IT-Security 1

Chapter 7: IPsec

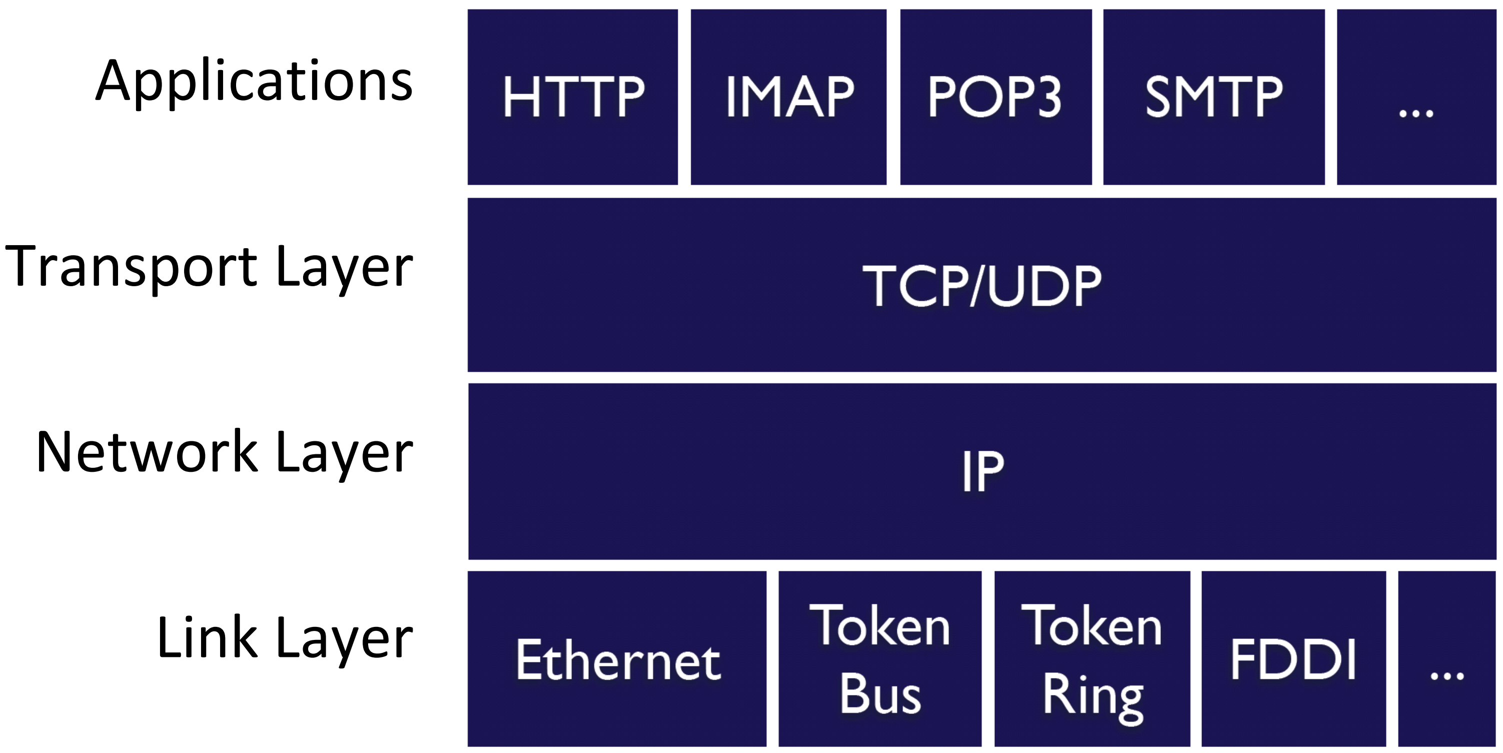
Prof. Dr.-Ing. Ulrike Meyer

WS 15/16

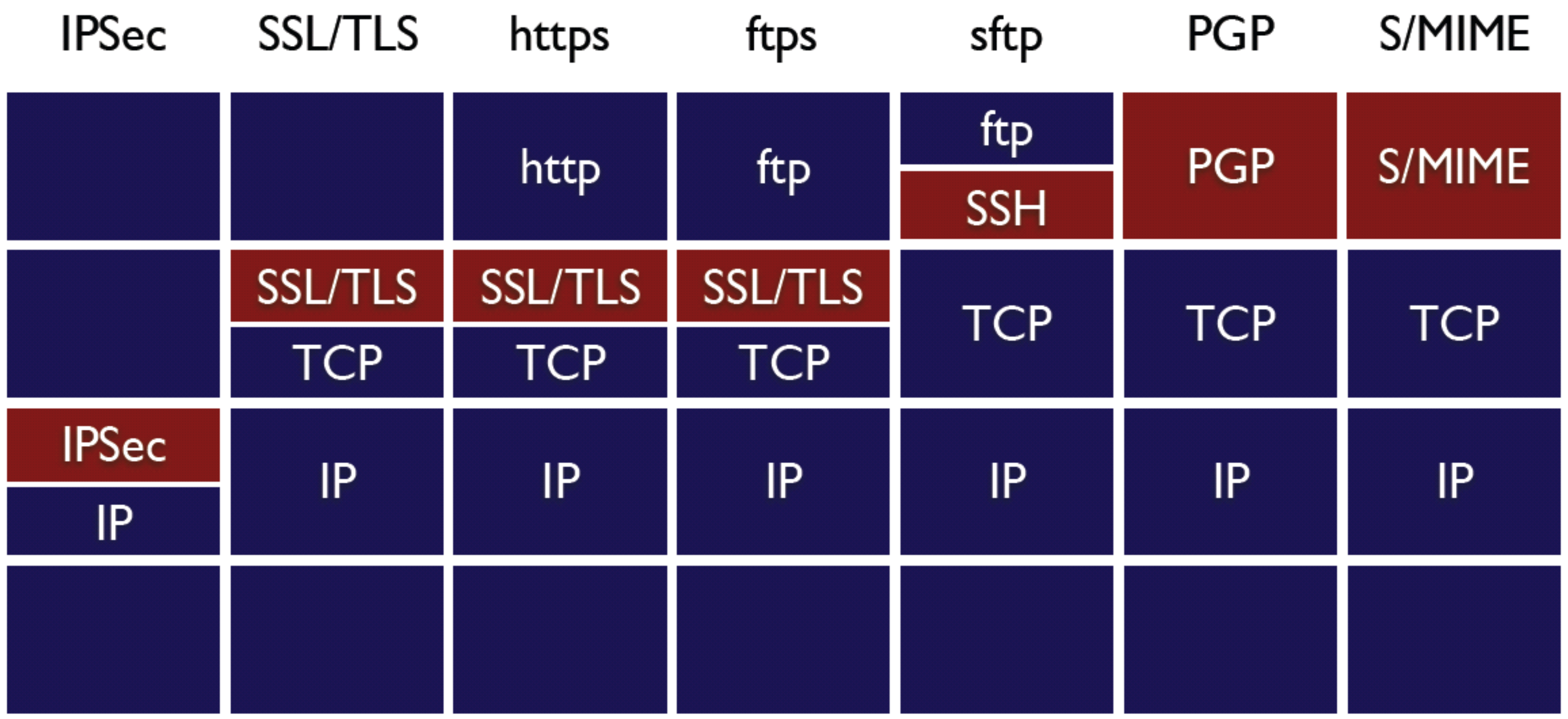
Chapter Overview

- Use Cases for IPsec
- IPsec RFCs
- Tunnel and transport mode
- Authentication header protocol
- Encapsulating security payload protocol
- Concepts and Processing
- Key Management for IPsec
  - IKE v1 as well as IKE v2

The TCP/IP Protocol Stack



Security Mechanisms on Different Layers



- IPSec = IP Security
- SSL/TLS = Secure Socket Layer / Transport Layer Security
- SSH = Secure Shell
- PGP = Pretty Good Privacy
- S/MIME = Secure Multipurpose Mail Extensions

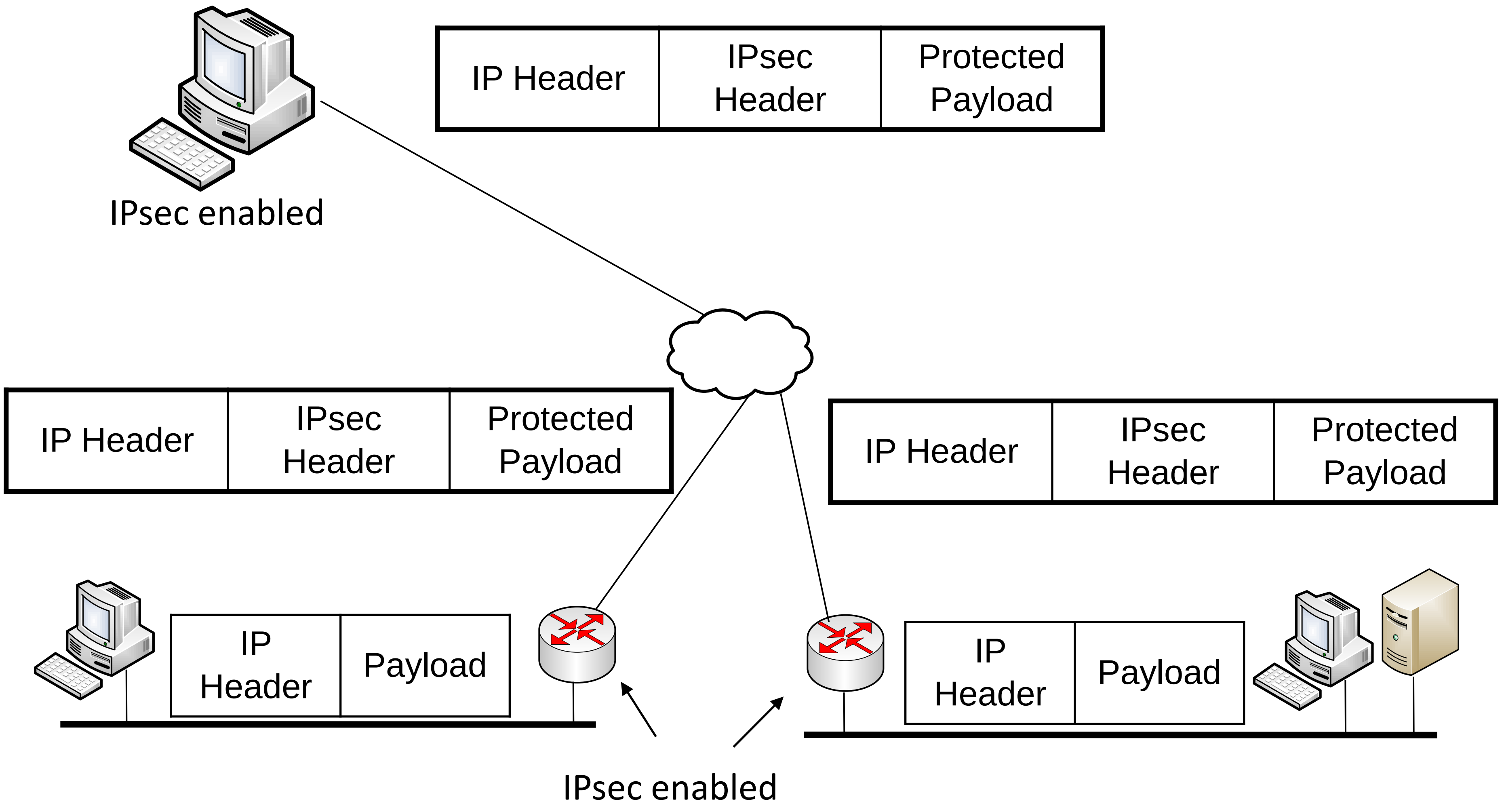
(Very) High-Level View

- IPsec is a capability that can be added to IPv4 or IPv6 by means of additional headers
- IPsec encompasses authentication, confidentiality and key management
- Authentication makes use of a MAC computed over the payload or the payload and the header of an IP packet
- Confidentiality is provided by encryption of the payload or the payload and the header of an IP packet
- IPsec defines a number of techniques for key management
- IPsec is transparent to applications above the transport layer

Example Applications of IPsec

- Secure connection between LANs at different branches of a company over a public WAN
  - IPsec typically implemented in access routers / firewalls at the edge of each LAN
- Secure remote access to a company’s Intranet over the Internet
  - IPsec typically implemented on the remote host and in an access router / firewall at the edge of the company’s private network
- Secure connections between individual hosts
- Secure connections between routers

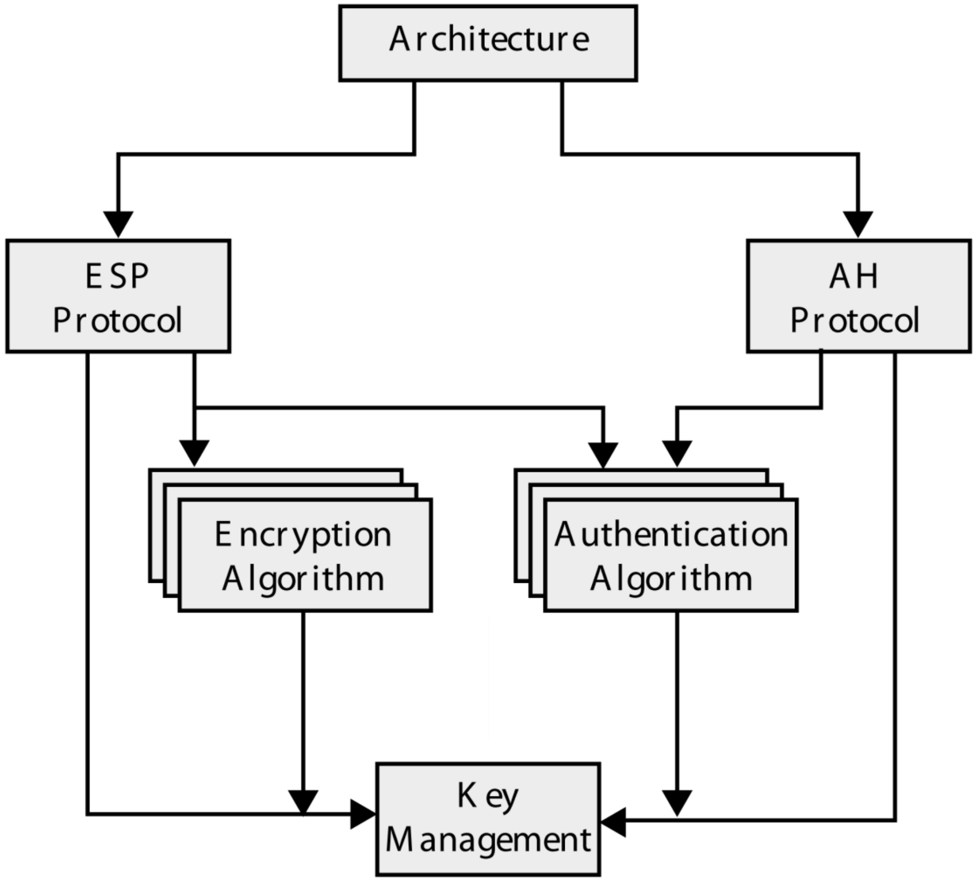
Example Use Case



The IPsec Protocol Family

- IPsec is a collection of protocols designed by IETF
- The base RFC "Security Architecture for IP
  - First version: RFC 2401, New version: RFC 4301
- The Encapsulating Security Payload protocol (ESP)
  - First version: RFC 2406, New version: RFC 4303
- The Authentication Header protocol (AH)
  - First version: RFC 2402, New version: RFC 4302
- The Key Management (and Authentication)
  - RFC 2409 (IKEv1), RFC 2407, 2408 (ISAKMP), RFC 5996 (IKEv2)
- Several other RFCs specifying the exact use of cryptographic algorithms, NAT traversal etc.

