



Leibniz-Rechenzentrum
der Bayerischen Akademie der Wissenschaften

The background of the slide is a photograph of a modern, multi-story building with a glass and metal facade, likely a data center or university building. The image is overlaid with a semi-transparent blue filter. A dark blue horizontal bar is positioned across the middle of the image, containing the chapter title in white text.

Kapitel 13: Netzicherheit - Schicht 3: Network Layer - IPSec

- Schwächen des Internet-Protokolls (IP)

- IPSec: Sicherheitserweiterung des IP-Protokolls
 - Authentication Header (AH)
 - Encapsulating Security Payload (ESP)
 - Anwendungsbeispiele

- Schlüsselverteilung mit IKEv2 (Internet Key Exchange)
 - Aufbau einer IKE SA
 - Authentisierung der Partner
 - Aufbau der IPSec SA
 - Erzeugung von Schlüsselmaterial

IP: Gefahren und Schwächen

- Vertraulichkeit:
 - Mithören relativ einfach möglich
 - Man-in-the-middle-Angriffe
 - Verkehrsfluss-Analyse
- Integrität:
 - Veränderung der Daten
 - Session Hijacking
 - Replay-Angriffe
- Authentisierung:
 - IP Spoofing
- Lösung: IPSec (Sicherheitserweiterungen für IP)
 - Fester Bestandteil von IPv6
 - Als Erweiterungs-Header auch für IPv4 einsetzbar
 - Motivation: Erspart den Aufwand für entsprechende Gegenmaßnahmen in jeder einzelnen Anwendung (d.h. auf höheren Schichten)

IPSec Überblick



- IP Authentication Header (AH)
 - Integrität des verbindungslosen Verkehrs
 - Authentisierung des Datenursprungs (genauer: des IP-Headers)
 - Optional: Anti-Replay-Dienst

- IP Encapsulating Security Payload (ESP)
 - Vertraulichkeit (eingeschränkt auch für den Verkehrsfluss)
 - Integrität
 - Authentisierung (der sog. Security Association)
 - Anti-Replay Dienst

