

The IPsec protocols

Security Protocols (bmevihim132)

Dr. Levente Buttyán
associate professor

BME Híradástechnikai Tanszék
Lab of Cryptography and System Security (CrySyS)
buttyan@hit.bme.hu, buttyan@crysys.hu



Outline

- components and services
- modes of operation
- Security Associations
- Authenticated Header (AH)
- Encapsulated Security Payload (ESP)
- Internet Key Exchange (IKE) version 2





Overview

- IPsec is an Internet standard for network layer security
 - provides protection for IP and protocols above (ICMP, TCP, ...)
 - allows selection of the required security services and algorithms
 - puts in place the necessary cryptographic keys
 - can be applied between a pair of hosts, between a pair of security gateways (e.g., firewalls), and between a host and a gateway
- components:
 - an authentication protocol (Authentication Header – AH)
 - a combined encryption and authentication protocol (Encapsulated Security Payload – ESP)
 - Security Association and key establishment protocol (IKEv2)
- possible ways to implement IPsec:
 - integration into the native IP stack implementation
 - bump-in-the-stack (BITS): between IP and the network driver
 - bump-in-the-wire (BITW): a separate HW device (security gateway)



IPsec services

	AH	ESP (encryption only)	ESP (encryption and authentication)
integrity	✓		✓
data origin authentication	✓		✓
replay detection	✓	✓	✓
confidentiality		✓	✓
limited traffic flow confidentiality		✓	✓



Modes of operation

- transport mode
 - provides protection primarily for upper layer protocols
 - protection is applied to the payload of the IP packet
 - ESP in transport mode encrypts and optionally authenticates the IP payload but not the IP header
 - AH in transport mode authenticates the IP payload and selected fields of the IP header
 - typically used between end-systems
- tunnel mode
 - provides protection to the entire IP packet
 - the entire IP packet is considered as payload and encapsulated in another IP packet (with potentially different source and destination addresses)
 - ESP in tunnel mode encrypts and optionally authenticates the entire inner IP packet
 - AH in tunnel mode authenticates the entire inner IP packet and selected fields of the outer IP header
 - usually used between two security gateways, or between a host and a security gateway



Security Associations (SA)

- the IPsec state shared by two systems (hosts, gateways) is represented by SAs
- an SA is always one-way, and it is either created for AH or for ESP (or IKE)
- an SA includes the following information:
 - end-point identifiers (e.g., IP addresses)
 - protocol information (algorithms, keys, and related parameters)
 - operational mode (tunnel or transport)
 - sending sequence number (for the sender of this SA)
 - anti-replay window (for the receiver of this SA)
 - a counter and a bit-map (or equivalent)
 - used to determine whether an inbound packet is a replay
 - lifetime (a time interval or byte count)
 - ...
- SAs are referenced by an SPI (Security Parameter Index)
 - a bit string carried in AH and ESP headers to allow the receiving party to select the SA which must be used to process the packet

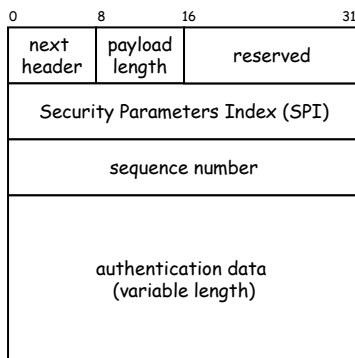


Selecting SAs

- Security Policy Database (SPD)
 - two separate databases for inbound and outbound traffic
 - each entry defines a subset of IP traffic and a set of SAs to be applied for that traffic
 - subset of IP traffic is defined in terms of selectors (header fields)
 - destination IP address (single, enumerated list, range, or mask)
 - source IP address (single, enumerated list, range, or mask)
 - transport layer protocol (single, enumerated list, or range)
 - source or destination port (single, enumerated list, range, or wildcard)
 - ...
- outbound processing
 - compare the selector fields of the packet to the values in the SPD
 - determine which SAs should be used for the packet and their SPIs
 - do the required IPsec processing
- inbound processing
 - retrieve the SA that should be used based on the SPI in the inbound packet
 - do the IPsec processing
 - verify in the SPD if the right SAs have been applied