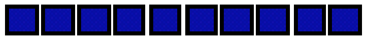
Overview on Types of Documents

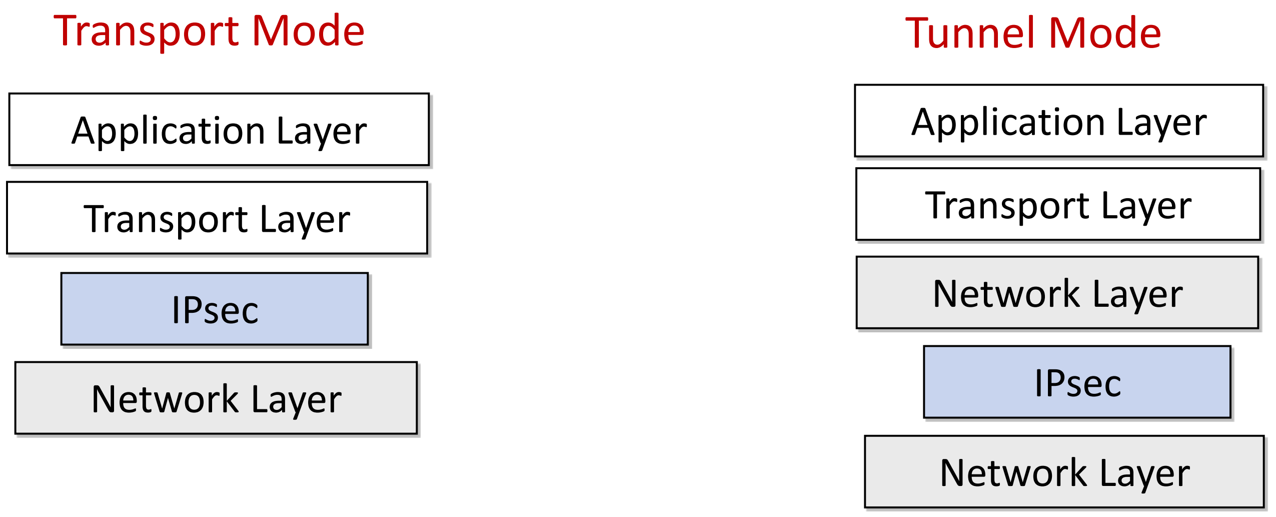


Security Services Provided by IPsec

- IPsec allows a system to
  - Select required security protocols (AH, ESP)
  - Determine the algorithms to be used
  - Put in place the required keying material
- IPsec provides the following services
  - IP-packet-level origin authentication
  - IP-packet-level integrity
  - Protection against replay of IP packets
  - IP-packet-level encryption
  - Limited traffic flow confidentiality
    - Hiding the IP address of the communicating parties to some extend

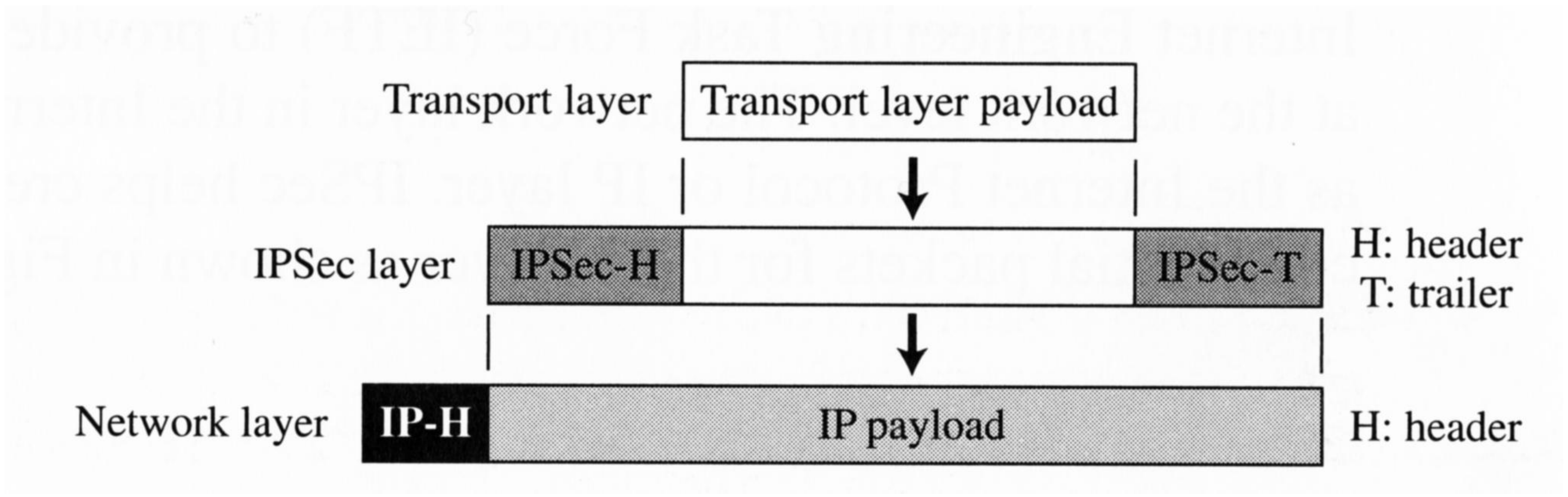
Tunnel Mode and Transport Mode

- IPsec operates in one of two modes
  - Transport mode 
    - The payload of an IP packet (without the header) is protected before it is handled by the IP layer
  - Tunnel mode
    - The complete IP packet is protected including the IP header



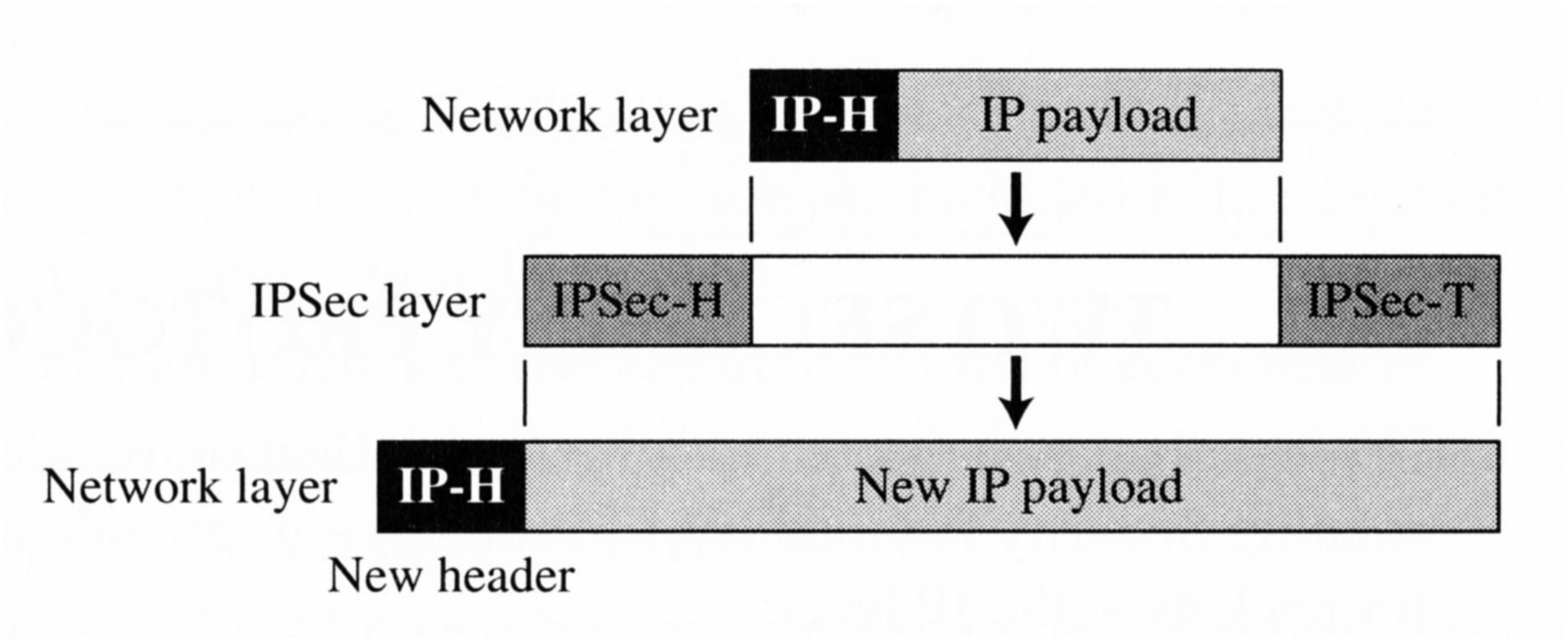
Transport Mode

- In Transport mode new header or a new header and a trailer are inserted between IP-header and payload of an IP packet

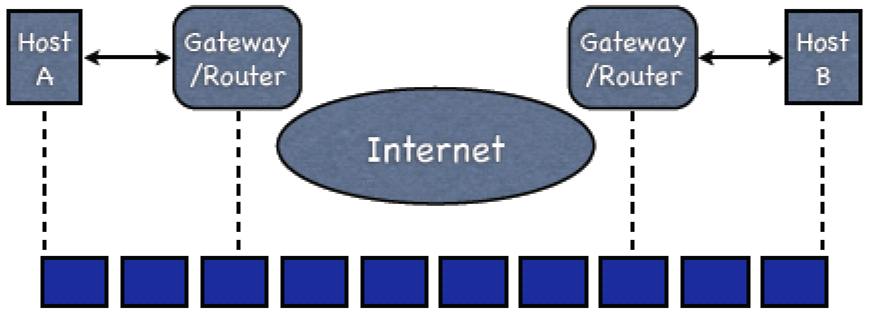


Tunnel Mode

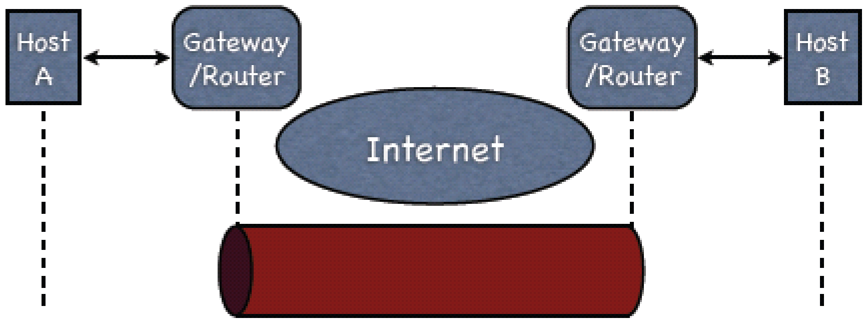
- In tunnel mode a header (and trailer) is added to the IP packet and a new IP header is added to the protected packet



Example Use Cases for the Modes

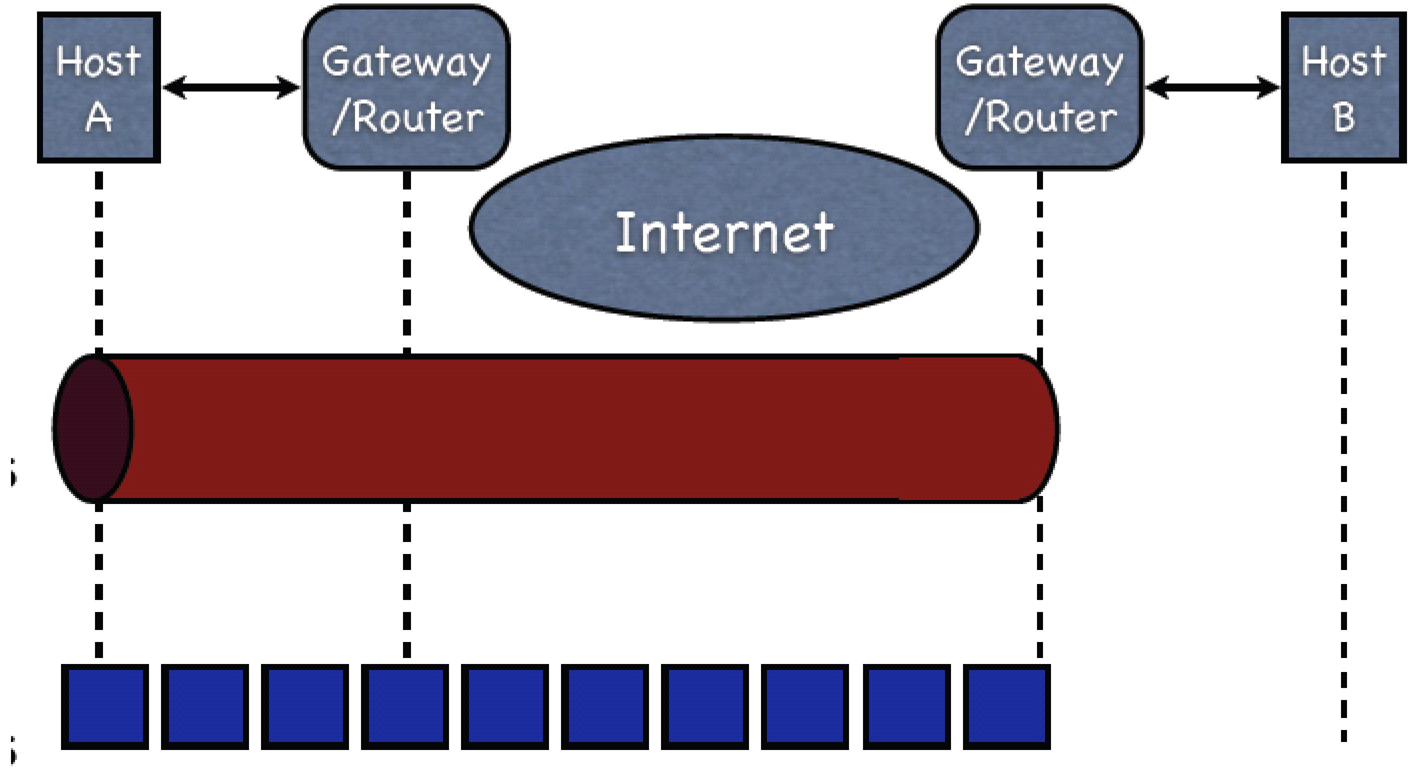


Transport Mode between Hosts



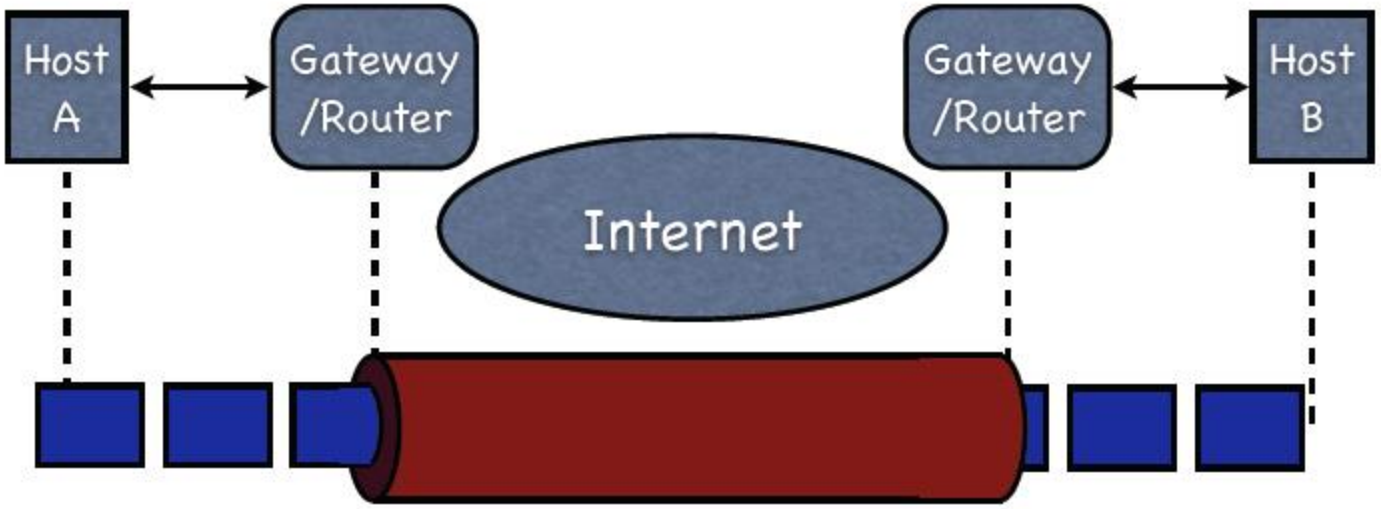
Tunnel Mode between Gateways
e.g. all traffic between two sites of a company

