

**KHT / Einführung
in**

Kryptologie

IPSec

VPN

Inhalt

Kryptologie:

Geschichte,
Schlüssel, symmetrisch/asymmetrisch
DES (Data Encryption Standard)
Triple-DES
A5 Algorithmus (D-Netz)
Signaturen am Bsp. MD5 (Message Digest Version 5)

IPSec:

Architektur (Tunnel-/Transport-Mode)
ESP (Encapsulating Security Payload)
AH (Authentication Header)
DH (Diffie-Hellman)
IKE (Internet Key Exchange)

VPN:

Überblick
End-to-End Security
Verschachtelte/Verkettete Tunnel
FreeS/WAN
Check Point Firewall-1
Astaro Security Linux

Part 1

Kryptologie

Kryptologie: Geschichte I

Caesar-Methode:

a -> d
b -> e
c -> f
...
z -> c

Beispiel:

FDHVDU LQ URQ ???

CAESAR IN ROM

Anwendung:

In C unter UNIX `ROT13()` !

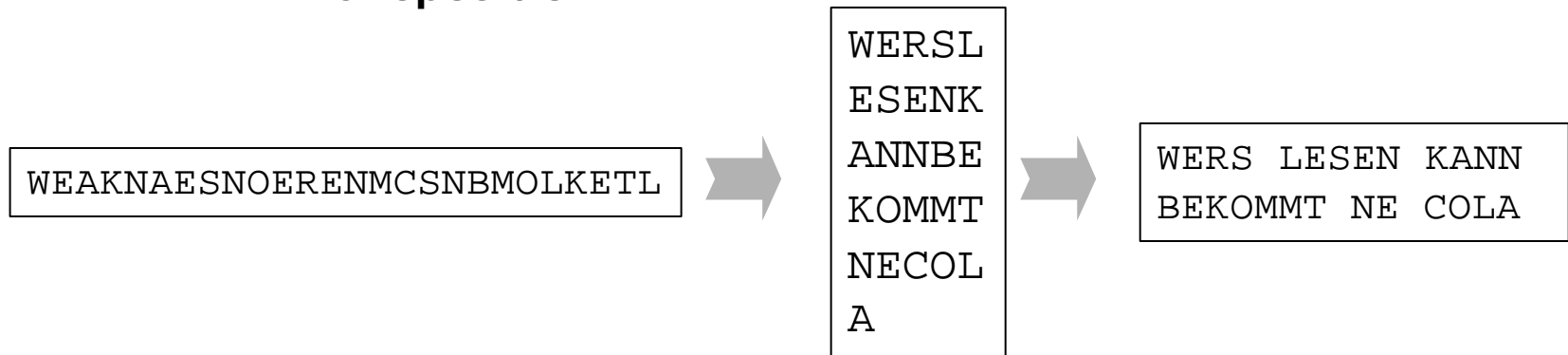
- Gegen unbeabsichtigtes Mitlesen !

- Dekodierung: `ROT13(ROT13(Text)) = Text`

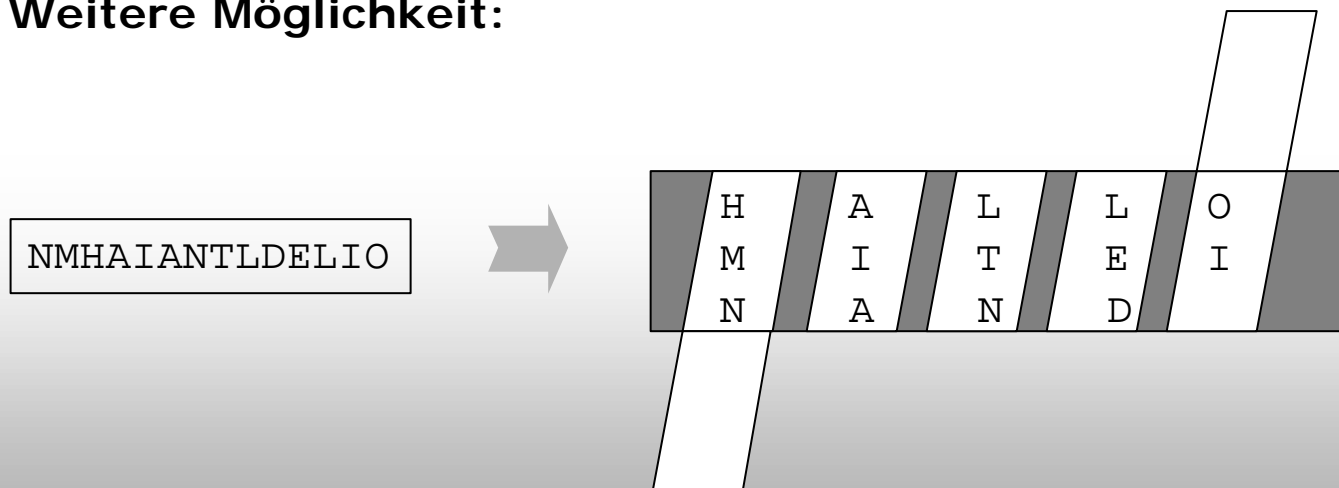
Kryptologie: Geschichte II

Zeitraum zwischen Caesar und Adolf:

-> Transposition



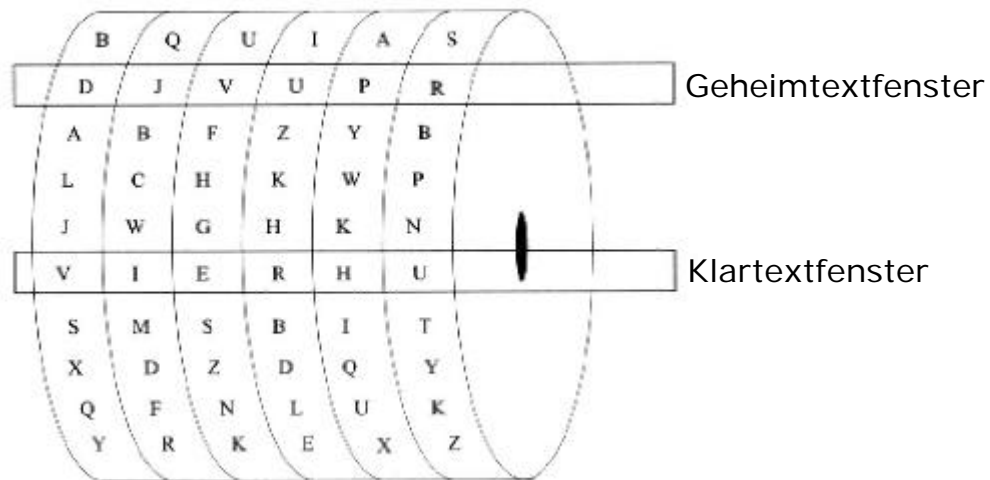
Weitere Möglichkeit:



Kryptologie: Geschichte III

Rotormaschinen:

-> **Die Enigma** (Willi Korn)



Prinzip:

- elekt. isolierende Scheiben mit 26 Kontakten und el. Magneten
- je Scheibe, das ABC in bestimmter Reihenfolge
- Eingabe über separate Tastatur
- Spannung der Tastatur wird links eingespeißt und plantz sich nach rechts fort
- je nach Zeichen einen Schritt weiterdrehen (vor/rück)

Schlüssel besteht aus der Anordnung der Scheiben !

Kryptologie: Geschichte IV

Die Enigma : (5 Scheiben)

„Kleines“ Beispiel:

3 Scheiben à 26 Zeichen → 26^3 Möglichkeiten !

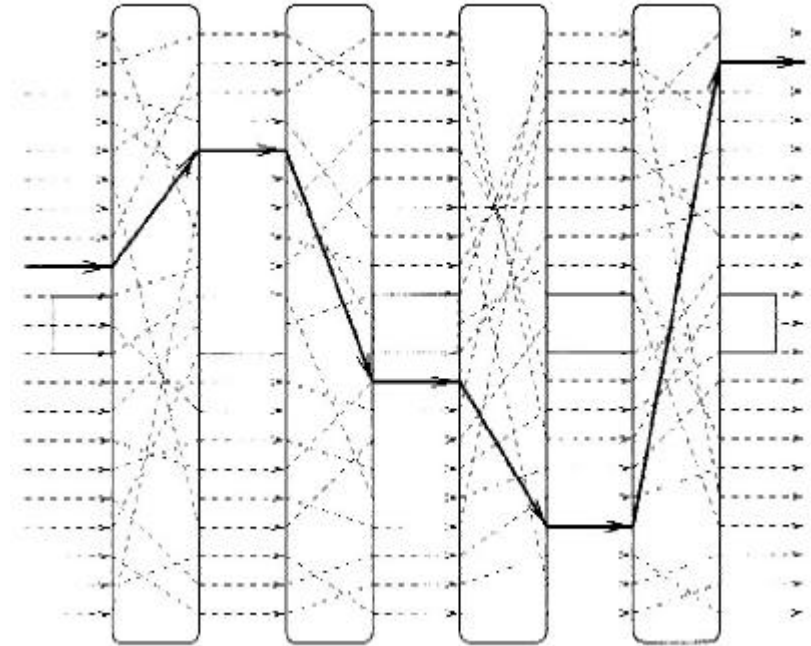
Multipliziert mit 10 Möglichkeiten der Grundeinstellung

Plus Steckbrett mit 26 Buchstaben
→ $26! / (13! * 2^{13})$ mögliche Paare

Insgesamt ca. :

8 Trillionen mögliche Schlüssel ($8 * 10^{18}$)

→ Wurde jedoch von einem Polen entschlüsselt (Okt. 1932) !



Stromlaufplan für 4 Scheiben

Kryptologie: Geschichte V

Enigma 1945

- 6 Scheiben
- Buchstaben und Zahlen
- 24-faches Steckbrett für Grundeinstellung
- 2-facher Schlüssel (alternierend)



Kryptologie: Schlüssel I

symmetrisch/asymmetrisch

Schlüssel - symmetrische

Beispiel aus der Mathematik:

$$Y = X * 22^{A-12}$$

x : Klar-„Text“

y : verschlüsselter „Text“

A : Key

→ Aus **y** ist nicht gleich auf **x** zu schließen

→ Jedoch kann ohne **A** nicht auf **x** zurückgeschlossen werden

Weitere Beispiele:

ROT13(), Stromchiffrierung, Caesar-Methode, crypt, ZIP, ...