- Jeweils zwei verschiedene Betriebsmodi:
 - Transport Mode
 - Tunnel Mode

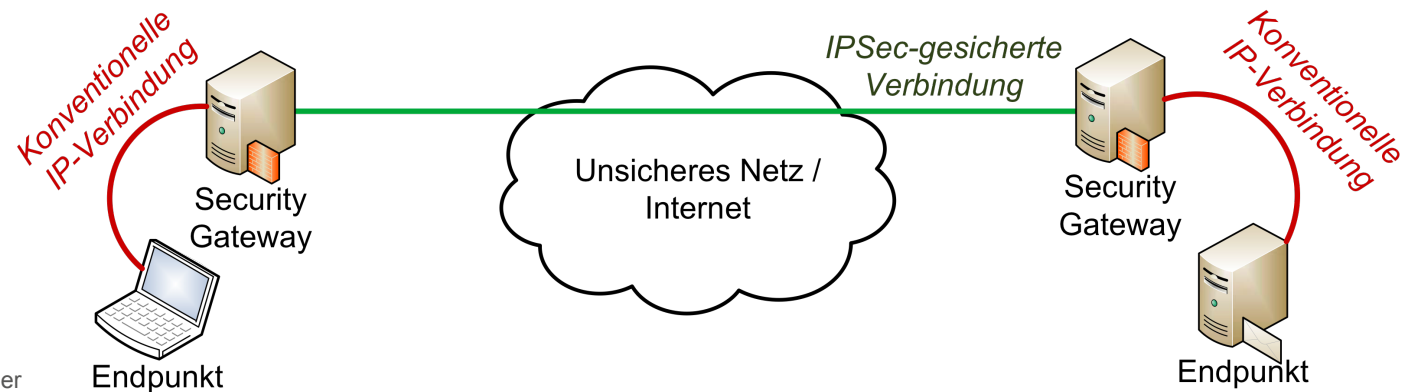
Transport Mode / Tunnel Mode

- In beiden Modi können AH und/oder ESP eingesetzt werden

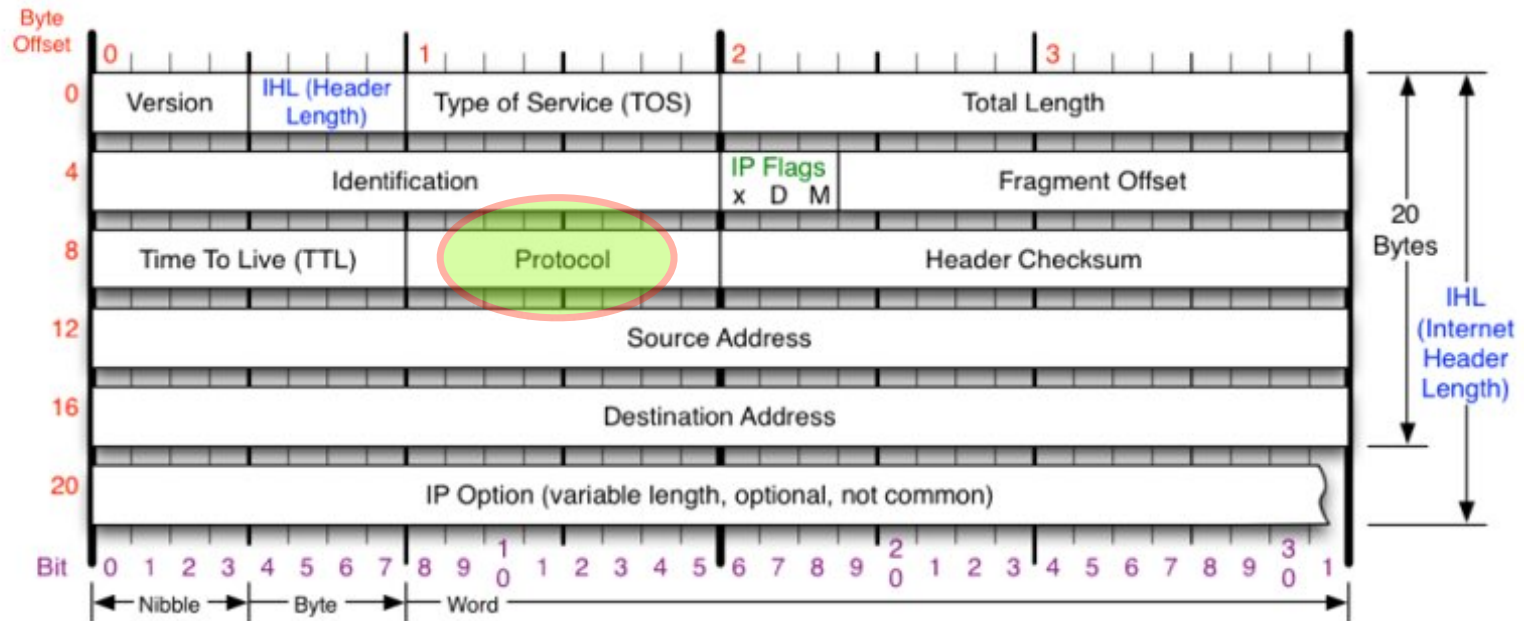
Transport Mode



Tunnel Mode



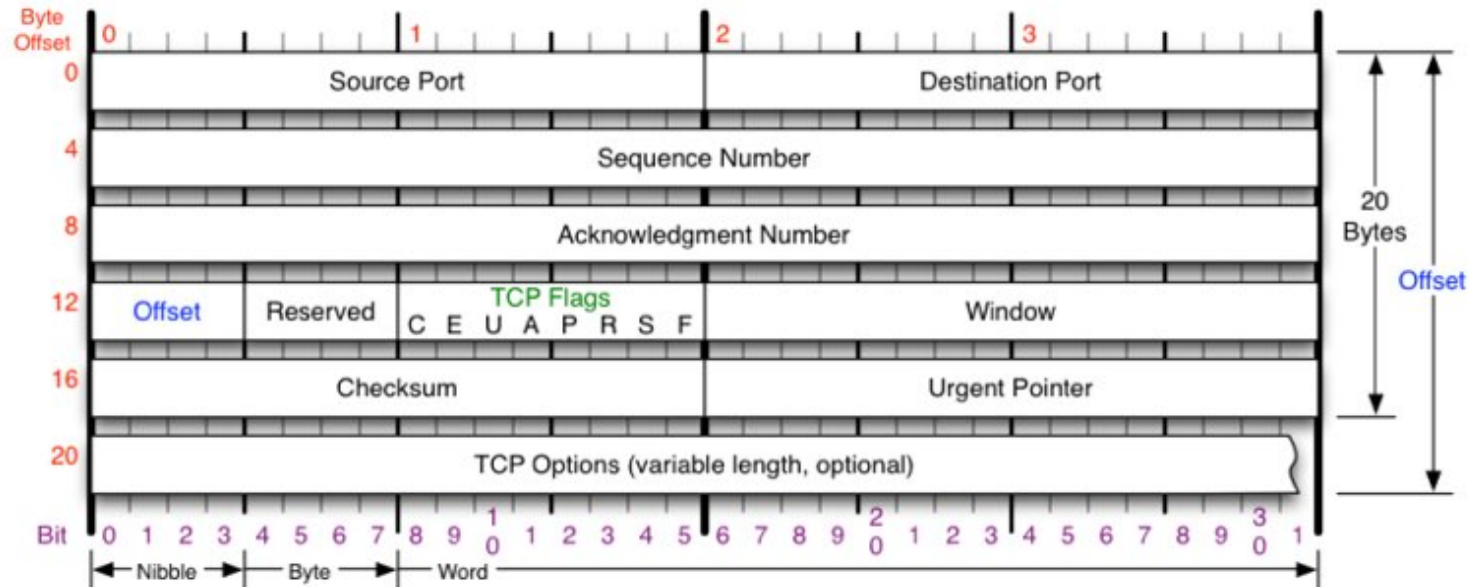
Einschub: Herkömmlicher IPv4 Header



<p>Version</p> <p>Version of IP Protocol. 4 and 6 are valid. This diagram represents version 4 structure only.</p>	<p>Protocol</p> <p>IP Protocol ID. Including (but not limited to):</p> <table border="0"> <tr> <td>1 ICMP</td> <td>17 UDP</td> <td>57 SKIP</td> </tr> <tr> <td>2 IGMP</td> <td>47 GRE</td> <td>88 EIGRP</td> </tr> <tr> <td>6 TCP</td> <td>50 ESP</td> <td>89 OSPF</td> </tr> <tr> <td>9 IGRP</td> <td>51 AH</td> <td>115 L2TP</td> </tr> </table>	1 ICMP	17 UDP	57 SKIP	2 IGMP	47 GRE	88 EIGRP	6 TCP	50 ESP	89 OSPF	9 IGRP	51 AH	115 L2TP	<p>Fragment Offset</p> <p>Fragment offset from start of IP datagram. Measured in 8 byte (2 words, 64 bits) increments. If IP datagram is fragmented, fragment size (Total Length) must be a multiple of 8 bytes.</p>	<p>IP Flags</p> <p>x D M</p> <p>x 0x80 reserved (evil bit) D 0x40 Do Not Fragment M 0x20 More Fragments follow</p>
1 ICMP	17 UDP	57 SKIP													
2 IGMP	47 GRE	88 EIGRP													
6 TCP	50 ESP	89 OSPF													
9 IGRP	51 AH	115 L2TP													
<p>Header Length</p> <p>Number of 32-bit words in TCP header, minimum value of 5. Multiply by 4 to get byte count.</p>	<p>Total Length</p> <p>Total length of IP datagram, or IP fragment if fragmented. Measured in Bytes.</p>	<p>Header Checksum</p> <p>Checksum of entire IP header</p>	<p>RFC 791</p> <p>Please refer to RFC 791 for the complete Internet Protocol (IP) Specification.</p>												

Bildquelle: nmap.org

Einschub: Herkömmlicher TCP-Header



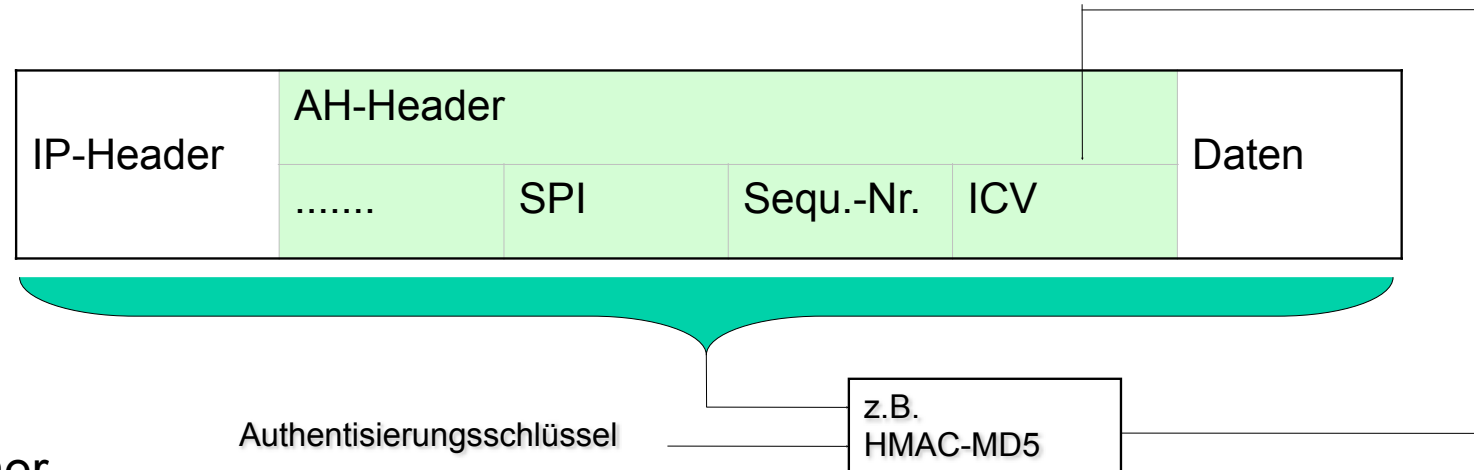
TCP Flags	Congestion Notification	TCP Options	Offset																											
C E U A P R S F	ECN (Explicit Congestion Notification). See RFC 3168 for full details, valid states below.	0 End of Options List 1 No Operation (NOP, Pad) 2 Maximum segment size 3 Window Scale 4 Selective ACK ok 8 Timestamp	Number of 32-bit words in TCP header, minimum value of 5. Multiply by 4 to get byte count.																											
Congestion Window C 0x80 Reduced (CWR) E 0x40 ECN Echo (ECE) U 0x20 Urgent A 0x10 Ack P 0x08 Push R 0x04 Reset S 0x02 Syn F 0x01 Fin	<table border="1"> <thead> <tr> <th>Packet State</th> <th>DSB</th> <th>ECN bits</th> </tr> </thead> <tbody> <tr> <td>Syn</td> <td>0 0</td> <td>1 1</td> </tr> <tr> <td>Syn-Ack</td> <td>0 0</td> <td>0 1</td> </tr> <tr> <td>Ack</td> <td>0 1</td> <td>0 0</td> </tr> <tr> <td>No Congestion</td> <td>0 1</td> <td>0 0</td> </tr> <tr> <td>No Congestion</td> <td>1 0</td> <td>0 0</td> </tr> <tr> <td>Congestion</td> <td>1 1</td> <td>0 0</td> </tr> <tr> <td>Receiver Response</td> <td>1 1</td> <td>0 1</td> </tr> <tr> <td>Sender Response</td> <td>1 1</td> <td>1 1</td> </tr> </tbody> </table>	Packet State	DSB	ECN bits	Syn	0 0	1 1	Syn-Ack	0 0	0 1	Ack	0 1	0 0	No Congestion	0 1	0 0	No Congestion	1 0	0 0	Congestion	1 1	0 0	Receiver Response	1 1	0 1	Sender Response	1 1	1 1	Checksum	RFC 793
Packet State	DSB	ECN bits																												
Syn	0 0	1 1																												
Syn-Ack	0 0	0 1																												
Ack	0 1	0 0																												
No Congestion	0 1	0 0																												
No Congestion	1 0	0 0																												
Congestion	1 1	0 0																												
Receiver Response	1 1	0 1																												
Sender Response	1 1	1 1																												
		Checksum of entire TCP segment and pseudo header (parts of IP header)	Please refer to RFC 793 for the complete Transmission Control Protocol (TCP) Specification.																											

