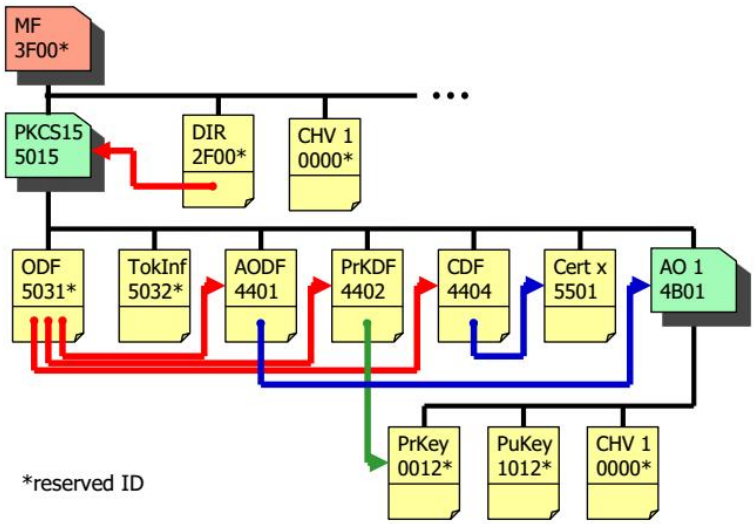

Abbildung 51: PKCS 15 Filesystem

MF Das Root File existiert immer (FID=3F00)

DF: Directory File Ein Ordner

EF: Elementary File Ein Daten File

Cert Beinhaltet das Zertifikat

PIN und PUK #1 Reservierte FID=0000

PIN und PUK #2 Reservierte FID=0100

FID=2F00 Dieses standardisierte Verzeichnis hält einen Zeiger auf das PKCS#15 Applikationsverzeichnis mit der FID=5015

PKCS15 PKCS#15 Applikationsverzeichis mit der FID=5015, enthält die eigentlichen Daten (Zertifikate und Schlüssel). Die Daten wiederum können sich sowohl im Verzeichnis 5015, als auch in jedem beliebigen anderen Verzeichnis auf der Chipkarte befinden.

ODF Objektverzeichnis ist die wichtigste Datei. Es enthält einen Zeiger auf das aktuell verwendete Verzeichnis, welches wiederum einen Pointer auf das eigentliche Objekt hält.

17.4. Auslesen

Angriffe auf Smartcards sind sehr aufwändig und schwierig.

- Möchte man den Chip auslesen muss die Passivationsschicht chemisch entfernt werden. Die Chiphersteller haben aber Funktionalitäten implementiert, damit dies bemerkt wird.
- Hat man physischen Zugriff auf eine Smartcard, kann man diese mit Kältespray besprayen und in flüssiges Helium werfen. Somit wird der RAM Zustand über mehrere Wochen haltbar machen, weshalb man das RAM zu einem späteren Zeitpunkt mit einem Elektroden Mikroskop analysieren kann.
- Mittels Leistungs- und Zeitanalyse kann auf die Operation geschlossen werden. Deshalb versuchen die Hersteller einen möglichst konstanten Leistungsverbrauch zu erreichen.

17.5. PKCS#11: Cryptographic Token Interface Standard

PKCS#11 ist ein C API, welches die Smartcard als Objekt modelliert.

17.6. EMV: Europay, Mastercard, Visa

EMV ist ein Standard für interoperable Kredit- und Debitkarten. Der Standard ist öffentlich verfügbar und setzt auf einen kopiergeschützen Chip anstatt auf einen Magnetstreifen. Es können RSA-, oder Elliptic Curve-Keys für die Authentifizierung der Karte verwendet werden. Die verbreitung ist mit Ausnahme der USA (2%) und Asien (40%) recht gut.