More Example Use Cases for the Modes



Tunnel or transport mode to access gateway
e.g. if mobile user accesses company
network

Combining Both Modes

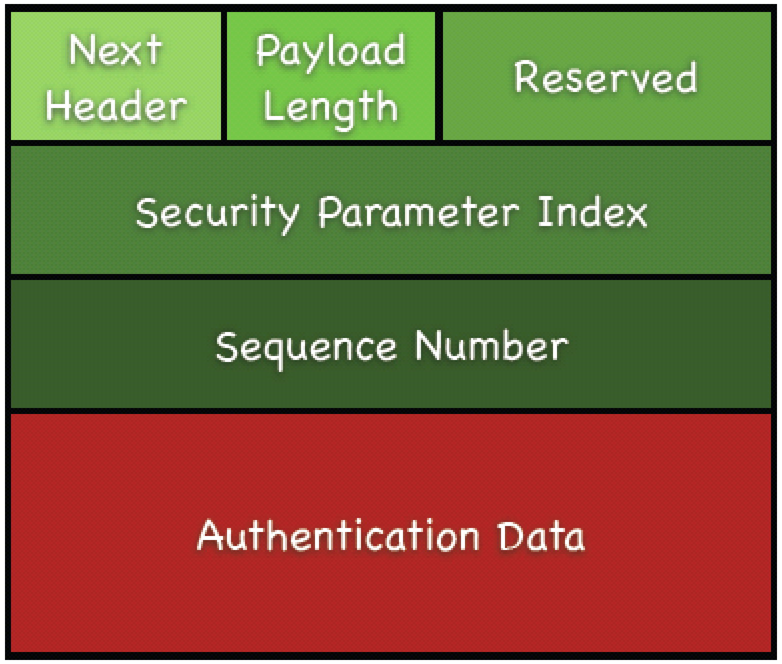


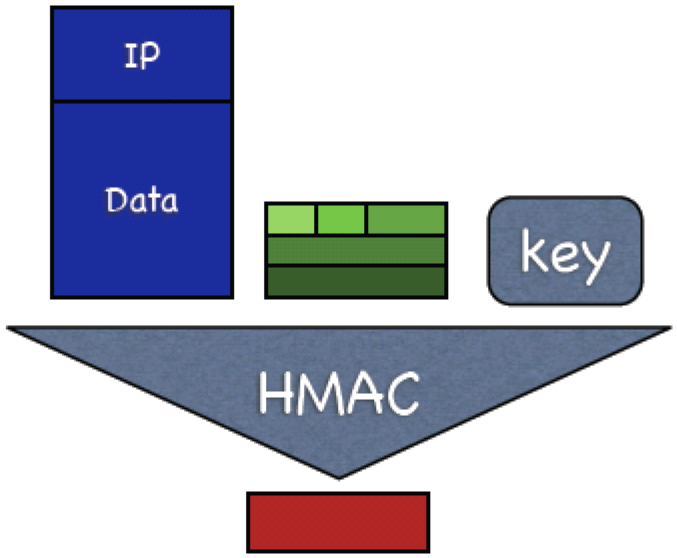
ESP in Tunnel Mode between Gateways hides IP addresses of communicating end hosts
Additional use of ESP in Transport mode between the hosts provides end-to-end confidentiality and authenticity

The Authentication Header Protocol

- Designed to
  - Authenticate packets from the source host to the destination host
  - Protect the integrity of the payload of the IP packet including the header
  - Protect against replay
  - Usable in both modes transport mode and tunnel mode
- Uses a keyed hash function to create a MAC
- Includes the MAC in the Authentication Header (AH)
- Inserts the AH
  - Between IP Header and Payload (transport mode)
  - Between new IP Header and original IP Header (tunnel mode)

Authentication Header and MAC Computation



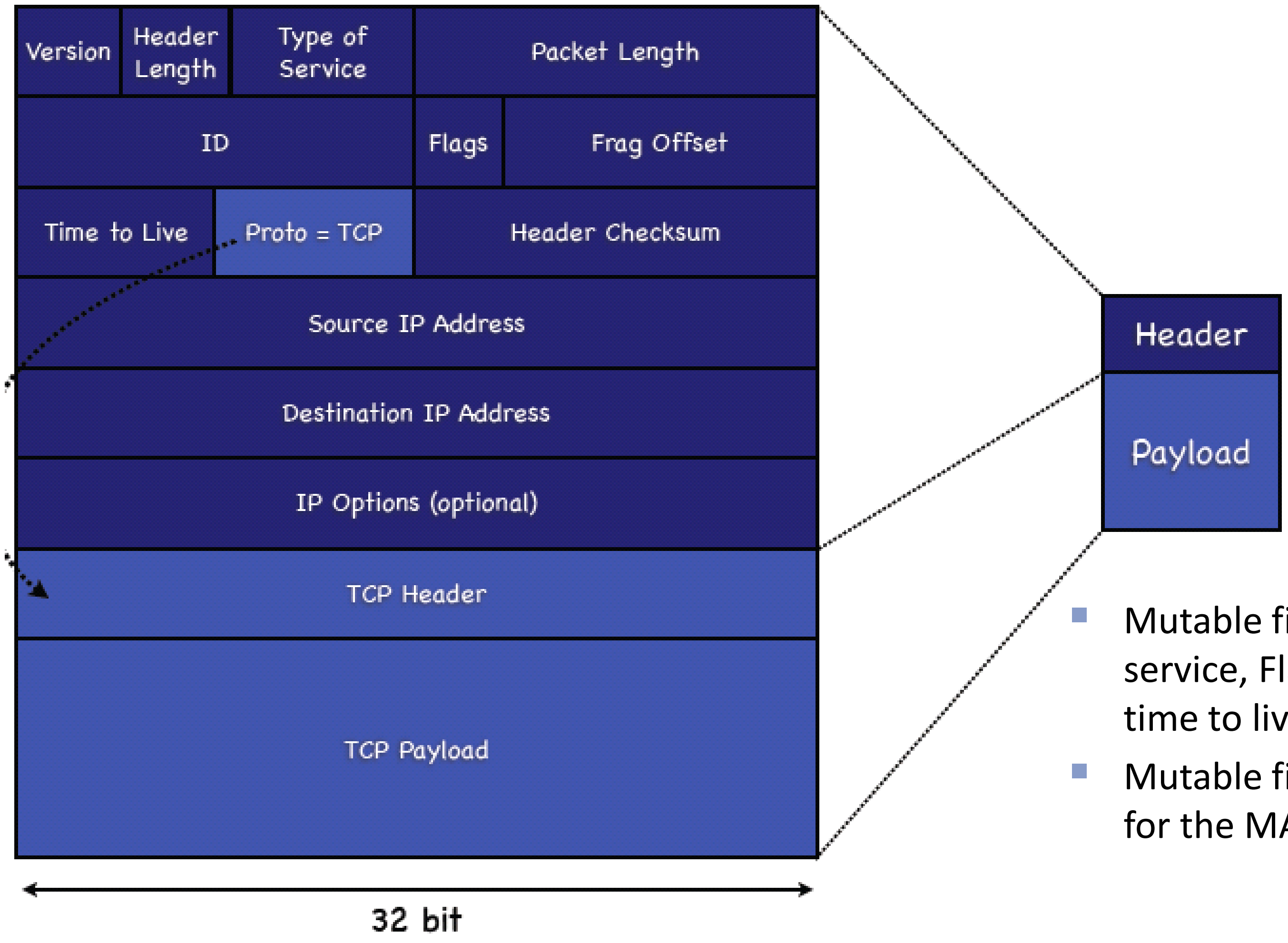


Fields in AH

- Next header field
  - 8-bit field that indicates type of payload carried by the IP packet
- Payload length
  - 8-bit field that defines the length of the authentication header
- Security parameter index (SPI)
  - 32-bit field that identifies a security association
  - Basically defines what security algorithms / keys were used to calculate the MAC
- Sequence number
  - 32-bit field incremented with each packet, used for replay protection
- Authentication data
  - MAC computed over the complete packet including (most of) the outer IP header

Will be explained later

Why only Most of the IP Header Fields?



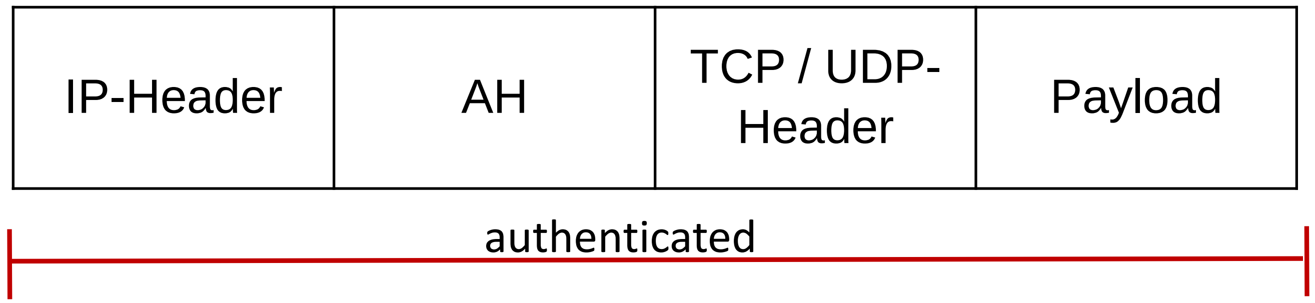
- Mutable fields are: type of service, Flags, Fragment Offset, time to live, Header Checksum
- Mutable fields are set to zero for the MAC calculation in AH

MACs defined for AH

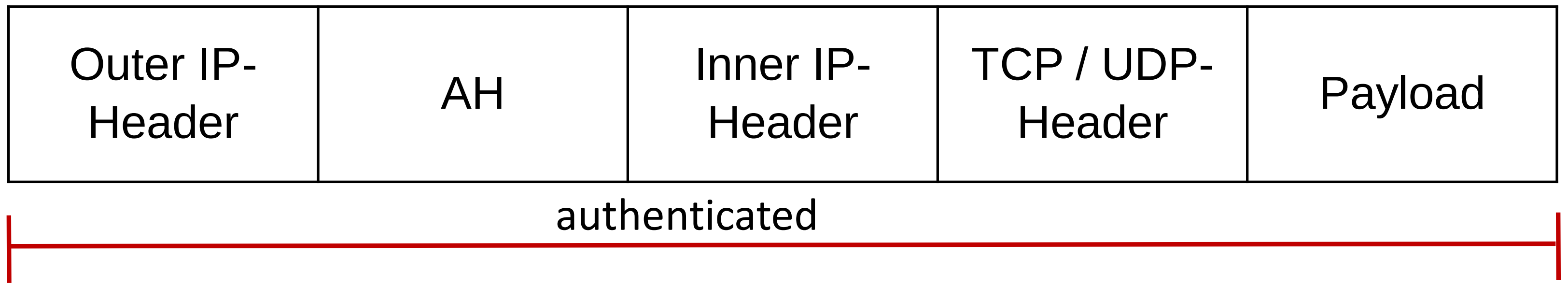
- AH implementations that comply to the latest version of the AH protocol (RFC 7321)
  - SHOULD implement AES-XCBC-MAC-96, AES-GMAC
  - MUST HMAC-SHA1-96
- In earlier versions HMAC-MD5-96 and HMAC-SHA1-96 were mandatory (RFC 2407)
- Note: XCBC is a predecessor of CMAC and differs from CMAC in generation of the masking keys