Bildquelle: nmap.org

Autentication Header (AH)

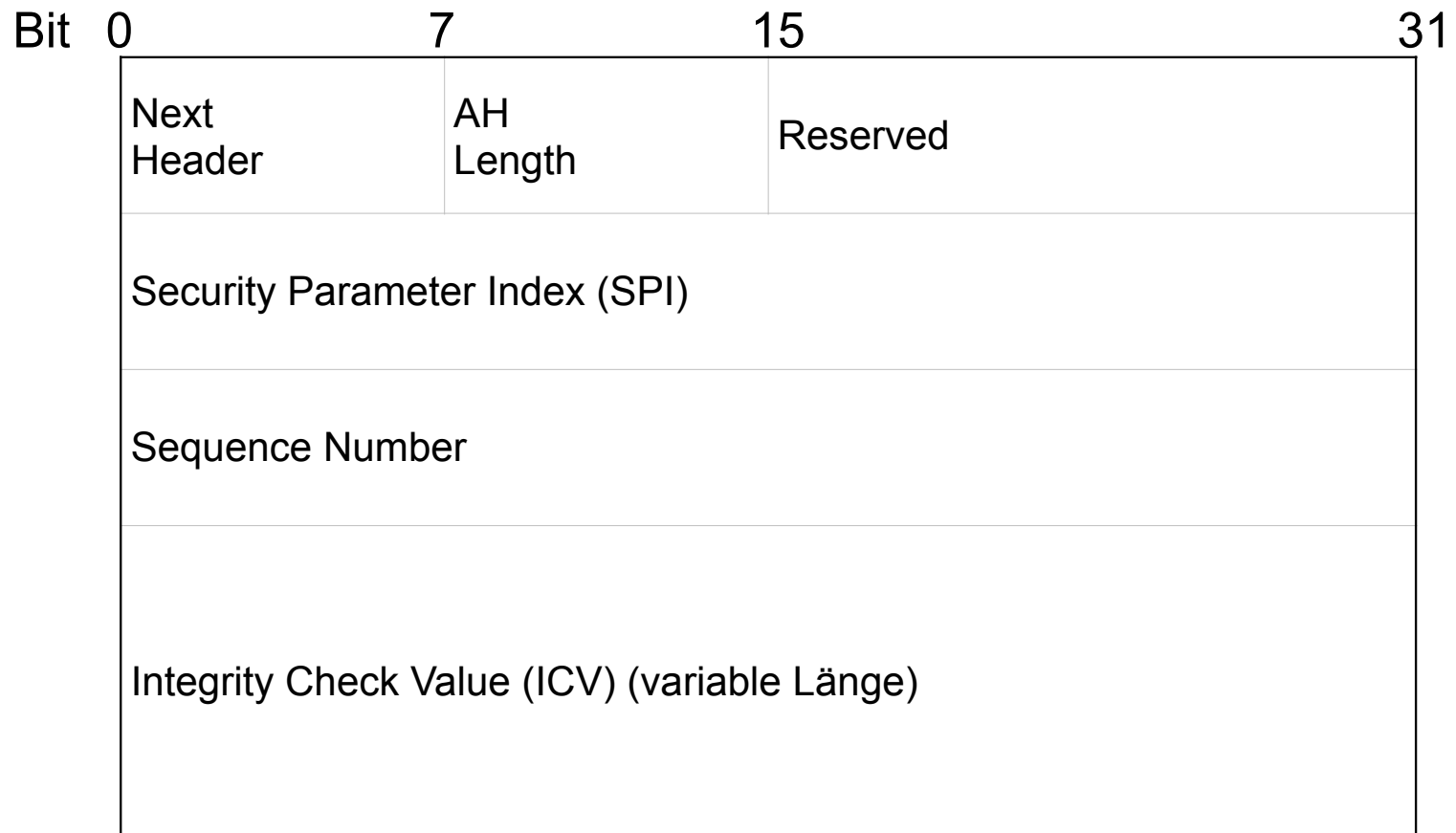
■ AH im Transport Mode

- ❑ Integrität durch MAC
- ❑ Authentisierung durch gemeinsamen Schlüssel
- ❑ Anti-Replay durch gesicherte Sequenznummer



