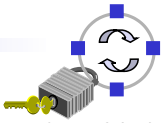
Network Security

Chapter 12

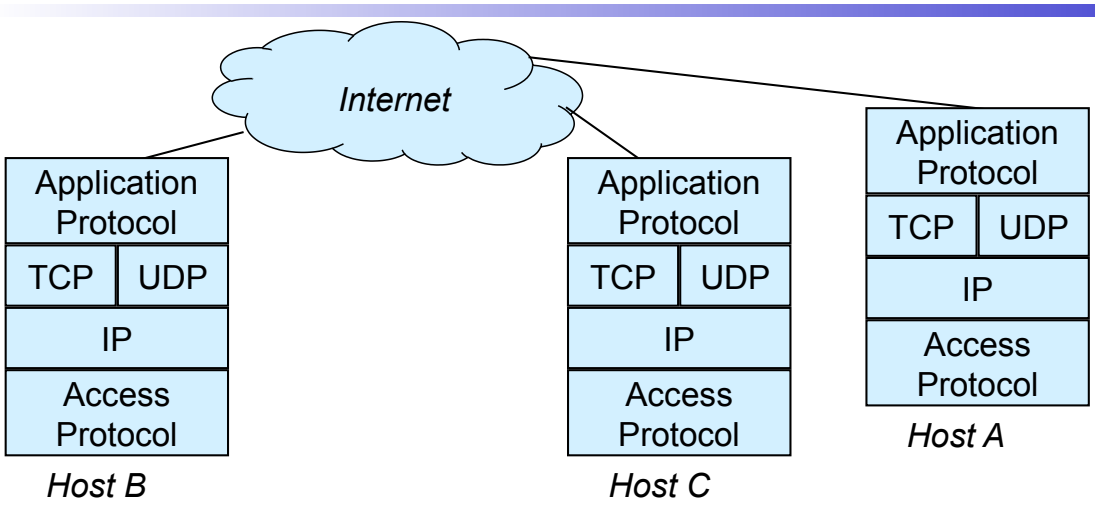
The *IPsec* Security Architecture for the Internet Protocol



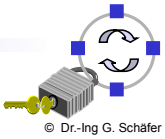
Overview

- Brief introduction to the Internet Protocol (IP) suite
- Security problems of IP and objectives of IPsec
- The IPsec architecture:
  - IPsec security protocol modes:
    - Transport mode
    - Tunnel mode
  - Implementation alternatives
  - IP Security Policy Database (SPD)
  - Security associations (SA) and the SA Database (SADB)
- IPsec security protocols:
  - Authentication Header (AH)
  - Encapsulating Security Payload (ESP)
- Entity Authentication and the Internet Key Exchange (IKE)

The TCP/IP Protocol Suite



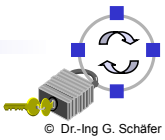
- *IP (Internet Protocol):* unreliable, connectionless network protocol
- *TCP (Transmission Control Protocol)*: reliable, connection-oriented transport protocol, realized over IP
- *UDP (User Datagram Protocol)*: unreliable, connectionless transport protocol, offers an application interface to IP
- Examples for *application protocols*:
  - HTTP: Hypertext Transfer Protocol
  - SMTP: Simple Mail Transfer Protocol



The IPv4 Packet Format (1)

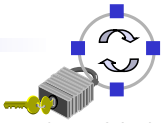
Ver.	IHL	TOS	Length	
IP Identification			Flags	Fragment Offset
TTL		Protocol	IP Checksum	
Source Address				
Destination Address				
IP Options (if any)				
TCP / UDP / ... Payload				

- *Version (Ver.):* 4 bit
  - Currently version 4 is widely deployed
  - Version 6 is already specified but is not yet clear if it will ever be deployed
- *IP header length (IHL):* 4 bit
  - Length of the IP header in 32-bit words
- *Type of service (TOS):* 8 bit
  - This field could be used to indicate the traffic requirements of a packet
  - Now: DCSP and Explicit Congestion (EC) Indication



The IPv4 Packet Format (2)

- *Length:* 16 bit
  - The length of the packet including the header in octets
  - This field is, like all other fields in the IP suite, in “big endian” representation
- *Identification:* 16 bit
  - Used to “uniquely” identify an IP datagram
  - Important for re-assembly of fragmented IP datagrams
- *Flags:* 3 bit
  - Bit 1: do not fragment
  - Bit 2: datagram fragmented
  - Bit 3: reserved for future use
- *Fragmentation offset:* 13 bit
  - The position of this packet in the corresponding IP datagram
- *Time to live (TTL):* 8 bit
  - At every processing network node, this field is decremented by one
  - When TTL reaches 0 the packet is discarded to avoid packet looping

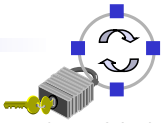


The IPv4 Packet Format (3)

- *Protocol:* 8 bit
  - Indicates the (transport) protocol of the payload
  - Used by the receiving end system to de-multiplex packets among various transport protocols like TCP, UDP, ...
- *Checksum:* 16 bit
  - Protection against transmission errors
  - As this is not a cryptographic checksum, it can easily be forged
- *Source address:* 32 bit
  - The IP address of sender of this packet
- *Destination address:* 32 bit
  - The IP address of the intended receiver of this packet
- *IP Options:* variable length
  - An IP header can optionally carry additional information
  - As they are not integral to IPsec, they will not be discussed in this course

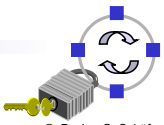
Security Problems of the Internet Protocol

- When an entity receives an IP packet, it has no assurance of:
  - *Data origin authentication / data integrity:*
    - The packet has actually been send by the entity which is referenced by the source address of the packet
    - The packet contains the original content the sender placed into it, so that it has not been modified during transport
    - The receiving entity is in fact the entity to which the sender wanted to send the packet
  - *Confidentiality:*
    - The original data was not inspected by a third party while the packet was sent from the sender to the receiver

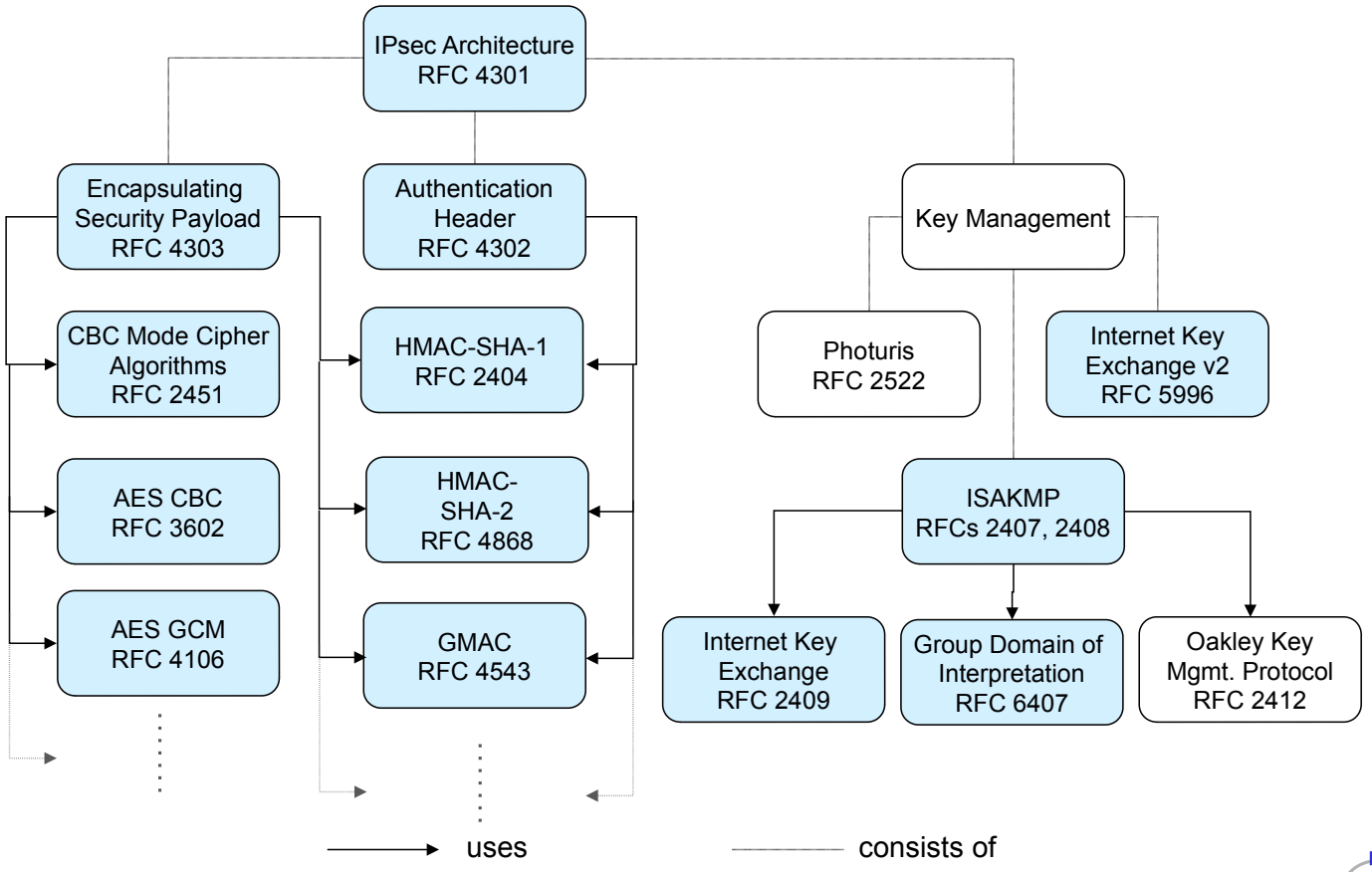


Security Objectives of IPsec

- IPsec aims to ensure the following security objectives:
  - *Data origin authentication / connectionless data integrity:*
    - It is not possible to send an IP datagram with neither a masqueraded IP source nor destination address without the receiver being able to detect this
    - It is not possible to modify an IP datagram in transit, without the receiver being able to detect the modification
    - *Replay protection:* it is not possible to later replay a recorded IP packet without the receiver being able to detect this
  - *Confidentiality:*
    - It is not possible to eavesdrop on the content of IP datagrams
    - Limited traffic flow confidentiality
- Security policy:
  - Sender, receiver and intermediate nodes can determine the required protection for an IP packet according to a local security policy
  - Intermediate nodes and the receiver will drop IP packets that do not meet these requirements



Overview of the IPsec Standardization

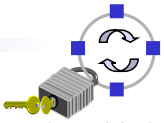


Overview of the IPsec Architecture (1)

- RFC 4301 defines the basic architecture of IPsec:
  - Concepts:
    - Security association (SA), security association database (SADB)
    - Security policy, security policy database (SPD)
  - Fundamental IPsec Protocols:
    - Authentication Header (AH)
    - Encapsulating Security Payload (ESP)
  - Protocol Modes:
    - Transport Mode
    - Tunnel Mode
  - Key Management Procedures:
    - IKE & IKEv2

Overview of the IPsec Architecture (2)

- RFC 4301 defines the basic architecture of IPsec:
  - Use of various cryptographic primitives with AH and ESP:
    - Encryption: 3DES-CBC, AES & other CBC mode cipher algorithms, AES counter mode
    - Integrity: HMAC-MD5, HMAC-SHA-1, HMAC-SHA-2, HMAC-RIPEMD-160, AES-GMAC, AES-CMAC, AES-XCBC…
    - Authenticated encryption: GCM and `Counter with CBC-MAC' (CCM), both defined for AES



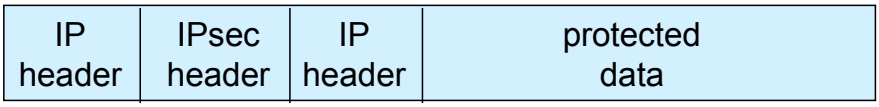
Overview of the IPsec Architecture (3)

- A *security association (SA)* is a simplex “connection” that provides security services to the traffic carried by it
  - Security services are provided to one SA by the use of either AH or ESP, but not both
  - For bi-directional communication two security associations are needed
  - An SA is uniquely identified by a triple consisting of a *security parameter index (SPI)*, an IP destination address, and a security protocol identifier (AH / ESP)
  - An SA can be set up between the following peers:
    - Host ↔ Host
    - Host ↔ Gateway (or vice versa)
    - Gateway ↔ Gateway
  - There are two conceptual databases associated with SAs:
    - The security policy database (SPD) specifies, what security services are to be provided to which IP packets and in what fashion
    - The security association database (SADB)

Overview of the IPsec Architecture (4)

- Protocol modes – An SA is always of one of the following types:
  - *Transport mode* can only be used between end-points of a communication:
    - host ↔ host, or
    - host ↔ gateway, if the gateway is a communication end-point (e.g. for network management)
  - *Tunnel mode* can be used with arbitrary peers
- The difference between the two modes is, that:
  - Transport mode just adds a security specific header (+ eventual trailer):

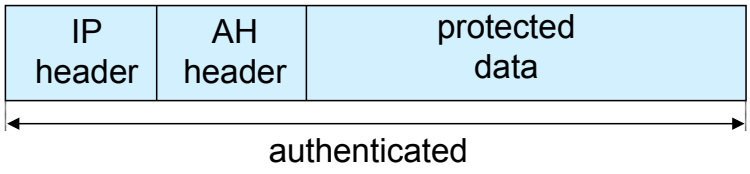
  - Tunnel mode encapsulates IP packets:



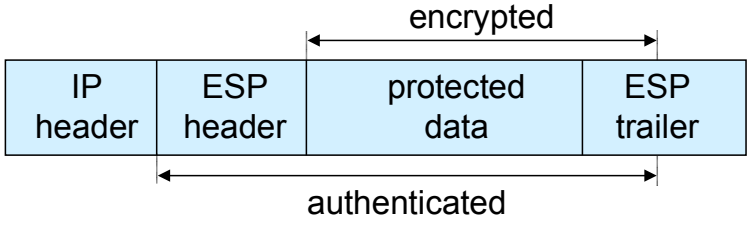
Encapsulation of IP packets allows for a gateway protecting traffic on behalf of other entities (e.g. hosts of a subnetwork, etc.)

Overview of the IPsec Architecture (5)

- The authentication header (AH):
  - Provides data origin authentication and replay protection
  - Is realized as a header which is inserted between the IP header and the data to be protected

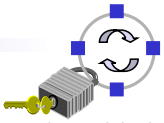


- The encapsulating security payload (ESP):
  - Provides data origin authentication, confidentiality and replay protection
  - Is realized with a header and a trailer encapsulating the data to be protected



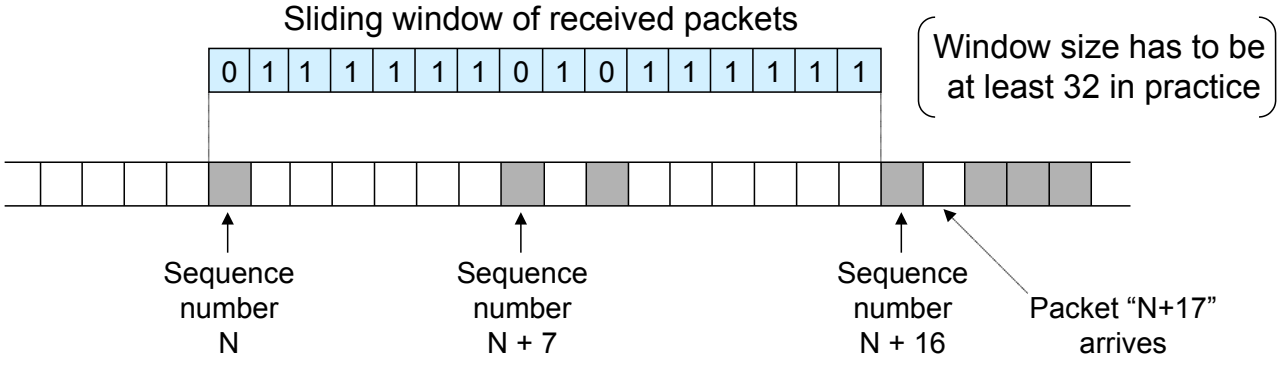
Overview of the IPsec Architecture (6)

- Setup of security associations is realized with:
  - Internet Security Association Key Management Protocol (ISAKMP):
    - Defines generic framework for key authentication, key exchange and negotiation of security association parameters [RFC2408]
    - Does not define a specific authentication protocol, but specifies:
      - Packet formats
      - Retransmission timers
      - Message construction requirements
    - Use of ISAKMP for IPsec is further detailed in [RFC2407]
  - Internet Key Exchange (IKE):
    - Defines an authentication and key exchange protocol [RFC2409]
    - Is conformant to ISAKMP and may be used for different applications
    - Setup of IPsec SAs between two entities is realized in two phases:
      - Establishment of an IKE SA (defines how to setup IPsec SAs)
      - Setup of IPsec SAs

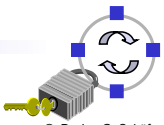


IPsec Replay Protection (1)

- Both AH- and ESP-protected IP packets carry a sequence number which realizes a replay protection:
  - When setting up an SA this sequence number is initialized to zero
  - The sequence number is increased with every IP packet sent
  - The sequence number is 32 bit long, a new session key is needed before a wrap-around occurs
  - The receiver of an IP packet checks, if the sequence number is contained in a window of acceptable numbers