Placement of AH in the IP packet

- AH in transport mode

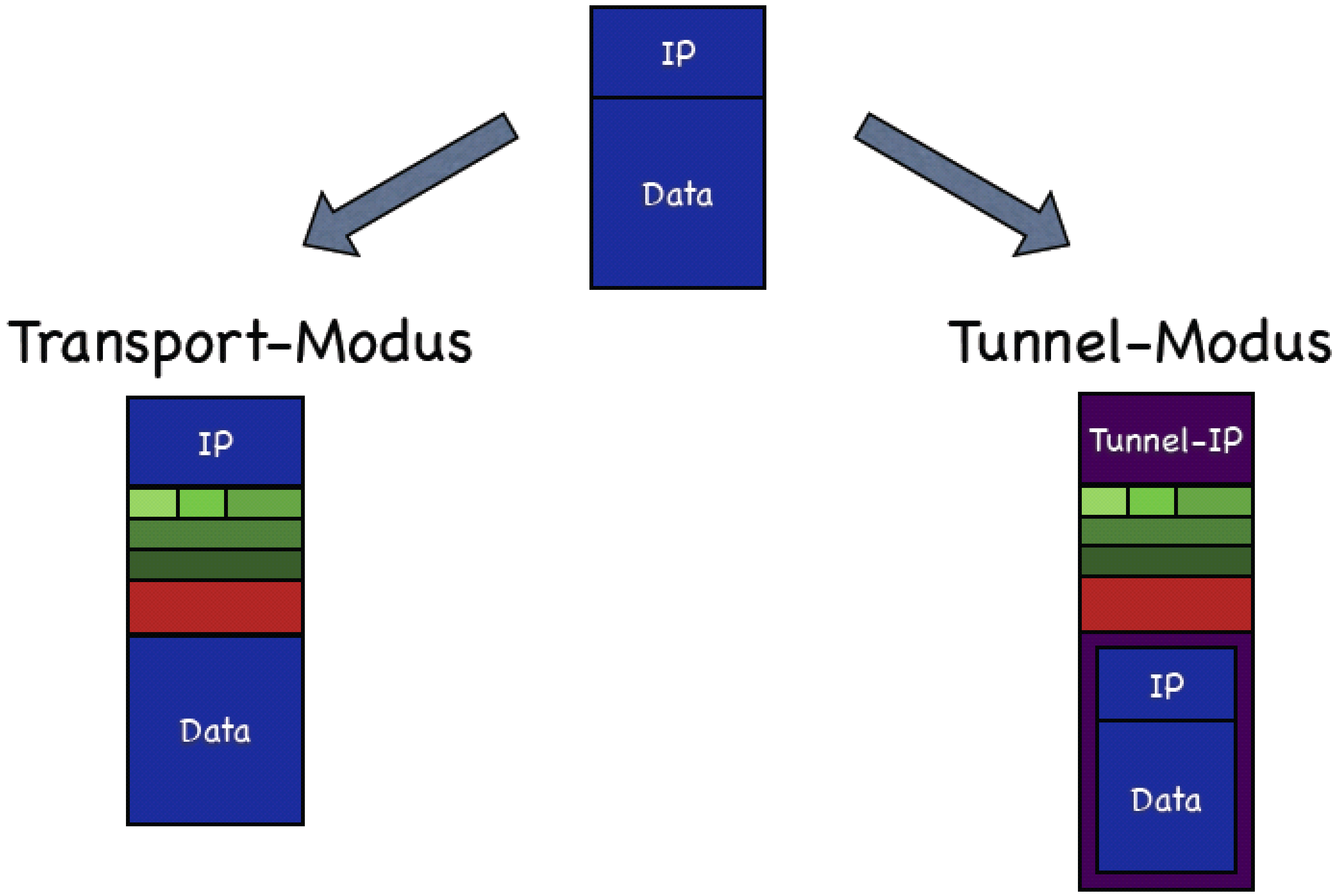


- AH in tunnel mode



- Authentication header fields except for the MAC itself are also authenticated

Using AH in Tunnel and Transport Mode



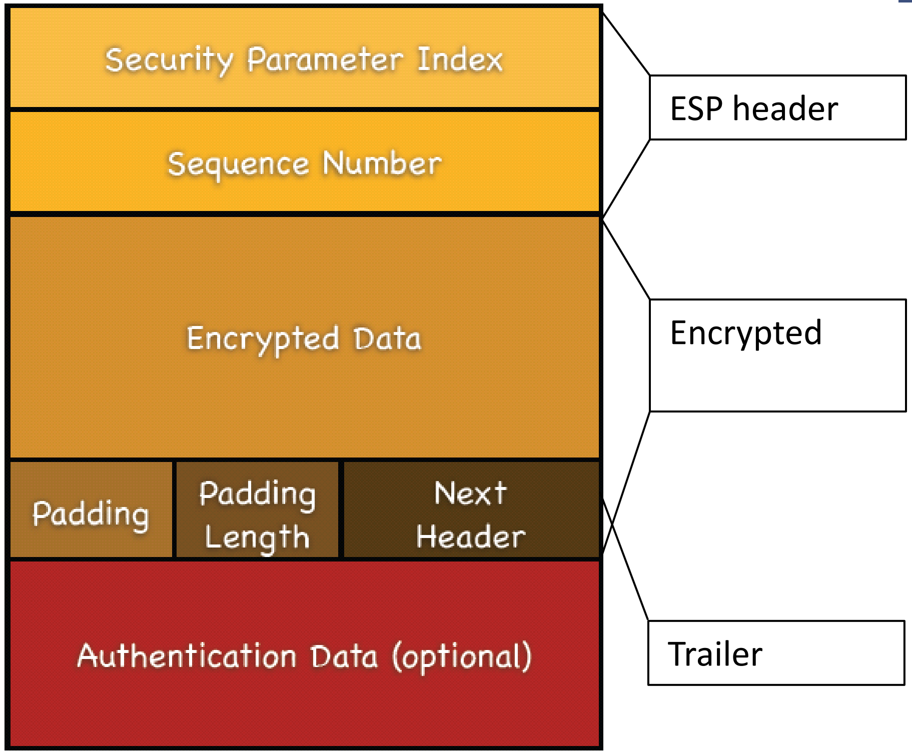
Additional Explanations

- Next header field (IPv6) or protocol field (IPv4) of the outer IP header is set to 51
- In transport mode the original next header field value is included in the AH
- In tunnel mode the next header field in AH is set to IP and the inner IP header carries the original next header field
- The MAC is computed over the complete packet including (most of) the outer IP header
  - Fields that change during transmission are not protected

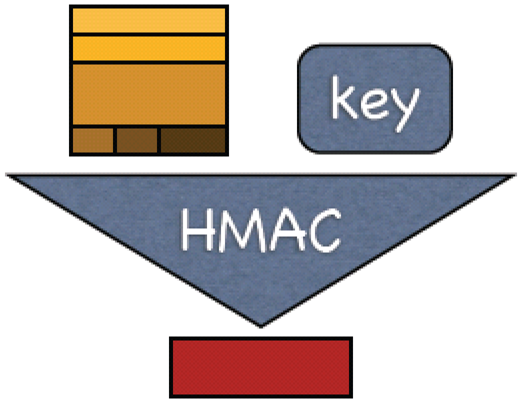
The ESP Protocol

- Provides packet-level source authentication, integrity, replay protection, and confidentiality
- Adds a header, a trailer and a MAC to a packet
- Encrypts the payload of the packet
- In tunnel mode, ESP also encrypts the inner IP header
  - Provides privacy of, e.g., the destination IP address in case all traffic to a destination is tunneled through an IPsec gateway

Encapsulated Security Payload



- Authentication Data (MAC) computation



- ESP header, payload, and trailer
  - The Payload is variable
  - The Padding depends on the payload length (0 - 255 bit)

Fields in the ESP Header and Trailer

- ESP Header
  - Security Parameter Index
    - 32 bit long, same meaning as in AH
  - Sequence Number
    - 32 bit long, same meaning as in AH
- ESP Trailer
  - Padding
    - 0 – 255 padding bits
  - Padding length
    - 8 bit pad-length filed
  - Next header
    - 8 bit field indicating the type of payload
- MAC
  - Is not calculated over the outer IP header

Algorithms Defined for ESP

- With respect to encryption, implementations that comply to the latest version of ESP (RFC 7321)
  - MUST support NULL encryption, AES-CBC-128, AES-CTR
  - MAY support 3DES-CBC
  - MUST NOT support DES-CBC
- With respect to message authentication codes, implementations of ESP
  - MUST support HMAC-SHA1-96
  - SHOULD support AES-XCBC-MAC-96, AES-GMAC
  - MAY support NULL integrity

NULL Encryption

- Specified in RFC 2410: The NULL Encryption Algorithm and Its Use With IPsec http://tools.ietf.org/html/rfc2410

„NULL does nothing to alter plaintext data. In fact, NULL by itself does nothing.

NULL is a block cipher the origins of which appear to be lost in antiquity. Despite rumors that the NSA suppressed publication of this algorithm, there is no evidence of such action on their part. Rather, recent archaeological evidence suggests that the NULL algorithm was developed in Roman times, as an exportable alternative to Caesar ciphers. However, because Roman numerals lack a symbol of zero, written records of the algorithm‘s development were lost to historians for over two millennia.

2.1. Keying Material

Like other modern ciphers, e.g., RC5, the NULL encryption algorithm can make use of keys of varying lengths. However, no measurable increase in security is afforded by the use of longer key lengths.“

NULL Encryption Cont’d

2.4. Performance

The NULL encryption algorithm is significantly faster than other commonly used symmetric encryption algorithms and implementations of the base algorithm are available for all commonly used hardware and OS platforms.

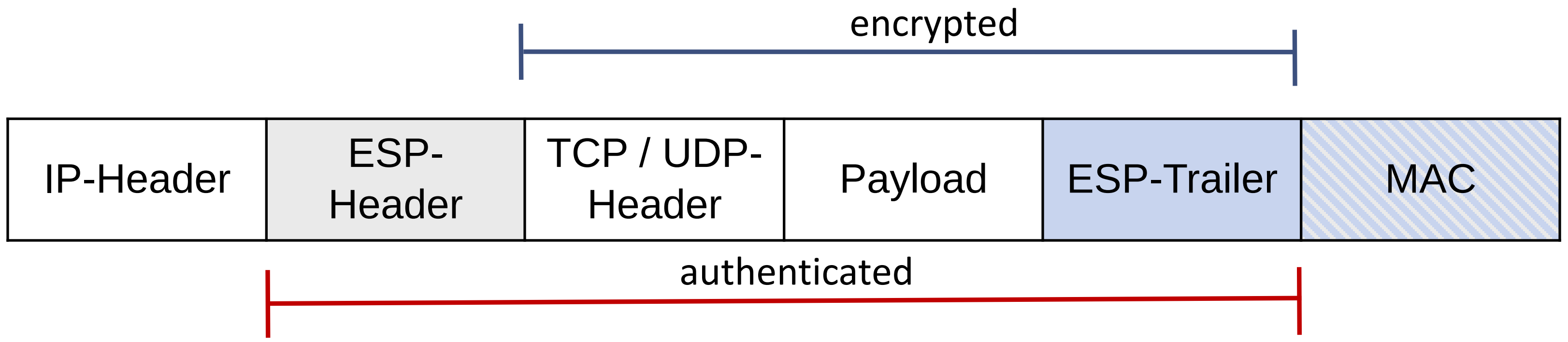
2.5 Test Vectors

The following is a set of test vectors to facilitate in the development of interoperable NULL implementations.

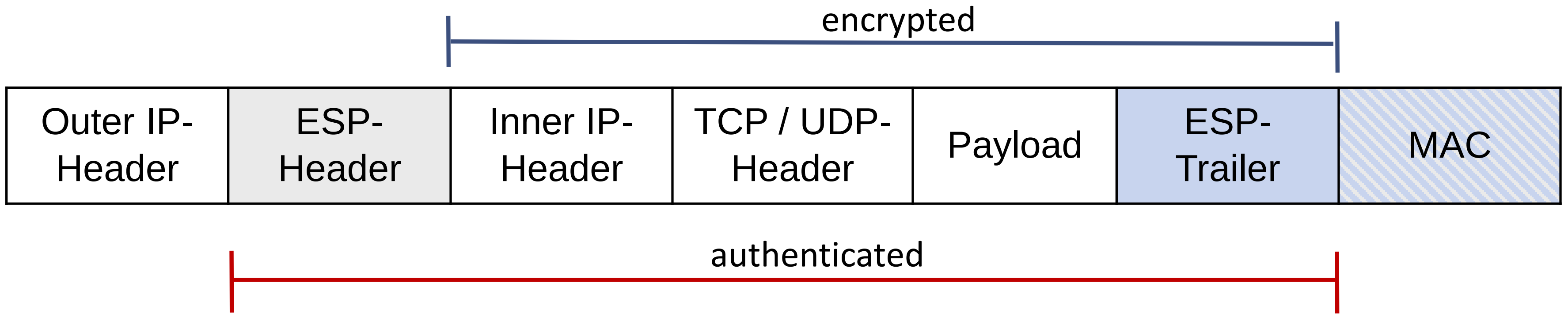
```
test_case = 1 data = 0x123456789abcdef
data_len = 8
NULL_data = 0x123456789abcdef

test_case = 2
data = "Network Security People Have A Strange Sense Of Humor"
data_len = 53
NULL_data = "Network Security People Have A Strange Sense Of  Humor"
```