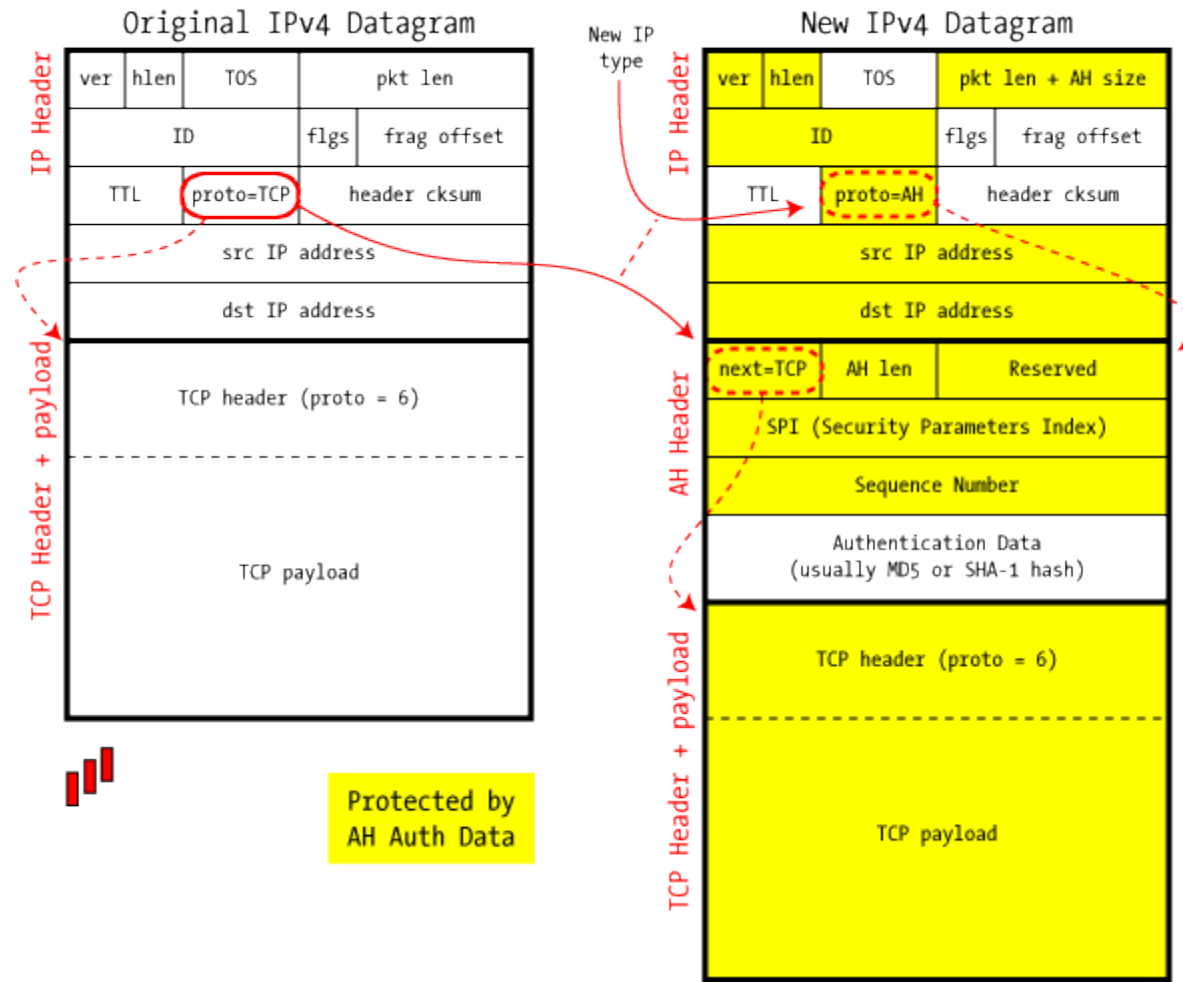
■ AH im Tunnel Mode





AH Transport Mode - Details

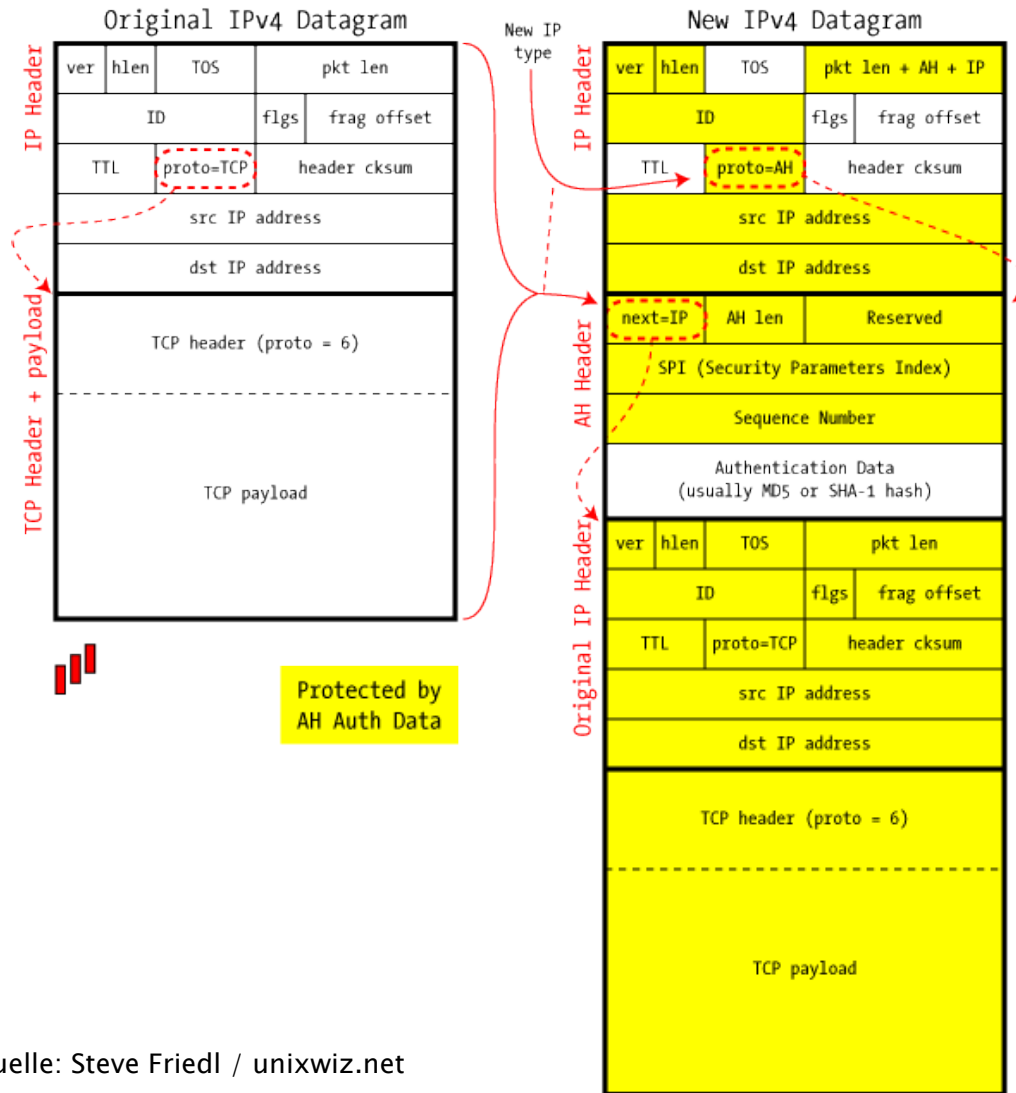
IPSec in AH Transport Mode



Bildquelle: Steve Friedl / unixwiz.net

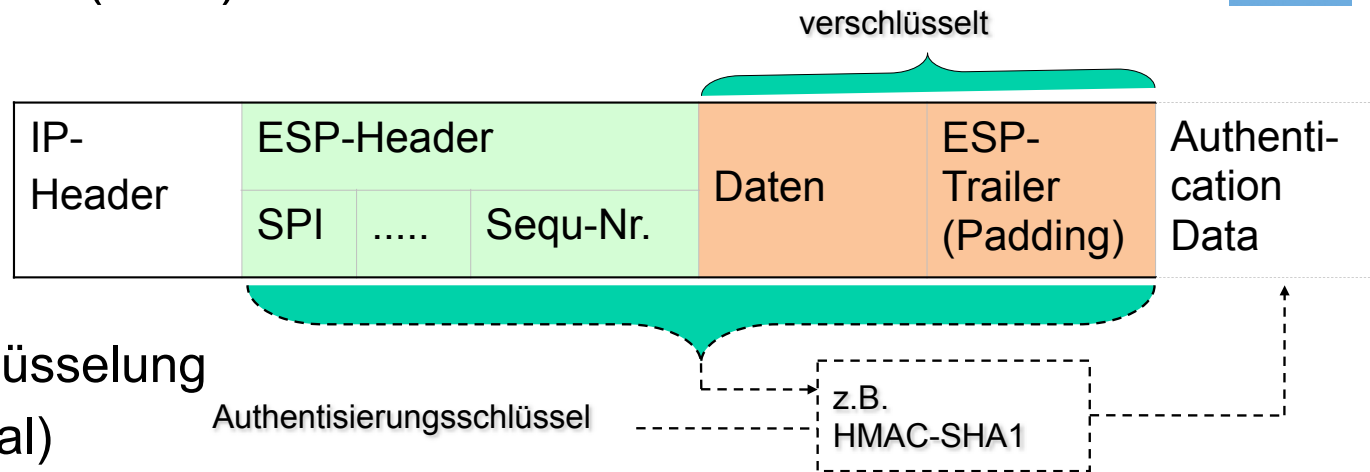
Tunnel Mode im Detail

IPSec in AH Tunnel Mode



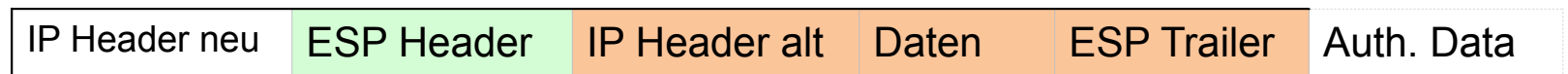
Encapsulating Security Payload (ESP) - Überblick

■ ESP Transport Mode



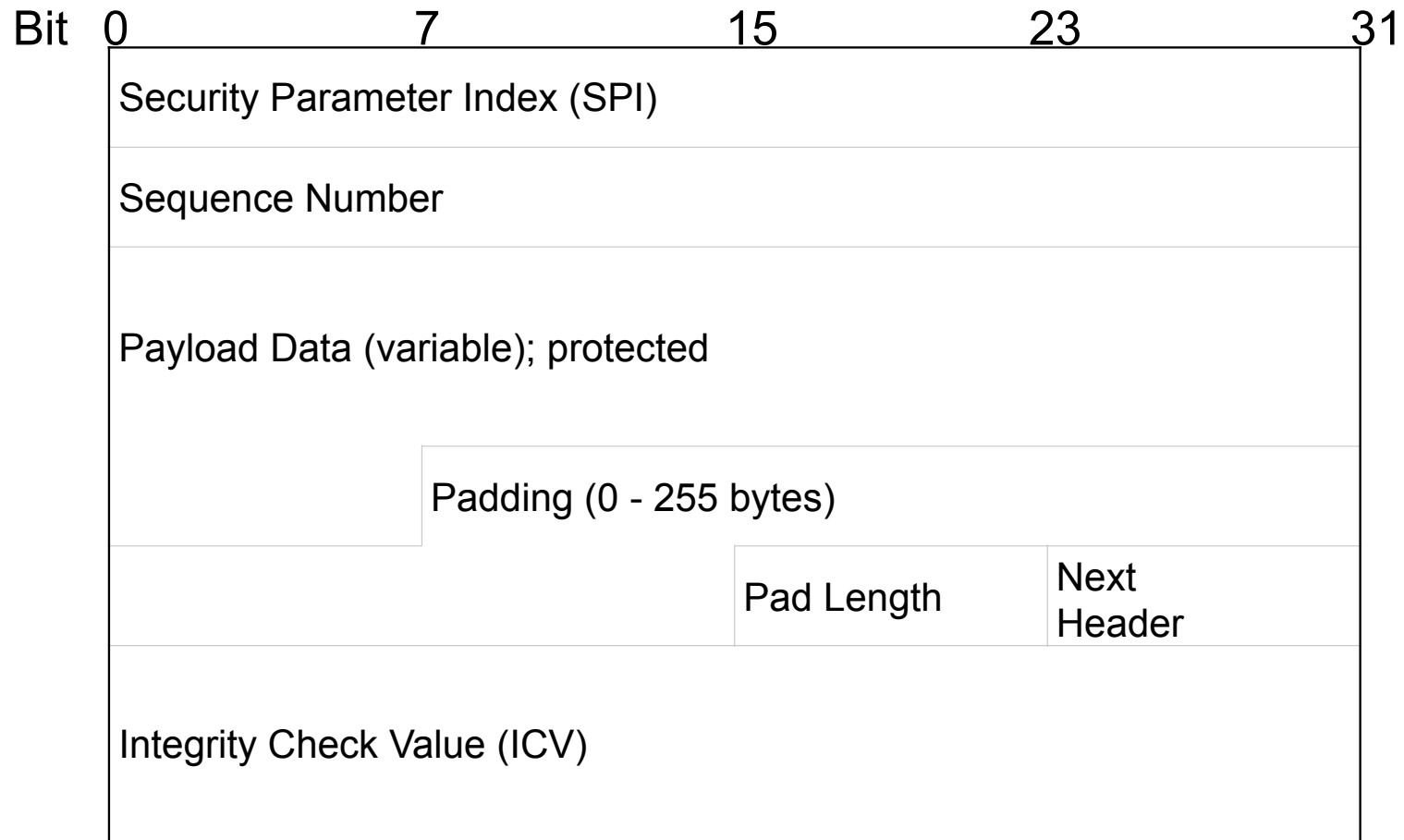
- ❑ Vertraulichkeit durch Verschlüsselung
- ❑ Integrität durch MAC (optional)
- ❑ Authentisierung durch HMAC (optional)
- ❑ Anti-Replay durch gesicherte Sequenznummer (optional)