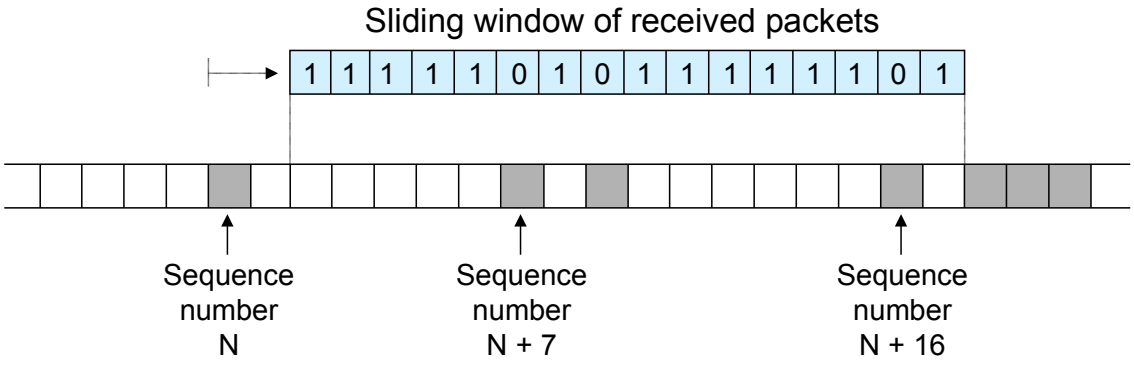
Packet with sequence number N can still be accepted



IPsec Replay Protection (2)

- If a received packet has a sequence number which:
  - is to the left of the current window ⇒ the receiver rejects the packet
  - is inside the current window ⇒ the receiver accepts the packet
  - is to the right of the current window ⇒ the receiver accepts the packet and advances the window

  Of course IP packets are only accepted if they pass the authentication verification and the window is never advanced before this verification
- The minimum window size is 32 packets (64 packets is recommended)



Packet with sequence number N can no longer be accepted

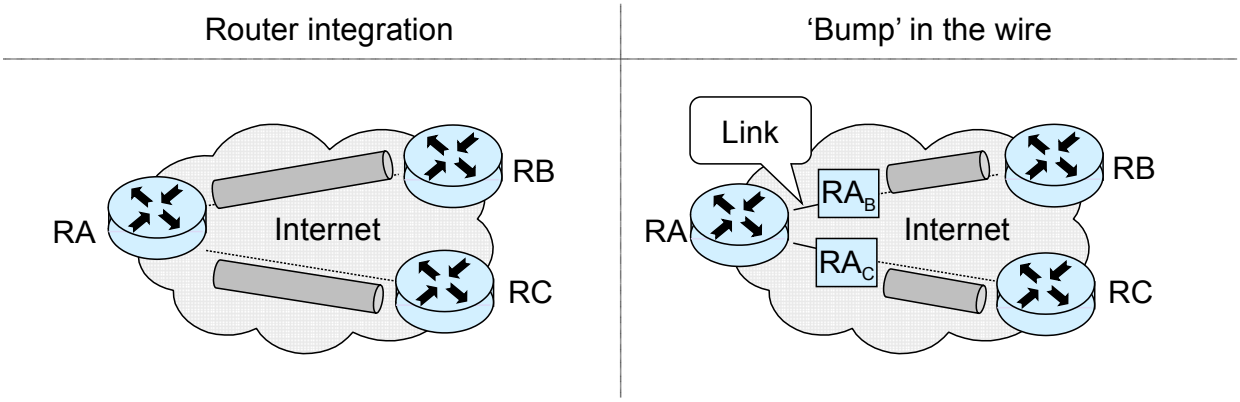
IPsec Implementation Alternatives: Host Implementation

- Advantages of IPsec implementation in end systems:
  - Provision of end-to-end security services
  - Provision of security services on a per-flow basis
  - Ability to implement all modes of IPsec
- Two main integration alternatives:

OS integrated	"Bump" in the stack
Application Transport Network + IPsec Data Link	Application Transport Network IPsec Data Link
True OS integration is the method of choice, as it avoids duplication of functionality	If the OS can not be modified, IPsec is inserted above the data link driver

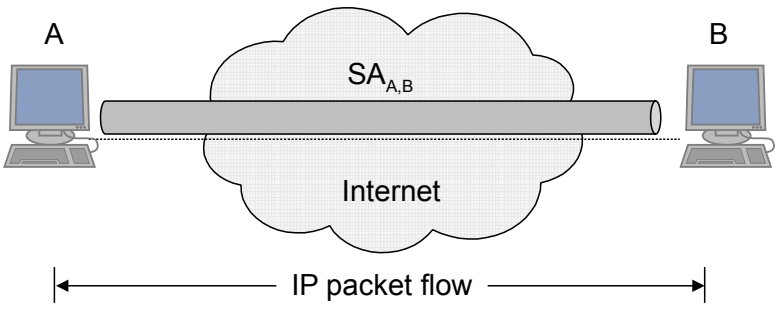
IPsec Implementation Alternatives: Router Implementation

- Advantages of IPsec implementation in routers:
  - Ability to secure IP packets flowing between two networks over a public network such as the Internet:
    - Allows to create *virtual private networks (VPNs)*
    - No need to integrate IPsec in every end system
  - Ability to authenticate and authorize IP traffic coming in from remote users
- Two main implementation alternatives:



When to use which IPsec Mode? (1)

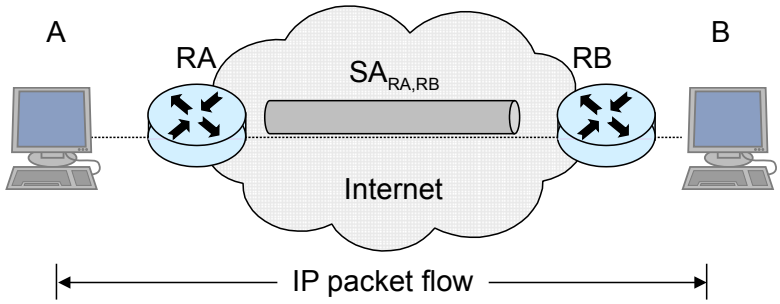
- Transport mode is used when the "cryptographic endpoints" are also the "communication endpoints" of the secured IP packets
  - Cryptographic endpoints: the entities that generate / process an IPsec header (AH or ESP)
  - Communication endpoints: source and destination of an IP packet



- In most cases, communication endpoints are hosts (workstations, servers), but this is not necessarily the case:
  - Example: a gateway being managed via SNMP by a workstation

When to use which IPsec Mode? (2)

- Tunnel mode is used when at least one “cryptographic endpoint” is not a “communication endpoint” of the secured IP packets
  - This allows for gateways securing IP traffic on behalf of other entities

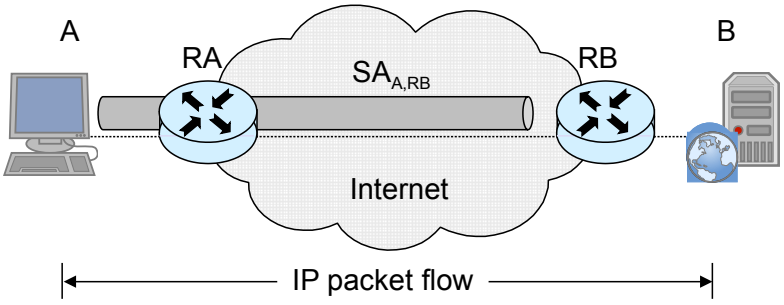


Packet structure

IP Header	IPsec Header	IP Header	Protected Data
Src = RA Dst = RB		Src = A Dst = B	

When to use which IPsec Mode? (3)

- The above description of application scenarios for tunnel mode includes the case in which only one cryptographic endpoint is not a communication endpoint:
  - Example: a security gateway ensuring authentication and / or confidentiality of IP traffic between a local subnetwork and a host connected via the Internet (“road warrior scenario”)

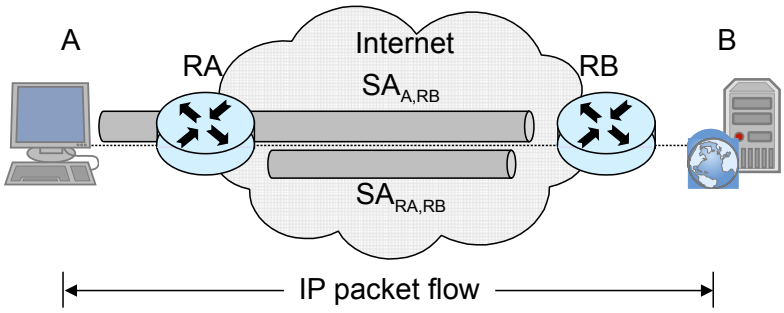


Packet structure

IP Header	IPsec Header	IP Header	Protected Data
Src = A Dst = RB		Src = A Dst = B	

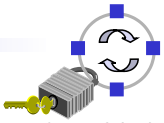
Nesting of Security Associations (1)

- Security associations may be nested:
  - Example: Host A and gateway RB perform data origin authentication and gateways RA and RB perform subnetwork-to-subnetwork confidentiality



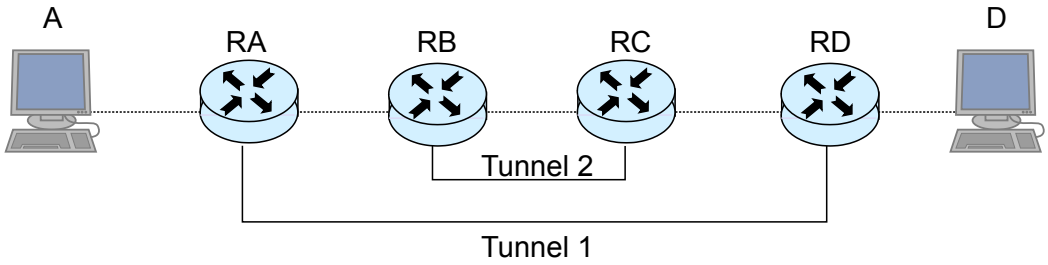
Packet structure

IP Header	IPsec Header	IP Header	IPsec Header	IP Header	Protected Data
Src = RA Dst = RB		Src = A Dst = RB		Src = A Dst = B	



Nesting of Security Associations (2)

- However, one has to take care when nesting SAs that there occurs no "incorrect bracketing" of SAs, like "[(])"
- One example of valid SA nesting:

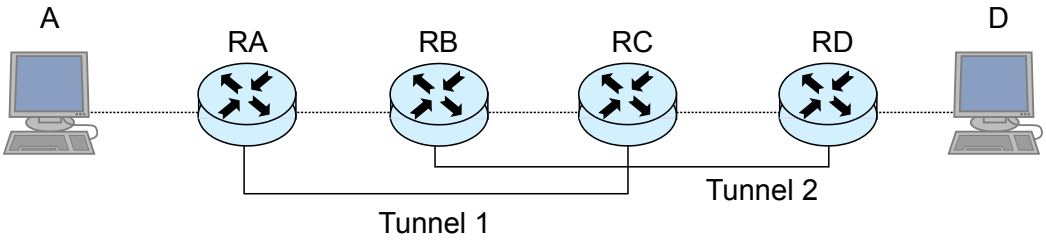


Packet structure

IP Header	IPsec Header	IP Header	IPsec Header	IP Header	Protected Data
Src = RB Dst = RC		Src = RA Dst = RD		Src = A Dst = D	

Nesting of Security Associations (3)

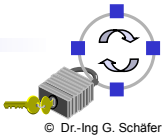
- One example of invalid SA nesting:



Packet structure

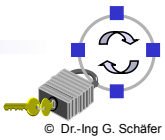
IP Header	IPsec Header	IP Header	IPsec Header	IP Header	Protected Data
Src = RB Dst = RD		Src = RA Dst = RC		Src = A Dst = D	

- As the packet is tunneled from RB to RD the gateway RC can not process the inner IPsec header
- A possible result of this faulty configuration could be that the packet is routed back to RC



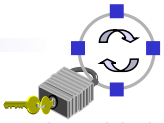
Basic Scheme of IPsec Processing: Outgoing Packets

- Consider, the IP layer of one node (host / gateway) is told to send an IP packet to another node (host / gateway)
- In order to support IPsec it has to perform the following steps:
  - Determine if and how the outgoing packet needs to be secured:
    - This is realized by performing a lookup in the SPD
    - If the policy specifies “discard” then drop the packet ⇒ done
    - If the packet does not need to be secured, then send it ⇒ done
  - Determine which SA should be applied to the packet:
    - If there is not yet an appropriate SA established with the corresponding node, then ask the key management demon to perform an IKE
  - Look up the determined (and eventually freshly created) SA in the SADB
  - Perform the security transform determined by the SA by using the algorithm, its’ parameters and the key as specified in the SA
    - This results in the construction of an AH or an ESP header
    - Eventually also a new (outer) IP header will be created (tunnel mode)
  - Send the resulting IP packet ⇒ done



Basic Scheme of IPsec Processing: Incoming Packets

- Consider now, the IP layer of one node (host / gateway) receives an IP packet from another node (host / gateway)
- In order to support IPsec it has to perform the following steps:
  - Determine if the packet contains an IPsec header this entity is supposed to process:
    - If there is such an IPsec header then look up the SA in the SADB which is specified by the SPI of the IPsec header and perform the appropriate IPsec processing
    - If the SA referenced by the SPI does not (yet) exist, drop the packet
  - Determine if and how the packet should have been protected:
    - This is again realized by performing a lookup in the SPD, with the lookup being performed by evaluating the inner IP header in case of tunneled packets
    - If the policy specifies “discard” then drop the packet
    - If the protection of the packet did not match the policy, drop the packet
    - If the packet had been properly secured, then deliver it to the appropriate protocol entity (network / transport layer)

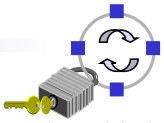


IPsec Security Policy Selection

- The following selectors to be extracted from the network and transport layer headers allow to select a specific policy in the SPD:
  - *IP source address:*
    - Specific host , network prefix, address range, or wildcard
  - *IP destination address:*
    - Specific host , network prefix, address range, or wildcard
    - In case of incoming tunneled packets the inner header is evaluated
  - *Protocol:*
    - The protocol identifier of the transport protocol for this packet
    - This may not be accessible when a packet is secured with ESP
  - *Upper layer ports:*
    - If accessible, the upper layer ports for session oriented policy selection

IPsec Security Policy Definition

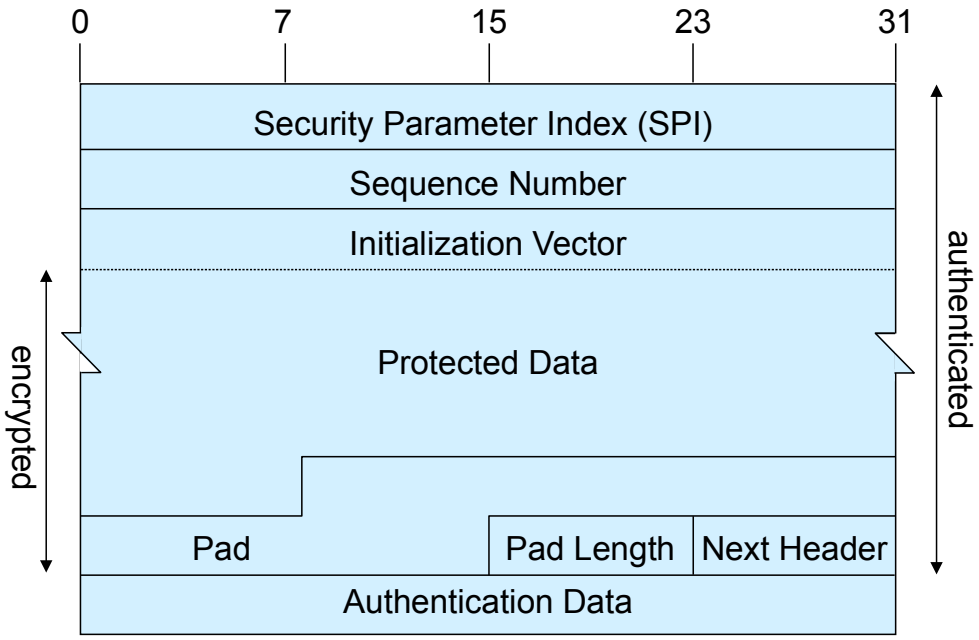
- Policy selectors are used to select specific policy definitions, specifies:
  - How to perform setup of an IKE SA between two nodes:
    - *Identification:* DNS name or other name types as defined in IPsec domain of interpretation of a protocol for setting up SAs
    - *Phase I mode:* main mode or aggressive mode (see below)
    - *Protection suite(s):* specify how IKE authentication is performed
  - Which and how security services should be provided to IP packets:
    - *Selectors,* that identify specific flows
    - *Security attributes* for each flow:
      - *Security protocol:* AH or ESP
      - *Protocol mode:* transport or tunnel mode
      - *Security transforms:* cryptographic algorithms and parameters
      - *Other parameters:* SA lifetime, replay window
    - *Action:* discard, secure, bypass
- If an SA is already established with a corresponding security endpoint, it is referenced in the SPD



The Encapsulating Security Payload (1)

- ESP constitutes a generic security protocol that provides to IP packets replay protection and one or both of the following security services:
  - Confidentiality, by encrypting encapsulated packets or just their payload
  - Data origin authentication, by creating and adding MACs to packets
- The ESP definition is divided up into two parts:
  - The definition of the base protocol [RFC4303]:
    - Definition of the header and trailer format
    - Basic protocol processing
    - Tunnel and transport mode operation
  - The use of specific cryptographic algorithms with ESP:
    - Encryption: 3DES-CBC, AES-CBC, AES counter mode, use of other ciphers in CBC mode
    - Authentication: HMAC-MD5-96, HMAC-SHA-96,...