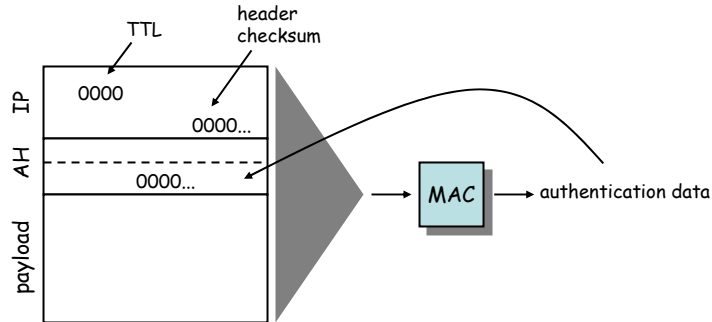


Authentication Header – AH



- next header
 - type of header immediately following this header (e.g., TCP, IP, etc.)
- payload length
 - length of AH (in 32 bit words) minus 2
 - e.g., 4 if Authentication data is 3x32 bits long
- Security Parameters Index
 - identifies the SA used to generate this header
- sequence number
 - sequence number of the packet
- authentication data
 - a (truncated) MAC (default length is 3x32 bits)

- the MAC is calculated over
 - IP header fields that do not change in transit
 - the AH header fields (authentication data field is set to 0)
 - entire upper layer protocol data
- the fields not covered by the MAC are set to 0 for the calculation



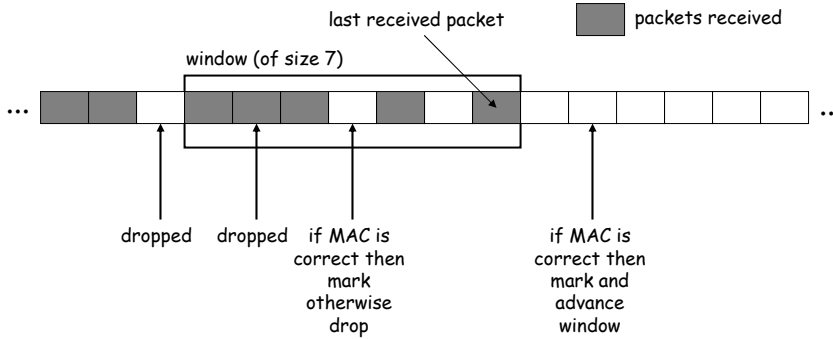
<u>Requirement</u>	<u>Algorithm</u> (notes)
MUST	HMAC-SHA1-96 [RFC2404]
SHOULD+	AES-XCBC-MAC-96 [RFC3566]
MAY	HMAC-MD5-96 [RFC2403] (1)

Note:

(1) Weaknesses have become apparent in MD5; however, these should not affect the use of MD5 with HMAC.

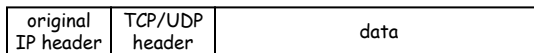
Replay detection

- replay: the attacker obtains an authenticated packet and later transmits (replays) it to the intended destination
- receiver has an anti-replay window of default size $W = 64$

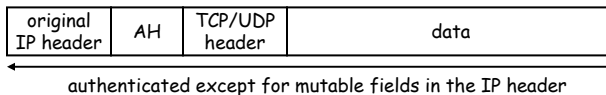


AH in transport and tunnel mode

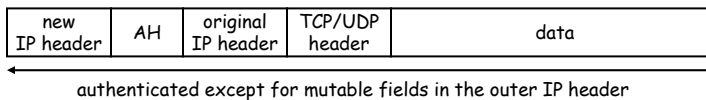
original IPv4 packet



AH in transport mode

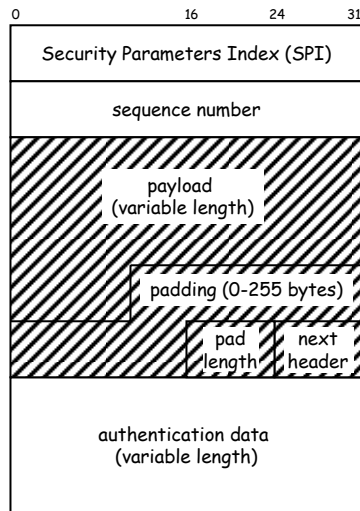


AH in tunnel mode





Encapsulating Sec Payload – ESP



- Security Parameters Index
 - identifies the SA used to generate this encrypted packet
- sequence number
- payload
 - transport level segment (transfer mode) or encapsulated IP packet (tunnel mode)
- padding
 - variable length padding
- pad length
- next header
 - identifies the type of data contained in the payload
- authentication data
 - a (truncated) MAC computed over the ESP packet (SPI ... next header)



Encryption and MAC

- encryption
 - applied to the payload, padding, pad length, and next header fields
 - if an IV is needed, then it is explicitly carried at the beginning of the payload data (the IV is not encrypted)
- MAC
 - default length is 3x32 bits
 - MAC is computed over the SPI, sequence number, and encrypted payload, padding, pad length, and next header fields
 - unlike in AH, here the MAC does not cover the preceding IP header