■ ESP Tunnel Mode



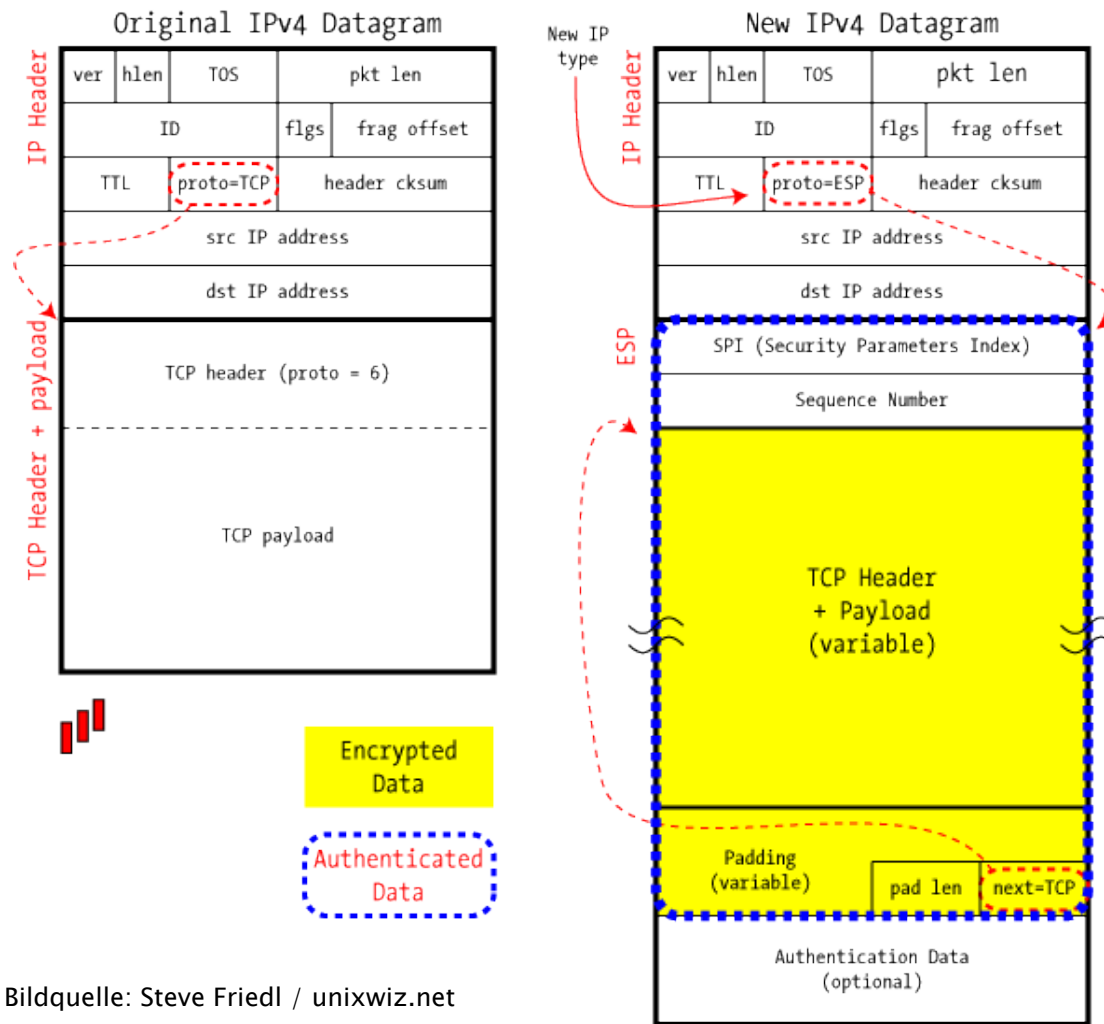
- ❑ Schutz vor Traffic-Analysen durch verschlüsselten IP-Header „alt“