The Encapsulating Security Payload (2)

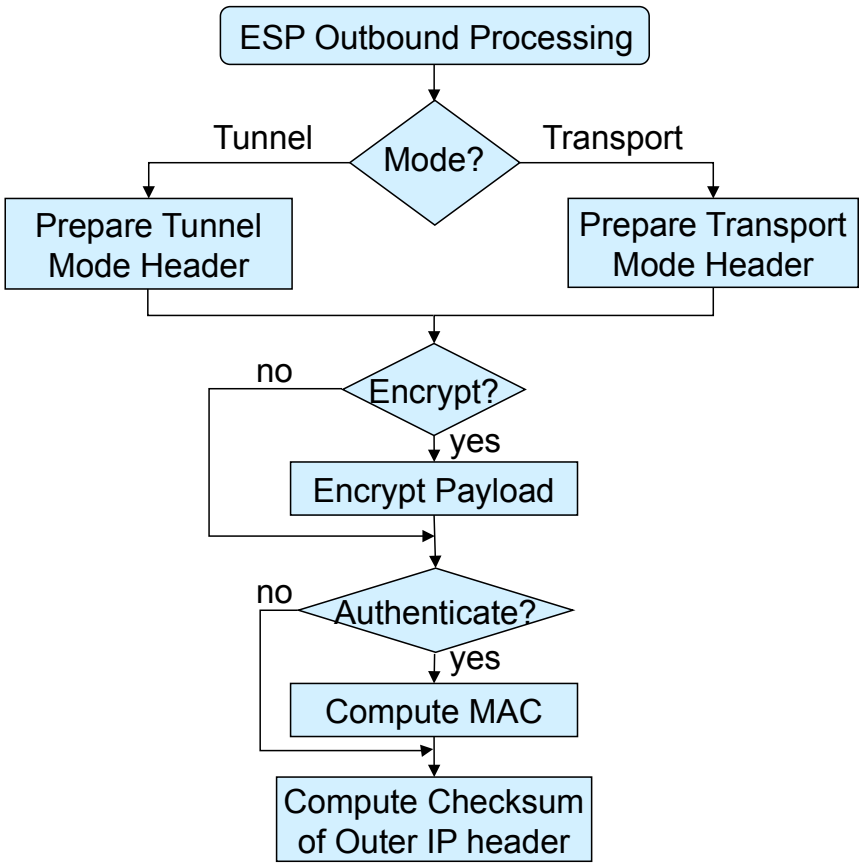


- The ESP header immediately follows an IP header or an AH header
- The next-header field of the preceding header indicates “50” for ESP

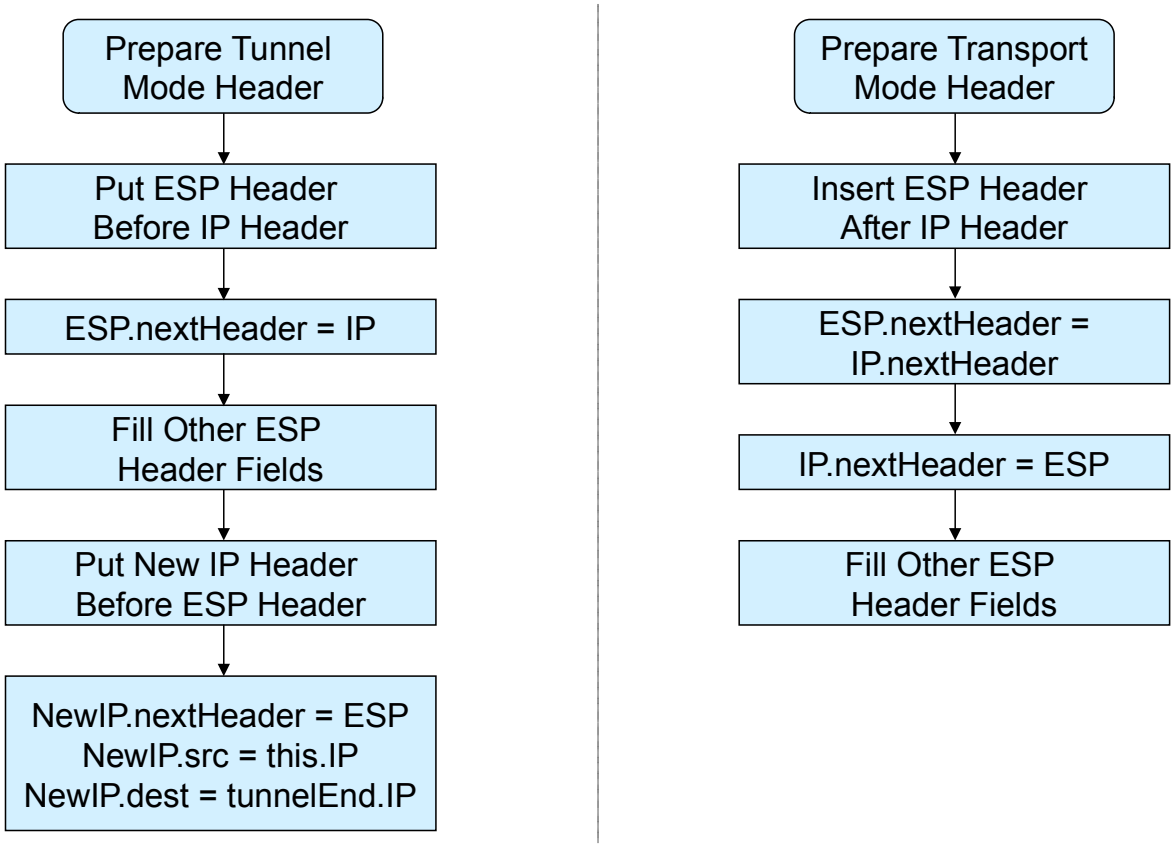
The Encapsulating Security Payload (3)

- The SPI field indicates the SA to be used for this packet:
  - The SPI value is always determined by the receiving side during SA negotiation as the receiver has to process the packet
- The sequence number provides replay protection as explained before
- If the cryptographic algorithm in use requires an initialization vector, it is transmitted in the clear in every packet in the beginning of the payload
- The pad field serves to ensure:
  - padding of the payload up to the required block length of the cipher in use
  - padding of the payload to right-justify the pad-length and next-header fields into the high-order 16 bit of a 32-bit word
- The pad length indicates the amount of padding bytes added
- The next-header field of the ESP header indicates the encapsulated payload:
  - In case of tunnel mode: IP
  - In case of transport mode: any higher-layer protocol as TCP, UDP, ...
- The optional authentication-data field contains a MAC, if present

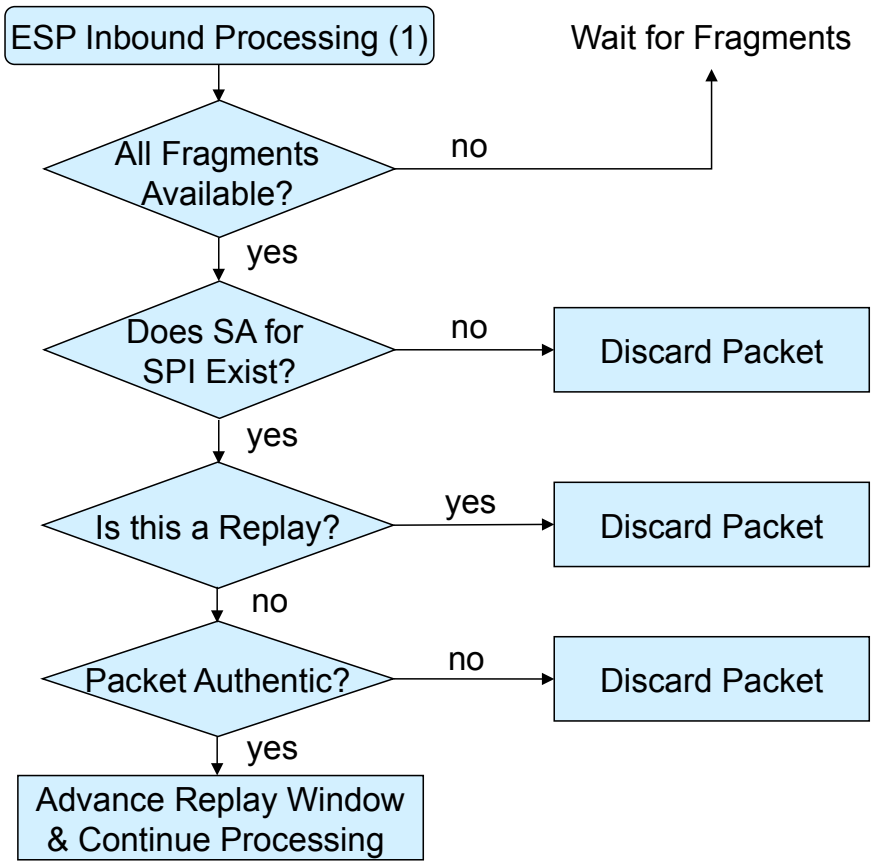
The Encapsulating Security Payload (4)



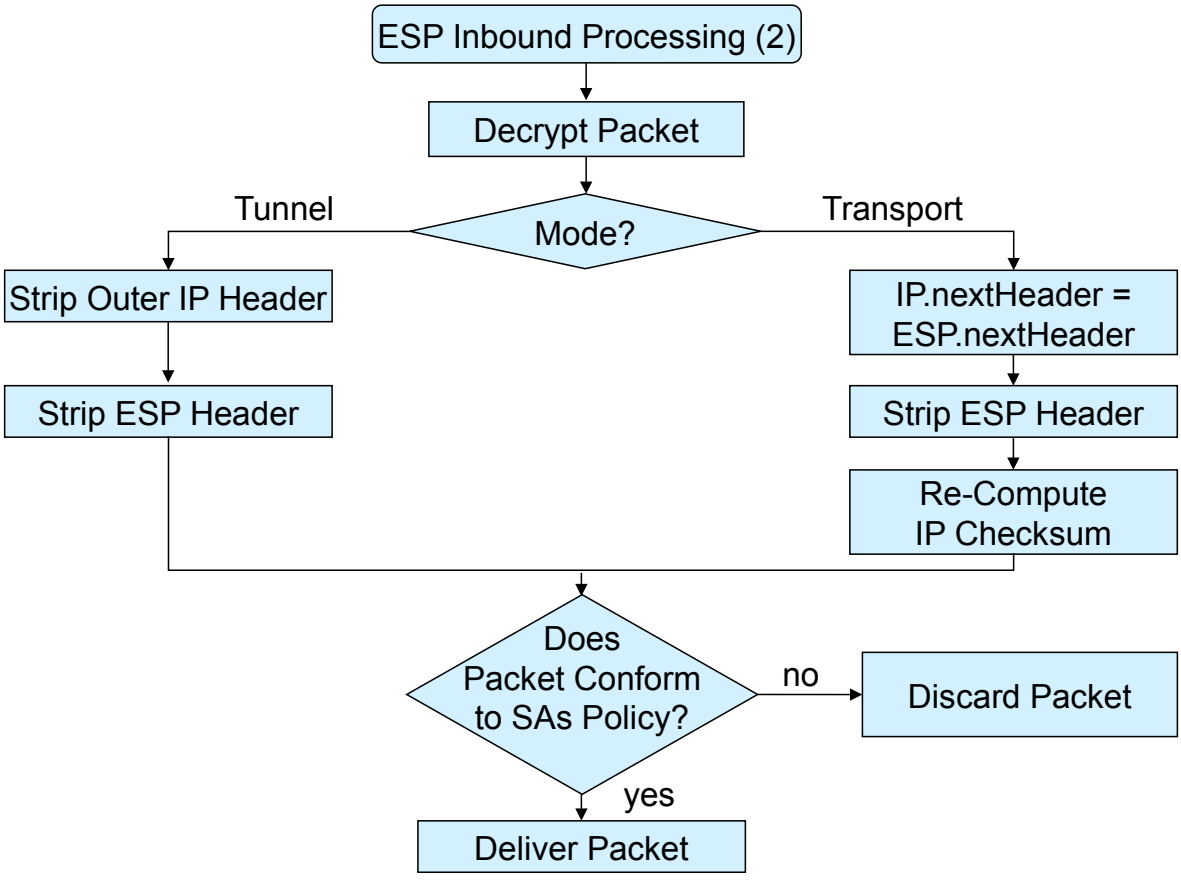
The Encapsulating Security Payload (5)



The Encapsulating Security Payload (6)

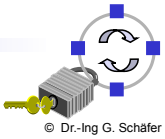


The Encapsulating Security Payload (7)



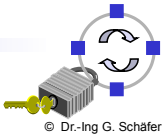
The Encapsulating Security Payload (8)

- Note, that the de-capsulated IP packet can be a fragmented packet:
  - This might occur when ESP was applied in tunnel mode by a router
  - In order to correctly check conformance to the SA's policy, all fragments belonging to that packet might have to be received by the router before the check can be applied
  - Example: only packets to a specific port are allowed in one SA
    - The required port information is only available in the first fragment of the IP packet
- Packet delivery means delivery to the appropriate processing entity:
  - If another IPsec header for this entity is present ⇒ IPsec processing
  - In tunnel mode ⇒ convey packet
  - In transport mode ⇒ call appropriate protocol header (TCP, UDP, etc.)



The Encapsulating Security Payload (9)

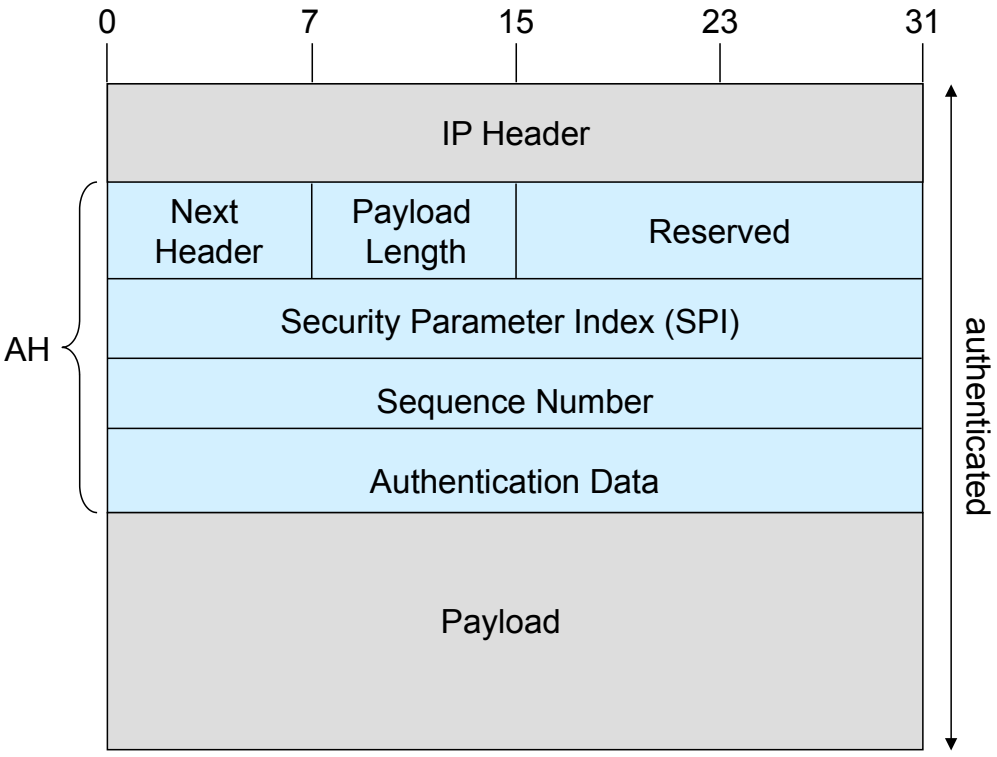
- If ESP provides both confidentiality and authentication then different keys may be used for both services
  - This needs to be negotiated during establishment of the ESP SA
- Note, using ESP without authentication is insecure...
  - No reliable replay protection
  - If at least if used in CBC mode:
    - Active attacks allow to recover messages
    - Example: flip bits and check if error messages are produced
    - Complete recovery of plaintext blocks



The Authentication Header (1)

- AH constitutes a generic security protocol that provides to IP packets:
  - Replay protection
  - Data origin authentication, by creating and adding MACs to packets
- Like with ESP the AH definition is divided up into two parts:
  - The definition of the base protocol [RFC4302]:
    - Definition of the header format
    - Basic protocol processing
    - Tunnel and transport mode operation
  - The use of specific cryptographic algorithms with AH:
    - Authentication: HMAC-MD5-96, HMAC-SHA1-96, HMAC-SHA2, …
    - If both ESP and AH are to be applied *by one entity*, then ESP is always applied first:
  - This results in AH being the outer header
  - "Advantage": the IP header can also be protected by AH
  - Remark: two SAs (one for each AH, ESP) are needed for each direction

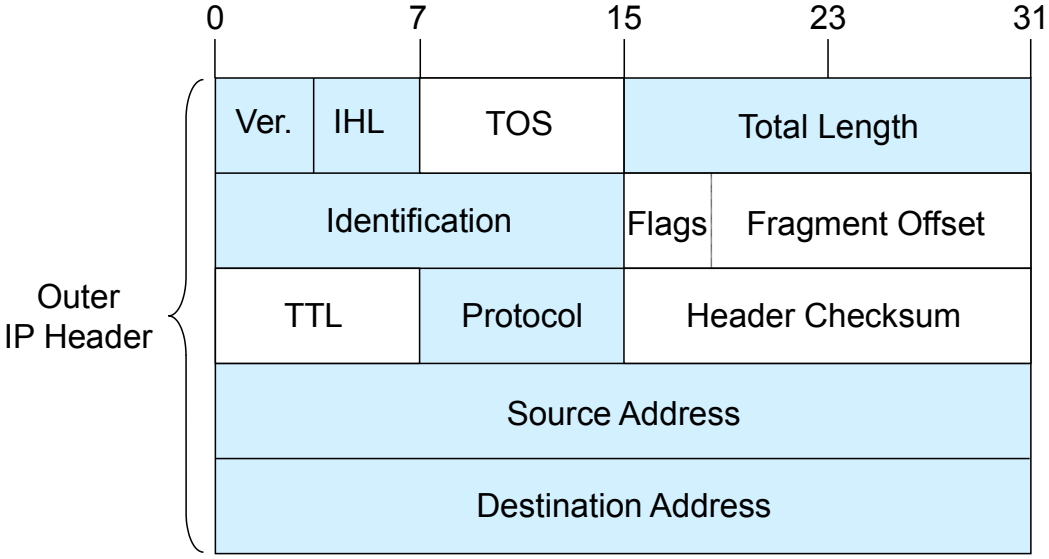
The Authentication Header (2)