ESP in Transport and Tunnel Mode

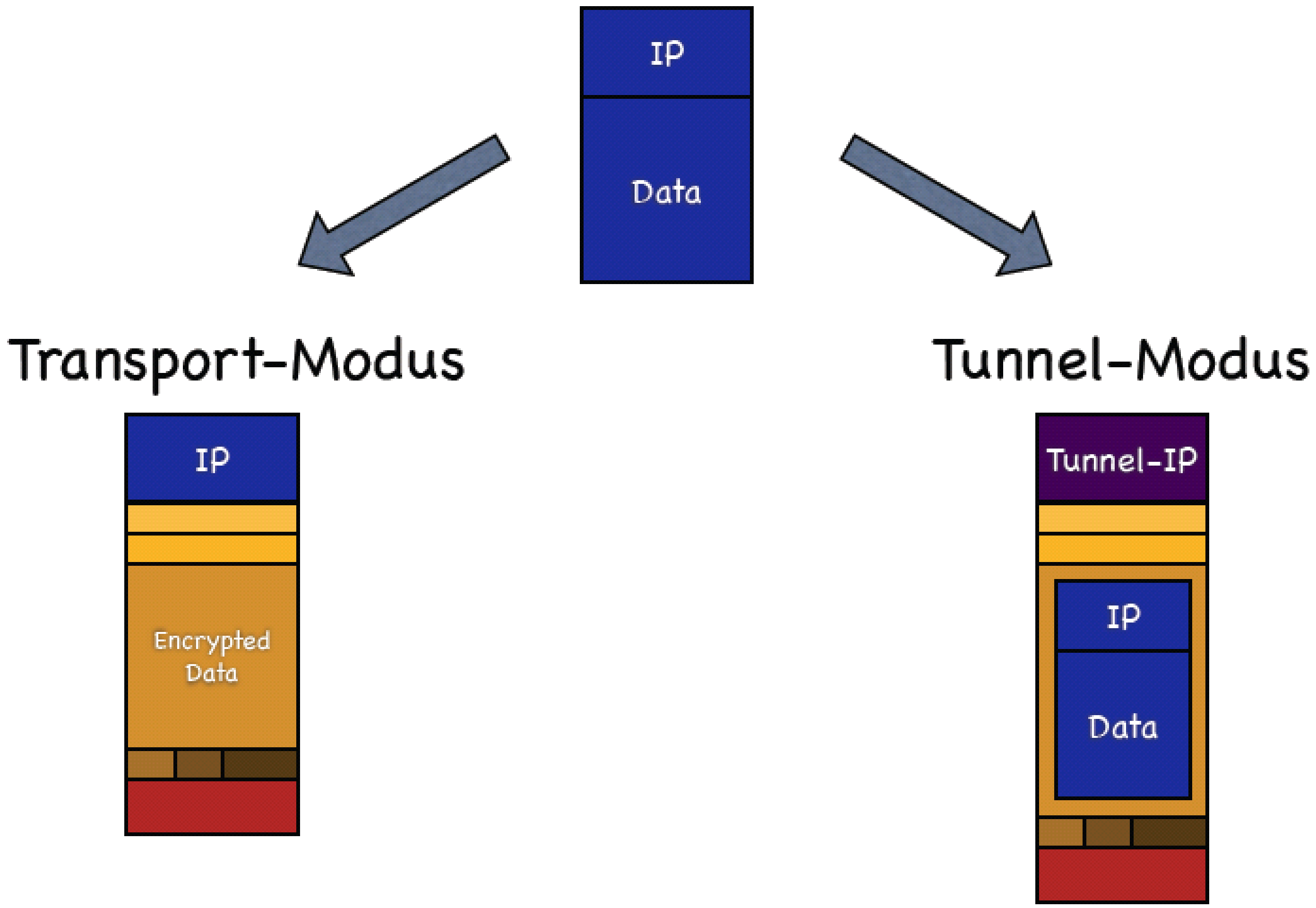
- ESP in Transport Mode



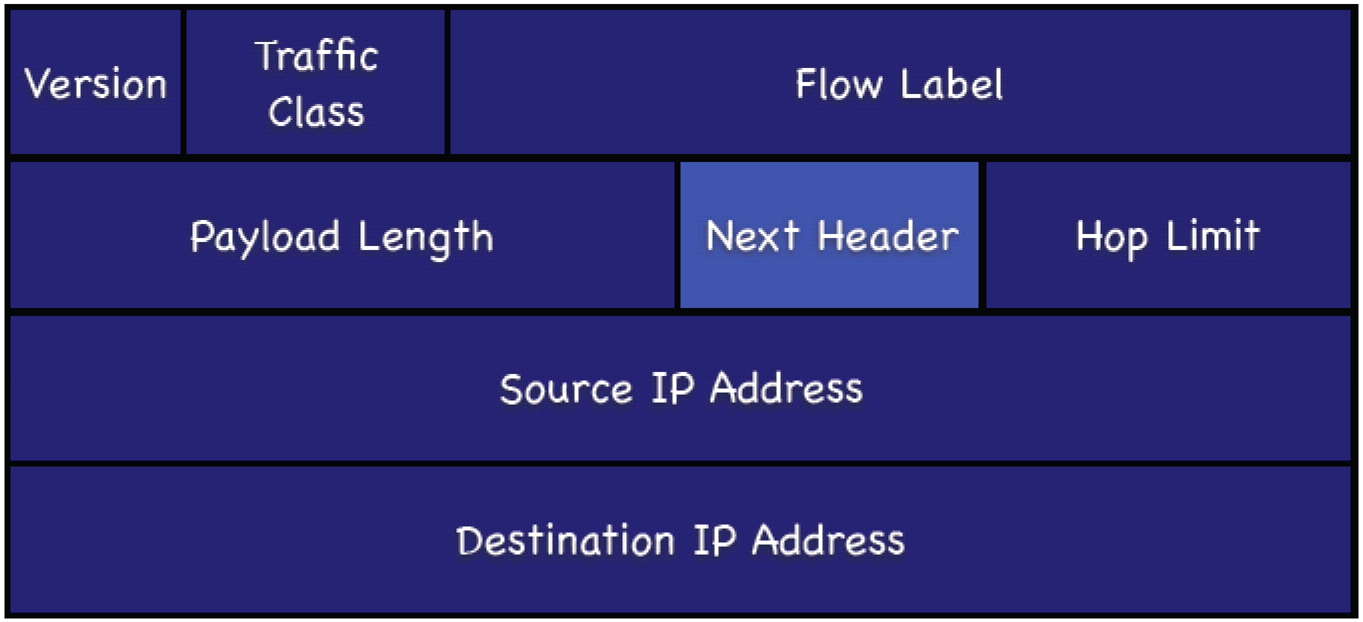
- ESP in Tunnel Mode



ESP Modes

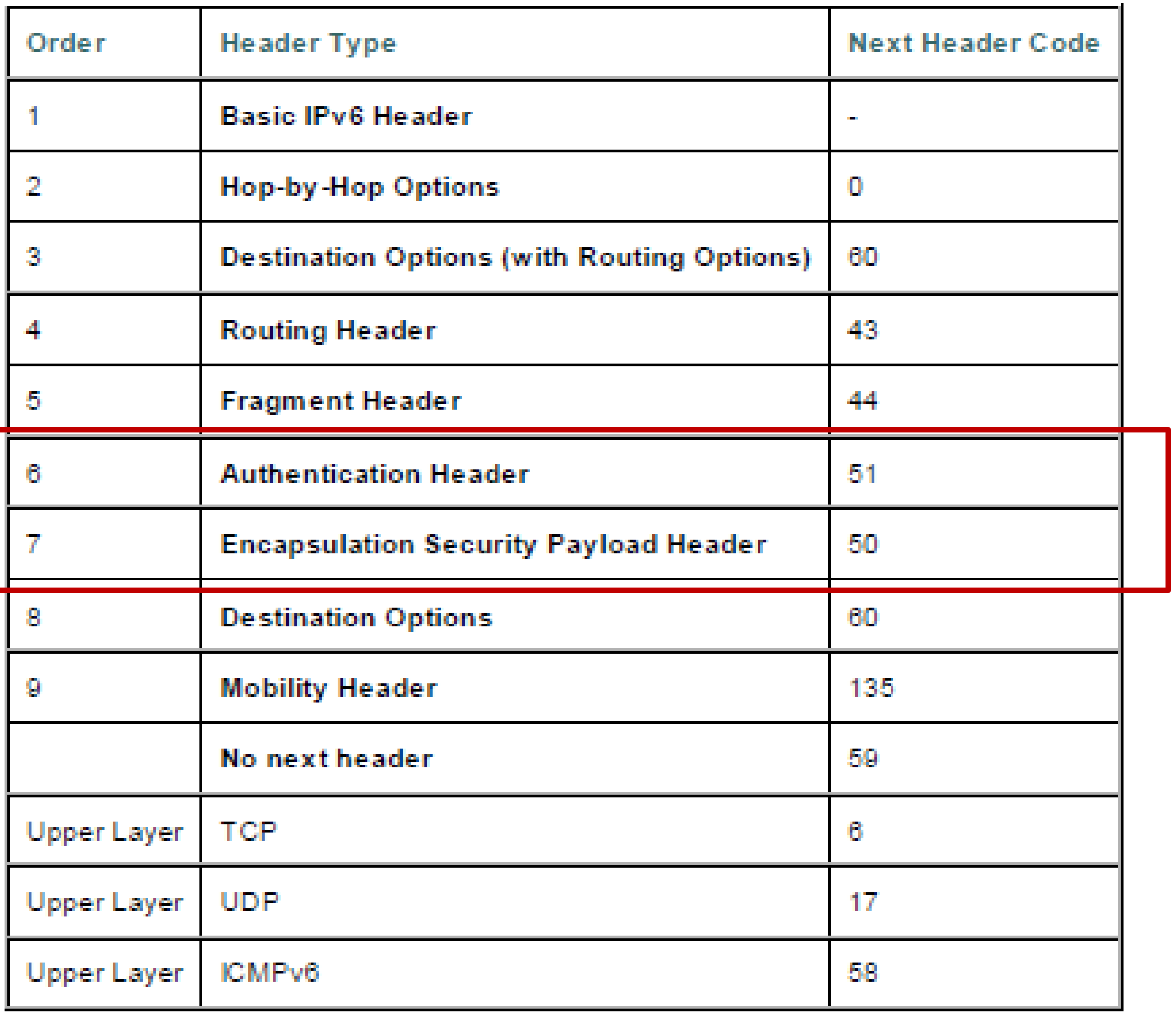

IP
Data
Transport-Modus
IP
Encrypted Data
Tunnel-Modus
Tunnel-IP
IP
Data

Basic IPv6 Header



- Next header field indicates the header type immediately following the basic IPv6 header. This can be an extension header or a higher layer header.

Recommended Order of IPv6 extension headers



Order	Header Type	Next Header Code
1	Basic IPv6 Header	-
2	Hop-by-Hop Options	0
3	Destination Options (with Routing Options)	60
4	Routing Header	43
5	Fragment Header	44
6	Authentication Header	51
7	Encapsulation Security Payload Header	50
8	Destination Options	60
9	Mobility Header	135
	No next header	59
Upper Layer	TCP	6
Upper Layer	UDP	17
Upper Layer	ICMPv6	58

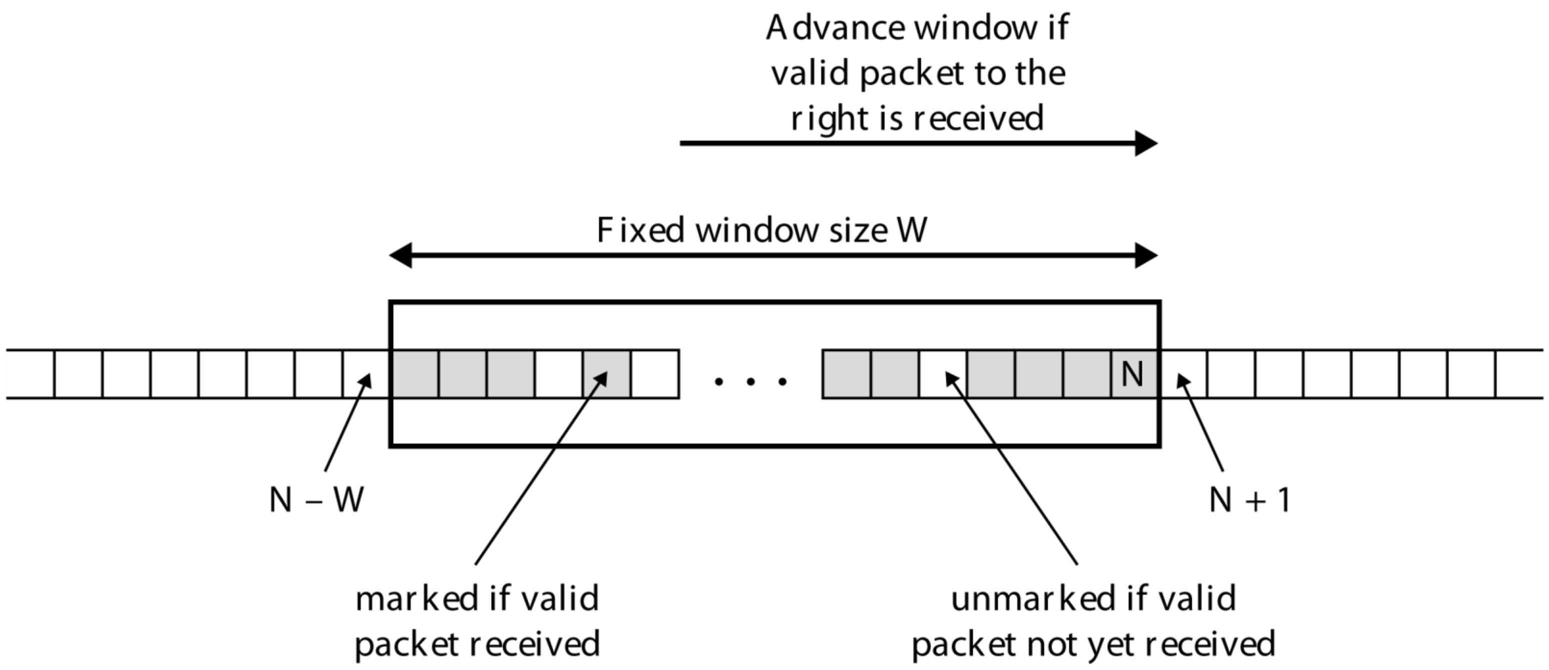
IPv4 vs IPv6

- In IPv6 the header extensions following and including the ESP header are protected
- In IPv6 the last header extension before the ESP header extension of the outer IP header is set to 50 (=ESP header)
- In IPv4 the protocol field of the outer IP header is set to 50

AH vs. ESP

- ESP provides payload confidentiality which AH does not
- AH provides integrity protection of the outer IP header which ESP does not
- ESP in tunnel mode, however, does provide integrity protection of the inner IP header
- Both provide
  - Payload integrity
  - Replay protection
  - Data source authentication
- AH often causes problems in contexts where IP headers naturally change
  - E.g. in case of NAT traversal

Replay protection in AH and ESP


Advance window if valid packet to the right is received
Fixed window size W
N
N − W
N + 1
marked if valid packet received
unmarked if valid packet not yet received

- Provided with the help of the sequence number and a window of acceptable sequence numbers of size W
- The window slides to the right whenever a packet with a sequence number higher than the rightmost one in the window is received

Replay Protection cont'd

- If a packet arrives, the receiver checks the sequence number in the AH or ESP header
  - If the sequence number is in the current window and the number is not yet marked the number is marked and the packet further processed. If the number is already marked the packet is dropped
  - If the sequence number is lower then the left border of the window the packet is dropped and an auditable event occurs
  - If the sequence number is higher then the right border, the number is marked and the window slides to the right such that the received sequence number is the rightmost number in the window

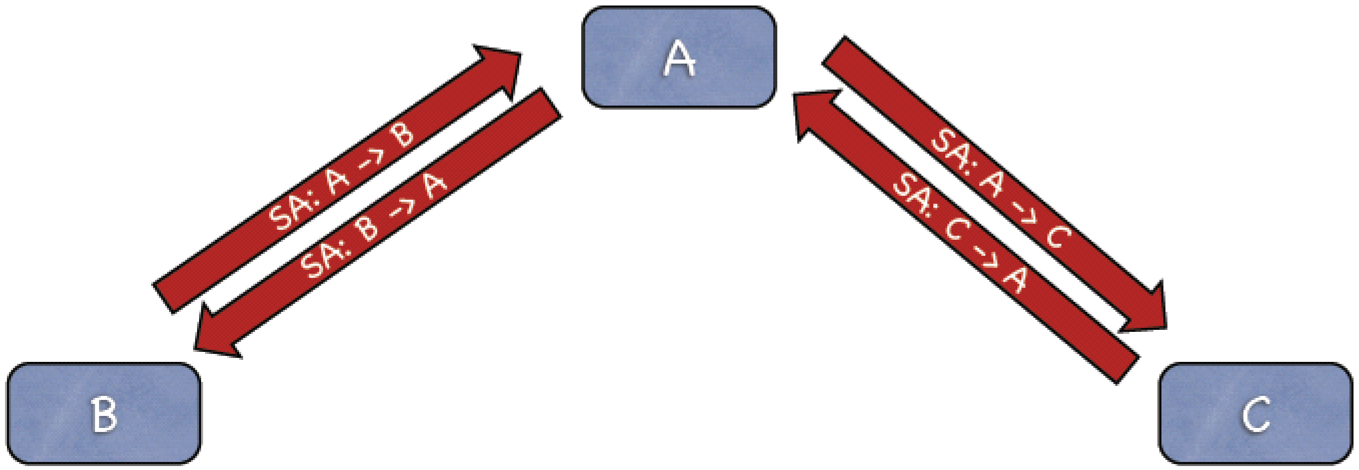
Window Size

- IPsec specifies that
  - The window size should be 32 or larger
  - The window size should be set to 64 as default
- A Receiver may, however, choose any window size larger than 32

Security Associations (SAs)

- One-way relationship between a sender and a receiver of IPsec protected traffic
  - Two SAs are required for two-way communication
- Each SA is uniquely identified by
  - Security Parameter Index (SPI) – Bit string assigned to the SA, carried in the corresponding AH or ESP header to enable receiver to identify SA used to protect the packet
  - Security Protocol Identifier – Indicates if the SA is meant for AH or for ESP protection
- Each SA contains a list of parameters (see below)
- Each host may store several SAs for different destinations, different traffic types, different directions
- SAs are stored in an SA – database (SAD)

One SA for Each Direction



SA Parameters

- Sequence Number Counter – A 32 bit value used to generate the Sequence Number Field in AH or ESP
- Sequence Counter Overflow – A flag indicating if a sequence number overflow should generate an auditable event
- Anti-Replay Window – Defines start point and size of the anti-replay window for this SA
- AH Information – MAC algorithm, key, key lifetime
- ESP Information – Encryption and MAC algorithms, keys, initialization vector, key lifetimes

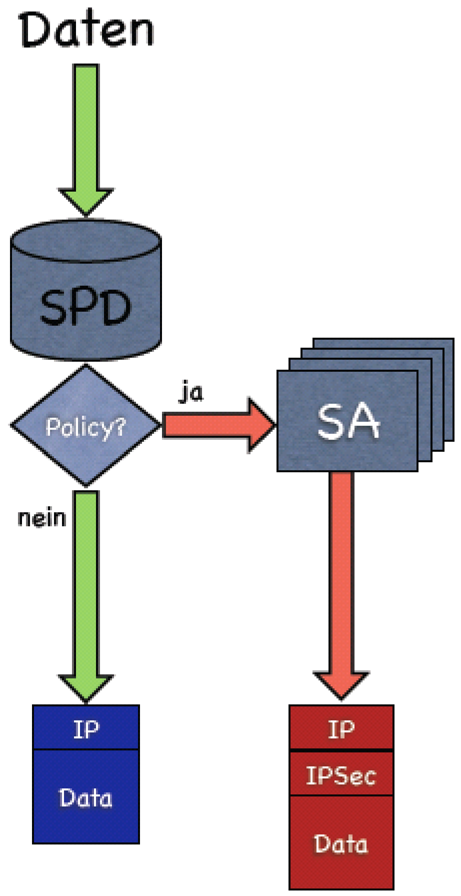
SA Parameters Cont'd

- Lifetime of this SA – A time interval or byte count after which an SA must be replaced with a new SA or terminated, plus an indication of which of these options is used
- IPsec Protocol Mode – Tunnel mode, transport mode or wildcard (i.e. any of the two)
- Path MTU – Any observed maximum transmission unit (max size of packet that can be transmitted without fragmentation)