ESP im Detail



ESP Transport Mode - Details

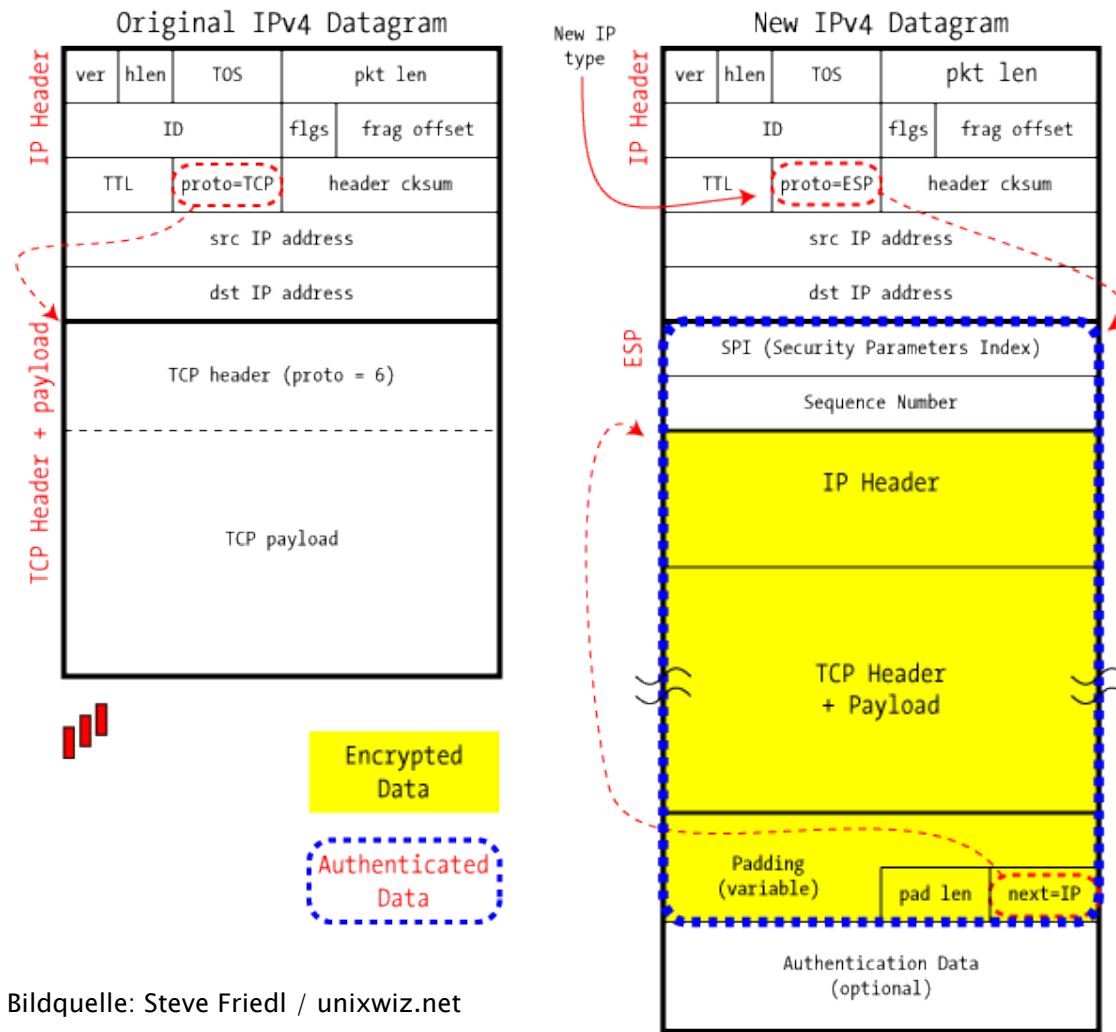
IPSec in ESP Transport Mode



Bildquelle: Steve Friedl / unixwiz.net

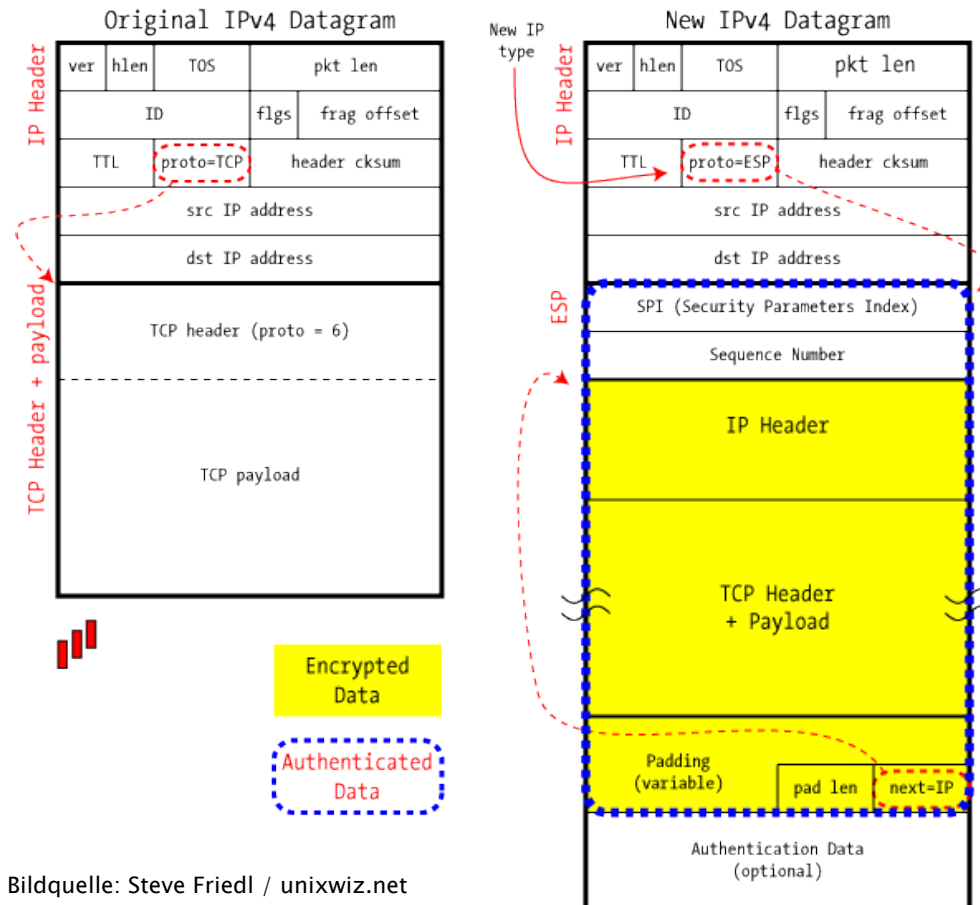
ESP Tunnel Mode - Details

IPSec in ESP Tunnel Mode



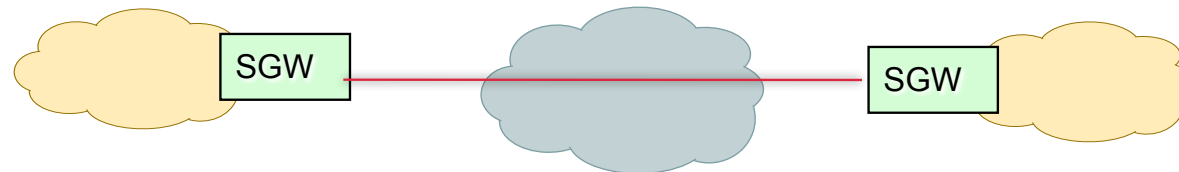
ESP Tunnel Mode - Details

IPSec in ESP Tunnel Mode

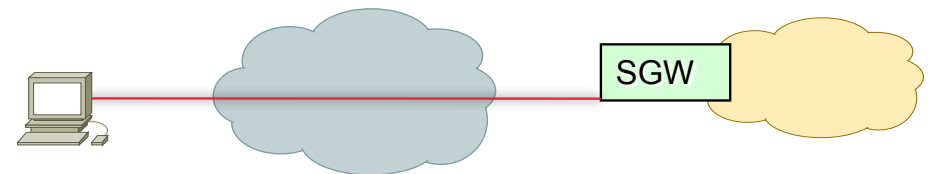


IPSec Anwendungsszenarien

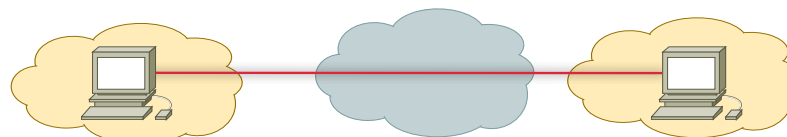
- AH und ESP können kombiniert verwendet werden
- Auch Tunnel und Transport Mode können kombiniert werden
- Mögliche Einsatzszenarien
 - Kopplung von verschiedenen Unternehmensstandorten
Verbindung von Security Gateway (SGW) zu Security Gateway



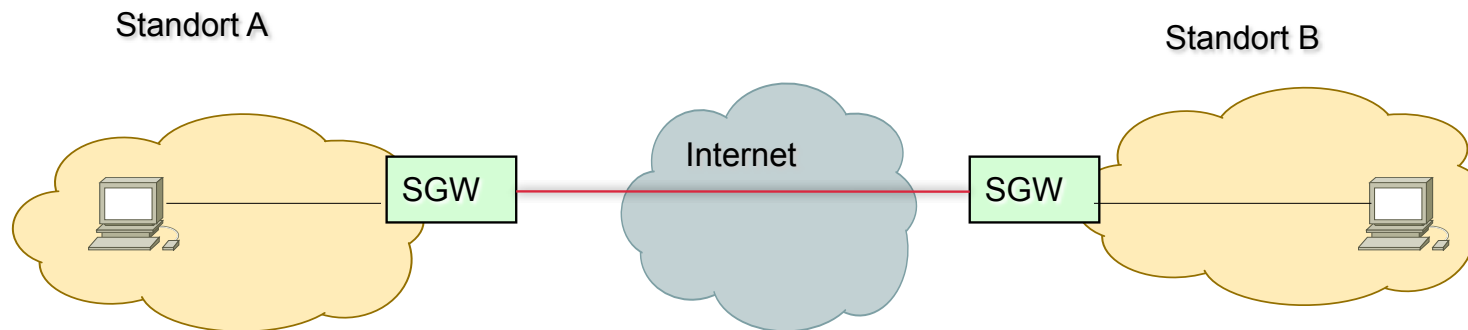
- Telearbeitsplätze; Remote Access („Road Warrior“)Endsystem zu SGW



- End-to-End



Szenario: Standortvernetzung



- Mögliche Anforderungen:
 - ❑ Authentisierung SGW-to-SGW oder End-to-End
 - ❑ Integritätssicherung SGW-to-SGW oder End-to-End
 - ❑ Schutz gegen Replay-Angriffe
 - ❑ Vertraulichkeit auch im (jeweils) internen Netz
 - ❑ SGW realisiert auch Firewall-Funktionen
 - ❑ Verwendung privater IP-Adressen in den Standorten
 - ❑ Verschattung interner Netzstrukturen

Protokollkombinationen

■ AH Tunnel Mode am Security Gateway

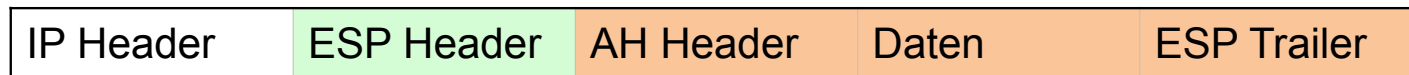
- Integritätssicherung
- Authentisierung SGW to SGW
- Private Adressen im internen Netz

■ ESP Tunnel Mode am Security Gateway

- Vertraulichkeit (auch der privaten Adressen)

■ AH Transport am Endsystem / ESP Transport am SGW

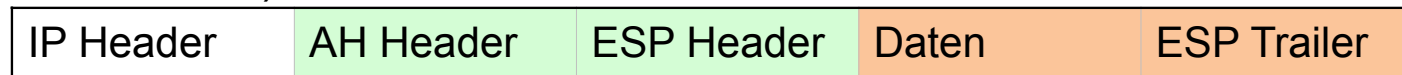
- Integritätssicherung
- Authentisierung End to End
- Vertraulichkeit ab SGW
- Private Adressen nicht möglich