Kryptologie: Schlüssel II

symmetrisch/asymmetrisch

Schlüssel - asymmetrische

-> Auch Public-Key-Verfahren genannt

Bestehen aus einem Key-Ring

- privaten Schlüssel (private key)

- öffentlichen Schlüssel (public key)

Verschlüsselung mit private key, Entschlüsselung mit public key

→ Aus dem public key kann man nicht* den private key ableiten

* „nicht“ ist hier im kryptologische Sinn zu verstehen. D.h. nicht mit bekannten Mitteln innerhalb einer praktisch realisierbaren Zeit !

Kryptologie: Schlüssel II

symmetrisch/asymmetrisch

Schlüssel – asymmetrische

Am Beispiel des **RSA**-Verfahren (**R**ivest, **S**hamir und **A**dleman)

Schlüsselerzeugung:

- Wähle zwei große Primzahlen p und q (Standard: 512 Bit)
- Bilde $n=pq$ (n sei N Bit lang)
- Wähle ein $e > 1$, daß zu $(p-1)(q-1)$ teilerfremd ist
- Berechne ein d mit $de=1 \pmod{(p-1)(q-1)}$
- n und e bilden den öffentlichen Schlüssel
- d den privaten Schlüssel

Kryptologie: Schlüssel III

symmetrisch/asymmetrisch

RSA – Verfahren

Chiffrierung:

- Zerlege den Klartext in Blöcke (m) zu je $N-1$ Bit (wenn das nicht aufgeht, wird sogenanntes *padding* angewandt)
 - Berechne zu jedem Block mit Wert $m < n$ den Rest c von m^e bei Teilung durch n
- c ist der Geheimtextblock mit Länge N Bit.

Dechiffrierung:

- Zerlege den Geheimtext (c) in N -Bit Blöcke
- Zu jedem Block mit Wert $c < n$ ist der Rest von c^d bei Teilung durch n der zugehörige Klartext

ALLES KLAR ????

Kryptologie: Schlüssel IV symmetrisch/asymmetrisch

Klarheit am Beispiel (mathematisch) der Schlüsselerzeugung:

- Wähle zwei große Primzahlen p und q (Standard: 512 Bit)

$$p = 5, q = 3 \text{ (je 3 Bit)}$$

- Bilde $n=pq$ (n sei N Bit lang)

$$n = pq = 5 \cdot 3 = 15 \text{ (8 Bit)}$$

- Wähle ein $e > 1$, daß zu $(p-1)(q-1)$ teilerfremd ist

$$e = 3$$

- Berechne ein d mit $de=1 \pmod{(p-1)(q-1)}$

$$(p-1)(q-1)=8 \rightarrow de=1 \pmod{8} = 1 \rightarrow d = 1/3$$

→ n und e bilden den öffentlichen Schlüssel

$$n = 15, e = 3 \rightarrow \text{public key}$$

→ d den privaten Schlüssel

$$d = 1/3 \rightarrow \text{private key}$$

Kryptologie: DES I

DES – Data Encryption Standard

Entwickelt 1974 von einem IBM-Team:
Horst Feistel und Don Coppersmith

Später Weiterentwicklung von NBS (*National Bureau of Standards*) und
NSA (*National Security Agency*)

Wichtiges zusammengefaßt:

- Blocklänge: 64 Bit
- Schlüssellänge: 56 Bit effektiv
- Feistelchiffre mit 16 Runden

- Eingangsgrößen:
 - 64-Bit-Klartextblock m
 - 64-Bit-Schlüssel K , wovon 8 Bit nicht genutzt werden
- Ausgangsgröße:
 - 64-Bit-Geheimtextblock c

Kryptologie: DES II

DES – Funktionsweise/Sicherheit/Probleme

Funktionsweise (vereinfacht):

$$L_i = R_{i-1}, \quad R_i = L_{i-1} \text{ XOR } f(R_{i-1}, K_i), \quad i = 0, 1, 2, \dots, 15$$

Kombinatorische Komplexität – Schlüsselvielfalt:

Es gibt nur 2^{56} , also etwa $7,2 \cdot 10^{16}$ verschiedene Schlüssel !!

Probleme – Schwache Schlüssel:

0101010101010101

FEFEFEFEFEFEFEFE

1F1F1F1F1F1F1F1F

E0E0E0E0E0E0E0E0

→ diese können zur Involution führen (also Rückbildung der Schlüssellänge)

Kryptologie: DES III

DES – Bewertung:

Positiv :

- weit verbreitet,
- sehr gründlich untersucht,
- offener Standard,
- gute Performance,

Negativ:

- schwache Schlüssel möglich,
- 40-Bit DES ist per Software mit Brute-Force in ca. 40 Sekunden geknackt
- 56-Bit DES per Software (PIII 500) mit Brute-Force in 30 Tagen knackbar
- 56-Bit DES per Hardware (Deep-Black, ASIC, 50\$) in 20h knackbar

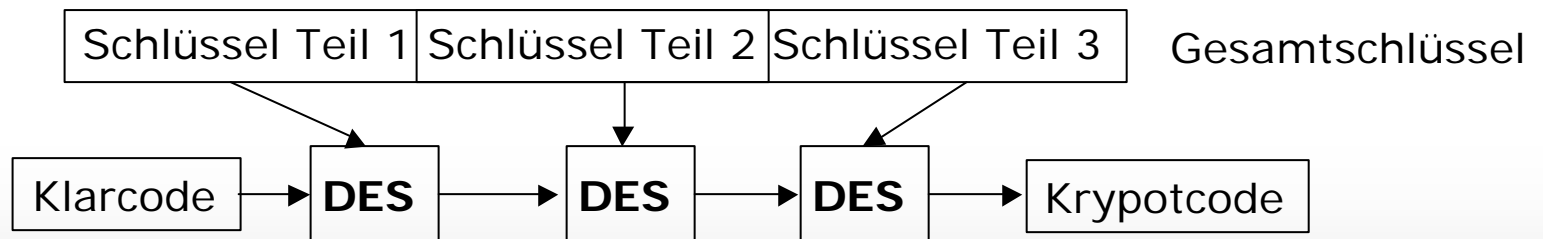
Kryptologie: Triple-DES I

Triple-DES:

Erweiterung von DES durch Tuchman (1979)

Im Prinzip

- Mehrfachchiffrierung (3-mal) mit DES
- jeweils verschiedenen Schlüsseln (Teil des Gesamtschlüssels)



Kryptologie: Triple-DES II

Triple-DES - Bewertung

- 112 Bit effektive Schlüssellänge
- 72 Billionen mal größere Sicherheit (Brute-Force)
- Kryptologisch gesehen als unknackbar in einem endlichen Zeitraum !
- keine Möglichkeit von schwachen Schlüsseln
- „zähe“ Performance

Kryptologie: A5 I

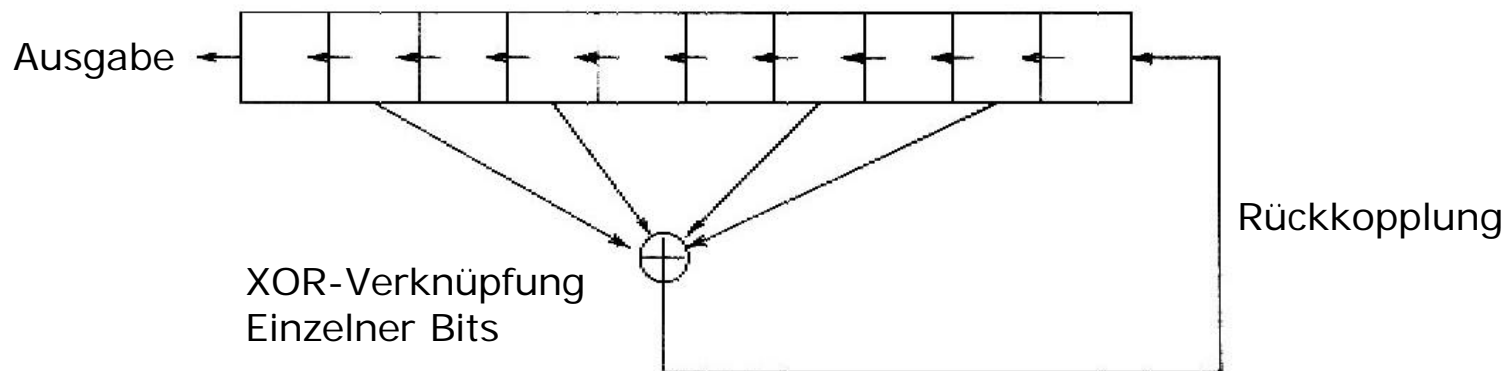
A5 – Verschlüsselung für das D-Netz

GSM-Standard:

- Sprache wird digitalisiert
- in 114 Bit große Datenpakete verpackt und zur Basisstation gefunkt
- Schlüssellänge 57 Bit, effektiv 56 Bit (Bit 57 ist immer 0)
- Im Prinzip nur eine LFSR (line feedback shift register)

Kryptologie: A5 II

LFSR - Line Feedback Shift Register



Frage: Wo wird ein Schlüssel benutzt ???

Antwort: Gar nicht !! Der Schlüssel ist das eingegebene Bitmuster selbst !