- In tunnel mode the payload constitutes a complete IP packet

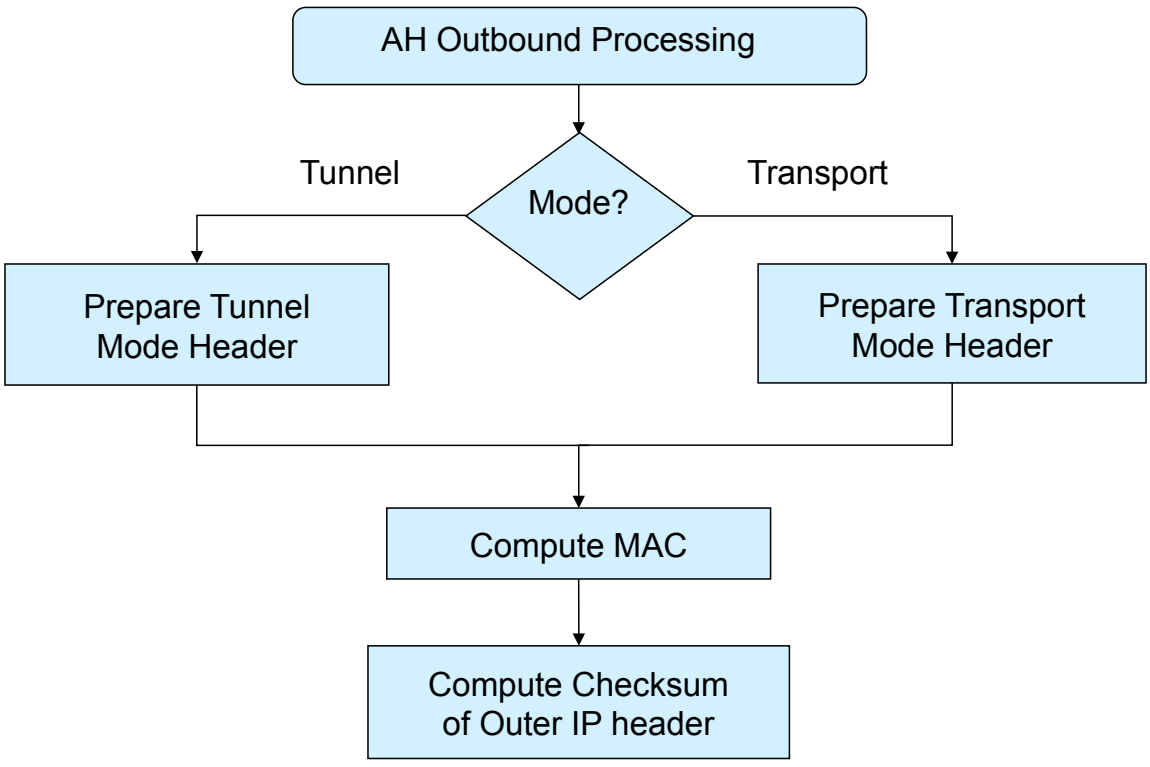
The Authentication Header (3)

- Although AH also protects the outer IP header, some of its' fields must not be protected as they are subject to change during transit:
  - This also applies to mutable IPv4 options or IPv6 extensions
  - Such fields are assumed being zero when computing the MAC

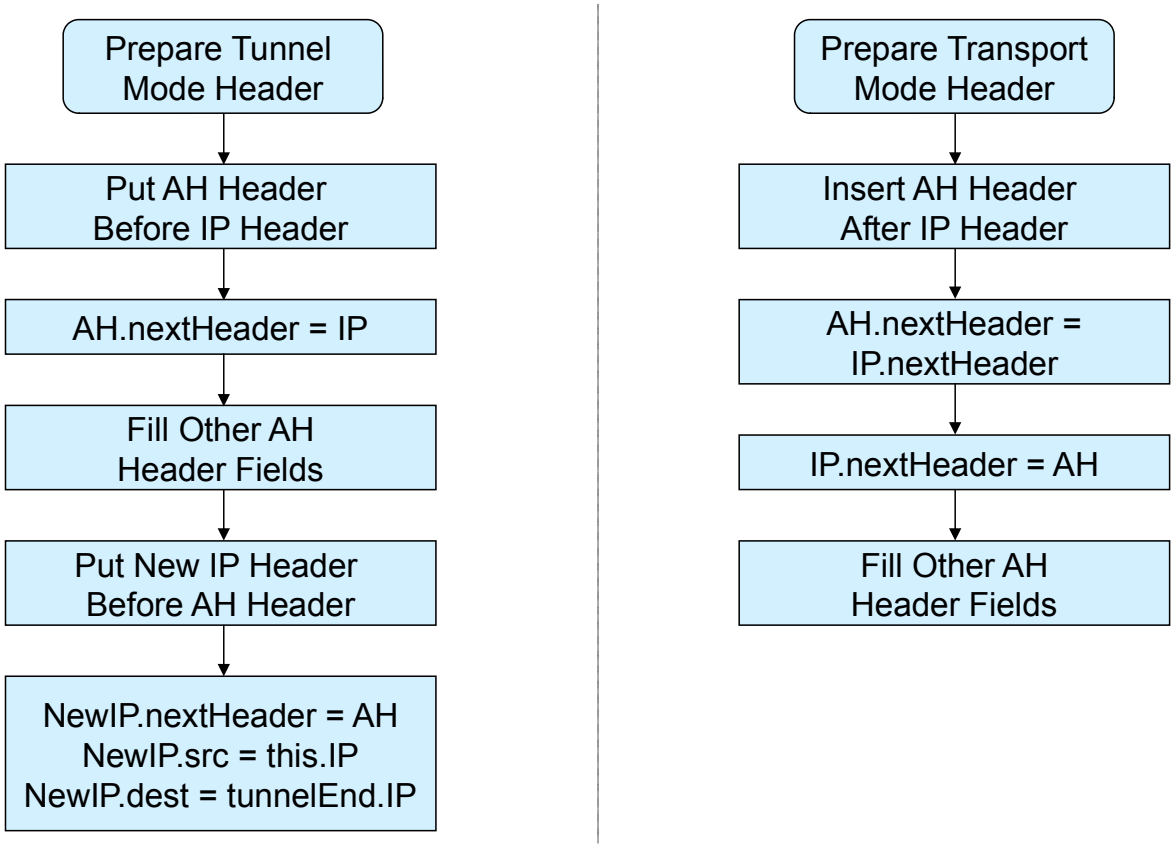


- All immutable fields, options and extensions (gray) are protected

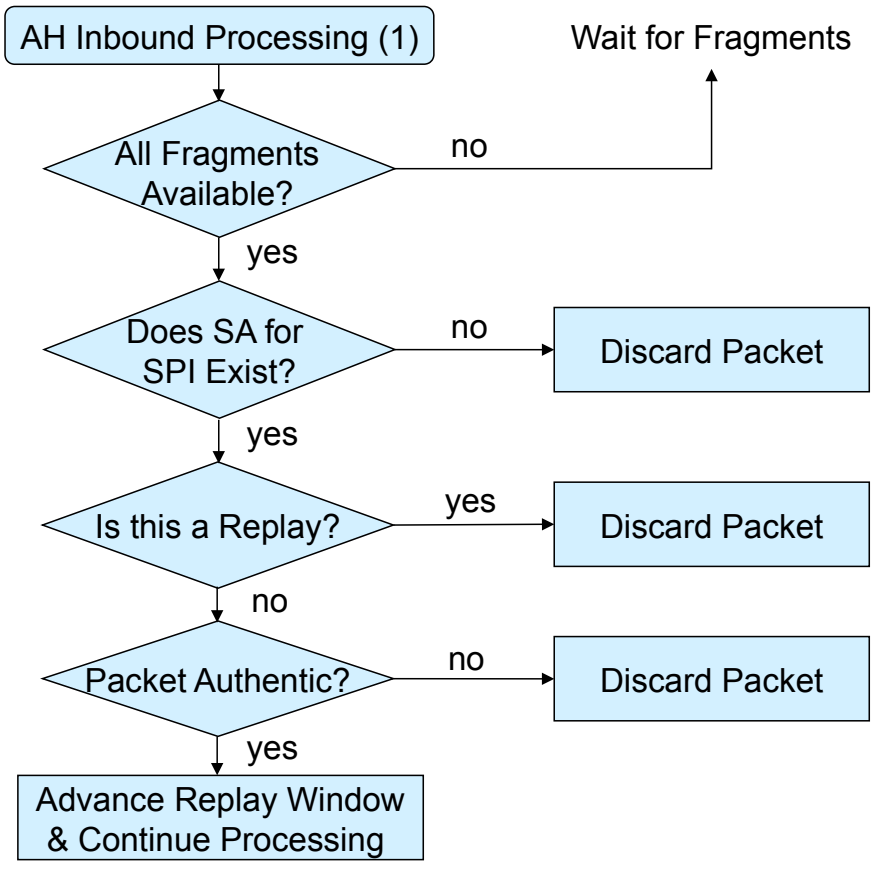
The Authentication Header (4)



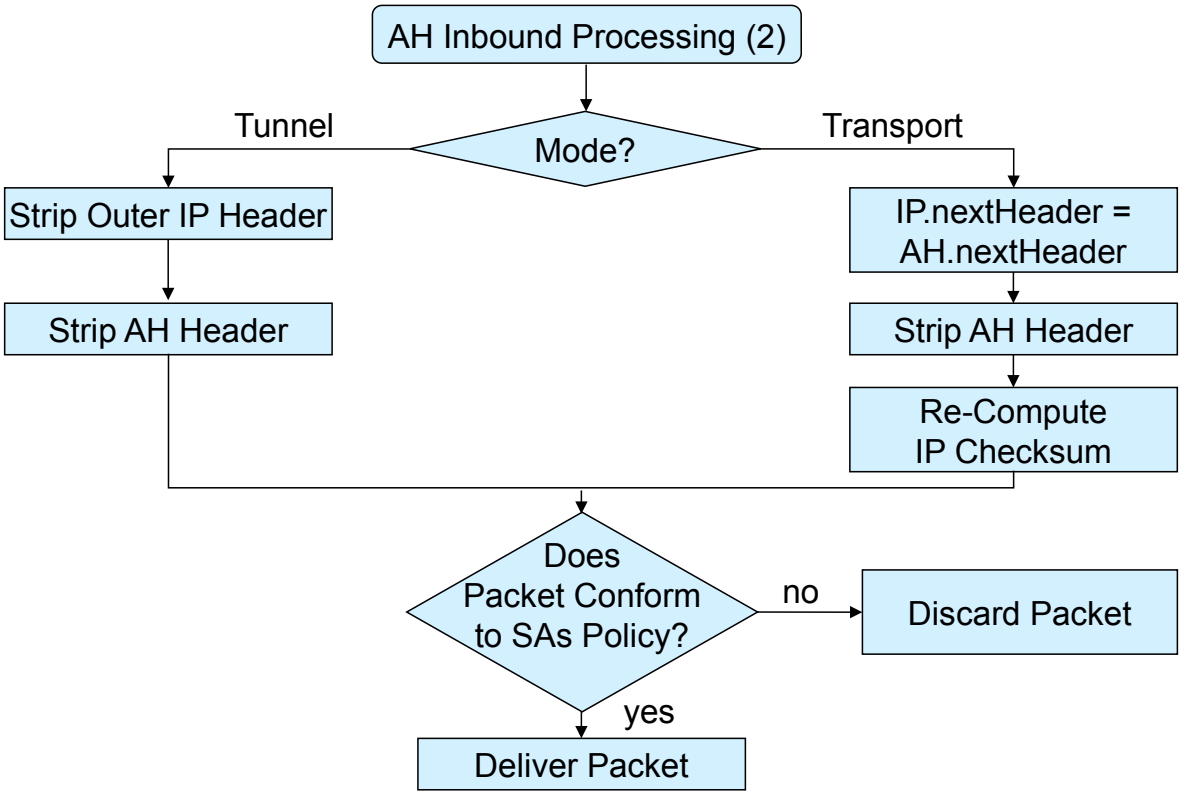
The Authentication Header (5)



The Authentication Header (6)



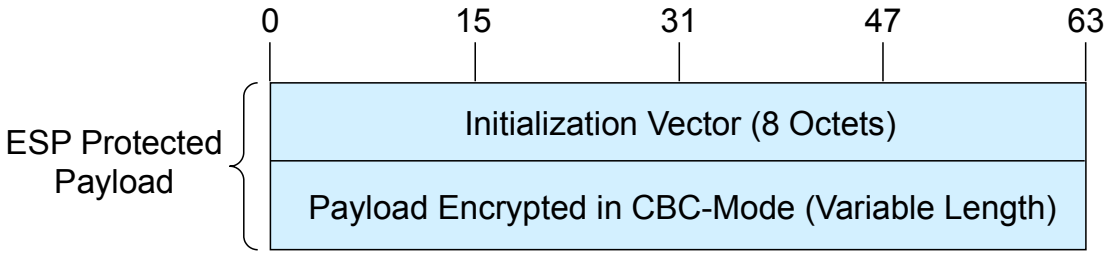
The Authentication Header (7)



IPsec's Use of Cryptographic Algorithms (1)

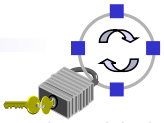
- Confidentiality (ESP only):
  - The use of DES with ESP [RFC4303] is no longer recommended
  - AES-CBC defined in RFC 3602 perhaps `the' standard algorithm

  - The initialization vector (IV) is always included in the clear, in order to avoid synchronization problems
  - The all IV is supposed to be random
  - Do NOT take further IVs from earlier ciphertexts!
    - Security issues
    - Synchronization issues



IPsec's Use of Cryptographic Algorithms (2)

- Data origin authentication (AH and ESP):
  - Some of the algorithms for authentication are defined so far:
    - HMAC-MD5-96 with key length 128 bit
    - HMAC-SHA1-96 with key length 160 bit
    - HMAC-RIPEMD160-96 with key length 160 bit
    - HMAC-SHA2 with key length 256, 384 & 512 bit
  - All of these algorithms use the HMAC construction defined in [RFC2104]:
    - ipad = 0x36 repeated B times (B = 64 for the above algorithms)
    - opad = 0x5C repeated B times
    - HMAC = H(Key XOR opad, H(Key XOR ipad, data))
      with H denoting the cryptographic hash function in use
  - The "-96" in the algorithms mentioned above means, that the output of the hash function is truncated to the 96 leftmost bits
  - SHA2 truncated to half of key length
  - This value fulfills most security requirements well

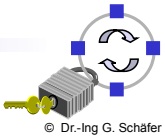


Establishment of Security Associations

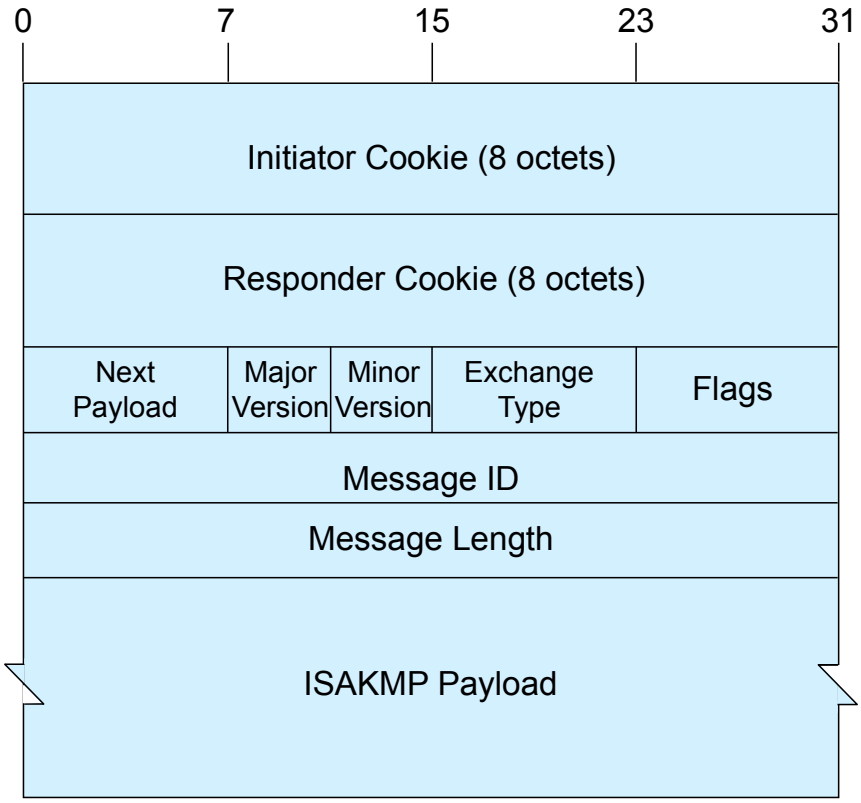
- Prior to any packet being protected by IPsec, an SA has to be established between the two "cryptographic endpoints" providing the protection
- SA establishment can be realized:
  - Manually, by proprietary methods of systems management
  - Dynamically, by a standardized authentication & key management protocol
  - Manual establishment is supposed to be used only in very restricted configurations (e.g. between two encrypting firewalls of a VPN) and during a transition phase
- IPsec defines a standardized method for SA establishment:
  - *Internet Security Association and Key Management Protocol (ISAKMP)*
    - Defines protocol formats and procedures for security negotiation
  - *Internet Key Exchange (IKE)*
    - Defines IPsec's standard authentication and key exchange protocol