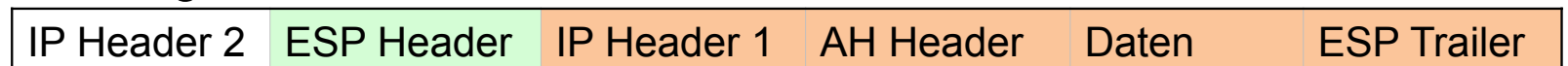
- Nur theoretische Kombination; praktisch schwer realisierbar (Empfänger SGW nicht adressierbar)

Protokollkombinationen (2)

- ESP Transport am Endsystem, AH Transport am SGW
 - Vertraulichkeit End to End
 - Authentisierung SGW to SGW
 - Private Adressen nicht möglich
 - SGW kann nicht mehr filtern (wegen Verschlüsselung)
 - Theoretisches Beispiel, in der Praxis schwer realisierbar, SGW nicht adressiert (transparentes SGW)



- AH Transport am Endsystem / ESP Tunnel am SGW
 - Integritätssicherung
 - Authentisierung End to End
 - Vertraulichkeit ab SGW
 - Private Adressen möglich



IPSec Security Association (SA)

- Inhalt einer SA
 - IPSec Protokoll Modus (Tunnel oder Transport)
 - Parameter (Algorithmen, Schlüssel, Zertifikat, Initialisierungsvektor,...)
 - Lebensdauer der SA
 - Sequenznummernzähler mit –overflow
 - Anti-Replay-Window
 -
- Identifikation einer SA per Kombination aus:
 - Security Parameter Index (SPI); 32-Bit Zahl
 - Ziel-Adresse
 - Verwendetes Protokoll (AH, ESP)
- D.h. in jede Kommunikationsrichtung wird eine eigene SA vereinbart
- Jeder IPSec-Teilnehmer hat eine lokale Security Policy Database (SPD) mit SAs

- Schwächen des Internet-Protokolls (IP)

- IPSec: Sicherheitserweiterung des IP-Protokolls
 - Authentication Header (AH)
 - Encapsulating Security Payload (ESP)
 - Anwendungsbeispiele

- Schlüsselverteilung mit IKEv2 (Internet Key Exchange)
 - Aufbau einer IKE SA
 - Authentisierung der Partner
 - Aufbau der IPSec SA
 - Erzeugung von Schlüsselmaterial

Grundlage: Diffie-Hellman Schlüsselaustausch



- Ermöglicht den sicheren Austausch eines Schlüssels über einen unsicheren Kanal:
- Primzahl p und eine primitive Wurzel $g \pmod{p}$ dürfen öffentlich bekannt gemacht werden
(oft als Diffie-Hellman Group bezeichnet)

- Alice wählt ein x aus $[1..p-2]$
- Bob wählt ein y aus $[1..p-2]$
- Alice schickt $A = g^x \pmod{p}$ an Bob
- Bob schickt $B = g^y \pmod{p}$ an Alice

- Beide verwenden den folgenden Schlüssel:

$$Key = A^y = (g^x)^y = g^{xy} = (g^y)^x = B^x \pmod{p}$$

Diffie-Hellman Beispiel

- Achtung: Üblicherweise Zahlen mit mehreren hundert Stellen!
- Alice und Bob einigen sich auf $p=13$ und $g=2$
- Alice wählt zufällig $x=5$, Bob wählt zufällig $y=7$
- Alice berechnet $A = 2^5 \bmod 13 = 6$, schickt dies an Bob
- Bob berechnet $B = 2^7 \bmod 13 = 11$, schickt dies an Alice
- Alice berechnet $11^5 \bmod 13 = 7$
- Bob berechnet $6^7 \bmod 13 = 7$
- Beide erhalten also das Ergebnis 7

- Angreifer kann die Zahlen 13, 2, 6 und 11 mithören, den Wert 7 aber nicht berechnen, da g^{xy} aufwendig zu berechnen ist, selbst wenn g , g^x und g^y bekannt sind.
(Eng verwandt mit dem Diskreten-Logarithmus-Problem)

IPSec Schlüsselaustausch über IKEv2



■ Protokollprimitive

1. IKE_INIT

- Aufbau einer bidirektionalen IKE SA

2. IKE_AUTH

- Authentisierung der Partner
- Aufbau der ersten (und oft einzigen) bidirektionalen IPSec SA

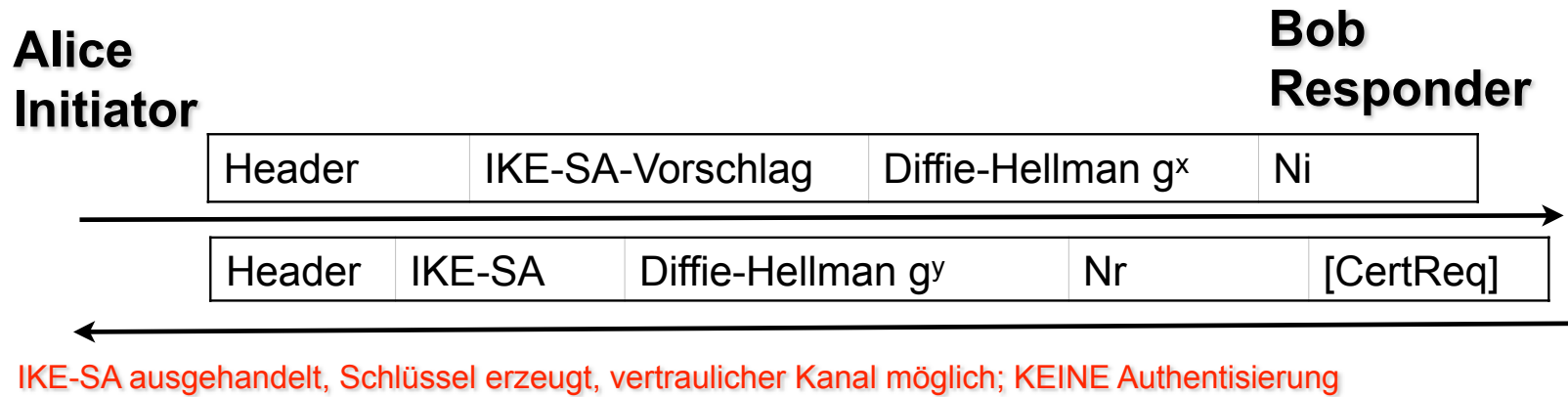
3. IKE_CHILD_SA

- Aushandeln weiterer IPSec SAs
- Re-Keying einer bestehenden SA

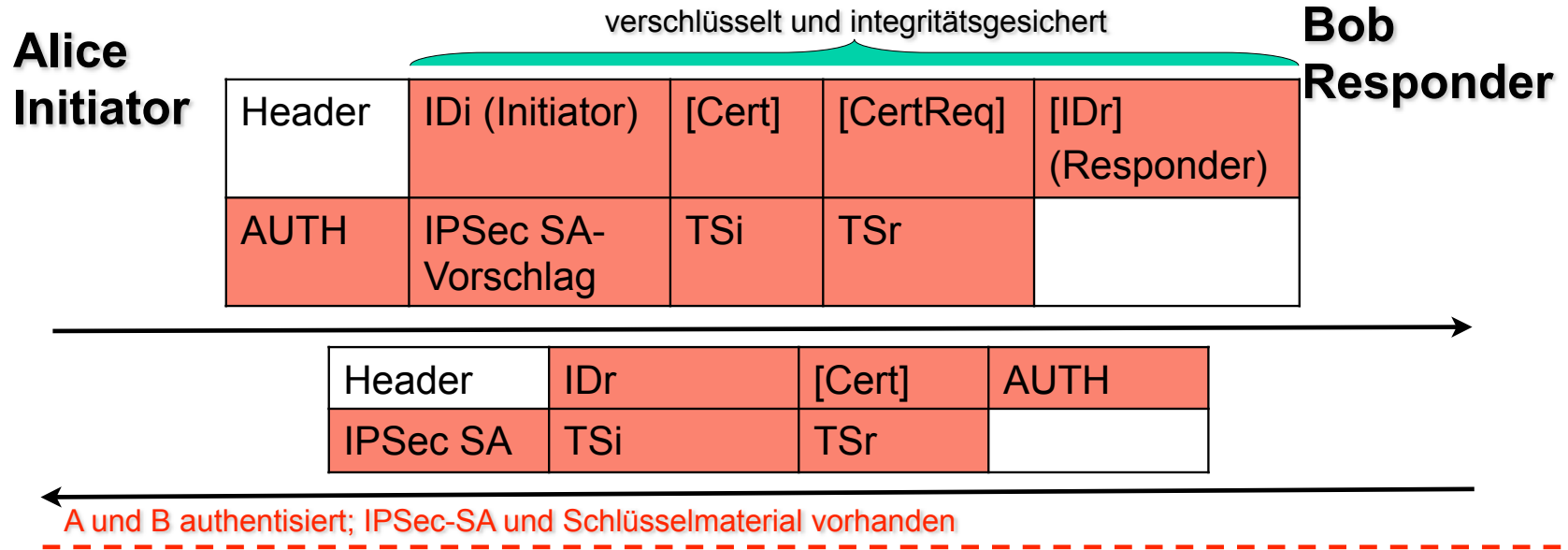
- Ein durch IKE_AUTH etablierter Kanal kann für mehrere IKE_CHILD_SA Exchanges verwendet werden