SA Selectors

- IPsec provides user with flexibility with respect to which IPsec services are applied to which IP-traffic
- Different types of IP-traffic are related to specific SAs by means of a Security Policy Database (SPD)
- Each entry (= selector) in the SPD is defined by
  - Source IP address (one, list, range, or wildcard)
  - Destination IP address (one, list, range, or wildcard)
  - Transport layer protocol number (one, list, range)
  - Source and Destination ports (one, list, or wildcard)
- The selectors determine how inbound and outbound packets are processed

Inbound and Outbound Processing

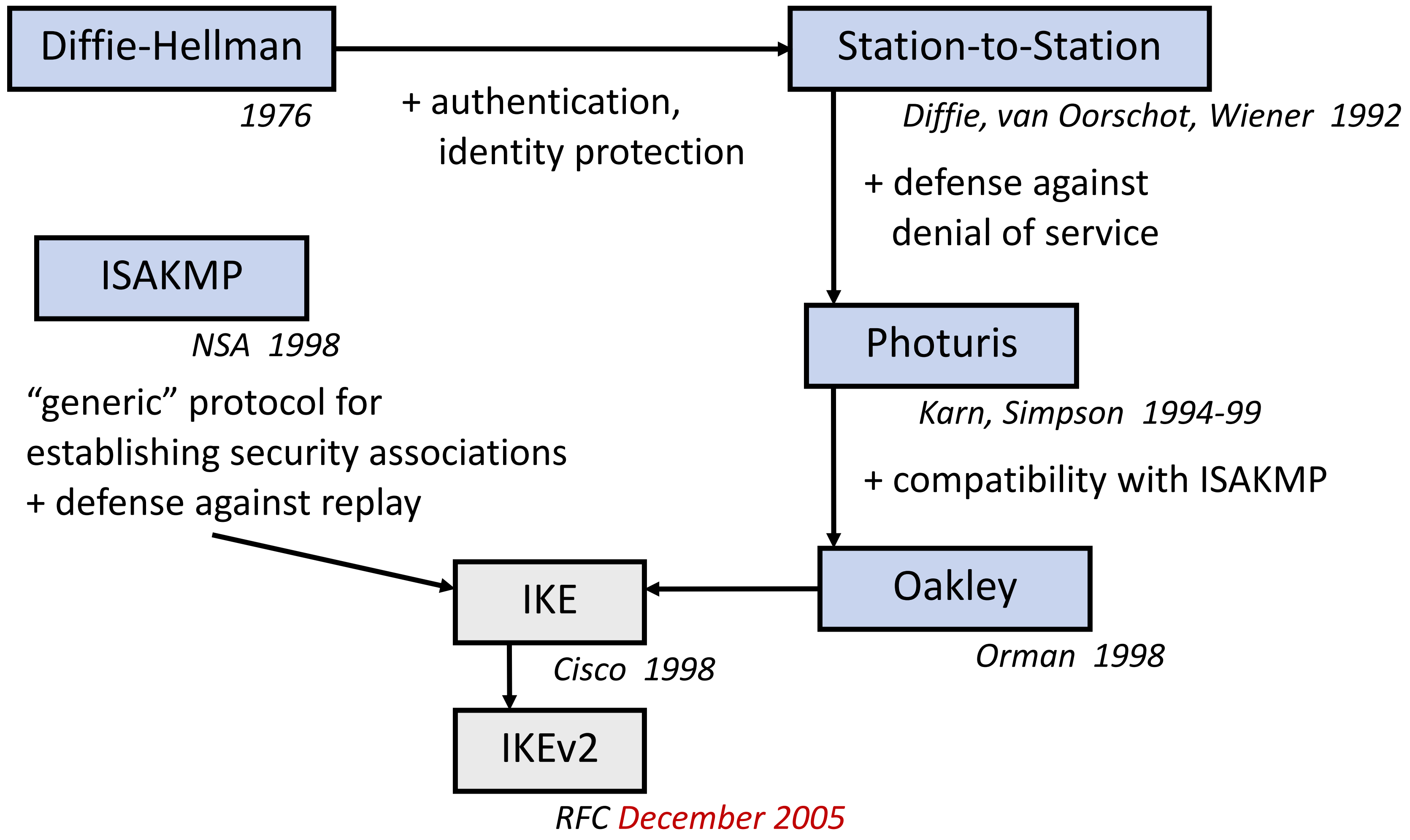


- The Security Policy Data base indicates if and how certain traffic types are to be protected
  - SPD consists of traffic selectors
  - A traffic selector points to an SA
- Inbound and outbound traffic is checked against the SPD
  - If no selector applies to the IP packet, it bypasses the rest of IPSec
  - If a selector applies, the security association is retrieved and the AH/ESP is applied to the packet
  - If no SA exists yet, the endpoints negotiate an SA with the help of the Internet Key Exchange Protocol

Key Management for IPsec

- IPsec supports manual and automated SA establishment / negotiation between peers
- The default automated SA establishment is the Internet Key Exchange (IKE)
- IKE (RFC 2409) describes a protocol for authenticated key exchange to establish SAs
  - Defines actual content of ISAKMP payloads
- ISAKMP (RFC 2408) provides a framework for authentication and key exchange
  - Message formats for authenticated key exchange but not the actual content
- IKE v2 (RFC 7427) describes both message formats and payloads for automated authenticated key exchange to establish SAs for the latest IPsec version. It is the default key management for the latest IPsec version

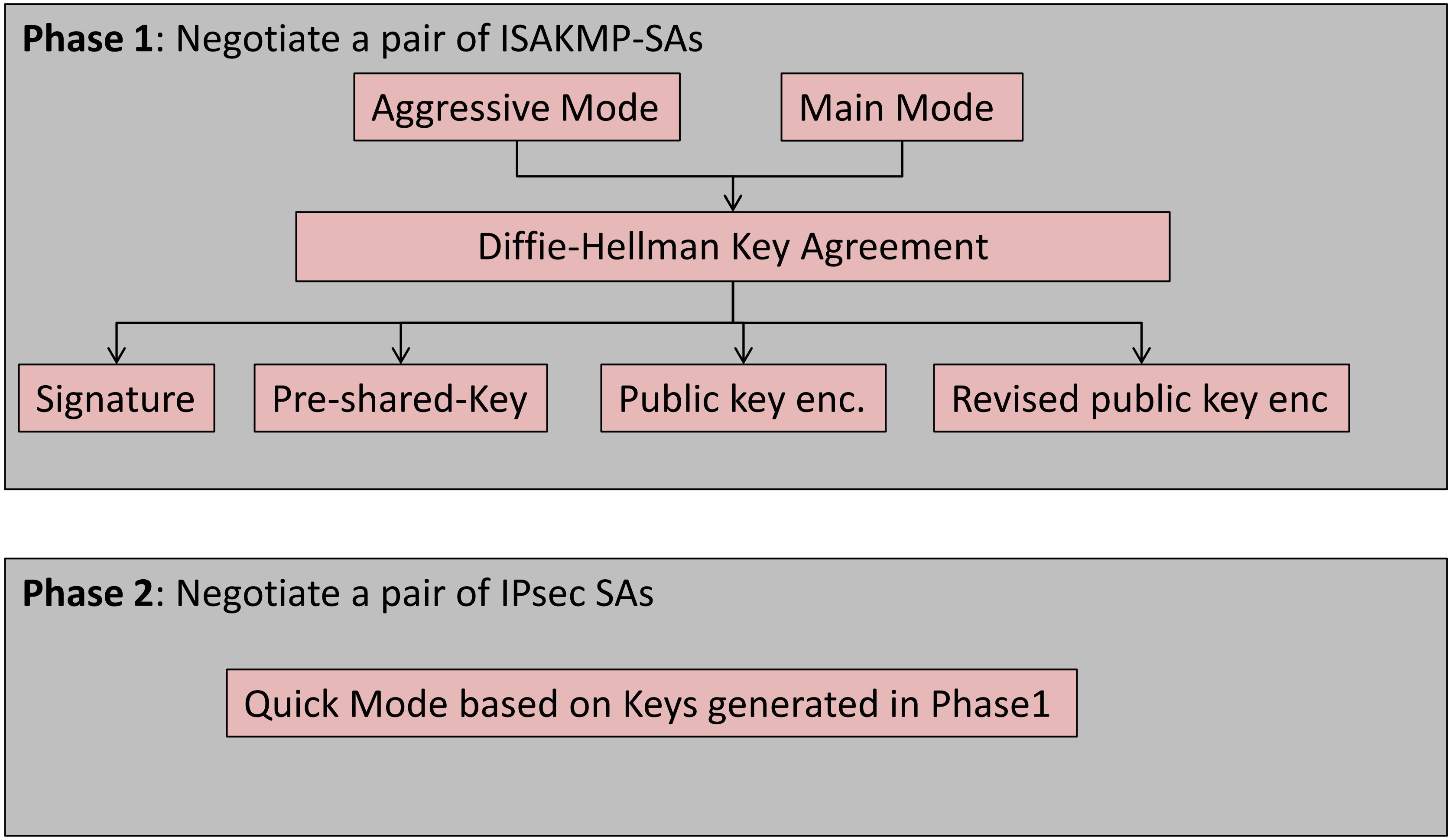
IKE Genealogy


Diffie-Hellman
1976
+ authentication, identity protection
Station-to-Station
Diffie, van Oorschot, Wiener 1992
+ defense against denial of service
ISAKMP
NSA 1998
"generic" protocol for establishing security associations + defense against replay
Photuris
Karn, Simpson 1994-99
+ compatibility with ISAKMP
IKE
Cisco 1998
Oakley
Orman 1998
IKEv2
RFC December 2005

ISAKMP

- Stands for "Internet Security Association and Key Management Protocol", RFC 2408
- Can be used with different key establishment protocols, e.g. IKEv1
- Defines message formats for authenticated key exchange
- Supports the negotiation of SAs between peers
- Defines two phases
  - Phase 1 allows for the negotiation of a pair of SAs to protect phase 2 of ISAKMP. An SA established in phase one is called ISAKMP SA
  - Phase 2 allows for the protected negotiation of further SAs for use in other protocols (ESP, AH)

The IKE/ISAKMP Phases (IKEv1)


Phase 1: Negotiate a pair of ISAKMP-SAs
Aggressive Mode
Main Mode
Diffie-Hellman Key Agreement
Signature
Pre-shared-Key
Public key enc.
Revised public key enc
Phase 2: Negotiate a pair of IPsec SAs
Quick Mode based on Keys generated in Phase1

Why Two Phases?

- Same first phase can be used for several second phases
  - I.e. can be used to establish several SAs between peers
  - As the second phase is inexpensive, the hope was that this may reduce the overall overhead
- Different SAs between the same security gateways allow for traffic separation
  - E.g. protect traffic originating from different hosts with different SAs (i.e. different keys, different algorithms)
  - Use different SAs for different port pairs
- Key refreshment for any SAs can be accomplished efficiently by phase 2 alone

Phase 1 of IKE v1

- Phase 1 can be run in two different modes
  - Aggressive mode
    - provides authentication and session key establishment
    - in 3 messages, based on Diffie-Hellman
  - Main mode
    - provides authentication and session key establishment in
    - 6 messages, based on Diffie-Hellman
    - Additionally protects the endpoint identifiers
- In each mode one of 4 mechanisms can be used to authenticate the Diffie-Hellman key-exchange

Main Mode vs. Aggressive Mode

- Main Mode offers identity protection
- Aggressive mode is faster as it requires only 3 messages
- Main Mode is mandatory to implement, aggressive mode is not
- There is a problem with aggressive mode and DoS protection (see Slide 60)
    - The expensive operation needs to be done before the cookie is verified
    - DoS protection provided by cookies useless in aggressive mode

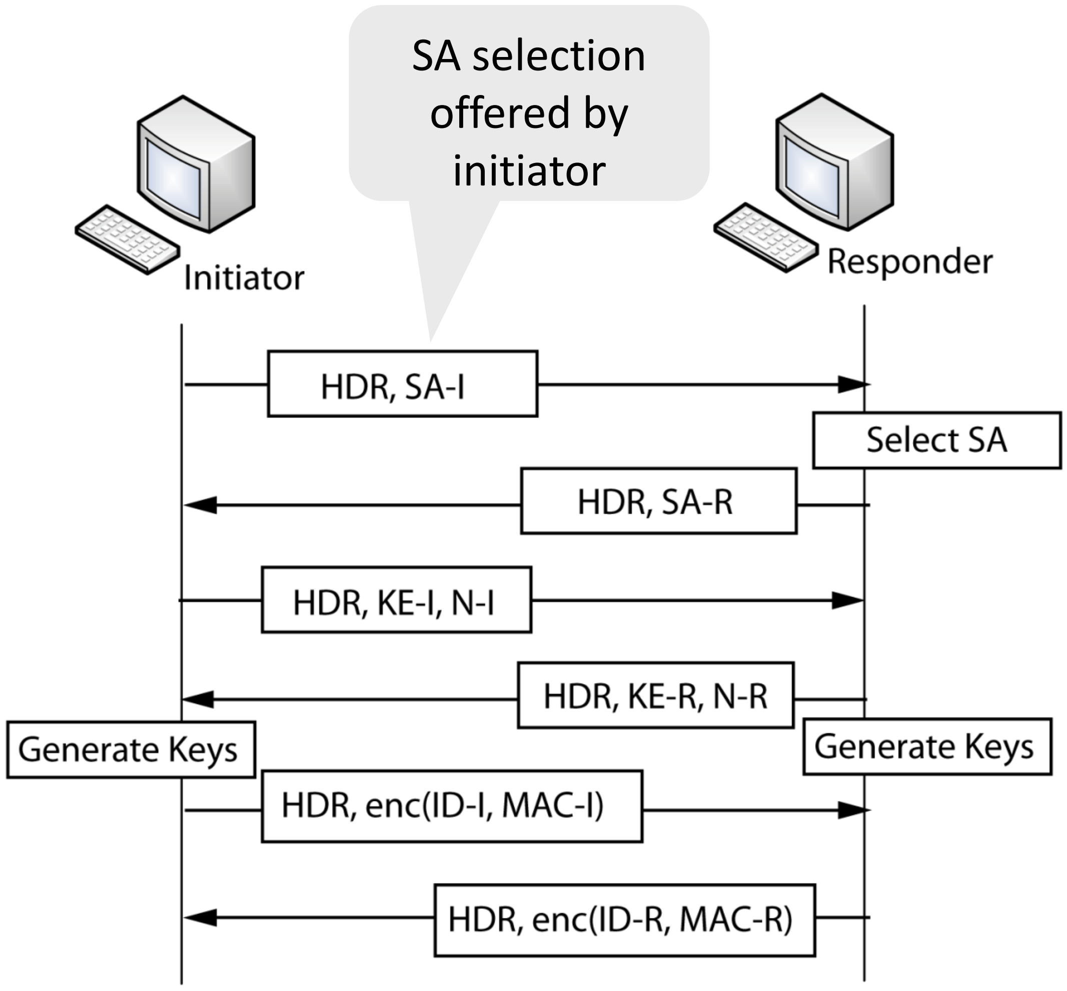
Authentication Mechanisms

- Based on pre-shared key
- Based on public encryption key
- Based on revised public encryption key
- Based on public signature verification key

➢ Eight different options for Phase 1 of IKE

- In the following we will discuss three examples
  - Pre-shared key based, main mode
  - Public signature key based, main mode
  - Public signature key based, aggressive mode