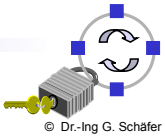
ISAKMP – Introduction

- The IETF has adopted two RFCs on ISAKMP for IPsec:
  - RFC 2408, which defines the ISAKMP base protocol
  - RFC 2407, which defines IPsec's *"domain of interpretation" (DOI)* for ISAKMP further detailing message formats specific for IPsec
- The ISAKMP base protocol is a generic protocol, that can be used for various purposes:
  - The procedures specific for one application of ISAKMP are detailed in a *DOI document*
  - Other DOI documents have been produced:
    - Group DOI for secure group communication [RFC6407]
    - MAP DOI for use of ISAKMP to establish SAs for securing the *Mobile Application Protocol (MAP)* of GSM (Internet Draft, Nov. 2000)
- ISAKMP defines two fundamental categories of exchanges:
  - Phase 1 exchanges, which negotiate some kind of "Master SA"
  - Phase 2 exchanges, which use the "Master SA" to establish other SAs



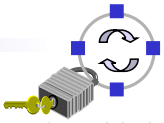
ISAKMP – Basic Message Format (1)





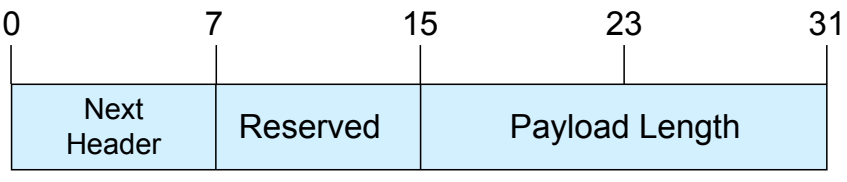
ISAKMP – Basic Message Format (2)

- *Initiator & responder cookie:*
  - Identify an ISAKMP exchange, or security association, respectively
  - Also serve as a limited protection against denial of service attacks (explained below)
- *Next payload:* specifies which ISAKMP payload type is the first payload of the message
- *Major & minor version:* identify the version of the ISAKMP protocol
- *Exchange type:*
  - Indicates the type of exchange being used
  - There are five pre-defined generic exchange types, further types can be defined per DOI
- *Flags:*
  - Encrypt: if set to one, then the payload following the header is encrypted
  - Commit: used for key synchronization purposes
  - Authenticate only: if set to one, only data origin authentication protection is applied to the ISAKMP payload and no encryption is performed

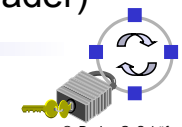


ISAKMP – Basic Message Format (3)

- *Message ID:*
  - Used to identify messages belonging to different exchanges
- *Message Length:*
  - Total length of the message (header + payload)
- *Payload:*
  - The payload of one ISAKMP message can, in fact, contain multiple “chained” payloads
  - The payload type of the first payload in the message is indicated in the next payload field of the ISAKMP header
  - All ISAKMP payloads have a common payload header:

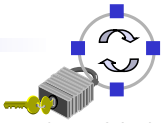


  - *Next Header:* the payload type of the next payload in the message
  - *Payload Length:* total length of current payload (including this header)



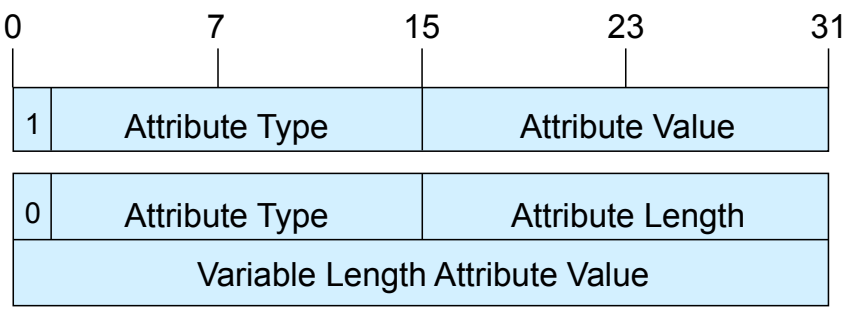
ISAKMP – Limited Denial of Service Protection

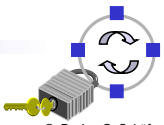
- The initiator and responder cookies also serve as a protection against simple denial of service attacks:
  - Authentication and key exchange often requires “expensive” computations, e.g. exponentiation (for Diffie-Hellman key exchange)
  - In order to avoid, that an attacker can easily flood an ISAKMP entity with bogus messages from forged source addresses and cause these expensive operations, the following scheme is used:
    - The initiating ISAKMP entity generates an initiator cookie:
      $CKY\text{-}I = H(Secret_{Initiator}, Address_{Responder}, t_{Initiator})$
    - The responder generates his own cookie:
      $CKY\text{-}R = H(Secret_{Responder}, Address_{Initiator}, t_{Responder})$
    - Both entities always include both cookies, and always check *their own cookie* before performing any expensive operation
    - The attack mentioned above will, therefore, not be successful as the attacker needs to receive a response from the attacked system in order to obtain a cookie from it
  - ISAKMP does not specify the exact cookie generation method



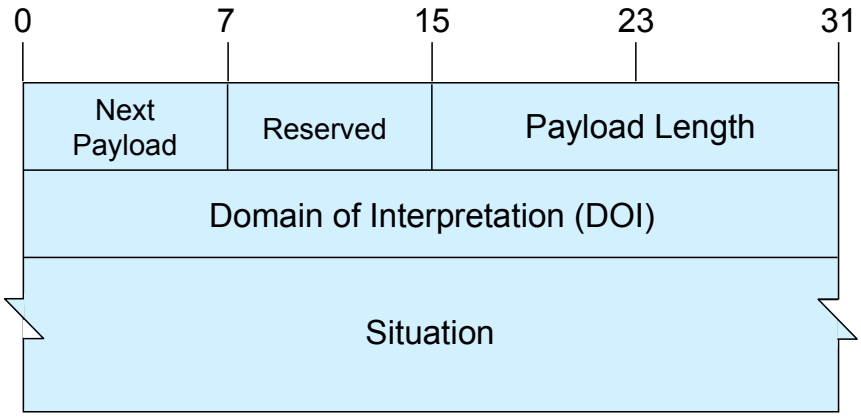
ISAKMP – Payload Types

- RFC 2408 defines various payloads of ISAKMP (list is not exhaustive):
  - Generic payloads: *hash, signature, nonce, vendor ID, key exchange*
  - Specific payloads: *SA, certificate, certificate request, identification*
  - Dependent and encapsulated payloads:
    - *Proposal payload:* describes a proposal for SA negotiation
    - *Transform payload:* describes one transform of a proposal
  - Furthermore, there is one generic *attribute payload:*
    - This is actually not an ISAKMP payload, but a payload which appears inside ISAKMP’s payloads
    - All attribute payloads have a common structure:



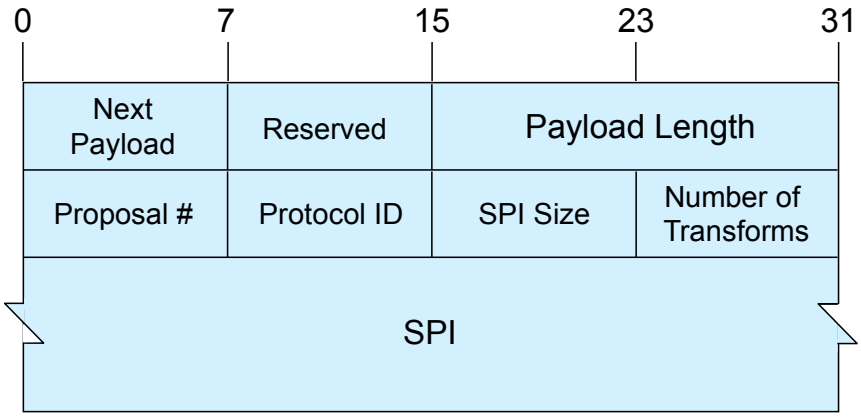


ISAKMP – The Security Association Payload



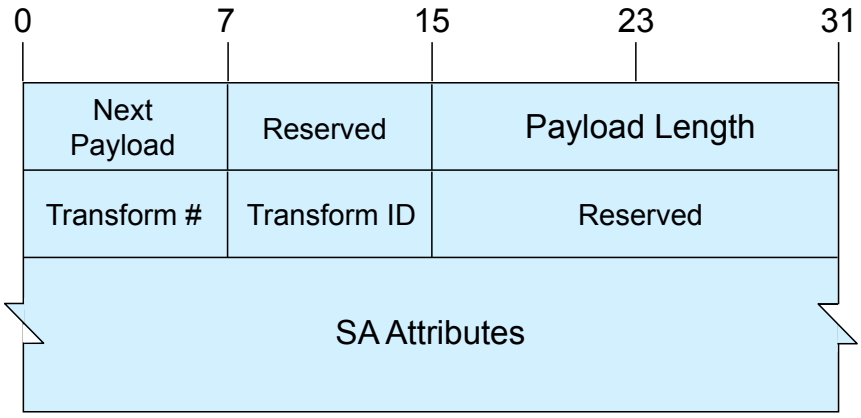
- *Domain of Interpretation* defines the application domain for the SA to be negotiated, e.g. IPsec
- *Situation* is a DOI specific field which identifies the situation under which the current negotiation is taking place (e.g. emergency vs. normal call)
- The SA payload is followed by one or multiple proposal payloads

ISAKMP – The Proposal Payload

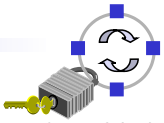


- *Proposal #* is used to express policy and negotiate proposals:
  - If two or more proposals carry the same number this realizes logical AND
  - Different values for Proposal # realize logical OR with descending priority
- *Protocol ID* specifies the protocol identifier of the current negotiation, e.g. AH or ESP (for IPsec)
- *SPI Size* specifies the length of the contained SPI value
- *Number of Transforms* specifies how many transforms belong to this proposal (these immediately follow the proposal payload)

ISAKMP – The Transform Payload



- A transform payload specifies a specific security mechanism, also called transform, to be used to secure the communications channel
- Each transform listed in a proposal has a unique *Transform #*
- Each transform is uniquely identified by a *Transform ID*, e.g. 3DES, AES, MD5, SHA-1, etc.
  - Transform IDs are specified in a DOI document
- The *SA Attributes* specifies attributes as defined for the transform given in the Transform ID field



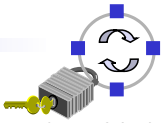
ISAKMP – SA Negotiation (1)

- Content of next payload field of SA, proposal, and transform payloads:
  - The next payload field of an SA payload does not specify the proposal payload which is following immediately, as this is implicit
  - The same applies to proposal and transform payloads
- The proposal payload provides the initiating entity with the capability to present to the responding entity the security protocols and associated security mechanisms for use with the security association being negotiated
- If the SA establishment negotiation is for a combined *protection suite* consisting of multiple protocols, then there must be multiple proposal payloads each with the same proposal number
- These proposals must be considered as a unit and must not be separated by a proposal with a different proposal number

ISAKMP – SA Negotiation (2)

- This first example shows an ESP AND AH protection suite:
  - The first protocol is presented with two transforms supported by the proposing entity, ESP with:
    - Transform 1 as 3DES
    - Transform 2 as AES
    - The responder must select from the two transforms proposed for ESP
  - The second protocol is AH and is presented with a single transform:
    - Transform 1 as SHA
  - The resulting protection suite will be either:
    - 3DES and SHA, or
    - AES and SHA,

    depending on which ESP transform was selected by the responder
  - In this case, the SA payload will be followed by the following payloads:
    - [Proposal 1, ESP, (Transform 1, 3DES, ...), (Transform 2, AES)]
      [Proposal 1, AH, (Transform 1, SHA)]
  - Please remark, that this will result in two SAs per direction!

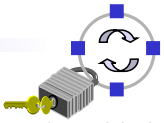


ISAKMP – SA Negotiation (3)

- This second example shows a proposal for two different protection suites:
  - The first protection suite is presented with:
    - one transform (MD5) for the first protocol (AH), and
    - one transform (3DES) for the second protocol (ESP)
  - The second protection suite is presented with two transforms for a single protocol (ESP):
    - 3DES, or
    - AES
  - Please note, that it is not possible to specify that transform 1 and transform 2 have to be used for one instance of a protocol specification
  - In this case, the SA payload will be followed by the following payloads:
    - [Proposal 1, AH, (Transform 1, MD5, ...)]
      [Proposal 1, ESP, (Transform 1, 3DES, ...)]
      [Proposal 2, ESP, (Transform1, 3DES, ...), (Transform 2, AES, ...)]
  - Please note, that proposal 1 results in two SAs per direction.

ISAKMP – SA Negotiation (4)

- When responding to a security association payload, the responder must send a Security Association payload with the selected proposal, which may consist of multiple proposal payloads and their associated transform payloads
- Each of the proposal payloads must contain a single transform payload associated with the protocol
- The responder should retain the *Proposal #* field in the proposal payload and the *Transform #* field in each transform payload of the selected proposal.
  - Retention of proposal and transform numbers should speed up the initiator's protocol processing by avoiding the need to compare the responder's selection with every offered option
  - These values enable the initiator to perform the comparison directly and quickly.
- The initiator must verify that the SA payload received from the responder matches one of the proposals sent initially



ISAKMP – Session Key Establishment

- ISAKMP establishes 4 different keys with an authentication exchange:
  - *SKEYID* is a string derived from secret material known only to the active players in the exchange, serves as a “master key”
    - The computation of SKEYID is dependent on the authentication method
  - *SKEYID_e* is the keying material used by the ISAKMP SA to protect the confidentiality of its messages
  - *SKEYID_a* is the keying material used by the ISAKMP SA to authenticate its messages
  - *SKEYID_d* is the keying material used to derive keys for non-ISAKMP security associations

IKE – Introduction

- Whereas ISAKMP defines the basic data formats and procedures to negotiate arbitrary SAs, the *Internet Key Exchange* specifies the standardized protocol to negotiate IPsec SAs
- IKE defines five exchanges:
  - Phase 1 exchanges for establishment of an IKE SA :
    - *Main mode exchange* which is realized by 6 exchanged messages
    - *Aggressive mode exchange* which needs only 3 messages
  - Phase 2 exchange for establishment of IPsec SAs:
    - *Quick mode exchange* which is realized with 3 messages
  - Other exchanges:
    - *Informational exchange* to communicate status and error messages
    - *New group exchange* to agree upon private Diffie-Hellman groups
- Note: On the following slides HMAC(K, x | y | …) denotes $H(K, p_1, H(K, p_2, x, y, \ldots))$ with p_1 and p_2 denoting padding patterns (cf. chapter 5)

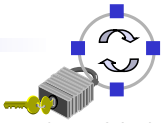
IKE – Computation of IKE Session Keys

- IKE establishes four different keys with an authentication exchange:
  - *SKEYID* is a string derived from secret material known only to the active players in the exchange and it serves as a “master key”
    - The computation of SKEYID is dependent on the authentication method
  - *SKEYID_d* is the keying material used to derive keys for non-IKE SAs
    - $SKEYID\_d = HMAC(SKEYID, g^{xy} \mid CKY\text{-}I \mid CKY\text{-}R \mid 0)$ with g^{xy} denoting the shared Diffie-Hellman secret
  - *SKEYID_a* is the keying material used by the IKE SA to authenticate its messages
    - $SKEYID\_a = HMAC(SKEYID, SKEYID\_d \mid g^{xy} \mid CKY\text{-}I \mid CKY\text{-}R \mid 1)$
  - *SKEYID_e* is the keying material used by the IKE SA to protect the confidentiality of its messages
    - $SKEYID\_e = HMAC(SKEYID, SKEYID\_a \mid g^{xy} \mid CKY\text{-}I \mid CKY\text{-}R \mid 2)$
- If required, keys are expanded by the following method:
  - $K = (K_1 \mid K_2 \mid \ldots)$ with $K_i = HMAC(SKEYID, K_{i-1})$ and $K_0 = 0$

IKE – Authentication Methods

- Phase 1 IKE exchanges are authenticated with the help of two hash values *Hash-I* and *Hash-R*, created by the initiator and responder:
  - *Hash-I* = $HMAC(SKEYID, g^x \mid g^y \mid CKY\text{-}I \mid CKY\text{-}R \mid SA\text{-}offer \mid ID\text{-}I)$
  - *Hash-R* = $HMAC(SKEYID, g^y \mid g^x \mid CKY\text{-}R \mid CKY\text{-}I \mid SA\text{-}offer \mid ID\text{-}R)$

  where g^x, g^y denote the exchanged public Diffie-Hellman values
  ID-I, ID-R denote the identity of the initiator and the responder
  SA-offer denotes the payloads concerning SA negotiation
- IKE supports four different methods of authentication:
  - Pre-shared key:
    - $SKYEID = HMAC(K_{Initiator,Responder}, r_{Initiator} \mid r_{Responder})$
  - Two different forms of authentication with public-key encryption:
    - $SKEYID = HMAC(H(r_{Initiator}, r_{Responder}), CKY\text{-}I \mid CKY\text{-}R)$
  - Digital Signature:
    - $SKEYID = HMAC((r_{Initiator} \mid r_{Responder}), g^{xy})$
    - As in this case *SKEYID* itself provides no authentication, the values *Hash-I* and *Hash-R* are signed by the initiator / responder



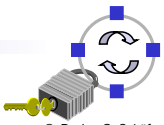
IKE – Main Mode Exchange with Pre-Shared Key

- The following descriptions list the exchanged ISAKMP- and IKE-payloads when performing different “flavors” of IKE authentication:

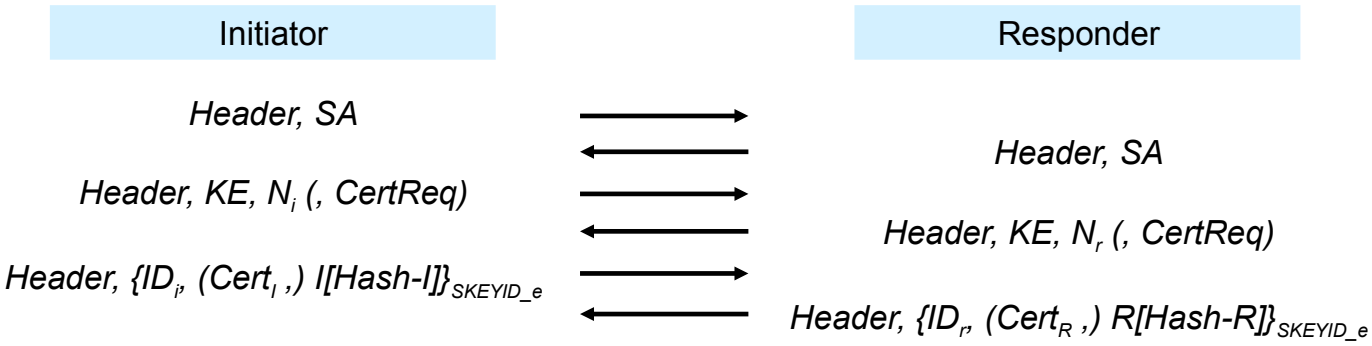
Initiator		Responder
Header, SA	→	
	←	*Header, SA*
Header, KE, N_i	→	
	←	*Header, KE, N_r*
Header, $\{ID_i$, Hash-I$\}_{SKEYID_e}$	→	
	←	*Header, $\{ID_r$, Hash-R$\}_{SKEYID_e}$*

where: N_i, N_r denote $r_{Initiiator}$, $r_{Responder}$ (IKE notation)
ID_i, ID_r denote the identity of the initiator and the responder
KE denotes the public values of a DH-exchange

- Please note that *Hash-I* and *Hash-R* need not to be signed, as they already “contain an authentic secret” (pre-shared key)

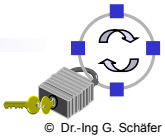


IKE – Main Mode Exchange with Signatures

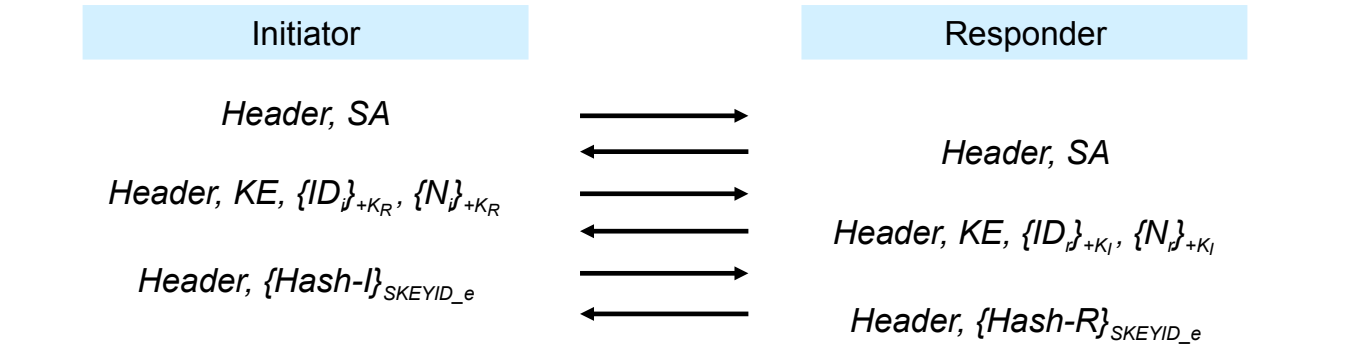


where: *(m)* denotes that *m* is optional
I[m] denotes that *I* signs *m*

- Please note that *Hash-I* and *Hash-R* need to be signed, as they do not contain anything which is known to be authentic

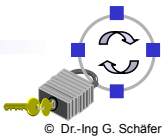


IKE – Main Mode Exchange with Public Key Encryption 1

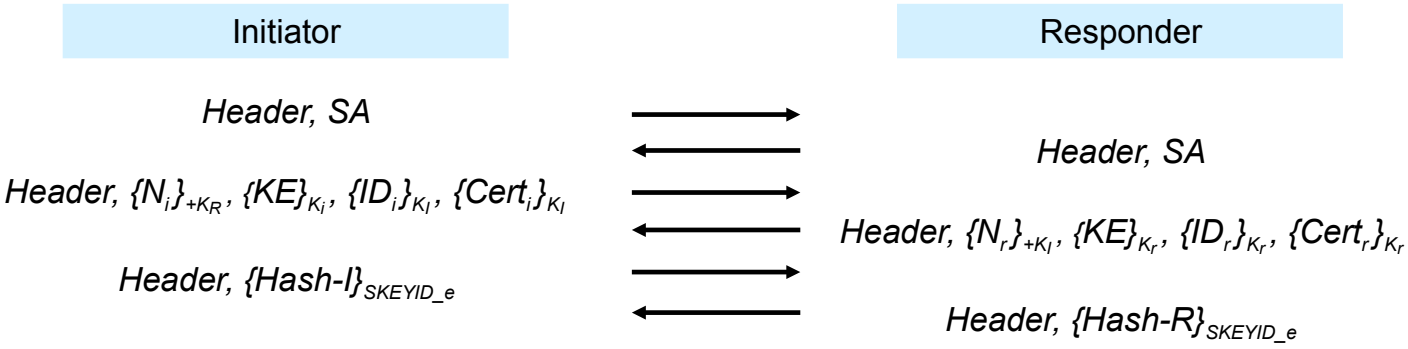


where: $\{m\}_{+K_I}$ denotes that *m* is encrypted with the public key $+K_I$

- Please note that *Hash-I* and *Hash-R* need not to be signed, as they "contain" the exchanged random numbers N_i or N_r, respectively
  - So, every entity proves his authenticity by decrypting the received random number (N_i or N_r) with its' private key



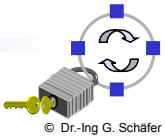
IKE – Main Mode Exchange with Public Key Encryption 2



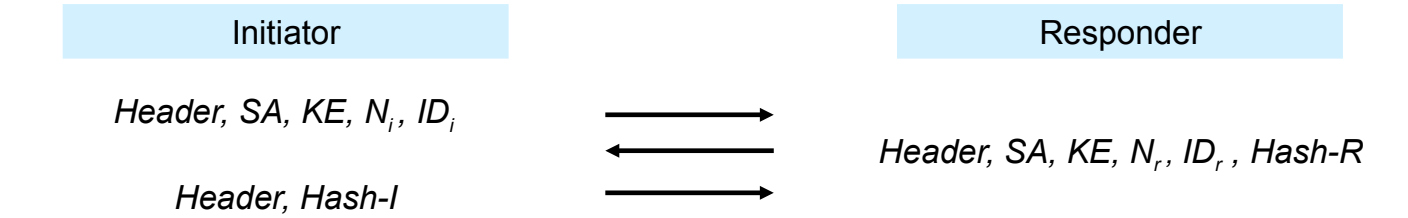
where: $\{m\}_{+K_I}$ denotes that m is encrypted with the public key $+K_I$

$\{m\}_{K_i}$ denotes that m is encrypted with the symmetric key K_i with $K_i = H(N_i, CKY\text{-}I)$ and $K_r = H(N_r, CKY\text{-}R)$

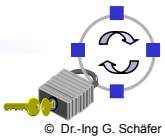
- Please note that all schemes described so far provide protection of identity against eavesdroppers in the Internet, as the IDs and certificates are not send in the clear:
  - However, the IP addresses of exchanged packets are always readable...



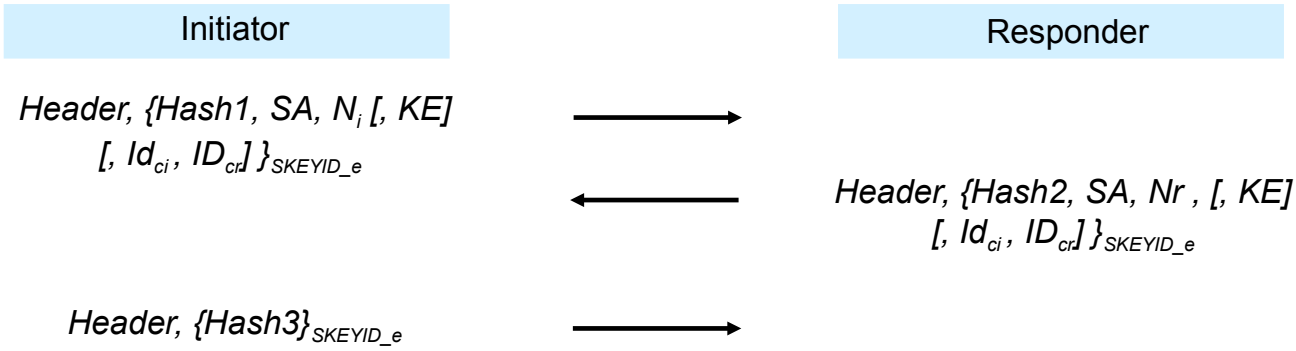
IKE – Aggressive Mode Exchange with Pre-Shared Key



- As the identity of the initiator and the responder have to be send before a session key can be established, the aggressive mode exchange can not provide identity protection against eavesdroppers
- There are similar aggressive mode variants for authentication with:
  - Digital signature
  - Public key encryption



IKE – Quick Mode Exchange



where: *Hash1* = HMAC(SKEYID$_a$, M-ID | SA | N_i | [| KE] [| ID_{ci} | ID_{cr}])
Hash2 = HMAC(SKEYID$_a$, M-ID | N_i | SA | N_r | [| KE] [| ID_{ci} | ID_{cr}])
Hash3 = HMAC(SKEYID$_a$, 0 | M-ID | N_i | N_r)

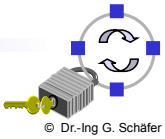
- The optional inclusion of the identities ID_{ci} and ID_{cr} allows to ISAKMP entities to establish an SA on behalf of other clients (gateway scenario)
- The optional key exchange payloads KE allow to perform a new DH-exchange if perfect forward secrecy is desired
- Session Key Material = HMAC(SKEYID_d, [g^{xy} |] protocol | SPI | N_i | N_r)

Further Issues with IPsec

- Compression:
  - If encryption is used, then the resulting IP packets can not be compressed in the link layer, e.g. when connecting to an ISP via Modem
  - Therefore, the *IP payload compression protocol (PCP)* has been defined
  - PCP can be used with IPsec:
    - IPsec policy definition allows to specify PCP
    - IKE SA negotiation allows to include PCP in proposals
- Interoperability problems of end-to-end security with header processing in intermediate nodes:
  - Interoperability with firewalls:
    - End-to-end encryption conflicts with the firewalls' need to inspect upper layers protocol headers in IP packets
  - Interoperability with network address translation (NAT):
    - Encrypted packets do neither permit analysis nor change of addresses
    - Authenticated packets will be discarded if source or destination address is changed

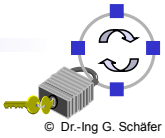
Conclusion

- IPsec is IETF's security architecture for the Internet Protocol
- It provides the following security services to IP packets:
  - Data origin authentication
  - Replay protection
  - Confidentiality
- It can be realized in end systems or intermediate systems:
  - End system implementation: OS integrated or "bump in the stack"
  - Gateway implementation: Router integrated or "bump in the wire"
- Two fundamental security protocols have been defined:
  - Authentication header (AH)
  - Encapsulating security payload (ESP)
- SA negotiation and key management is realized with:
  - Internet security association key management protocol (ISAKMP)
  - Internet key exchange (IKE)



Recent Directions in IPsec Development

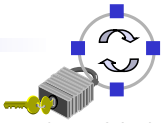
- Internet Key Exchange version 2
  - Based on lessons learned from IKEv1
  - Major simplifications
- Network Address Translation (NAT)
  - Example for issues with NAT and IPsec
  - NAT-Traversal
  - Bound-End-to-End Tunnel Mode (BEET)
- Configuration of large IPsec infrastructures



Internet Key Exchange Protocol version 2 [RFC5996]

Additional design goals to IKEv1

- Consolidation of several IKEv1 RFCs (and several extensions)
  - Makes things easier for developers & testers
  - Clarifies several unspecific points
- Simplifications
  - Number of different key exchanges reduced to one
  - Encryption like in ESP
  - Simple Request/Response mechanism
- Decrease Latency
- Negotiation of traffic selectors
- Graceful changes to allow existing IKEv1 software to be upgraded



IKEv2 – Key Exchange Procedure

Initiator		Responder
Header, g^i, Algorithms, N_i (, CertReq)	→	
	←	*Header, g^r, Algorithms, N_r (, CertReq)*
Header, $\{ID_i$, ($Cert_i$,) SIG, SAEx$\}_K$	→	
	←	*Header, $\{ID_r$, ($Cert_r$,) SIG, SAEx$\}_K$*

where: K key derived by PRF(PRF(N_i || N_r, g^{ir}), N_i || N_r || SPI_i || SPI_r)

PRF "some" pseudo-random function – usually an HMAC

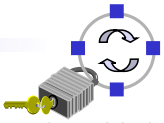
SIG asymmetric signature or MAC over the first two messages

SAEx a piggybacked "Quick-Mode-Exchange"

- Only a single exchange type
- Four messages exchanged (= 2 * RTT)
- Initiator triggers all retransmissions

IKEv2 – Properties of the Key Exchange Procedure

- First SA exchange is piggybacked
  - Lower latency, as it saves one RTT
- Message 4 was discussed to be piggybacked to message 2, but
  - Message 3 verifies that initiator received message 2 (SPI ~ Cookie)
    - Serves as a DoS protection if computational intensive tasks are performed afterwards
  - Identity of responder only disclosed after verification of initiator
    - Protects from scanning for a party with a specific ID
  - Initiator would not know when it is safe to send data
    - (Packets may be received out of order)
  - Would require more complicated retransmission strategy
  - Responder cannot decide on a policy for the child SA

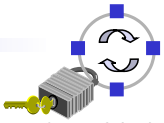


IKEv2 – Additional Features

- Further DoS Protection
  - In case of DoS attack Responder may require Initiator to send a stateless cookie
  - Adds 2 extra messages to the exchange
- Dead Peer Detection
  - Periodic IKE requests to determine whether SA can be deleted
- More flexible negotiation techniques
  - Ability to state: “use one of these ciphers with one of these authentication algorithms” (no more need to enumerate all combinations)
  - Traffic selectors may be narrowed down
    - Initiator: “I want to use 192.168.0.0/16 for my tunnel mode”
    - Responder: “OK, but you may only use 192.168.78.0/24”
    - Can be used to have responder assign address range to initiator (do without / help DHCP in simple situations; see also below)

Network Address Translation (NAT)

- Common problem nowadays: ISP provides only a single IP address, but multiple devices shall be connected
- Solution: A router is used to map several internal (private) addresses to a single external (public) address
- Most common approach (simplified):
  - For packets coming from the private side:
    - Router rewrites TCP/UDP source ports to unique value per IP flow
    - Stores the new source port in a table with the source address and old source port
    - Replaces source IP address with the external address
  - For packets coming from the public side:
    - Router looks up IP flow by TCP/UDP destination port
    - Replaces destination address and port to the old values



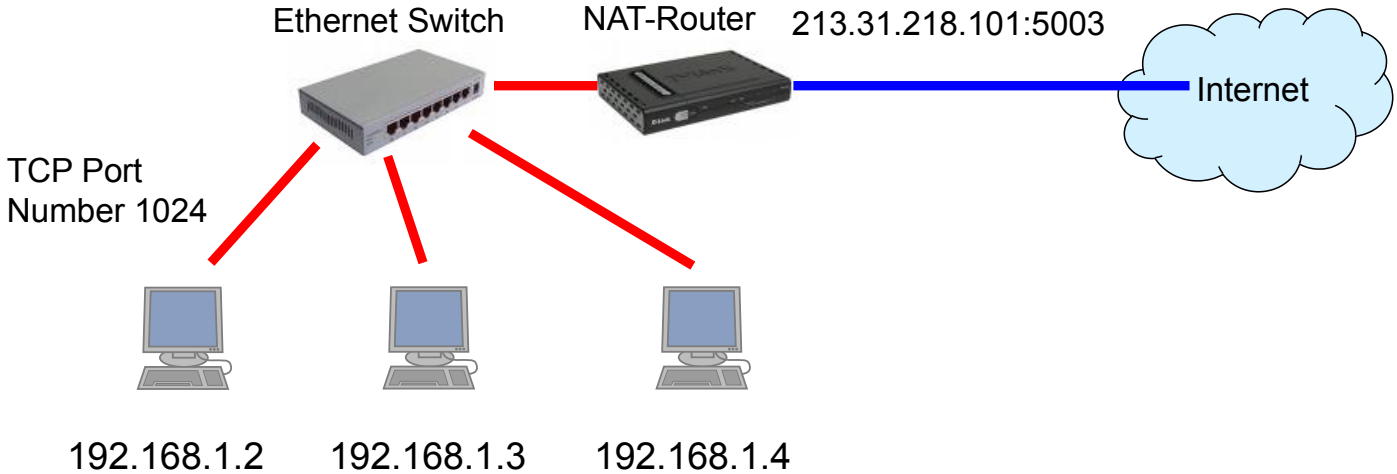
NAT – An Example

NAT changes the sources address of each packet to a public IP address with different („rewritten“) source ports