■ Ziele:

- Erzeugung des für IPSec benötigten Schlüsselmaterials
- Authentisierung der Gegenseite schon in IKE (nicht erst in IPSec)



- IKE-SA-Vorschlag:
 - enthält die vom Initiator unterstützten Algorithmen
- Ni, Nr Zufallszahlen
- Diffie-Hellman Verfahren zur Berechnung von SKEYSEED
- Ableitung aus SKEYSEED (für jede Richtung separat)
 - SK_a: Authentisierungsschlüssel
 - SK_e: Schlüssel für Kryptoverfahren
- CertReq: Anforderung von Zertifikat(en); Optional

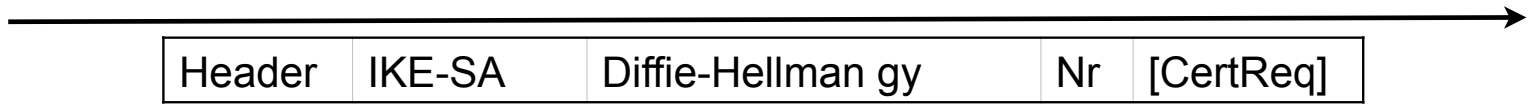


- Initiator und Responder können mehrere IDs haben; IDi und IDr bestimmen die jeweils gewählte ID
- Authentisierung über Public Key in AUTH
- Zertifikat und entsprechende Kette in Cert (Optional)
- TSx enthält Informationen aus lokaler Security Policy Database

- Falls IP-Paket verarbeitet wird, für das „protect“ in der SPD gesetzt ist:
 - Paket muss verschlüsselt werden
 - Mögliches Problem: Es existiert keine SA
 - SPD-Verwaltung ist keine Aufgabe von IKE
 - Aber IKE dient zur Aushandlung von SAs
 - Informationen aus lokaler SPD können über TSx weitergegeben werden
 - Damit Wahrung der Konsistenz

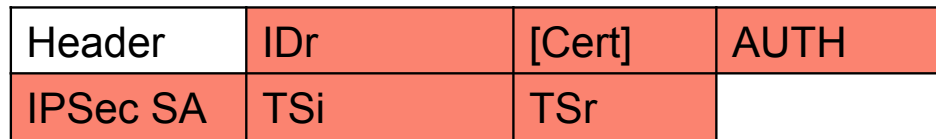
- Bsp.: Bob ist Gateway für privates Subnetz
 - Alice will Verkehr ins Subnetz 10.11.12.* tunneln
 - TSi enthält Adress-Range: 10.11.12.0 - 10.11.12.255
 - Bob kann Adress-Range in TSr einschränken

Zusammenfassung



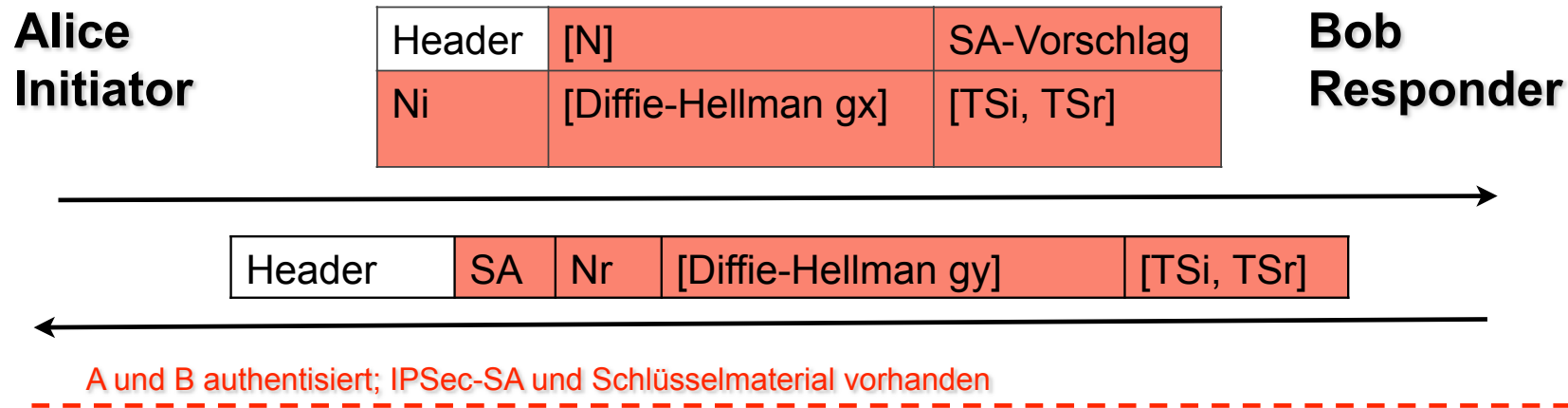
IKE-SA ausgehandelt, Schlüssel erzeugt, vertraulicher Kanal möglich; KEINE Authentisierung

verschlüsselt und Integrität gesichert



A und B authentisiert; IPSec-SA und Schlüsselmaterial vorhanden

CREATE_CHILD_SA



- Optional, da SA bereits mit IKE_AUTH ausgehandelt wird
- N enthält existierende SA, für die neues Schlüsselmaterial berechnet werden soll
- Optionaler Diffie-Hellman Key Exchange für Forward Security
- Nx sind von Initiator / Responder gewählte Zufallszahlen

Schlüsselgenerierung

- IKE-SA legt fest:
 - Verschlüsselungsalgorithmus
 - Integritätssicherungsalgorithmus
 - Diffie-Hellman Group (p und g)
 - Zufallszahlenfunktion (Pseudo-random function, prf)
- prf wird zur Schlüsselerzeugung verwendet;
- Abhängig von der benötigten Schlüssellänge wird prf iteriert
 - $\text{prf}^+(K, S)$
 - $\text{prf}^+ = T1 \mid T2 \mid T3 \mid T4 \mid \dots$ mit $K = \text{Key}$
 $S = \text{Seed}$
 - $T1 = \text{prf}(K, S \mid 0x01)$
 - $T2 = \text{prf}(K, S \mid 0x02)$
 -
 - $Tn = \text{prf}(K, S \mid 0x n)$

IKE-Schlüsselmaterial

■ IKE-SA Schlüsselmaterial:

- SK_d verwendet zur Ableitung neuer Schlüssel für CHILD_SA
- SK_{ai} Schlüssel für Integritätssicherung des Initiators
- SK_{ar} Schlüssel für Integritätssicherung des Responders
- SK_{ei} und SK_{er} Schlüssel für Verschlüsselung
- SK_{pi} und SK_{pr} Erzeugung der AUTH Payload

■ $SKEYSEED = \text{prf} (Ni | Nr , g^{xy})$

■ IKE-SA Schlüsselmaterial:

$$\{SK_d | SK_{ai} | SK_{ar} | SK_{ei} | SK_{er} | SK_{pi} | SK_{pr}\} = \text{prf+} (SKEYSEED, Ni | Nr | SPI_i | SPI_r)$$

■ CHILD_SA Schlüsselmaterial:

- $KEYMAT = \text{prf+} (SK_d , Ni | Nr)$ bzw.
- $KEYMAT = \text{prf+} (SK_d, g^{xy} | Ni | Nr)$

- mehrere Alternativen:

- Durch digitale Signatur eines vordefinierten Datenblocks
 - Verifikation durch Empfänger
 - Zertifikat (und evtl. entsprechende Kette) erforderlich
 - Optionale Anforderung und Übertragung: CertReq und Cert
 - Zertifikat kann auch schon bekannt sein

- Durch HMAC des Datenblocks

- Durch Verwendung des Extensible Authentication Protocol (EAP, vgl. Kap. 9)