18. PTS: Platform Trust Service

18.1. TPM: Trusted Platform Module

Das TPM ist ein Mikrochip mit dem Manipulationen an Software erkannt werden können. Zu diesem Zweck sammelt und speichert das TPM Daten, über vertrauenswürdige, extern ausgeführte Software. Stellt das TPM Veränderungen von Merkmalen an der Software fest, verweigert es den Zugriff auf den Schlüssel, mit der die Daten ver- und entschlüsselt werden. TPM Module waren früher eigene Chips auf dem Motherboard, sie werden jedoch immer mehr in die CPU integriert.

Ist ein TPM Chip auf dem PC Motherboard vorhanden, nimmt das OS während des gesamten Boot-Prozesses Messungen an den Treibern, ausgeführten Programmen, Libraries und geladenen Dateien vor und hasht die Messwerte in die Platform Configuration Registers (PCR) des TPMs. Die gehashten Messwerte werden für die Attestation bei einem Firmennetzwerk gebraucht. Das Netzwerk setzt Policies vorraus, die ein Client erfüllen muss. Die PCR Werte werden z.B über das TNC Protokoll übertragen.

Endorsement Key Der Endorsment Key ist fixes 2048 Bit langes RSA Schlüsselpaar, dass einem TPM zugeordnet und kann nicht verändert werden kann. Das Schlüsselpaar wird innerhalb des Chips verwaltet, wobei der private Teil von aussen nicht zugreifbar ist.

SRK: Storage Root Key Vom SRK werden alle weiteren Keys abgeleitet. Der 2048 Bit lange RSA Key basiert auf dem Endorsement Key und einem Nutzer spezifischen Passwort.

- AIK: **Attestation Identiy Keys**: Der Attestation Key wird verwendet, da der Endorsement Key einzigartig ist und somit Rückschlüsse auf den Benutzer gezogen werden könnten. Attestation Keys sind nicht migrierbar und dürfen vom TPM nur für die **Digitale Signatur** von Werten eingesetzt werden, welche im Platform Configuration Register (PCR) abgelegt werden (Attestation). PCR sind ein Teil des flüchtigen Speichers im TPM und für die Speicherung von Zustandsabbildern der aktuellen Konfiguration von Soft- und Hardware zuständig.
- AIK Zertifikat: Wird für die verfikation einer AIK Signatur benötigt. Das Zertifikat wird von einer vertrauenswürdigen Privacy CA signiert und enthält einen Public Key. Mit dem Public Key kann die Signatur überprüft werden. Das Zertifikat wird nur ausgestellt, wenn die CA eine eindeutige Zuordnung zwischen AIK und TPM-EK vorgenommen werden kann. Dazu wird
- SigK: Signing Keys
- BindK: Binding Keys
- MigrK: Migration Keys
- SymK: Symmetric Keys

Abbildung 52: Measure Boot

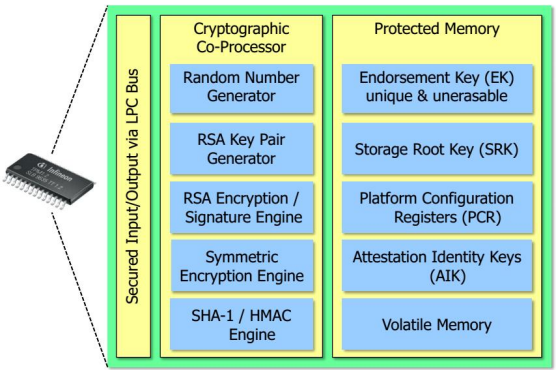

Abbildung 53: TPM Architecture

18.2. Binding

Das TPM kann Schlüssel auch ausserhalb des Trust Storage (z. B. auf der Festplatte) speichern. Diese werden ebenfalls in einem Schlüssel-Baum organisiert und deren Wurzel mit einem Key im TPM verschlüsselt. Somit ist die Anzahl der sicher gespeicherten Schlüssel nahezu unbegrenzt.