Pre-Shared Key, Main Mode



- Header (HDR) includes both cookies, except first message (includes only initiator cookie), SPI
- KE-I, KE-R: public DH values of I and R
- N-I, N-R: Nonces picked by I and R
- enc: encryption with SKEYe derived from pre-shared key and DH key
- MAC-I, MAC-R: see next slide
- ID-I, ID-R: additional identifiers that allow to determine policies w.r.t. traffic selectors

Calculation of Keys and MACs

Session key derived from pre-shared key
SKEY = PRF(PSK, N-I || N-R)

Public DH values: KE-I = g^i mod p, KE-R = g^r mod p
DH-key = g^{ir} mod p

Key derivation key used to derive further keys
SKEYd = PRF(SKEY, DH-key|| Cookie-I || Cookie-R || 0)

Authentication key used for MAC calculation
SKEYa = PRF(SKEY, SKEYd || DH-key || Cookie-I || Cookie-R || 1)

Encryption key
SKEYe = PRF(SKEY, SKEYa || DH-key || Cookie-I || Cookie-R || 2)

MAC-I = PRF(SKEYa, KE-I || Cookie-I || Cookie-R || SA-I || ID-I)
MAC-R = PRF(SKEYa, KE-R || Cookie-I || Cookie-R || SA-R || ID-R)

PRF is negotiable and can be based or XCBC or HMAC

Explanation

- The secret keys can only be calculated by the initiator and the responder as only they know the pre-shared key and the DH key
- Mutual Authentication:
  - MAC-I guarantees the responder that all previous messages originated from the initiator and were received unaltered
  - MAC-R guarantees the initiator that all previous messages originated from the responder and were received unaltered
- Encryption is used to protect the identities of initiator and responder
- The HDR includes cookies which are meant to provide DoS protection:

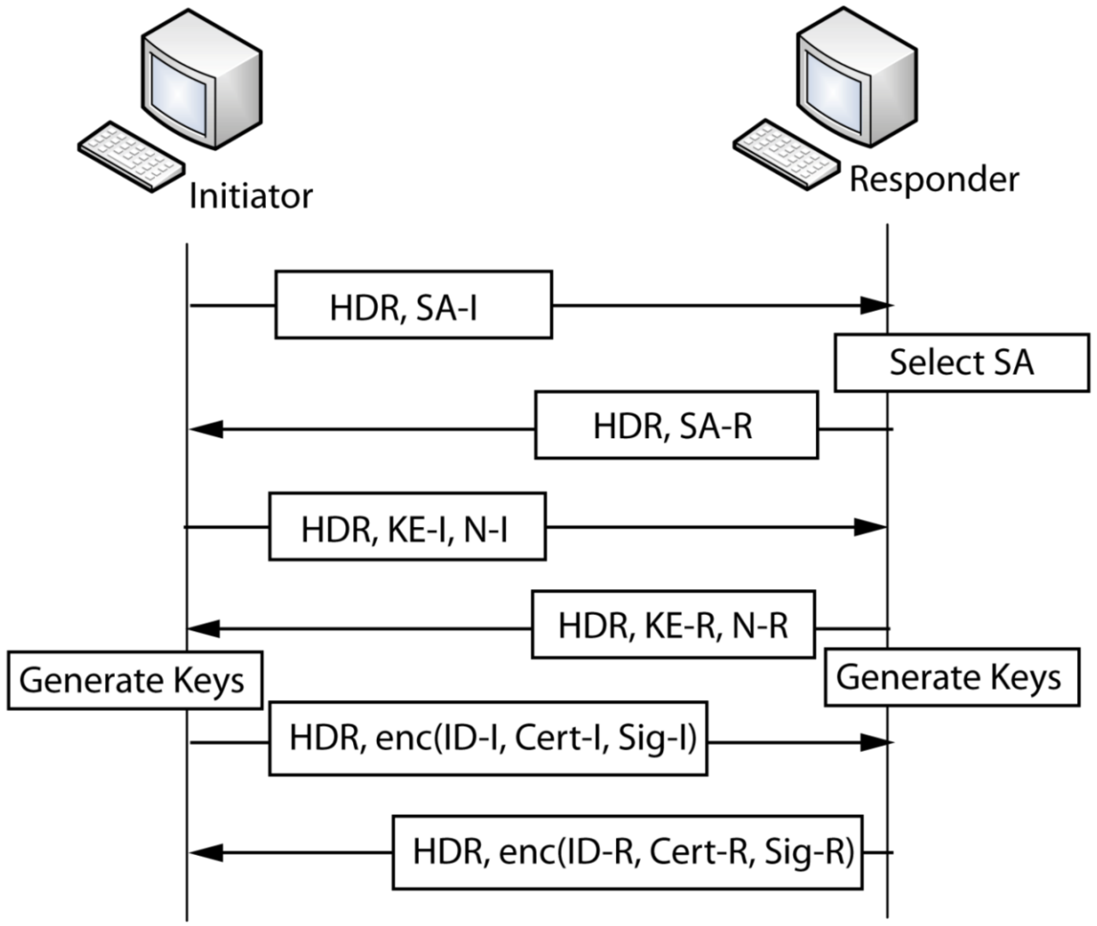
DoS Protection with Cookies

- Denial of service (DoS) due to resource clogging
  - If responder opens a state for each connection attempt, attacker can initiate thousands of connections from bogus or forged IP addresses
- Cookies are included in the HDR to ensure that the responder is stateless until initiator produced at least 2 messages
  - Responder's state (IP addresses and ports) is stored in an unforgeable cookie and sent to initiator
    - The exact cookie generation is left to the implementer
  - After initiator responds, cookie is regenerated and compared with the cookie returned by the initiator
  - The cost is 2 extra messages in each execution

Excursion: Typical Anti-DoS Cookie

- Typical protocol:
  - Client sends request (message #1) to server
  - Server sets up connection, responds with message #2
  - Client may complete session or not (potential DoS)
- Cookie version:
  - Client sends request to server
  - Server sends connection data hashed with a key known only to the server back to the client
    - Send message #2 later, after client confirms his address
  - Client confirms by returning hashed data
  - Need an extra step to send postponed message #2

Signatures, Main Mode



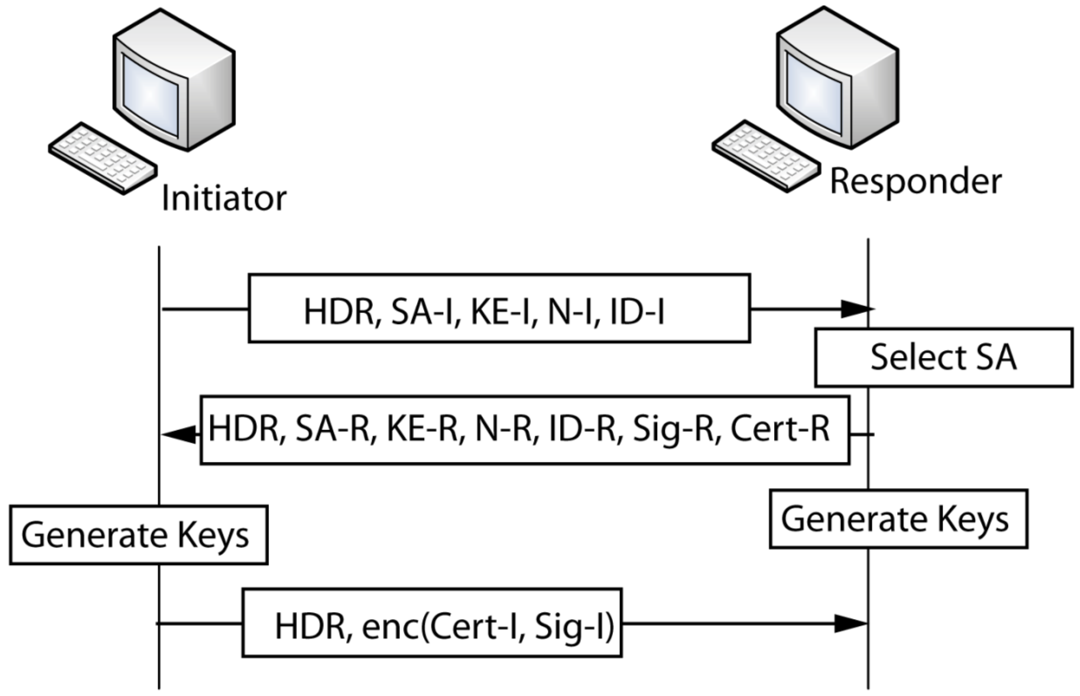
- Cert-I, Cert-R: certificates on the signature verification keys of I and R
- Sig-I: signature of I on hash of messages 1-4
- Sig-R: signature of R on hash of messages 1-5
- enc: encrypted with SKEYe

Explanation

- Keys are generated in the same way as in the pre-shared key case, except SKEY:

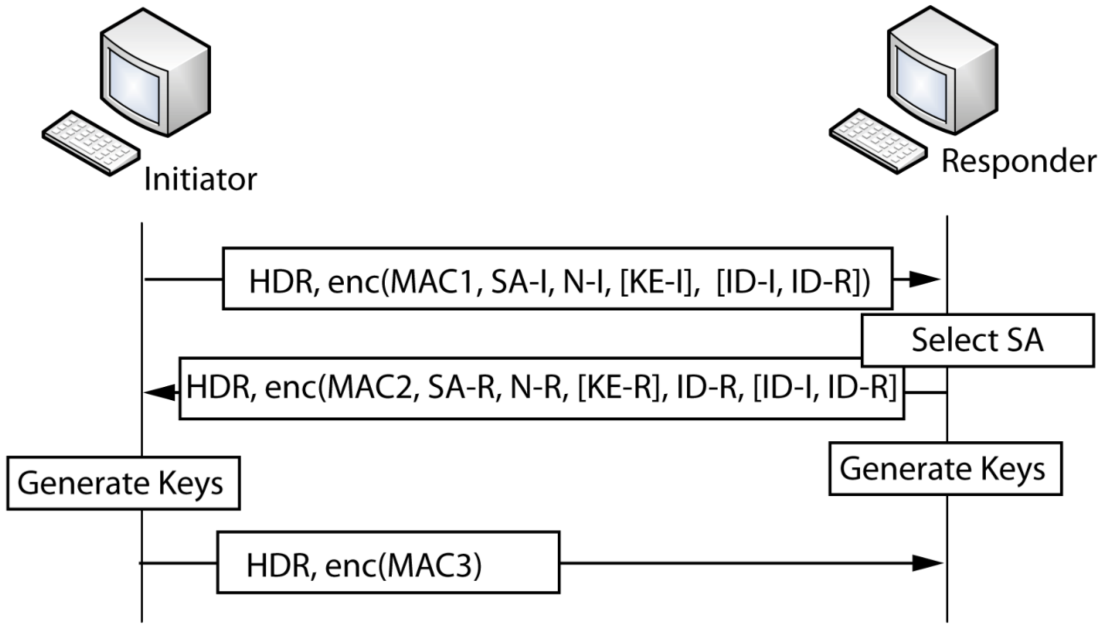
  SKEY = PRF(hash(N-I, N-R), Cookie-I, Cookie-R)
- Sig-I guarantees to responder that previous messages (in particular KE-I) originated from initiator and responder messages were correctly received
- Sig-R guarantees to initiator that previous messages (in particular KE-R) originated from responder and initiator messages were correctly received
- As KE-I and KE-R are verifiably authentic, the DH-key and thereby all derived keys shared with the right entities
- Again the encryption key SKEYe is used to hide the identities of responder and receiver and to provide key confirmation

Signatures, Aggressive Mode



- Sig-R computed on first message
- Sig-I computed on first and second message
- Note: signature has to be calculated before the responder's cookie can be checked -> DoS potential!

Phase 2: Quick Mode


Initiator
Responder
HDR, enc(MAC1, SA-I, N-I, [KE-I], [ID-I, ID-R])
Select SA
HDR, enc(MAC2, SA-R, N-R, [KE-R], ID-R, [ID-I, ID-R]
Generate Keys
Generate Keys
HDR, enc(MAC3)

- enc: encryption with SKEYe from phase 1
- MAC1 = PRF(SKEYa, MsgID, SA-I, N-I, [KE-I])
- MAC2 = PRF(SKEYa, MsgID, SA-R, N-R, [KE-R])
- MAC3 = PRF(SKEYa, MsgID, N-I, N-R)

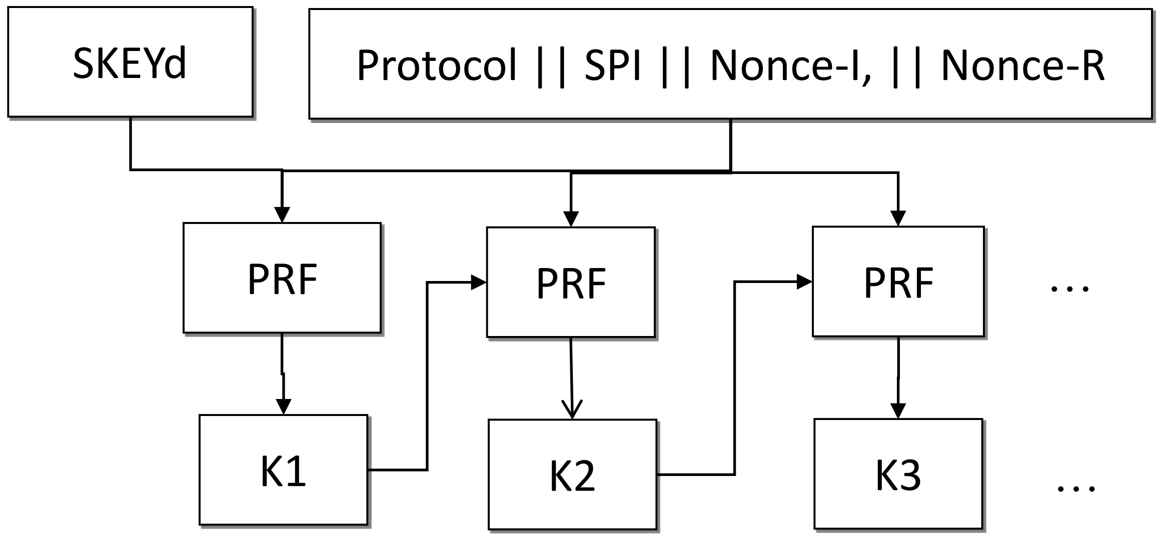
IPsec SA Keying Material

- After the quick mode exchange the keying material for the IPsec SAs is generated
- There are two ways to establish the keying material
  - With perfect forward secrecy
    - Fresh DH values included in the quick mode exchange
  - Without perfect forward secrecy

K = PRF(SKEYd, protocol ||SPI || Nonce-I || Nonce-R) (without PFS)
K = PRF(SKEYd, fresh DH-key || protocol ||SPI || Nonce-I || Nonce-R) (with PFS)

Expending the Keying Material

- If the keying material needed for encryption and / or integrity protection in an SA is longer than K, K can be expanded by chaining