Sicherheit: Periodenlänge: $2^n - 1$, d.h. im Prinzip ein primitives Polynom modulo 2

Anwendung: A5 Algorithmus des D-Netzes, Militär, Cray (eigener Assembler-Befehl)

Kryptologie: Signaturen I

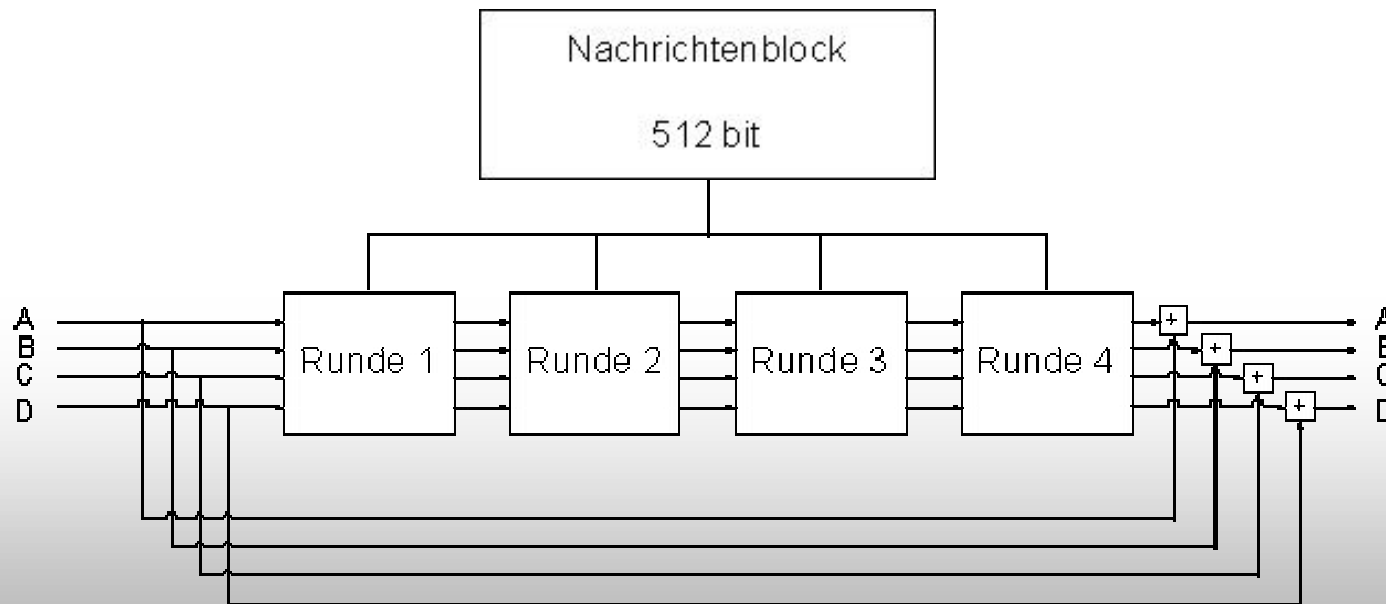
Signaturen:

- Signaturen sind keine Verschlüsselung !!!
- Signaturen entsprechen einer Prüfsumme über einen Bit-Block,
- Prüfsumme wird immer Bit-Blockweise erstellt,
- Signaturen können sowohl mit als auch ohne Schlüssel erzeugt werden (CRC)
- Signaturen werden meistens an den Bit-Strom angehängt

Kryptologie: Signaturen – MD5 II

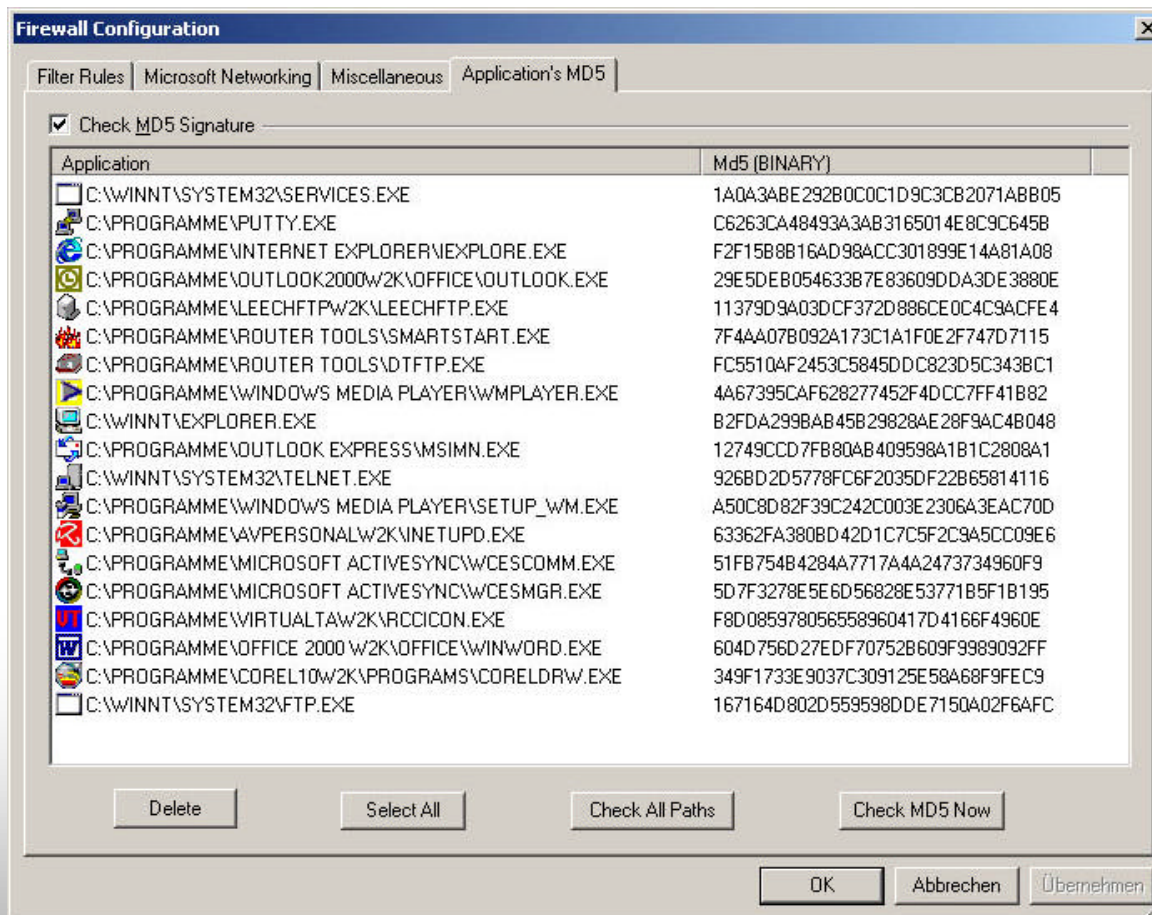
Signaturen – Am Beispiel **MD5** (Message Digest Version 5)

- Einweg – Hashfunktion,
- Entworfen von Ron Rivest (!!!)
- Ausgabe besteht aus vier 32-Bit-Blöcken
- Ausgabe-hash hat immer 128 Bit-Länge
- Eingabeblocke haben 512 Bit, aufgeteilt in 16 Teilblöcke (je 32 Bit)
- Der Algorithmus besteht aus vier Runden



Kryptologie: Signaturen – MD5 III

MD5 – Beispiel:



Tiny Personal Firewall:

Erstellt von allen Netzapplikationen eine MD5-Signatur. Dadurch können Veränderungen Durch Viren, Trojaner usw. festgestellt werden.

Part 2

IPSec

IPSec: Architektur I

IPSec - Sec für Sicherheit ? **IPSec** ist keine Garantie für absolute Sicherheit !

IPSec bietet folgende Methoden:

- Authentifizierung der Datenquelle
- Authentifizierung der Datenintegrität, ohne daß eine Verbindung notwendig ist
- Schutz für vertraulichen Dateninhalt
- eine begrenzte Verkehrsflussvertraulichkeit

IPSec gibt keine Garantie:

- daß angekommene IP-Datagramme vom angegebenen Sender (der Ursprungsadresse im IP-Header) stammen
- daß sie die ursprünglichen Daten enthalten, die der Sender geschickt hat
- daß keine andere Person die Daten auf dem Weg vom Ausgangspunkt zum Ziel angeschaut hat (man-in-the-middle)

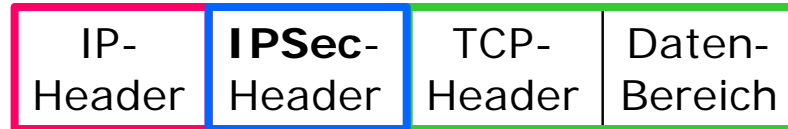
IPSec: Architektur II

Tunnel- und Transport-Modus:

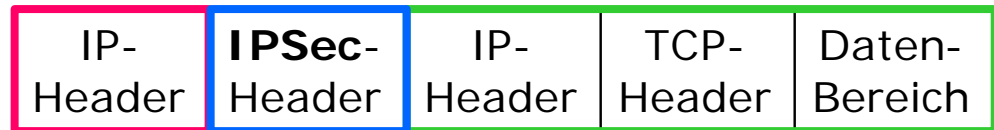
Original-
IP-Paket



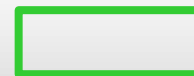
Geschütztes Paket
Im **Transport**-Modus



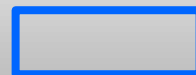
Geschütztes Paket
Im **Tunnel**-Modus



Ungeschützter Bereich



Geschützter Bereich



Je nach Anwendung

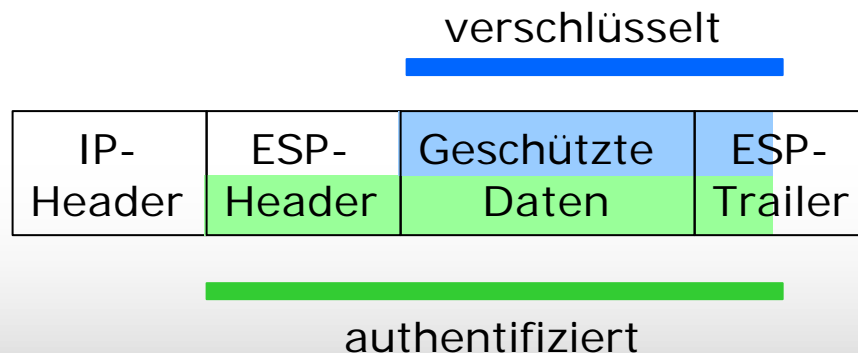
IPSec: ESP I

ESP – Encapsulating **S**ecurity **P**ayload (Zusätzlicher Header):

Sorgt bei IPSec für:

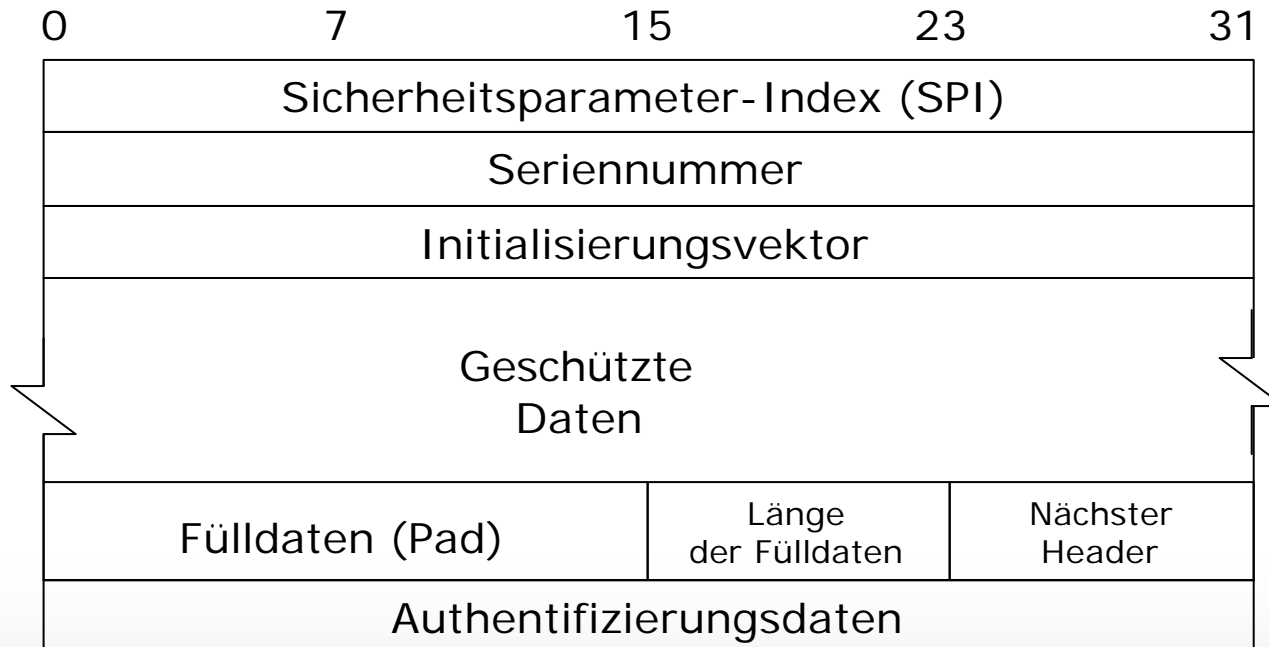
- Vertraulichkeit
- Datenintegrität
- Authentifizierung der Datenquellen von IP

→ Der ESP-Header ist selbst nicht verschlüsselt, der ESP-Trailer zum Teil !



IPSec: ESP II

ESP – Header und Trailer:



IPSec: ESP III

ESP – Headerinhalt:

ESP-Header

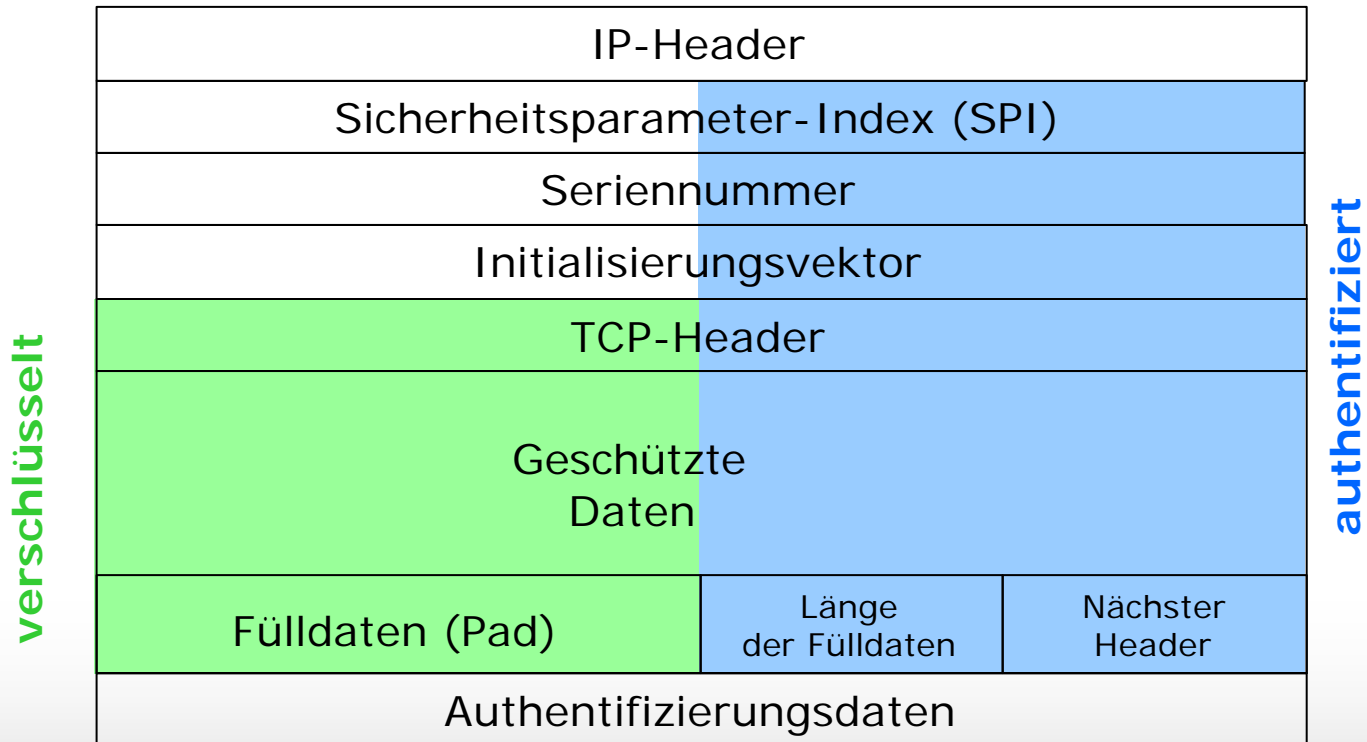
- enthält den so genannten Sicherheitsparameterindex (*SPI*)
- die Vertraulichkeit (*Cipher* genannt), den Verschlüsselungsalgorithmus
- die Authentifizierung, (*Authentifikator* genannt)
- Modus (Tunnel- oder Transport)

ESP-Trailer

- Seriennummer
- Datenintegrität
- Weitere SPI (verschlüsselt)

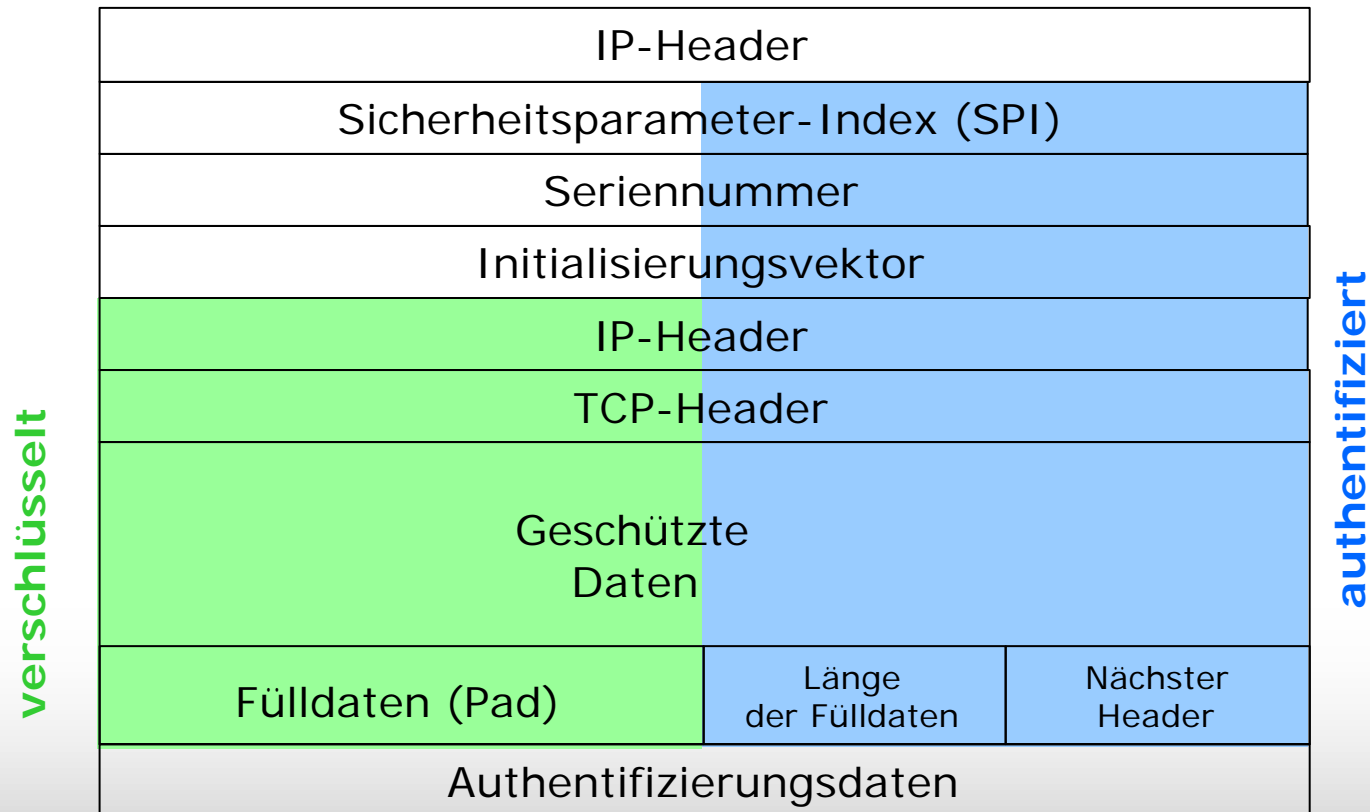
IPSec: ESP IV

ESP – Transport Modus:



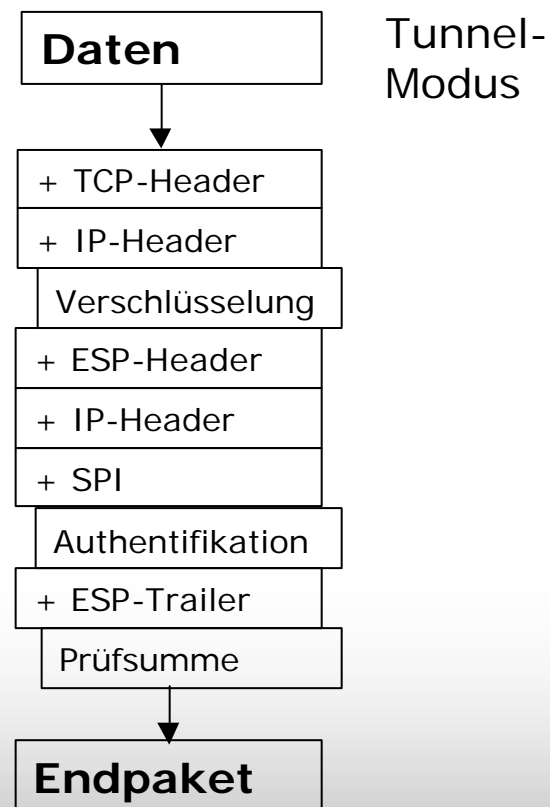
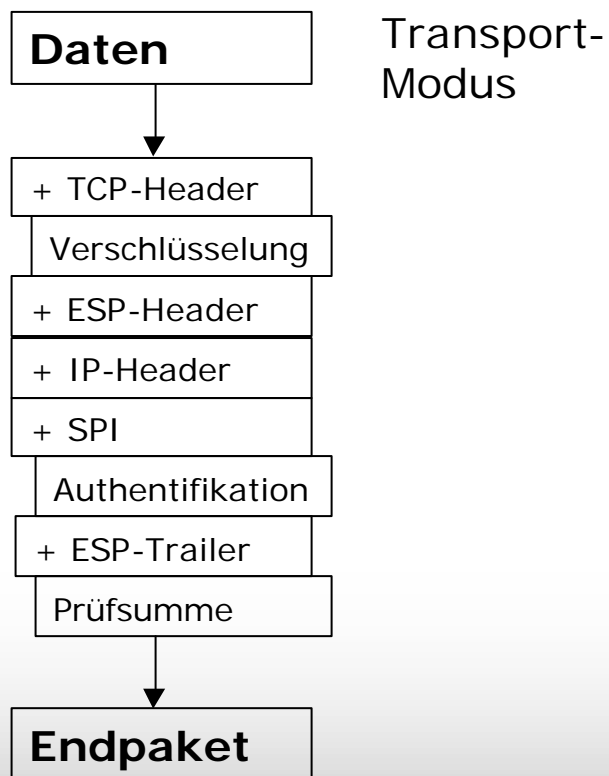
IPSec: ESP V

ESP – Tunnel Modus:



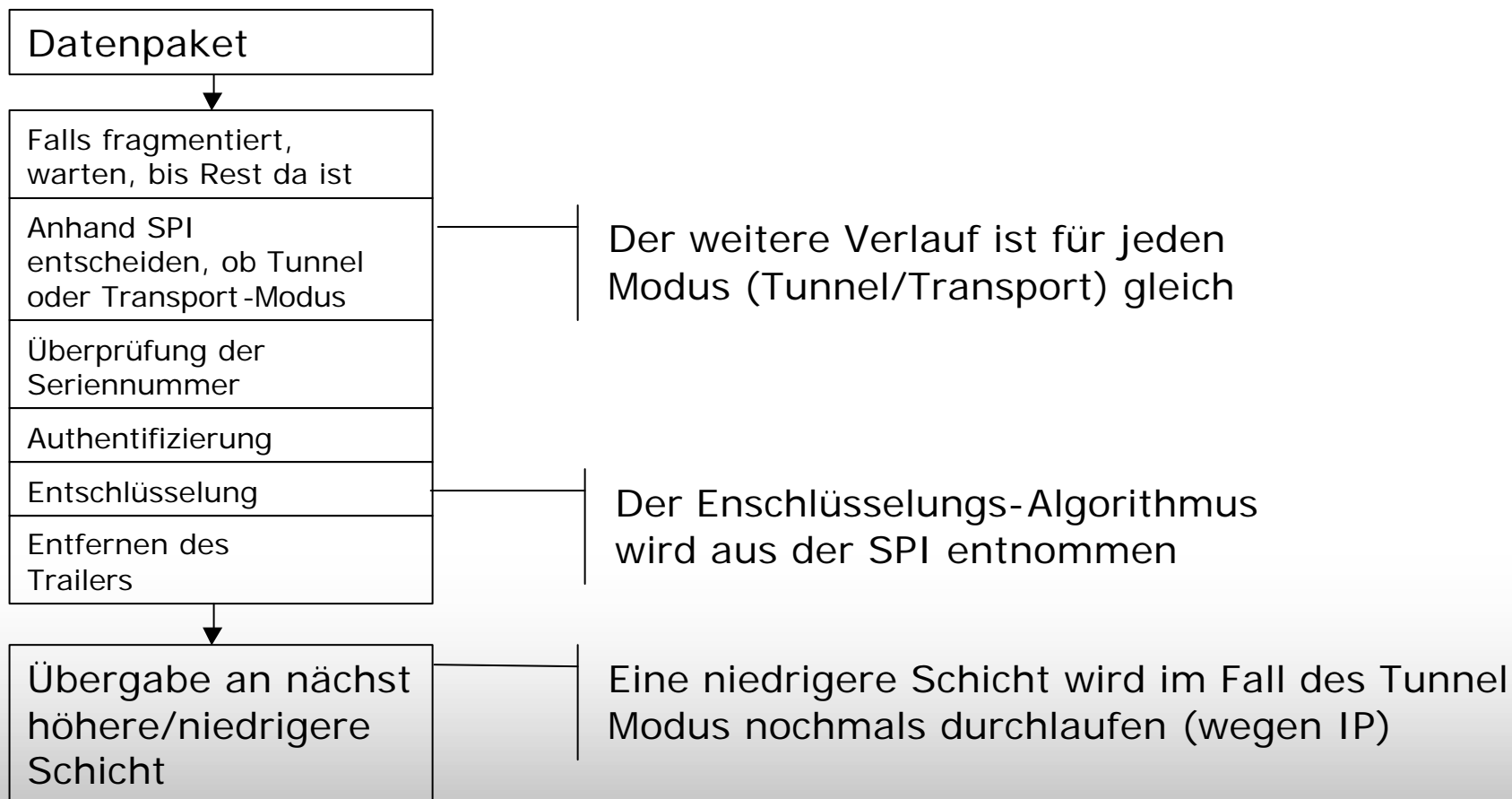
IPSec: ESP VI

ESP – Ausgehender Verkehr:



IPSec: ESP VII

ESP – Eingehender Verkehr:



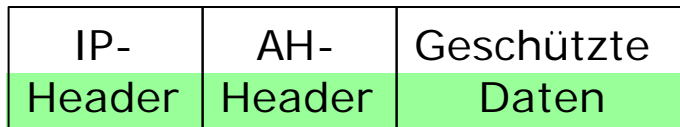
IPSec: AH I

AH – Authenticaion Header:

Sorgt für:

- Datenintegrität
- Authentifizierung

→ Enthält keine Verschlüsselung !



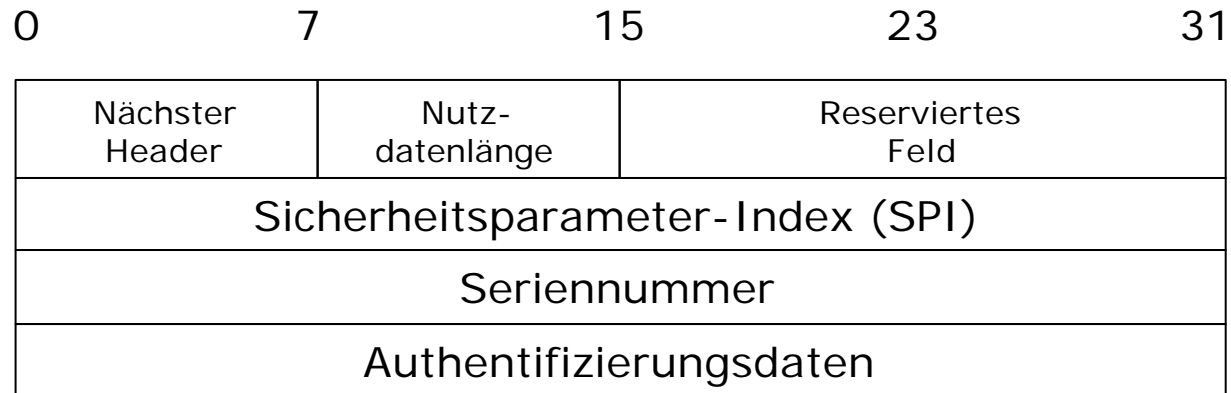
authentifiziert

Frage: Sinn des AH ??? – Das kann doch der ESP auch, oder ?

Antwort: Die Authentifizierung per AH wird über das komplette Datenpaket erstellt !!!