18.3. Sealing

Durch Bilden eines Hash-Wertes aus der System-Konfiguration (Hard- und Software) können Daten an ein einziges TPM gebunden werden. Hierbei werden die Daten mit diesem Hash-Wert verschlüsselt. Eine Entschlüsselung gelingt nur, wenn der gleiche Hash-Wert wieder ermittelt wird (was nur auf demselben System gelingen kann). Bei Defekt des TPM muss nach Aussagen von Intel die Anwendung, die Sealing-Funktionen nutzt, dafür sorgen, dass die Daten nicht verloren sind.

18.4. SRTM: Static Root of Trust for Measurement

Bei SRTM wird von unten her die geladenen DLL's gehashed und iterativ überschrieben. Dazu muss man in erster Linie dem TPM vertrauen. Dieses wird gehashed und so geht es über das CRTM, BIOS, Boot Loader, OS, Application, usw. weiter.

18.5. DRTM: Dynamic Root of Trust for Measurement

Der Ladeprozess eines Binaries wird erst ausgeführt, wenn die Signatur davon Stimmt. (Umsetzung z.Z. via Intel vPro / Trusted Execution Technology für VMs. Dadurch entsteht aber eine Abhängigkeit zur Hardware.)

18.6. Anwendungsbereiche

- Verschlüsselung von Daten mit symmetrischen Keys
- Authentifizierung bei Remote Systemem durch hinterlegte Zertifikate
- Überprüfung der Systemintegrität (Pre-Boot). Das Resultat von SRTM wird im TPM gespeichert und kann von autorisierten Dritten online abgefragt werden.

- Plattformbescheinigung (Remote Attestation)

18.7. PTT: Platform Trust Technology und TrustZone

PTT ist wie TPM aber in Software von Intel. Das gleiche gibt es auch von ARM und heisst dort TrustZone.

19. Secure Boot

Secure Boot ist ein Teil der UEFI-Spezifikation, der die Echtheit bzw. Unverfälschtheit von wichtigen Software-Teilen der Firmware garantieren soll. Kritische Teile der Firmware, wie der OS-Loader, sollen nur mehr dann ausgeführt werden, wenn sie zuvor durch eine vertrauenswürdige Institution dazu autorisiert wurden. Dadurch werden unter anderem Rootkits ausgeschlossen, die sich schon vor dem Boot des Betriebssystems (OS) einnisten.

Durch Signaturen von einem Zertifikat, dessen Public Key im UEFI hinterlegt ist, wird verhindert, dass nicht vertrauenswürdige Software-Teile ausgeführt werden.

Die Schlüssel in der UEFI-Firmware prüfen die Authentizität von z.B. Bootloadern. Ein Bootloader wird nur dann ausgeführt, wenn er eine gültige Signatur hat. Kann eine Signatur nicht überprüft werden bzw. ist sie nicht gültig wird das System am Starten gehindert. Entspricht z.B. die Signatur des Bootloaders nicht der, die die UEFI-Firmware erwartet, startet das System nicht.

PK: Platform Key
: Sobald der Platform Key in UEFI geschrieben wird, wechselt das UEFI von Setup Mode in den User Mode. Normalerweise wird ein Notebook bereits mit einem Platform Key des Herstellers ausgeliefert. Der PK stellt eine Vertrauensbasis zwischen dem Platform Besitzer und der Firmware her.

KEK: Key Exchange Keys
: Der KEK wird benötigt um Zertifikate in die Allowed/Forbidden Database speichern zu können. Der KEK muss mit dem private Platform Key signiert werden, damit der KEK verwendet werden kann. Der KEK stellt eine Vertrauens Basis zwischen dem OS und der Firmware her.

Platform Key Database Objekte zur Modifizierung der KEK's

Key Exchange Key Database Beinhaltet jene Trust-Objekte, welche die Allowed und Forbidden Signature Database modifizieren dürfen

Allowed Database
: Hier sind die Zertifikate von Software Signers hinterlegt (Microsoft).