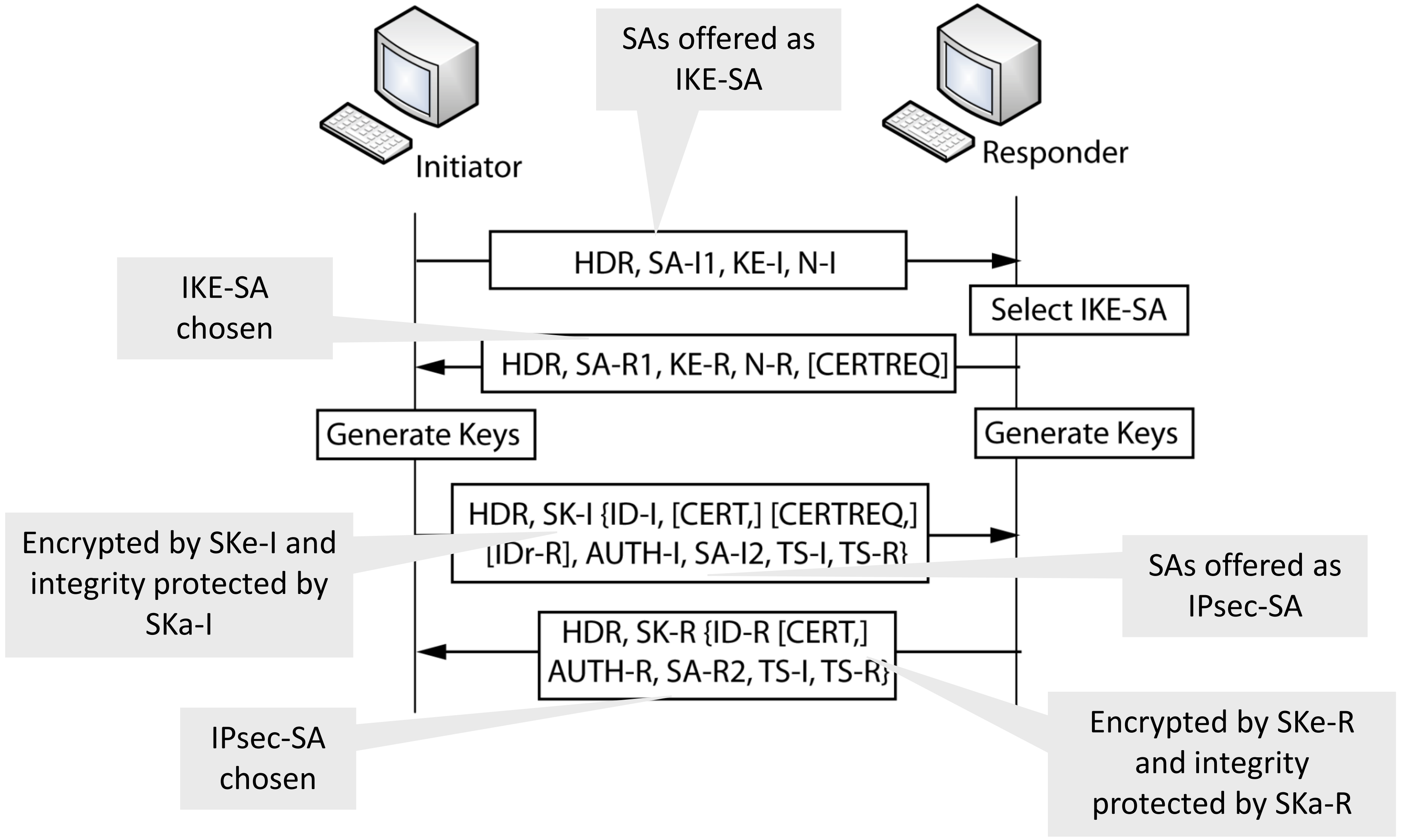
K = K1 || K2 || K3 …

IKEv2 Compared to IKEv1

- Fewer RFCs then IKE v1 (all in one)
  - I.e. a simpler specification and thus less prone to errors due to misunderstanding
- Less options to choose from
  - Instead of eight different phase 1 exchanges IKEv2 specifies only one initial exchange
- One pair of SAs for IPsec is already established during the initial IKE exchange (phase1)
  - More efficient SA establishment if only one is needed
- Additional child SAs can be generated with the create child SA exchange (2 messages)
- No mandatory use of cookies

IKE v2 Exchange

TS: Traffic selectors

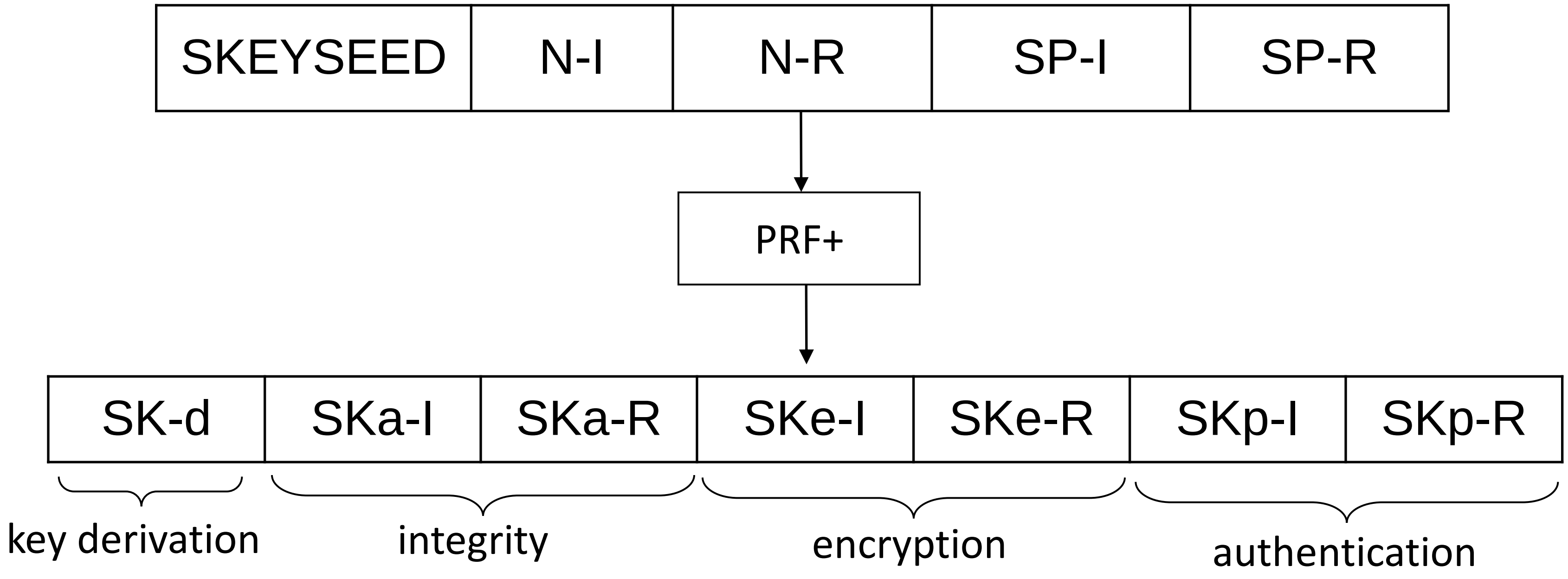

SAs offered as IKE-SA
Initiator
Responder
HDR, SA-I1, KE-I, N-I
IKE-SA chosen
Select IKE-SA
HDR, SA-R1, KE-R, N-R, [CERTREQ]
Generate Keys
Generate Keys
HDR, SK-I {ID-I, [CERT,] [CERTREQ,] [IDr-R], AUTH-I, SA-I2, TS-I, TS-R}
Encrypted by SKe-I and integrity protected by SKa-I
SAs offered as IPsec-SA
HDR, SK-R {ID-R [CERT,] AUTH-R, SA-R2, TS-I, TS-R}
IPsec-SA chosen
Encrypted by SKe-R and integrity protected by SKa-R

Key Generation

- All key generations in IKE are based on a PRF negotiated as part of the IKE-SA
- The PRF is used to generate keying material of arbitrary length by
  - PRF+(K,S) = K1 | K2 | K3 | K4 | ...
    - where: K1 = PRF (K, S | 0x01), K2 = PRF(K, K1 | S | 0x02), K3 = PRF(K, K2 | S | 0x03),…

Key Generation in IKE v2 Exchange

- Simplified but similar to IKEv1!
- SKEYSEED = PRF(N-I, N-R, g^{ir}), where g^{ir} is the DH key computed from the DH values KE-I, r and KE-R, i respectively



Authentication in IKE v2

- Authentication between initiator and responder can be based on
  - Pre-shared key
  - Signatures
- Signature case
  - If **R** authenticates to **I** with a signature, then **R** signs
    - Complete second message || N-I || PRF(SKp-R, ID-R payload)
  - **R** includes the signature in the AUTH field of the fourth message
  - If **I** authenticates to **R** with a signature, then **I** signs
    - Complete first message || N-R || PRF(SKp-I, ID-I payload)
  - **I** includes the signature in the AUTH field of the third message
  - Signatures can either be RSA or DSS signatures

Authentication in IKE v2 (cont’d)

- Pre-shared key case
  - If **R** authenticates to I with a pre-shared key PSK
    - **R** computes AUTH = PRF(PRF(PSK,"Key Pad for IKEv2"), message 2 || N-I || PRF(SKp-R, ID-R payload))
  - If **I** authenticates to **R** with a pre-shared key PSK
    - **R** computes AUTH = PRF(PRF(PSK,"Key Pad for IKEv2"), message 1 || N-R || PRF(SKp-I, ID-I payload))

- Note that pre-shared key and signature-based authentication may be mixed
- Other authentication methods (based on the extensible authentication protocol) may also be used but require more messages
- Note that PRF is negotiated as part of the IKE-SA

The Keying Material for First Child IPsec SAs

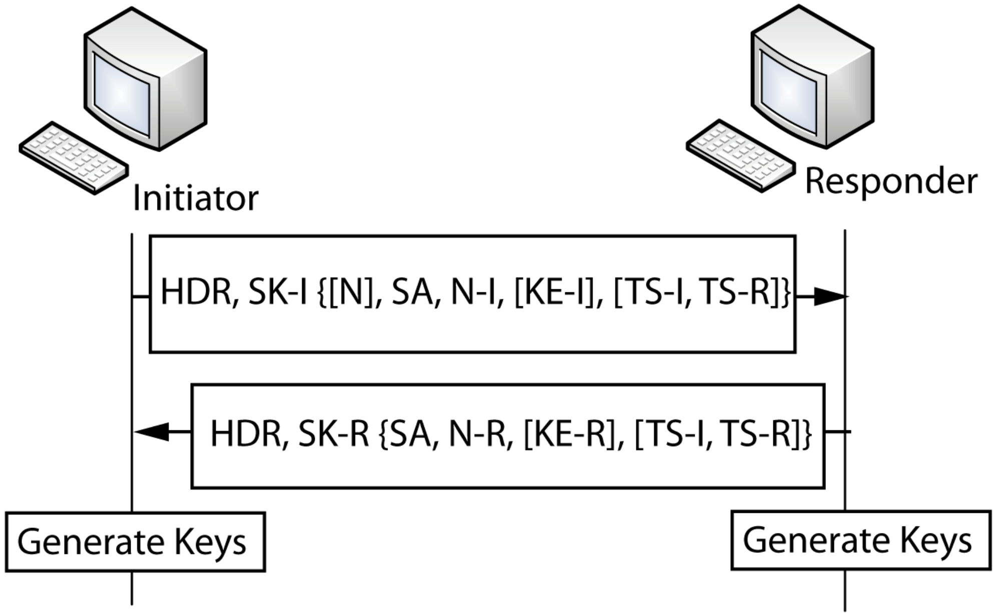
- Key material for the first IPsec child SAs is generated during the IKE exchange by
  - KEYMAT = PRF+(SKd, N-I || N-R)
- Up to two pairs of SAs may be created during the initial IKE exchange (one for each direction for AH and ESP)
- Keys for the SAs are generated from KEYMAT by
  - First taking the material for the direction from **I** to **R**
  - Take keys for the protocol first that will appear first in the encapsulated packet
  - If ESP with enc. and int. is used, take encryption keys first, then integrity keys

Negotiation of Traffic Selectors

- The traffic selectors for both directions of the IPsec SA pair generated during the IKE exchange are negotiated as part of message 3 and 4
- In message 3 the initiator makes a proposal for traffic selectors for each direction (TS-I, TS-R)
- The responder may choose a subset of the selectors suggested by the initiator and include them in message 4

Create Child Exchange

- After the IKE exchange further SAs may be negotiated between Initiator and Responder
- The key material is generated as during IKE exchange, using SKd of the IKE exchange and the fresh nonces
- IF new DH values are present in the child exchange a new DH key is generated and used in the key generation


Initiator
Responder
HDR, SK-I {[N], SA, N-I, [KE-I], [TS-I, TS-R]}
HDR, SK-R {SA, N-R, [KE-R], [TS-I, TS-R]}
Generate Keys
Generate Keys

Cookies

- As opposed to IKE v1 cookies are not included in a regular IKE v2 exchange
- However, the responder can request the use of cookies after receipt of the first message of the initiator
- This allows a responder under a flooding attack to react by requesting cookies

IKE v2 Header

8	8	8	8
Initiators IKE-SA SPI			
Responders IKE-SA SPI			
Next Payload	Minor Ver. / Major Ver.	Exchange Type	Flags
Message ID			
Length			

Generic Payload Header

8	8		8
Next Payload	c	Reserved	Payload Length

- Each IKE payload is appended with the payload header
- Next Payload – indicates type of next payload
- C – flag that indicates if the recipient should ignore the payload or drop the connection if he cannot interpret it

IKE Payloads

- Security Association
- Key Exchange
- Identification – Initiator
- Identification – Responder
- Certificate
- Certificate Request
- Authentication
- Nonce
- Notify
- Delete
- Vendor ID
- Traffic Selector - Initiator
- Traffic Selector - Responder
- Encrypted
- Configuration
- Extensible Authentication

PRFs specified for IKE v2

- Currently four different PRFs are specified for use with IKE v2
  - PRF_HMAC_MD5 (RFC2104),
  - PRF_HMAC_SHA1 (RFC2104),
  - PRF_HMAC_TIGER (RFC2104)
  - PRF_AES128_XCBC (RFC3664)
- Additional private ones can be defined by vendors

Encryption Algorithms in IKE v2

- The following encryption algorithms are specified within IKE v2

ENCR_DES_IV64	(RFC1827)
ENCR_DES	(RFC2405)
ENCR_3DES	(RFC2451)
ENCR_RC5	(RFC2451)
ENCR_IDEA	(RFC2451)
ENCR_CAST	(RFC2451)
ENCR_BLOWFISH	(RFC2451)
ENCR_3IDEA	(RFC2451)
ENCR_NULL	(RFC2410)
ENCR_AES_CBC	(RFC3602)
ENCR_AES_CTR	(RFC3664)

- Additional vendor specific algorithms can be specified

Integrity Algorithms in IKE v2

- The following integrity protection algorithms are specified for IKE v2
  - AUTH_HMAC_MD5_96 (RFC 2403)
  - AUTH_HMAC_SHA1_96 (RFC 2404)
  - AUTH_DES_MAC
  - AUTH_KPDK_MD5 (RFC 1826)
  - AUTH_AES_XCBC_96 (RFC 3566)
- Additional vendor specific algorithms can be specified

Recommended Reading

General reading on IPsec and IKEv1

- Kaufmann Chapter 18

For IKE v2:

- RFC 7296 https://tools.ietf.org/html/rfc7296
- RFC 2410: The NULL Encryption Algorithm and Its Use With IPsec http://tools.ietf.org/html/rfc2410

Additional reading

- N. Ferguson and B. Schneier: A cryptographic evaluation of IPsec, 2003:
- H.Soussi, M.Hussain, H.Afifi, D.Seret : IKEv1 and IKEv2: A Quantitative Analyses, 2006
- Perlman, R. and Kaufman, C.: Key Exchange in IPSec: Analysis of IKE, 2000
- Ran Canetti and Hugo Krawczyk: Security Analysis of IKE's Signature-based Key-Exchange Protocol , 2003
- Cremers, C.: Key Exchange in IPsec revisited: Formal Analysis of IKEv1 and IKEv2, 2010