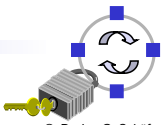
213.31.218.101:5001

213.31.218.101:5002

213.31.218.101:5003

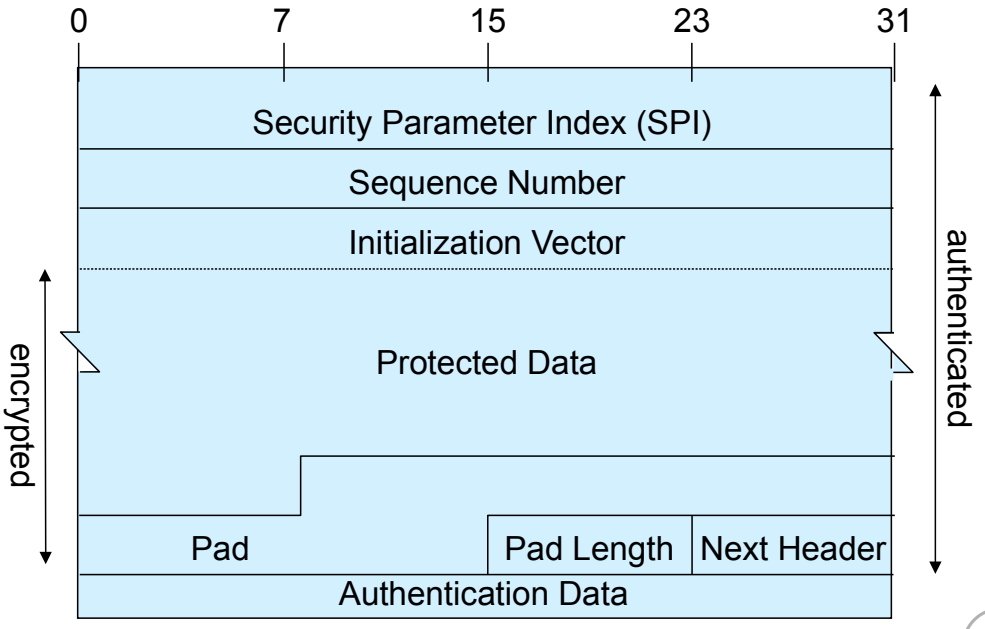


Private IP addresses on internal Network



Issues with NAT and IPsec - NAT-Traversal (I)

- Problems:
  - AH cannot be used with NAT by definition
  - ESP does not offer any “rewritable field” (like port number)
  - TCP/UDP port numbers are encrypted or authenticated (or both)



Issues with NAT and IPsec - NAT-Traversal (II)

- Solution for ESP: Encapsulate ESP packets in regular UDP packets [RFC3948]

NAT-T packet structure – Transport and Tunnel Mode

IP header	UDP header	ESP header	protected data

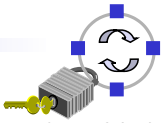
IP header	UDP header	ESP header	IP header	protected data

- UDP header only contains port numbers and empty checksum
  - Adds 8 byte overhead
  - Only purpose: give the NAT device something to “rewrite” (in order to be able to distinguish recipients of packets in response)
  - Port 4500 reserved for NAT-T (NAT-Traversal)
- In Transport mode:
  - Inner UDP/TCP checksum depends on original source address (layering violation in original TCP/IP suite)
  - Must be recovered

Issues with NAT and IPsec - NAT-Traversal (III)

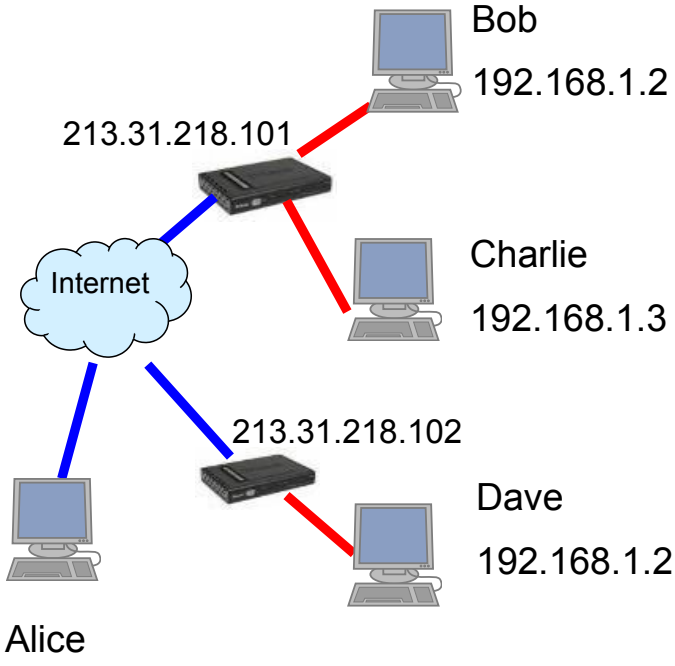
- When to use NAT-T?
  - NAT situation must be detected by IKE
  - Done by IKEv1 extension [RFC3947] and IKEv2
  - IKE uses NAT-T if the IKE source port is not 500
  - Not always working, then manual configuration is required

- Timeout issues and keep-alives
  - ESP packets are not periodically exchanged
  - NAT-T flows may timeout in router
  - Inbound packets can then not be delivered
  - Periodic keep-alive packets make sure router keeps state
  - Simple UDP packet on port 4500 containing a single 0xFF octet



Issues with NAT and IPsec – BEET-Mode (I)

- Which addresses shall Alice use to send packets to Bob, Charlie, and Dave?
- Neither the external nor the internal addresses must be unique!
  - Bob's and Charlie's packets both have the same external address
  - Bob's and Dave's packets both have the same internal adress
- Using either internal or external addresses is insecure (Why?)
- Distinguishing requires virtual addresses…

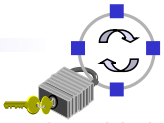


Issues with NAT and IPsec – BEET-Mode (II)

- Assigning or negotiating virtual IP addresses
  - Alice needs to assign unique virtual addresses to each of her peers
  - Can be done manually, or
  - By DHCP over IKE, or
  - By negotiation of traffic selectors (IKEv2)
  - Running L2TP over IPsec
- IPsec Tunnel Mode is required
  - External IP Header carries either public IP address, or private NAT address
  - Internal IP Header carries virtual IP address
  - Leads to (at least!) 28 bytes overhead per packet in NAT situations

IP header	UDP header	ESP header	IP header	protected data

- But actually only address fields in the inner IP header required (all other fields could be derived from external header)
- Both virtual address fields always use the same addresses (no multiplexing as in usual tunnel mode scenarios)



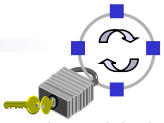
Issues with NAT and IPsec – BEET-Mode (III)

- The restriction to two addresses in tunnel allow a static binding during IKE negotiation
- The Bound-End-to-End-Tunnel (BEET) Mode [NiMe08] behaves semantically like a Tunnel Mode association with a traffic selector for a single host (/32)
- The transmitted ESP packets are equivalent to Transport (!) Mode packets (virtual addresses are never transmitted in packets)
- Inner header is recovered by ESP's decapsulation process

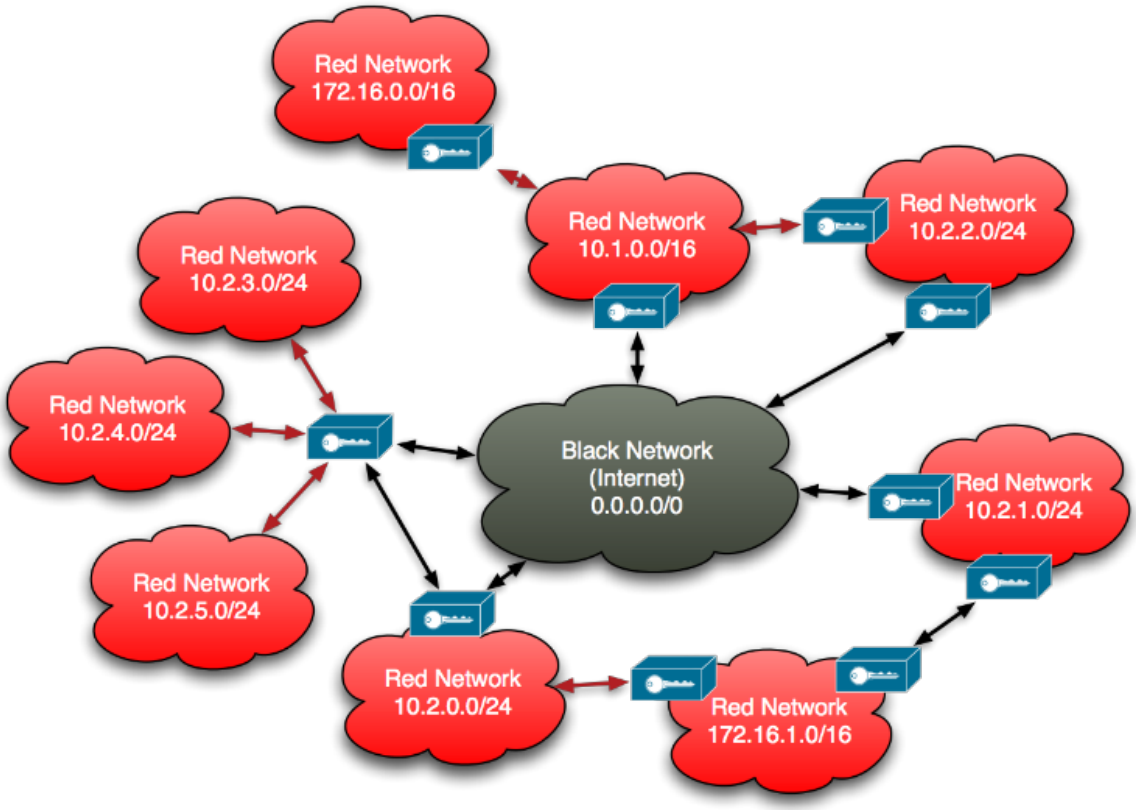
- Distinguishes between the reachability of a host (external IP address) and its identity (virtual IP address)
- Hosts may now roam between locations and keep their virtual IP address (additionally allows for better mobility support)

Configuration of Large IPsec Infrastructures

- Communication infrastructures of companies and authorities:
- May form complex overlay topologies
  - Nested
  - Cycles
  - Multiple security gateways per private network
  - Multiple private networks per gateway
  - Private address ranges in private networks
  - QoS and secure IP multicast may be required
- May have up to thousands of security gateways
- May be dynamically changing
  - Addition and removal of security gateways
  - Link and node failures
  - Denial of service attacks
  - Mobile security gateways (e.g. in disaster communication)
- Must be secure of course …

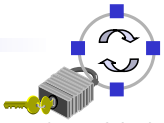


Example of an IPsec Infrastructure



Problems of Manual IPsec Infrastructure Configuration

- IETF did not define way to automatically configure and deploy IPsec in large scenarios
- Thus security gateways usually configured manually
  - Number of security policy entries grows quadratically to the number of security gateways
  - Scalability problem
    - Administration effort grows
    - ➡Expenses grow
    - Administrators potentially make more configuration errors, e.g. forget to delete an entry from an SPD or allow too large IP range, etc.
    - ➡Possible security problems
  - Agility problem
    - No dynamic adjustment of VPN topology
    - Limited support of mobile security gateways



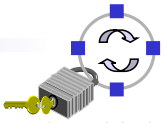
Automatic IPsec Configuration - Some Requirements

- Functional requirements
  - Must minimize manual intervention
  - Must support even complex infrastructures (nested topologies with private address ranges etc.)
  - Must use unicast only (as multicast etc. is not widely deployed)

- Non-functional requirements
  - Must be robust, thus react stable to severe network conditions
  - Must be secure, especially it must not be weaker than a manually configured IPsec infrastructure
  - Must be scalable with regards to the number of security gateways
  - Must be able to quickly adapt to new topologies

Different Approaches for Automatic IPsec Configuration

- IPsec policy distribution through central servers
- Group Encrypted Transport VPN (GET)
- Tunnel Endpoint Discovery (TED)
- Dynamic Multipoint VPN (DMVPN)
- Proactive Multicast-Based IPsec Discovery Protocol
- Social VPN
- Secure OverLay for IPsec Discovery (SOLID)

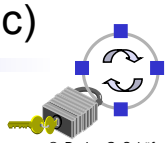


IPsec Policy Distribution Through Central Servers

- Straightforward, common approach to configure large numbers of security gateways
- Central policy server statically configured in each gateway
- Each gateway contacts policy server to update SPD

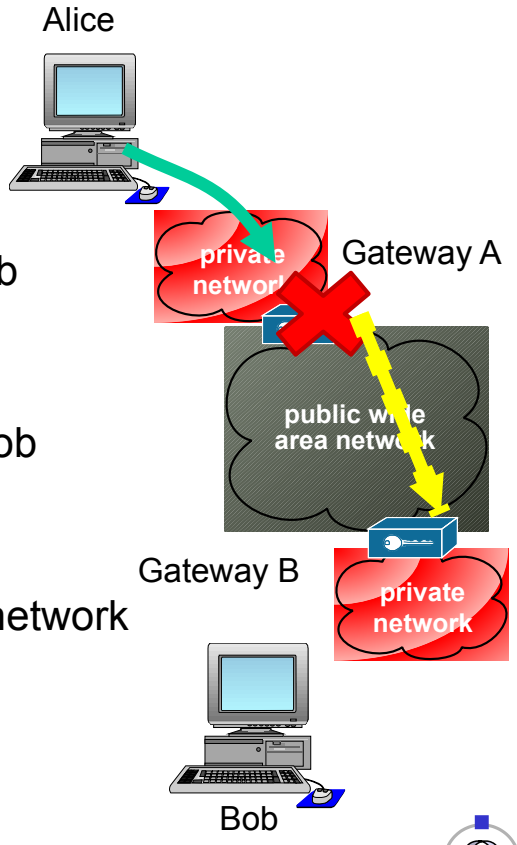
- Example: Microsoft Active Directory, several military products

- Some obvious problems:
  - Administrators need to manually edit central database
  - Nested topologies difficult to realize
  - Scalability problems due to bottleneck
  - Availability hard to guarantee (single point of failure)
  - Dynamic topologies require new policies to be proactively pushed to security gateways (even though they might not be used currently)
  - Many policy entries most probably will never be used (no traffic)



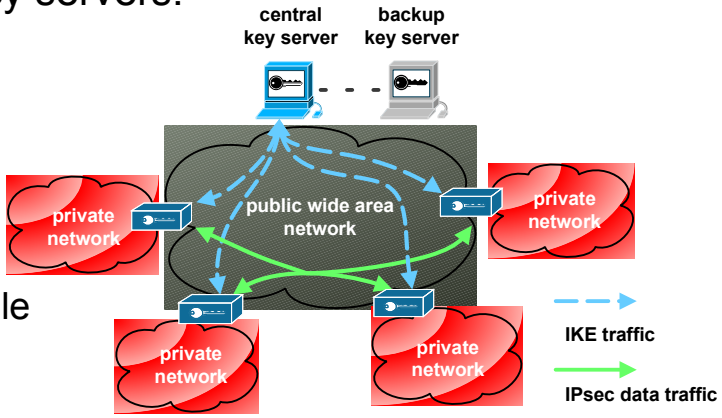
Tunnel Endpoint Discovery (TED)

- Proprietary Cisco approach [Fluh01]
- Security associations reactively created
  - Alice sends packet to Bob
  - Gateway A detects no valid SA present
  - Drops packet and sends IKE packet to Bob
  - Gateway B intercepts IKE packet
  - Establishes SA to gateway A
  - Subsequent packets between Alice and Bob can be transmitted
- Rather powerful, secure approach, but
  - Routing has to be performed in transport network
  - No private IP address ranges
  - No nested topologies



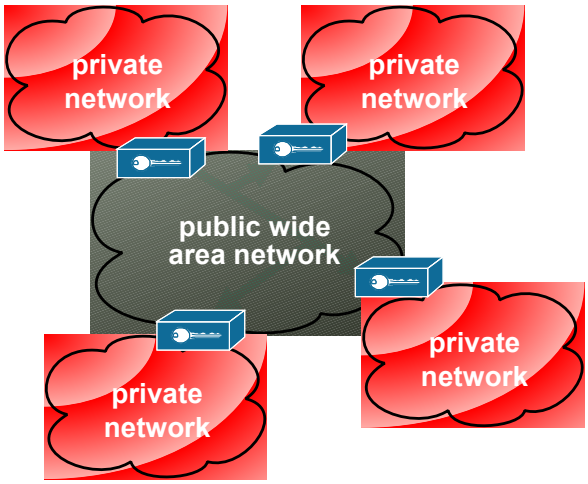
Group Encrypted Transport VPN (GET)

- Cisco product branding of several IPsec components [Bhai08]
- Security gateways contact central IKE server
- IKE server distributes symmetric keys (preferentially via multicast)
- All security gateways of a group use same SA (including SPI, keys)
- Replay protection by time window (1-100 seconds)
  - Sliding window mechanism does not work as multiple senders use same SPI
- Additional Problems to central policy servers:
  - weak replay protection
  - Compromise of a single gateway compromises whole VPN
  - Rekeying performed by symmetric exchanges ➡ cannot recover from compromised keys
  - Perfect forward secrecy not available
- Only advantage: Allows multicast



Proactive Multicast-Based IPsec Discovery Protocol

- Approach developed for military applications [Tran06]
- Security gateways periodically announce private networks
- Done by transport network multicast
- Messages protected by pre-shared symmetric key
- Advantages: Supports private address ranges, multicast within VPN
- Problems:
  - Requires transport network multicast
  - Nested topologies not working
  - Number of received messages may be rather large
  - Compromised gateway leads to non-recoverable compromise of VPN
  - Replay protection not addressed