Forbidden Database
: Enthält die Revoked Signing Zertifikate. Damit können gewisse Software in die Blacklist geschrieben werden.

19.1. Ablauf

1. Der OS Loader Code muss mit dem Private Key eines Software Signers signiert sein
2. Das Zertifikat des Software Signers muss sich in der Allowed Database des UEFIs befinden. Das Zertifikat kann nur im UEFI gespeichert werden, wenn es mit einem KEK signiert ist, dessen KEK Zertifikat sich schon im UEFI befindet.
3. Ein KEK Zertifikat kann nur ins UEFI importiert werden, wenn es mit dem PK des Hardwareherstellers signiert ist.

4. Das PK Zertifikat des Hardwarehersteller muss bei der Aktivierung des Secure Boot Modus ins UEFI importiert werden

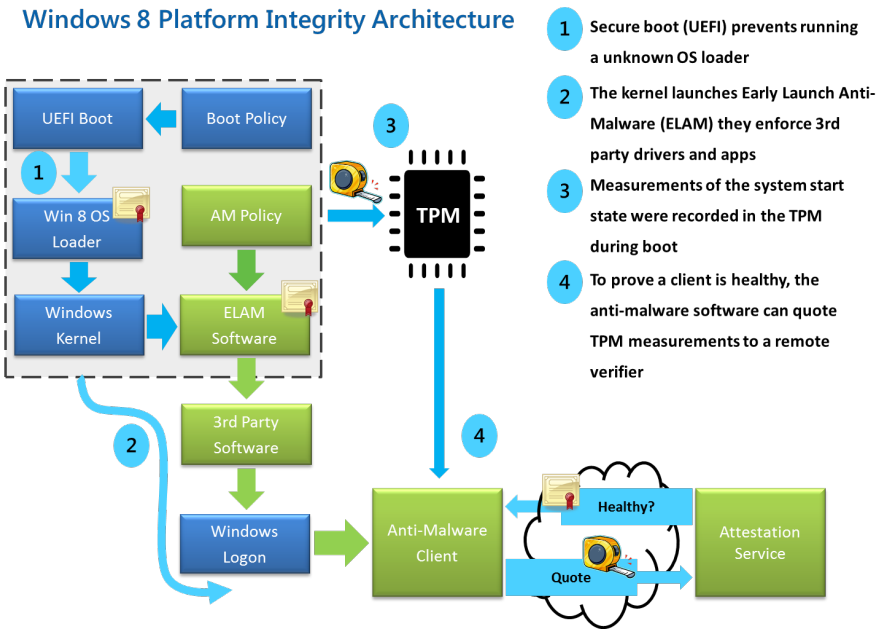

Abbildung 54: Secure Boot

Abbildung 55: Secure Boot unter Windows 8

20. Virtualization

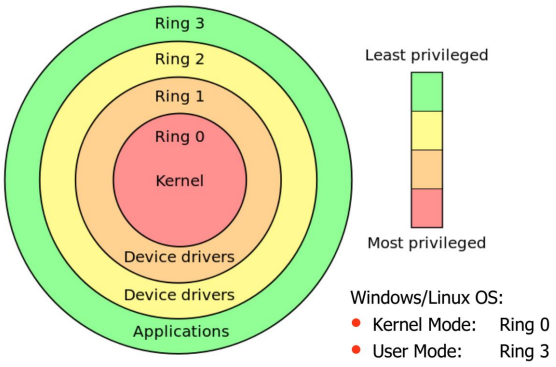

Abbildung 56: Virtualisierung: Protection Rings

20.1. Hypervisor Typen

Man unterscheidet zwischen zwei Typen von Hypervisor:

- Type1 (native): VMware ESX, Microsoft Hyper-V, Xen, Oracle VM Server (**Ring -1 (bare metal)**)
- Type2 (hosted): KVM, VirtualBox, VMWare Workstation