IPSec: AH II

AH – Authenticaion Header:



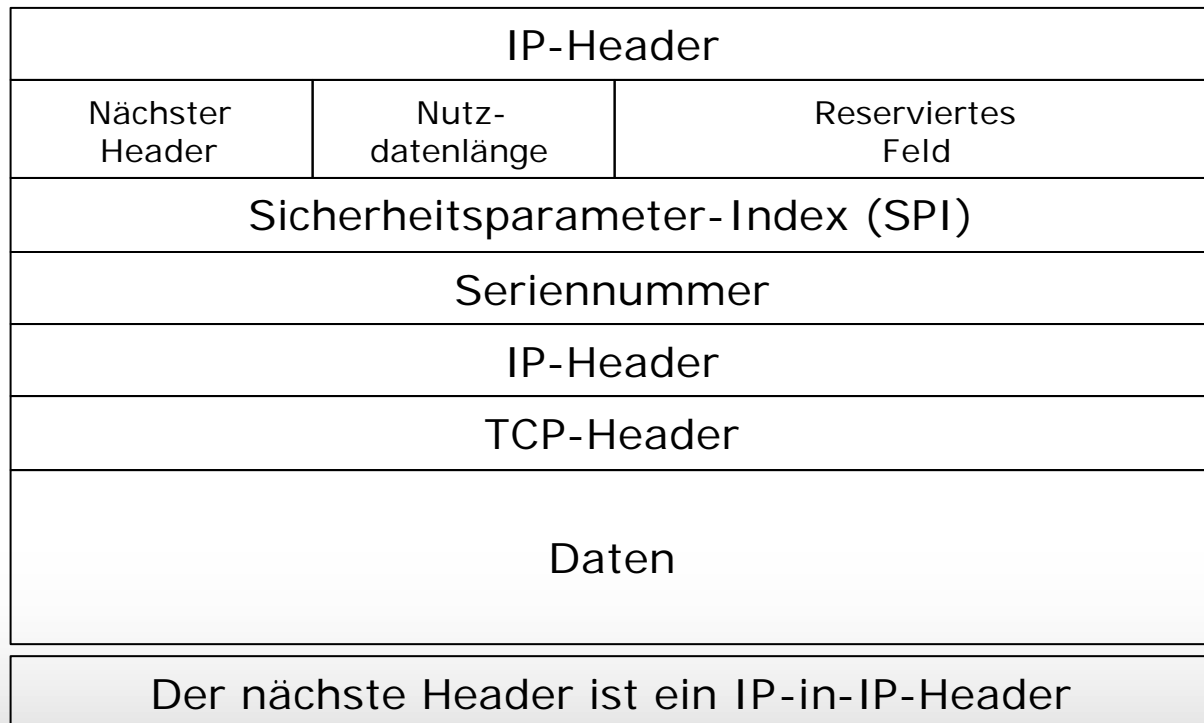
IPSec: AH III

AH – Transport-Modus:

IP-Header		
Nächster Header	Nutzdatenlänge	Reserviertes Feld
Sicherheitsparameter-Index (SPI)		
Seriennummer		
TCP-Header		
Daten		

IPSec: AH IV

AH – Tunnel-Modus:



IPSec: AH & ESP I

Datenintegrität bei AH und ESP:

Die Datenintegrität wird über sogenannte HMACs hergestellt.
HMAC steht für *Hashing Message Authentication Code*

→ HMAC ist keine Signatur !

→ Im Prinzip ein *Keyed Hashing*

$$\text{HMAC}(K, M) = H (K \text{ xor } \text{opad}, H (K \text{ xor } \text{ipad}, M))$$

ipad: Array, 64 Elemente mit Wert 0x36 K: Schlüssel
opad: Array, 64 Elemente mit Wert 0x5c M: Nachricht
H: Hash-Algorithmus

HMAC benutzt „normale“ Hash-Algorithmen:

- HMAC-MD5
- HMAC-MD4
- HMAC-SHA, usw.

→ diese sind kryptographisch stärker als „Ursprungs-Hash“, da zwingend Schlüssel verwendet werden !

IPSec: Schlüsselaustausch

Schlüsselaustausch:

Es gibt viele Protokolle, die einen Key-Change anbieten:

- SKEME
- ISAKMP (Internet Security Association and Key Management Protocol)
- Oakley
- DH (Diffie-Hellman)

→ Hier exemplarisch DH (Diffie-Hellman)

IKE (Internet Key Exchange) basiert auf drei Protokollen:

- ISAKMP
- Oakley
- SKEME

IPSec: Diffie-Hellman I

DH – Diffie-Hellman:

- Erstes Kryptosystem mit öffentlichen Schlüsseln
- Entwickelt von Whitfield Diffie und Martin Hellman
- Prinzip: Problem des diskreten Logarithmus

Funktionsweise: (Alice will mit Bob einen Schlüssel austauschen)

- I. Definition einer Primzahl p und einem Generator g
- II. Alice wählt frei a , Bob wählt frei b
- III. Alice berechnet $A = g^a \bmod p$
- IV. Bob berechnet $B = g^b \bmod p$
- V. Alice schickt A zu Bob, Bob schickt B zu Alice
- VI. Alice bildet $B^a \bmod p = k$, Bob bildet $A^b \bmod p = k$
- VII. **k** ist der geheime Schlüssel !

IPSec: Diffie-Hellman II

DH – Beispiel:

I. Definition einer Primzahl p und einem Generator g

$$g = 3, p = 7 \rightarrow \text{Austauschen}$$

II. Alice wählt frei a , Bob wählt frei b

$$\text{Alice wählt } a = 5, \text{ Bob wählt } b = 9$$

III. Alice berechnet $A = g^a \bmod p$

$$A = 3^5 \bmod 7 = 243 \bmod 7 = 5$$

IV. Bob berechnet $B = g^b \bmod p$

$$B = 3^9 \bmod 7 = 19683 \bmod 7 = 6$$

V. Alice schickt A zu Bob, Bob schickt B zu Alice

VI. Alice bildet $B^a \bmod p = k$, Bob bildet $A^b \bmod p = k$

$$\text{Alice : } k = 6^5 \bmod 7 = 7776 \bmod 7 = 6$$

$$\text{Bob : } k = 5^9 \bmod 7 = 1953125 \bmod 7 = 6$$

IPSec: Diffie-Hellman III

DH – Probleme:

Problematisch beim Diffie-Hellman Schlüsselaustausch ist der „*Man-in-the-middle*“-Angriff.

Dieser kann den Austausch von g , p , A und B mithören und somit den Schlüssel nachrechnen !!!



Lösung: Teilweise !!!

Als Lösung werden die öffentlichen Werte mit einer digitalen Signatur versehen.

Problem:

Wie wird die digitale Signatur ausgetauscht ?

→ Ei-Huhn-Problem !

IPSec: IKE I

IKE - Phasen:

IKE verwendet ISAKMP (Internet Security Association and Key Management Protocol) zum Schlüsselaustausch.

ISAKMP ist eine Weiterentwicklung von DH
(→ Genauer: die Protokol-Definition von DH)

Verlauf in Phasen:

Phase 1: Sicherheitsassoziation

Phase 2: Schlüsselaustausch

Phase 3: IDs (Wer der Initiator und wer der Responder der IPSec-Session ist)

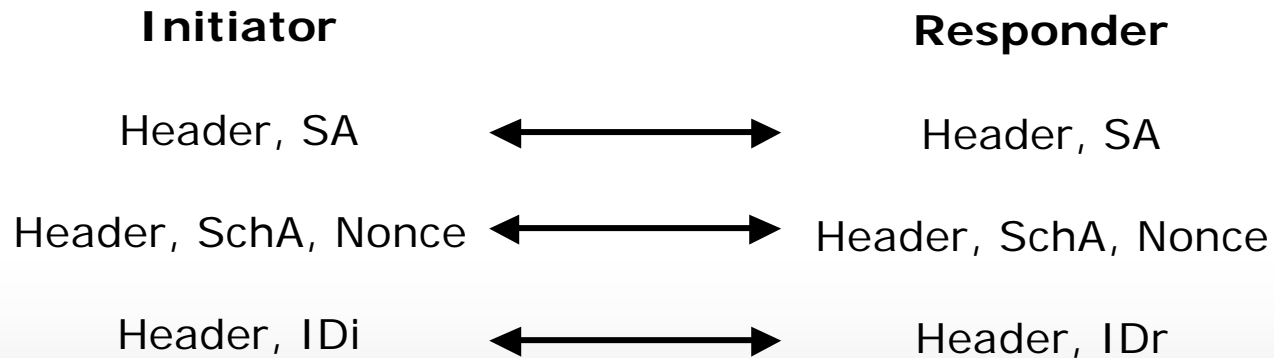
IPSec: IKE II

IKE - Modi:

IKE definiert 3 Modi zum Austausch:

- Haupt-Modus
- Aggressiv-Modus
- Schnell-Modus

Haupt-Modus:



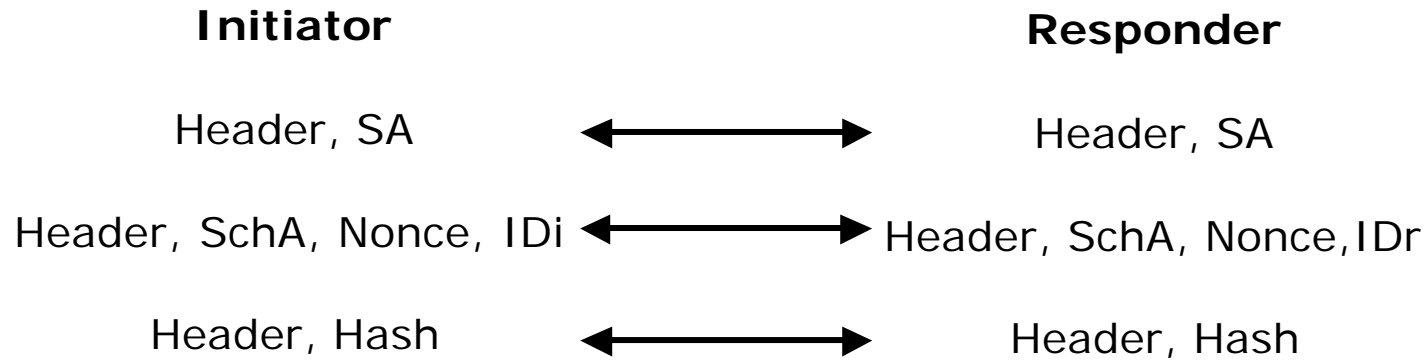
SA: Sicherheitsassoziation
IDi: Initiator ID
IDr: Responder ID

SchA: Schlüsselaustausch
Nonce: Zufallsdaten

IPSec: IKE III

IKE - Modi:

Aggressiv-Modus:



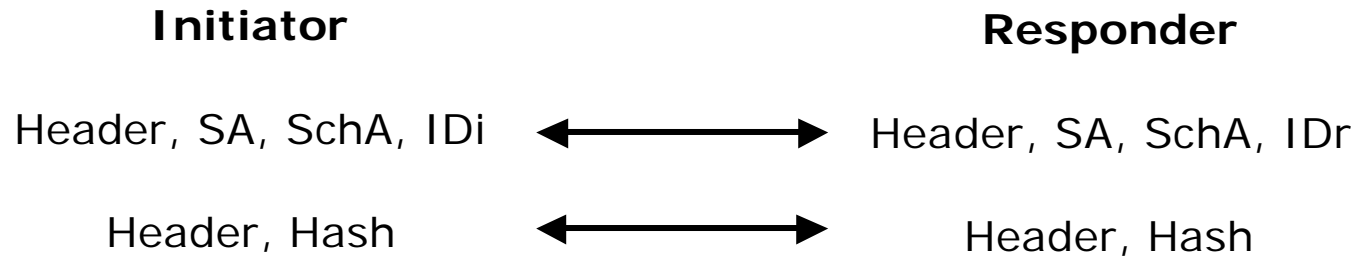
SA: Sicherheitsassoziation
IDi: Initiator ID
IDr: Responder ID

SchA: Schlüsselaustausch
Nonce: Zufallsdaten
Hash: zum verifizieren

IPSec: IKE IV

IKE - Modi:

Schnell-Modus:



SA: Sicherheitsassoziation
IDi: Initiator ID
IDr: Responder ID

SchA: Schlüsselaustausch
Nonce: Zufallsdaten
Hash: zum verifizieren

IPSec: Wo ? Was ? Wie ?