Supported algorithms

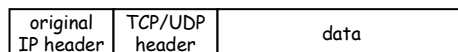
<u>Requirement</u>	<u>Encryption Algorithm</u>
MUST	NULL
MUST-	TripleDES-CBC [RFC2451]
SHOULD+	AES-CBC with 128-bit keys [RFC3602]
SHOULD	AES-CTR [RFC3686]
SHOULD NOT	DES-CBC [RFC2405]

<u>Requirement</u>	<u>Authentication Algorithm</u>
MUST	HMAC-SHA1-96 [RFC2404]
MUST	NULL
SHOULD+	AES-XCBC-MAC-96 [RFC3566]
MAY	HMAC-MD5-96 [RFC2403]



ESP in transport and tunnel mode

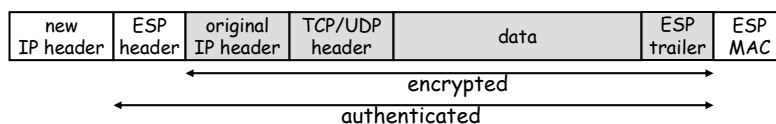
original IPv4 packet



ESP in transport mode

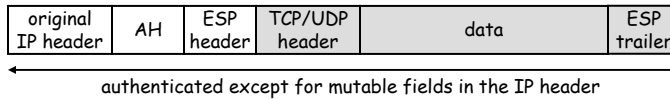


ESP in tunnel mode



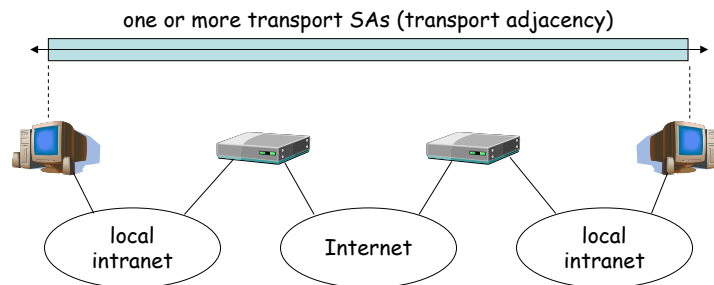
Combining SAs

- transport adjacency (basic ESP-AH combination)
 - first apply ESP in transport mode without authentication
 - then apply AH in transport mode

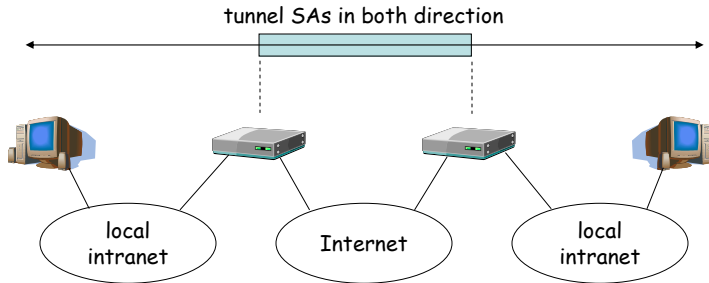


- iterated tunneling (multiple nested tunnels)
 - both end-points of the two tunnels are the same
 - one end-point of the two tunnels is the same
 - neither endpoint of the two tunnels is the same
- transport within tunnel