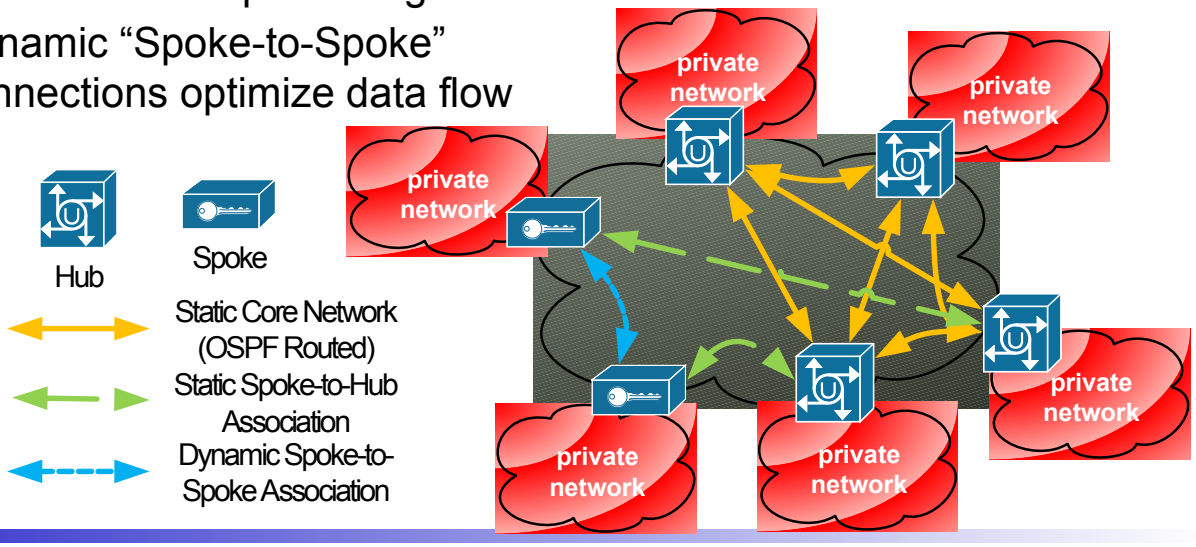


Social VPN

- Academic approach [FBJW08]
- Uses Facebook as “policy” server to exchange IKE certificates
  - One may communicate with friends
- Agility provided by peer-to-peer network
  - Looks up targets external IP address in distributed hash table
- Problems
  - No gateway functionality (only end-to-end)
  - No nested topologies
  - Rather large packet overhead
  - Bad scalability in case of many potential communication partners
  - Security
    - Do you trust Facebook?
    - Do you know, if the person in Facebook is really who it claims?
    - No verification available at all

Dynamic Multipoint VPN (DMVPN)

- Another approach by Cisco [Bhai08]
- VPN is split in
  - Static core gateways (“Hubs”)
  - Dynamic peripheral gateways (“Spokes”)
- Hubs may use OSPF routing between each others
- Spokes contact pre-configured Hubs to access VPN
- Dynamic “Spoke-to-Spoke” connections optimize data flow

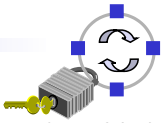


Dynamic Multipoint VPN (DMVPN) – Discussion

- Advantages
  - Approach allows for more dynamic topologies
  - Can use private addresses
- Disadvantages
  - Still requires substantial configuration effort
    - Core Network configured manually
    - Spokes must be configured with addresses of Hubs
    - Makes for example simple switching to new ISP impossible
  - Spokes cannot be nested
  - Spokes cannot move between “Hubs”
    - Hub behaves like MobileIP home agent for spoke
  - Failure of “Hubs” critical for their “Spokes”

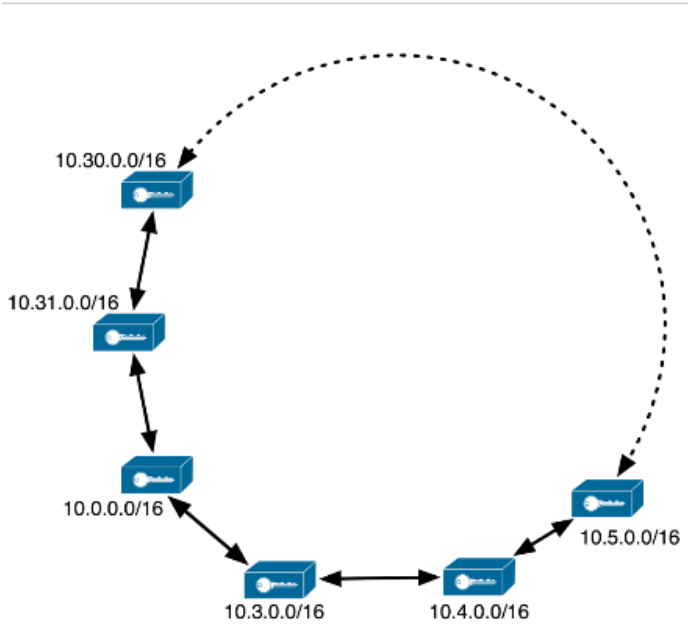
Secure OverLay for IPsec Discovery (SOLID)

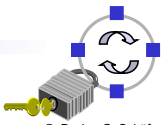
- Complex approach, promises simple deployment [RSS10]
- Security gateways form a structured overlay network
  - Interconnects security gateways so that the VPN can efficiently be searched for a target address
- Requires only very few proactively created IPsec associations
  - Minimal connectivity allows for reactive discovery of security gateways
  - Moving security gateways do not have to inform all others of current external IP address
- Three tasks to perform
  - Topology control
    - Proactively create a VPN structure to perform fast discovery
  - Discovery of security gateways
    - Every time a client computer sends a packet and no valid SA is found
    - Must find corresponding security gateway to create SA reactively
  - Routing of data packets
    - Find an efficient way to forward packets through the overlay



SOLID – Topology Control

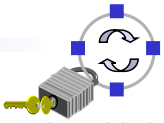
- Topology control mechanisms
  - Continuously update structure of VPN to adapt to changes
- In SOLID proactively creates SAs to form an artificial ring structure
- Security gateways are ordered by inner addresses
- Gateways that cannot directly communicate in transport network are connected by virtual paths
- ➡ Nested structures are flattened to allow for easy discovery





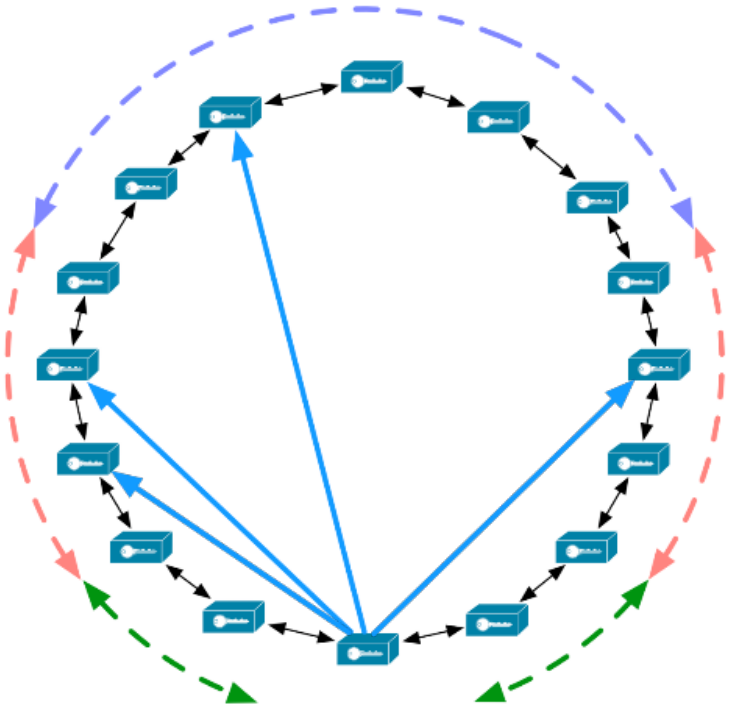
SOLID – Discovery

- Reactive discovery to find security gateway for a specific client IP address
- Search requests are forwarded to the (already associated) gateway whose inner IP address is “most similar” to the searched IP address
  - Simple mechanism assures correct corresponding security gateway is found
  - Packets will be sent along the ring structure
  - Needs O(n) overlay hops to reach target (where n is number of networks in VPN topology)
- ➡ Shorter “search paths” required



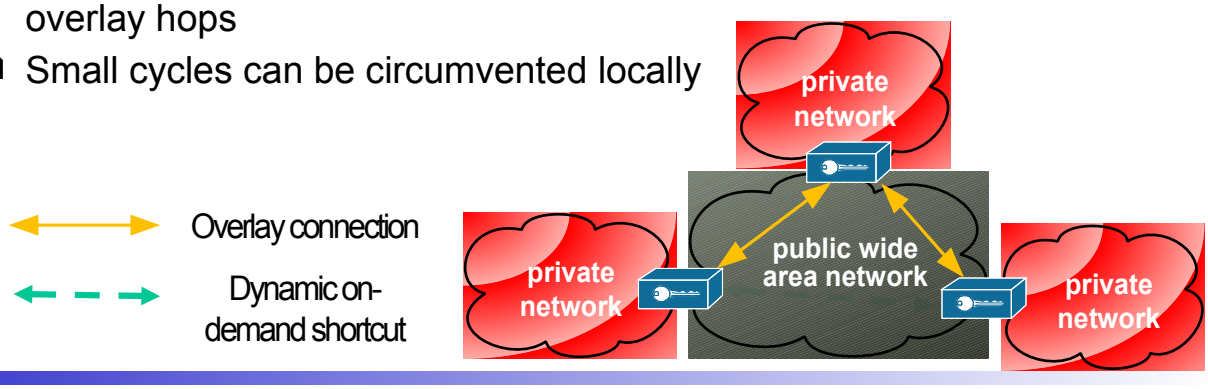
SOLID – More Topology Control

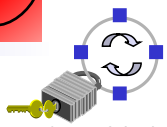
- More advanced topology control creates additional SAs
- IP address space of VPN is divided in ranges
  - Exponentially growing sizes of ranges
- To each range at least one SA is kept proactively by each gateway
- Number of additional SAs grows in O(log n)
- Due to construction technique discovery in O(log n) overlay hops
- ➡ Approach scales well with number of networks



SOLID – Routing of Data Packets

- After initial discovery data packets must be forwarded
- Sending data along discovery path possible
  - Length again O(log n) overlay hops
  - Too inefficient if many packets must be routed
  - Used initially only
- Subsequently path is optimized
  - Optimization done if gateway detects that it forwards packets for two gateways that are within the same network
  - In cycle-free VPNs leads to optimal routes with regards to number of overlay hops
  - Small cycles can be circumvented locally

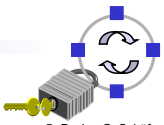




SOLID – Properties and Results

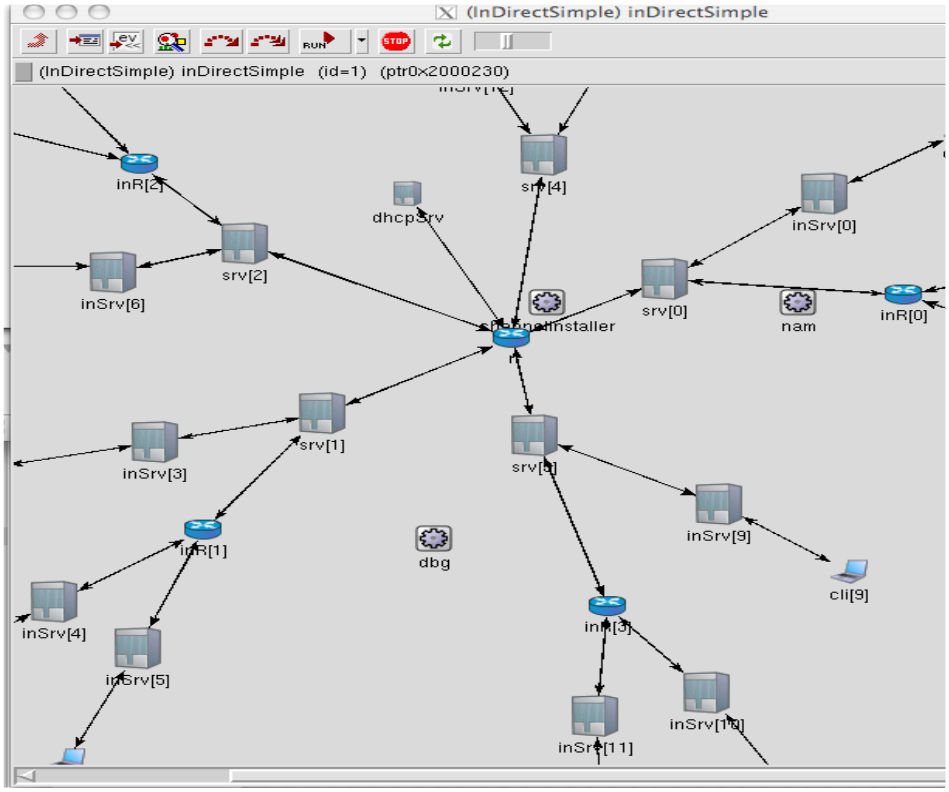
- Can configure complex infrastructures within seconds or minutes
- Requires no manual interaction
- Requires no special transport network features

- Robustness
  - No single point of failure
  - If network is split, parts can work independently
- Does not weaken security offered by standard IPsec
- Scales well with number of private networks, no bottlenecks
- If security gateways move, only two SAs must be re-established to ensure reachability



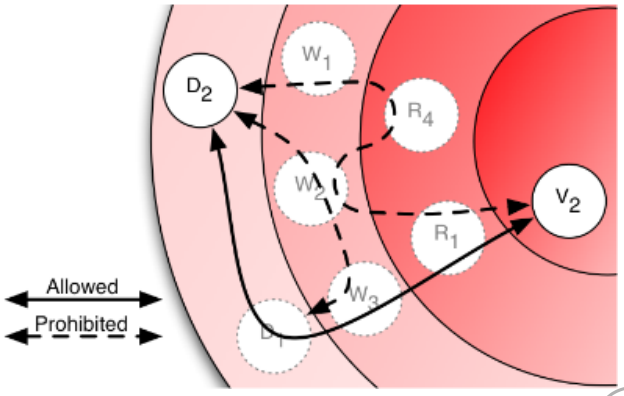
SOLID – Simulative Evaluation

- SOLID can be evaluated in OMNeT++
- Allows for tests of complex scenarios



SOLID – Other Research

- SOLID under research in Telematics/Computer Networks Group
- Prototype development
- Availability
  - Protect more important core network from DoS attacks
  - Creation of layered VPN, prevents certain traffic flows between security gateways
- Access control
- Robustness
  - Proactive recovery from network failur
- Application Layer Multicast
  - Allows secure multicast via unicast-only networks



Additional References

[RFC2401] R. Atkinson, S. Kent. *Security Architecture for the Internet Protocol.* RFC 2401, Internet Engineering Taskforce (IETF), 1998.

[RFC2402] R. Atkinson, S. Kent. *IP Authentication Header (AH).* RFC 2402, IETF, 1998.

[RFC2403] C. Madson, R. Glenn. *The Use of HMAC-MD5-96 within ESP and AH.* RFC 2403, IETF, 1998.

[RFC2404] C. Madson, R. Glenn. *The Use of HMAC-SHA-1-96 within ESP and AH.* RFC 2404, IETF, 1998.

[RFC2405] C. Madson, N. Doraswami. *The ESP DES-CBC Cipher Algorithm With Explicit IV.* RFC 2405, IETF, 1998.

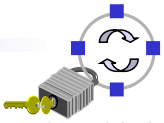
[RFC2406] R. Atkinson, S. Kent. *IP Encapsulating Security Payload (ESP).* RFC 2406, IETF, 1998.

[RFC2407] D. Piper. *The Internet IP Security Domain of Interpretation for ISAKMP.* RFC 2407, IETF, 1998.

[RFC2408] D. Maughan, M. Schertler, M. Schneider, J. Turner. *Internet Security Association and Key Management Protocol (ISAKMP).* RFC 2408, IETF, 1998.

[RFC2409] D. Harkins, D. Carrel. *The Internet Key Exchange (IKE).* RFC 2409, IETF, 1998.

[RFC2857] A. Keromytis, N. Provos. *The Use of HMAC-RIPEMD-160-96 within ESP and AH.* RFC 2857, IETF, 2000.



TELEMATIK Rechnernetze

Additional References

[RFC3947] T. Kivinen, B. Swander, A. Huttunen, V. Volpe: *Negotiation of NAT-Traversal in the IKE.* RFC 3947, IETF, 2005.

[RFC3948] A. Huttunen, B. Swander, V. Volpe, L. DiBurro, M. Stenberg: *UDP Encapsulation of IPsec ESP Packets.* RFC 3948, IETF, 2005.

[RFC4306] C. Kaufman: *Internet Key Exchange (IKEv2) Protocol.* RFC 4306, Internet Engineering Taskforce (IETF), 2005.

[NiMe08] P. Nikander, J. Melen: *A Bound End-to-End Tunnel (BEET) mode for ESP.* Internet-Draft, IETF Network Working Group, 2008.

[Bhai08] Y. Bhaiji: *Network Security Technologies and Solutions*, Cisco Press, 2008.

[Fluh01] S. Fluhrer: *Tunnel Endpoint Discovery.* Expired Internet-Draft, IETF IPSP Working Group, 2001.

[Tran06] T.H. Tran: *Proactive Multicast-Based IPSEC Discovery Protocol and Multicast Extension.* Military Communications Conference, 2006.

[FBJW08] R. Figueiredo, P. O. Boykin, P. St. Juste, D. Wolinsky: *Social VPNs: Integrating Overlay and Social Networks for Seamless P2P Networking'.* IEEE WETICE/COPS, 2008.

[RSS10] M. Rossberg, T. Strufe, G. Schaefer: *Distributed Automatic Configuration of Complex IPsec-Infrastructures.* Journal of Network and Systems Management, Volume 18, Issue 3, pp. 300-326, 2010.

[RSSM09] M. Rossberg, W. Steudel, G.Schaefer, M. Martius: *Eine Software-Architektur zur Konstruktion flexibler IPsec-Infrastrukturen.* BSI 11. Deutscher IT-Sicherheitskongress, 2009.