IPSec - Angewandt:

WO findet man IPSec ?

Implementation:

- im Host (eigener Prozess)
- im Betriebssystem (integriert)
- im BITS (*Bump in the Stack*)
- im Router

WAS macht man mit IPSec ?

IPsec in Aktion:

- Sicherheit von Endstelle zu Endstelle (z.B. RAS)
- VPNs (Virtuelle private Netzwerke) – kurz Tunnel

WIE arbeiten VPNs ? **WIE** implementiert man VPNs ?

Das WIE ist Thema des nächsten Kapitels :-)

Part 3

VPN

VPN: Überblick I

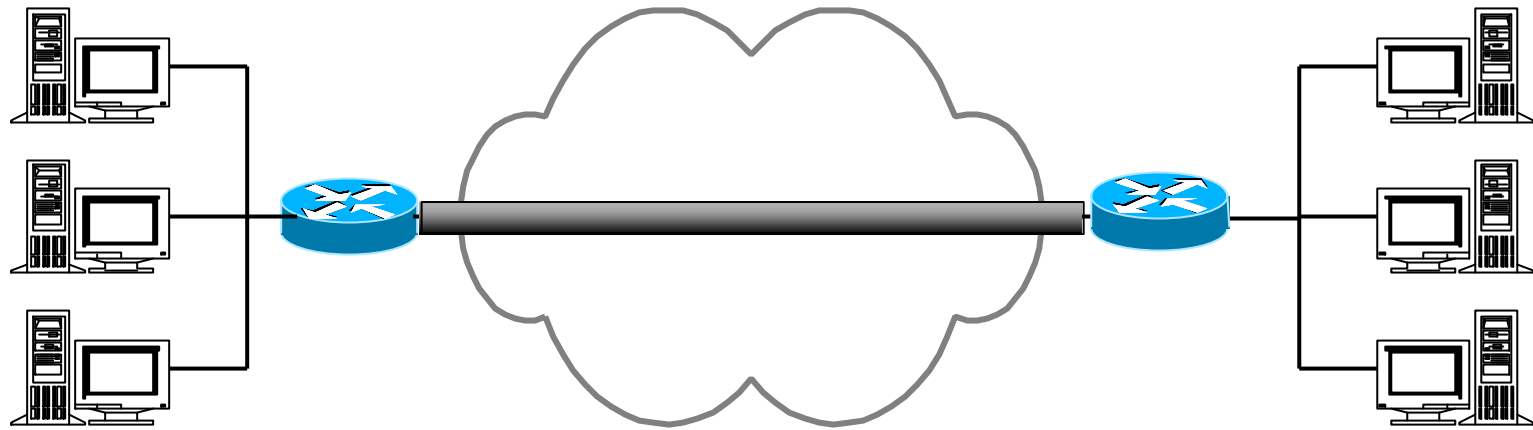
VPN – Virtuelle Private Netzwerke:

Eigenschaften von VPNs:

- werden auch Tunnel genannt,
- sind jegliche Art der Anwendung von IPSec (z.B. auch nur AH !!!)
- sind immer von Endstelle zu Endstelle (end-to-end security)
- können (nahezu) beliebig verschachtelt sein
- können beliebig verkettet werden

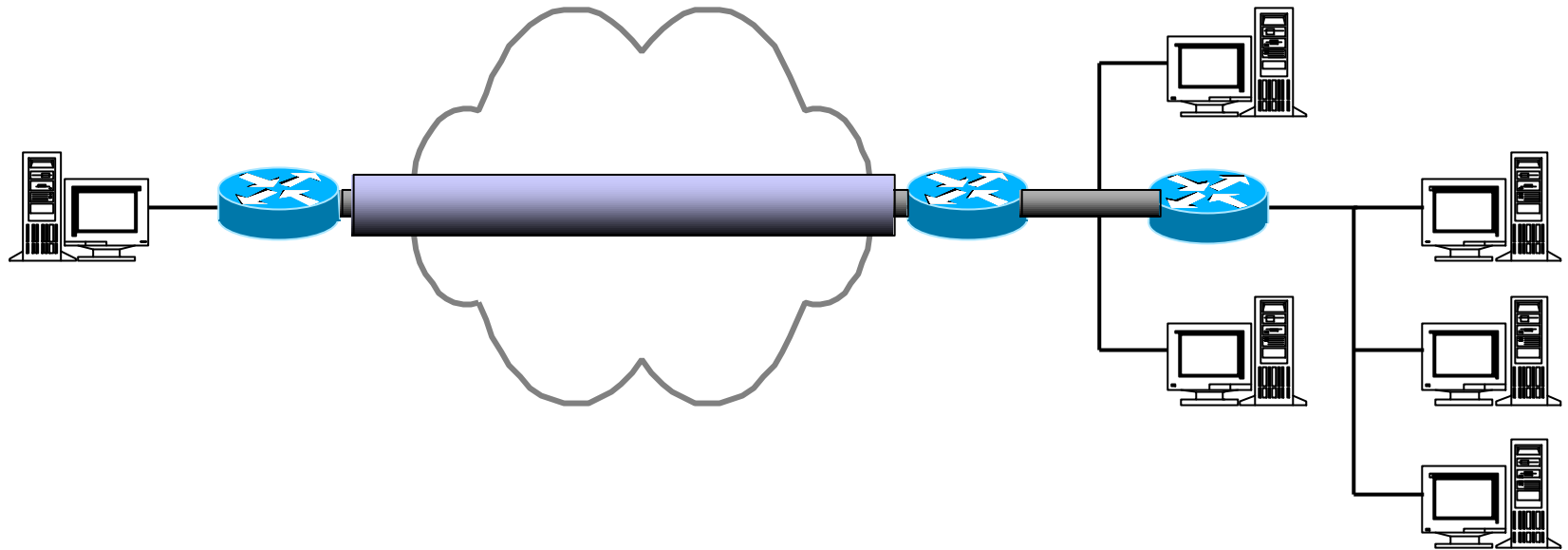
VPN: Überblick II

VPN – end-to-end security:

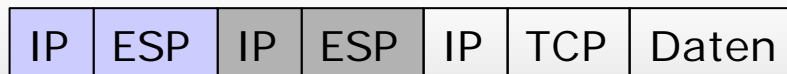


VPN: Überblick III

VPN – Verschachtelte Tunnel:

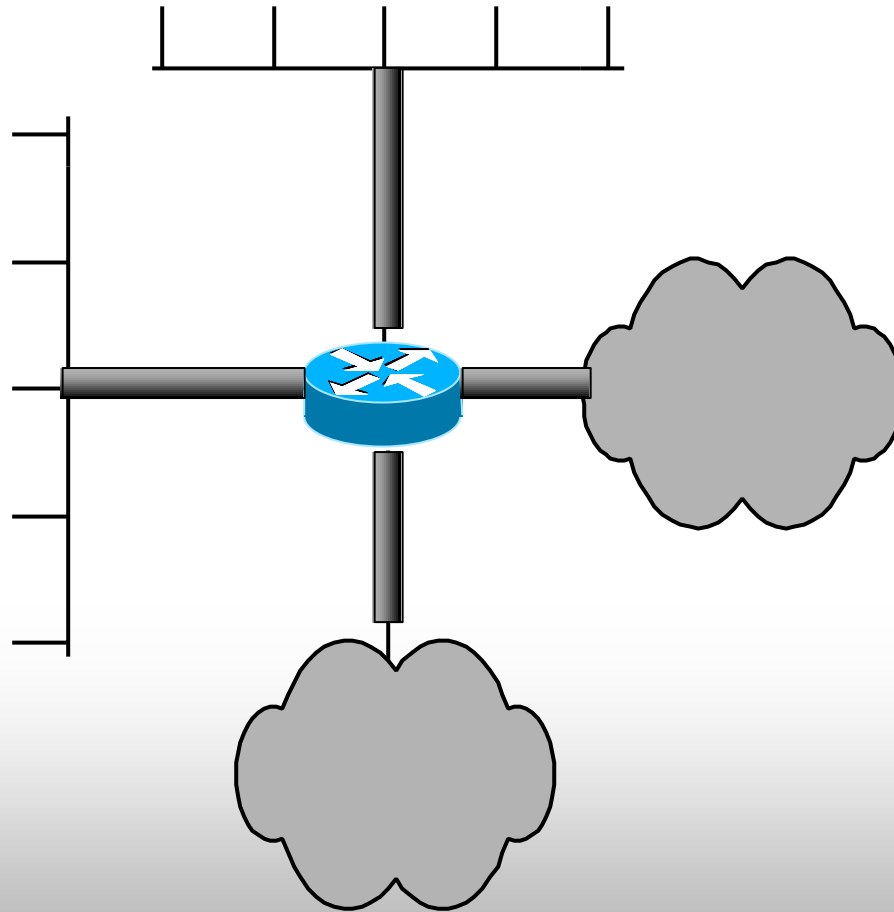


Paketformat



VPN: Überblick IV

VPN – Verkettete Tunnel:



Part 4

Anwendungs-Beispiele für VPN-Produkte

FreeS/WAN

Check Point Firewall-1

Astaro Security Linux

Produkt 1

FreeS/WAN

VPN: FreeS/WAN I

FreeS/WAN – IPSec für Unix und deren Derivate:

Unterstützt:

- AH (Authentication Header)
- ESP (Encapsulated Security Payload)
- IKE (Internet Key Exchange)
- automatische Schlüsselverteilung
- manuelle Schlüsselverteilung

Besteht aus:

- KLIPS (Kernel IPSec) implementiert AH und ESP in den Kernel
- Pluto (IKE Daemon) implementiert IKE

Bestandteil von einigen Linux Distributionen, z.B.