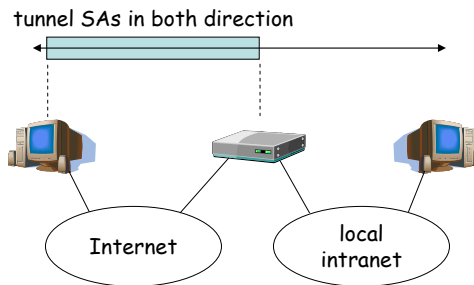
Application: secure e2e com



Application: corporate VPN

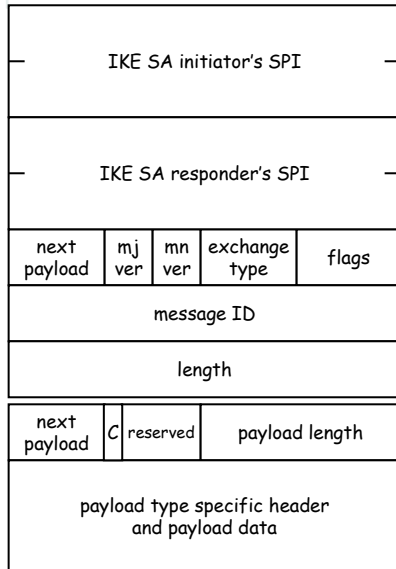


Application: remote access





IKEv2 generic message format



- initiator's SPI
 - an 8 byte value chosen by the initiator to identify the IKE SA
- responder's SPI
 - an 8 byte value chosen by the responder to identify the IKE SA
- major and minor version
 - 2.0
- exchange type
 - IKE_SA_INIT
 - IKE_AUTH
 - CREATE_CHILD_SA
 - INFORMATIONAL
- flags
 - I(nitiator) – set in messages sent by the original initiator
 - R(esponse) – indicates that this message is a response to a previous message with the same msg ID
- message ID
 - used to match responses and requests, and to detect retransmissions



Message IDs

- every IKE message contains a Message ID as part of its fixed header
- this Message ID is used
 - to match up requests and responses (responses always contain the same message ID as the corresponding request)
 - to identify retransmissions of messages
- each endpoint in the IKE SA maintains two "current" Message IDs:
 - the next one to be used for a request it sends and
 - the next one it expects to see in a request from the other end
- counters are incremented as requests are generated and received
- thus, each integer n may appear as the message ID in four distinct messages:
 - the n-th request from the original IKE initiator,
 - the corresponding response,
 - the n-th request from the original IKE responder, and
 - the corresponding response
- there is no ambiguity in the messages, because the (I)nitiator and (R)esponse bits in the message header specify which of the four messages a particular one is



Payload types

- Security Association (SA)
- Key Exchange (KE)
- Identification - Initiator (IDi)
- Identification - Responder (IDr)
- Certificate (CERT)
- Certificate Request (CERTREQ)
- Authentication (AUTH)
- Nonce (Ni, Nr)
- Notify (N)
- Delete (D)
- Vendor ID (V)
- Traffic Selector - Initiator (TSi)
- Traffic Selector - Responder (TSr)
- Encrypted (E)
- Configuration (CP)
- Extensible Authentication (EAP)



IKE exchange types

- all IKE exchanges consist of request/response pairs
- IKE_SA_INIT
 - negotiation of cryptographic algorithms (for the IKE SA)
 - exchange of nonces
 - execution of a Diffie-Hellman key exchange
- IKE_AUTH
 - authenticates the previous (IKE_SA_INIT) messages
 - exchange of identities and certificates
 - establishment of the first CHILD_SA (for AH and ESP)
- CREATE_CHILD_SA
 - may be initiated by either end of the IKE_SA after the initial exchanges are completed
 - establishes further SAs between the two ends if needed (for AH and ESP)



The IKE_SA_INIT exchange

messages:

I → R : HDR; SAi1, KEi, Ni

R → I : HDR; SAR1, KEr, Nr

where

SAi1 – proposed cryptographic algorithms for the IKE SA

- encryption alg
- integrity alg
- prf (pseudo-random function)
- DH group

SAR1 – cryptographic algorithms chosen by the responder

KEi – initiator's ephemeral DH value

KEr – responder's ephemeral DH value

Ni, Nr - nonces



Supported algorithms

▪ encryption

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_DES_IV32
ENCR_AES_CBC (RFC3602)
ENCR_AES_CTR (RFC3664)

▪ integrity

AUTH_HMAC_MD5_96
(RFC2403)
AUTH_HMAC_SHA1_96
(RFC2404)
AUTH_DES_MAC
AUTH_KPDK_MD5 (RFC1826)
AUTH_AES_XCBC_96
(RFC3566)

▪ prf

PRF_HMAC_MD5 (RFC2104)
PRF_HMAC_SHA1 (RFC2104)
PRF_HMAC_TIGER (RFC2104)
PRF_AES128_XCBC
(RFC3664)



Key derivation - preliminaries

- keying material is derived as the output of the negotiated prf algorithm

$$\text{prf+}(K,S) = T1 | T2 | T3 | \dots$$

where:

$$T1 = \text{prf}(K, S | 0x01)$$

$$T2 = \text{prf}(K, T1 | S | 0x02)$$

$$T3 = \text{prf}(K, T2 | S | 0x03)$$

- keys are taken from the output string T1|T2|T3|... without regard to boundaries