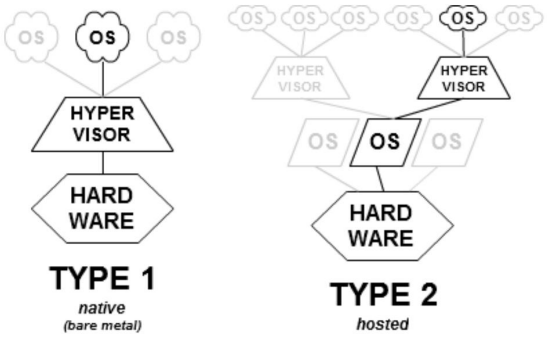

Abbildung 57: Hypervisor Types

20.2. Qubes OS

Qubes OS ist ein OpenSource Betriebssystem welches versucht durch Isolation den gewöhnlichen Desktop sicherer zu machen. Qubes OS basiert auf Xen und nutzt AppVM's für die Separierung. Dazu setzt es auf Virtualisierung, um Programme und Systemkomponenten (z.B Netzwerkstack) voneinander zu isolieren. Damit soll die Kompromittierung einese Teils des System, die Integrität des restlichen Systems nicht beeinflussen.

20.2.1. GUI Protokoll

Für die Kommunikation zwischen den Qubes Instanzen wird ein Shared Memory verwendet. Nachrichten werden über das Xen vchan Protokoll übertragen. Die eigentliche Grafikausgabe ist ebenfalls via Shared Memory implementiert. Die lokalen X Server der VMs verwenden einen speziellen Treiber der einfach in einen Framebuffer zeichnet, der nicht mit echter Hardware verknüpft ist. Dieser Memorybereich wird dann in dom0 in den dortigen X Server gemappt. Xen stellt dabei sicher, dass der Memory Bereich auch tatsächlich zur jeweiligen VM gehört. Dank diesem Vorgehen, hat nur dom0 auf den grafischen Output mehrere VM's Zugriff. Die jeweiligen VMs haben keinen direkten Zugang zur Grafikkarte oder die Framebuffer der anderen VMs. Damit werden Informationsleaks via Grafikausgabe verhindert. Dom0 hat als einzige direkten Zugriff auf die Tastatur.

21. Selbststudium

21.1. Tor

Siehe 12.3

21.2. Secure Boot

Was sind die Eigenschaften von UEFI, TPM, Virtual Smart Card?

- UEFI -> Secure Boot
- TPM -> Measure Boot
- XP SP2: ASLR, Firewall on by Default (erste wichtige Security Updates)
- Vista: TPM für Bitlocker
- Windows 7: Grösster Schritt wegen SDL: Secure Development Lifecycle, Biometric Authentication
- Windows 8: UEFI Secure Boot -> Measure Boot und Remote Attestation (UEFI ist Pflicht, sonst kein Lizenzkleber)
- Das UEFI 2.3.1 ist ein eigenes OS mit TCP/IP Stack -> Man muss es deshalb unbedingt mit TPM überprüfen. (Gefahr von nach Hause telefonieren)
- **VCS: Virtual Smart Card**: Beliebig viele Smartcards. Das TPM verhält sich wie eine ständig eingesteckte Smartcard.
- Seit Windows 8 können Anti Malware Programme über eine Hook eingehängt werden. Dies erlaubt es , dass die Anti Malware frühere Dinge im Boot Prozess zu überprüfen.
- Man unterscheidet zwei Typen von Unternehmen
  - Jene die merken, dass sie gehackt wurden
  - Jene die nicht merken, dass sie gehackt wurden.

Bekannte Threads

- Script Kitties
- Cybercrime
- Attacken auf die Fortune 500
- Hoffen, dass man nicht gehackt wird.

Neuartige Threads

- Cyber Spionage, Cyber Warfare, State Sponsored Actions
- Alle Organisationen werden angegriffen
- Man wird gehackt. Es geht darum, wie man damit umgeht.

A. Listings

B. Abbildungsverzeichnis

C. Tabellenverzeichnis