- SuSE
- Debian
- Mandrake
- u.a.

VPN: FreeS/WAN II

FreeS/WAN :

Installation:

- I. Kernel überprüfen. Größe 2.2.19 genügt
- II. Compilieren des Kernel-Moduls (KLIPS)
(besser, in den Kernel einkompilieren)
- III. Compilieren von Pluto

Konfiguration:

- I. Forwarding aktivieren (!!!)

VPN: FreeS/WAN III

FreeS/WAN :

Konfiguration: Schlüsselpaar in /etc/ipsec.secrets

```
# RSA 2048 bits   Firewall   Mon Jan 14 13:24:44 2002
# pubkey=0sAQ0q9Kgg9borGqqwPC4cTrBOBZY...
# (0x4200 = auth-only host-level, 4 = IPsec, 1 = RSA)

Modulus: 0xaaf4a82af5ba2...
PublicExponent: 0x03

# everything after this point is secret
PrivateExponent: 0x1c7e1c0728f45c847...
Prime1: 0xd6c6cbf64...
Prime2: 0xcbc4b0add5...
Exponent1: 0x8f2f32a42e78...
Exponent2: 0x87d875c93....
Coefficient: 0xc7f786245be9...
```

VPN: FreeS/WAN IV

Der eigentliche Tunnel wird in der Datei `/etc/ipsec.conf` konfiguriert

```
# basic configuration
config setup
    # THIS SETTING MUST BE CORRECT or almost nothing will work;
    # %defaultroute is okay for most simple cases.
    interfaces=%defaultroute
    # Debug-logging controls
    klipsdebug=none
    plutodebug=none
    plutoload=%search
    plutostart=%search
    # Close down old connection when new one using same ID.
    uniqueids=yes
```

VPN: FreeS/WAN V

FreeS/WAN : Beispiel Netz



/etc/ipsec.conf

```
# defaults for subsequent connection descriptions
conn %default
# How persistent to be in (re)keying negotiations.
keyingtries=0
# RSA authentication with keys from DNS.
authby=rsasig
leftrsasigkey=%dns
rightrsasigkey=%dns
```

VPN: FreeS/WAN VI

FreeS/WAN : Der eigentliche Tunnel

```
conn A-B
    # ID left
    leftid=@vpnleft
    leftrsasigkey=0sAQOq9Kgg9b...
    left=192.168.1.254           # Die externe IP von LEFT
    leftsubnet=192.168.2.0/24   # Internes Netz Links

    # ID right
    rightid=@vpnright
    rightrsasigkey=0sAQPk0pW/...
    right=172.17.1.254          # Die externe IP von RIGHT
    rightsubnet=172.17.2.0/24   # Internes Netz Rechts
    #
    auto=start
```

VPN: FreeS/WAN VII

FreeS/WAN : Kontrolle mit ifconfig

```
...
RX bytes:815158823 (777.3 Mb)  TX bytes:196895289 (187.7 Mb)
Interrupt:11 Basisadresse:0x1420 Speicher:d0979000-d0979c40

Ipsec0  Linkverkapselung:Ethernet  HWaddr 00:02:A5:87:44:F3
inet addr:192.168.1.254  Maske:255.255.255.0
UP RUNNING NOARP  MTU:16260  Metric:1
Empfangene Pakete:0 Fehler:0 Weggeworfen:0 Überlauf:0 Rahmen:0
Verschickte Pakete:0 Fehler:0 Weggeworfen:0 Überlauf:0 Rahmen:0
Kollisionen:0 Sendewarteschlangenlänge:10

Lo      Protokoll:Lokale Schleife
inet Adresse:127.0.0.1  Maske:255.0.0.0
...
```

VPN: FreeS/WAN VIII

FreeS/WAN : Bewertung

Positiv:

- einfache Konfiguration von FreeS/WAN zu FreeS/WAN,
- sehr kompatibel,

Negativ:

- schwierige Installation,
- Performance,
- komplizierte Konfiguration von FreeS/WAN zu anderen,

Check Point FireWall-1

VPN: Check Point Firewall-1 I

CheckPoint Firewall-1 : VPN-Überblick

Authenticaiton:

- FWZ,
- IKE,
- SKIP,

Encryption:

- FWZ-1,
- DES,
- Triple-DES,
- CAST,
- RC2,

VPN: Check Point Firewall-1 II

CheckPoint Firewall-1 : Prinzipielle Konfiguration

„Andocken“ an FreeS/WAN:

Konfiguration in 4 Schritten :

- Erzeugung manueller IPSec Schlüssel,
- Erzeugen eines Netzwerk-Objektes,
- Hinzufügen einer Encryption-Rule,
- Hinzufügen einer Translation-Rule,

VPN: Check Point Firewall-1 III

CheckPoint Firewall-1 : Prinzipielle Konfiguration

Erzeugen manueller IPsec Schlüssel:

1. Im *Policy Editor* -> *Manage/Keys* -> *New* -> *SPI*
2. Typischer *SPI-Wert*: 0x100 bis 0xfff,
3. *ESP* und *AH* aktivieren, Encryption auf *DES* oder *Triple-DES*,
4. Bei *AH* den *MD5* Algorithmus wählen,
5. Bei den *Keys* etwas „Müll“ generieren,
6. Auswahl *Manueller Schlüssel*,

Erzeugen eines Netzwerk-Objektes:

1. *Netzwerk-Objekte* für die LANs erzeugen, welche per VPN verbunden werden sollen,
2. Ein *Workstation-Objekt* für den *FreeS/WAN*-Rechner,
3. Das *VPN-Register* wählen, unter *Domain remote-lan* wählen,
4. Encryption-Schema *manual IPsec* wählen,
5. *Workstation-Objekt* der *FW-1* wählen -> *VPN Register*, unter *Domain other* wählen,
6. Encryption-Schema *manual IPsec* wählen,

VPN: Check Point Firewall-1 IV

CheckPoint Firewall-1 : Prinzipielle Konfiguration

Hinzufügen einer Encryption-Rule: (eigentlich 2)

1. Eine Regel für *Inbound*-Traffic,
Source: remote-lan, Dest: FW-1, Action: Encrypt,
2. Encryption-Properties anzeigen lassen,
3. Hier *manual IPSec* wählen,
4. Auswahl der *SPI* und des *FreeS/WAN*-Rechners, *Allowed-Peer Gateway*,
5. Hinzufügen einer *Outbound*-Regel,
Source: FW-1, Dest: remote-lan, Action: Encrypt,
6. Encryption-Properties anzeigen lassen,
7. Hier *manual IPSec* wählen,
8. Auswahl der *SPI* und der *FreeS/WAN*-Rechners, *Allowed-Peer Gateway*,

VPN: Check Point Firewall-1 V

CheckPoint Firewall-1 : Prinzipielle Konfiguration

Hinzufügen einer Translation-Rule: (eigentlich 2)

1. Im Policy Editor, Translation-Register wählen,
2. Add Rule:
Source: FW-1, Dest: remote-lan, Service: any
3. Add Rule:
Source: remote-lan, Dest: FW-1, Service: any

Übersetzen der Regeln

VPN: Check Point Firewall-1 VI

CheckPoint Firewall-1 : Bewertung

Positiv:

- Klick-Tool,
- Transparent im Log,

Negativ:

- Besch...eidene Performance,
- komplizierte Konfiguration,
- Nicht normgerecht,

Astaro Security Linux

VPN: Astaro Security Linux I


Astaro Security Linux: Überblick

- für „private use“ kostenlos
- vereint ein gehärtetes Linux mit Tools wie: Paketfilter, VPN, IDS, ...
 - Paketfilter,
 - VPN (FreeS/WAN),
 - IDS (Snort),
 - Proxies (Squid, Socks,...),
 - SMTP-Gateway mit Virens scanner,
 - NAT, Masquerading,
 - usw.
- komfortable Web-Oberfläche zur Administration,

VPN: Astaro Security Linux II

Astaro Security Linux: Überblick

Unregistered Version,
for evaluation purposes only.



System
Definitions
Network
Packet Filter
Proxies
VPN
IPSEC Configurations
IPSEC RSA Key
IPSEC LiveLog
PPTP Roadwarrior VPN
Reporting
Help
Exit

Definition of the Virtual Private Network connections

IPSEC Configurations
Definition of VPN connections and IKE debugging

IPSEC RSA Key
Administration of RSA key

IPSEC LiveLog
Display IPSEC connection status

PPTP Roadwarrior VPN
Enables PPTP remote access.

VPN: Astaro Security Linux III

Astaro Security Linux: Konfiguration

System	Edit rule	
Definitions	VPN status: <input checked="" type="checkbox"/> <input type="checkbox"/>	Disable
Network	IKE-debugging: <input type="checkbox"/> <input checked="" type="checkbox"/>	Enable
Packet Filter	New connection:	
Proxies	Name: <input type="text" value="Gregis-VPN"/>	Save
VPN	Perfect Forward Secrecy : <input type="radio"/> yes <input checked="" type="radio"/> no	
IPSEC Configurations	Secure Association: <input type="radio"/> ike <input checked="" type="radio"/> manual	
IPSEC RSA Key	SPIBASE: <input type="text" value="0x100"/>	
IPSEC LiveLog	ESP: <input type="text" value="3des-md5-96"/>	
PPTP Roadwarrior VPN	ESPENCKEY: <input type="text" value="testtesttesttesttesttesttest"/>	
Reporting	ESPAUTHKEY: <input type="text" value="testtesttesttesttesttesttest"/>	
Help	Local interface: <input type="text" value="Intern"/>	
Exit	Local subnet: <input type="text" value="Private Network 192.168.0.0"/>	
	Remote IP: <input type="text" value="Router"/>	
	Remote subnet: <input type="text" value="Outside"/>	

VPN: Astaro Security Linux IV

Astaro Security Linux: Bewertung

Positiv:

- einfache Installation,
- einfach Konfiguration,
- Debugging-Möglichkeit,
- normgerecht,
- performanter als SuSE 7.3 mit FreeS/WAN,

Negativ:

- in der „home-use“-Version nur HMAC-MD5,

Fragen ???

Bewertung/Meinungen !!!