Key derivation

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

$$\begin{aligned} &\text{SK}_d | \text{SK}_{ai} | \text{SK}_{ar} | \text{SK}_{ei} | \text{SK}_{er} | \text{SK}_{pi} | \text{SK}_{pr} = \\ &= \text{prf+}(\text{SKEYSEED}, Ni | Nr | \text{SPli} | \text{SPlr}) \end{aligned}$$

$$\begin{aligned} &\text{KEYMAT} = \text{prf+}(\text{SK}_d, Ni|Nr) \quad // \text{key material} \\ & \quad \quad \quad // \text{for child SAs} \end{aligned}$$

- SK_{ei}, SK_{er} – message encryption keys for the IKE SA
- SK_{ai}, SK_{ar} – message integrity (authentication) keys
- SK_{pi}, SK_{pr} – keys used to generate the AUTH payloads



The IKE_AUTH exchange

messages:

$I \rightarrow R : \text{HDR}; \{ \text{ID}_i, [\text{CERT},] \text{AUTH}_i, \text{SA}_i2, \text{TS}_i, \text{TS}_r \}$

$R \rightarrow I : \text{HDR}; \{ \text{ID}_r, [\text{CERT},] \text{AUTH}_r, \text{SA}_r2, \text{TS}_i, \text{TS}_r \}$

where

ID_i – identifier of the initiator (e.g., its IP address)

ID_r – identifier of the responder (e.g., its IP address)

CERT – public key certificate (optional)

AUTH – authentication payload used to authenticate the messages of the IKE_SA_INIT exchange

SA_i2 – proposals for the first child SA

SA_r2 – algorithms chosen by the responder for the child SA

TS_i, TS_r – traffic selectors

{ ... } denotes cryptographic protection (encryption and integrity) with the keys SK_{ai}, SK_{ar}, SK_{ei}, SK_{er}



The AUTH payload

- AUTH authenticates the IKE_SA_INIT exchange
- two options are supported:
 - signature based authentication
 - authentication with a pre-shared secret
- in case of signature based authentication:
 - AUTH_i : sig(msg1_of_IKE_SA_INIT | Nr | prf(SK_{pi}, ID_i))
 - AUTH_r : sig(msg2_of_IKE_SA_INIT | Ni | prf(SK_{pr}, ID_r))
 - the signature can be an RSA or a DSS signature
 - CERT contains the public signature verification key
- in case of shared secret based authentication:
 - AUTH = prf(prf(secret,"Key Pad for IKEv2"), msg_octets)
- AUTH_i and AUTH_r may not necessarily be computed with the same authentication approach



Protection against DoS attacks

messages:

$I \rightarrow R$: HDR; SAi1, KEi, Ni

$R \rightarrow I$: HDR; N(cookie)

$I \rightarrow R$: HDR; N(cookie), SAi1, KEi, Ni

$R \rightarrow I$: HDR; SAR1, KEr, Nr

$I \rightarrow R$: HDR; { IDi, [CERT,] AUTHi, SAi2, TSi, TSr }

$R \rightarrow I$: HDR; { IDr, [CERT,] AUTHr, SAR2, TSi, TSr }

- the responder sends a random cookie to the IP address from which the first message was received
- the initiator must repeat its first message with the cookie included
- an attacker cannot initiate an IKE exchange from a spoofed IP address



Protection against DoS attacks

- the responder shouldn't keep state for the purpose of verifying a previously sent cookie (because that could be exploited by a DoS attack)

- possible solution:

cookie = <VersionIDofSecret> | Hash(Ni | IPi | SPli | <secret>)

where

- <secret> is a randomly generated secret known only to the responder and periodically changed
- <VersionIDofSecret> is changed whenever <secret> is regenerated
- in this way, the cookie can be re-computed when the IKE_SA_INIT request arrives the second time and compared to the cookie in the received message
- in order to avoid problems when <secret> is changed to a new value during a pending IKE_SA_INIT, the responder may keep the old value of <secret> around for a short time and accept cookies computed from either value



The CREATE_CHILD_SA exchange

messages:

$I \rightarrow R : \text{HDR}; \{ \text{SA}_i, \text{Ni}, \text{TS}_i, \text{TS}_r \}$

$R \rightarrow I : \text{HDR}; \{ \text{SA}_r, \text{Nr}, \text{TS}_i, \text{TS}_r \}$

where

SA_i – contains proposals for the child SA to be created

SA_r – contains a single accepted offer

Ni, Nr – nonces (used in key derivation)

TS_i, TS_r – traffic selectors (specify which packets should be protected by the new SA)



Further readings

- RFC 4301: an overview of the IPsec security architecture
- RFC 4302: specification of AH
- RFC 4303: specification of ESP
- RFC 4305: specification of algorithms for AH and ESP
- RFC 4306: specification of IKEv2