# IPsec, IKEv2 and their use in Mobile IPv6

ANEMONE

**Author**:

Zoltán Faigl

June, 2007

# Table of contents

# Acknowledgment

This study was written as part of the work on the development of a large-scale test-bed offering support of mobile users and devices as well as enhanced services  within ANEMONE. The author is grateful to Stefan Lindskog and Anna Brunstrom, from the Privacy and Security Group of Karlstad University (KaU), for their useful comments that they made during the research visit of the first author at KAU, from September to December 2006.

# 1 Introduction

In this study, we deal with the general problem of how security services can be integrated into existing signaling protocols. It discusses the operation of Internet Security Protocol (IPsec) and Internet Key Exchange Protocol version 2 (IKEv2) , and especially, describe their use when protecting Mobile IPv6 (MIPv6) signaling . IPsec and IKEv2 can basically be used in the same way, with any other signaling protocol or data, which they protect. However, it is the security policies and configurations which will vary in function of the information sensitivity that they protect. We present the recommended security policies for the case of protecting MIPv6 signaling between the Mobile Node (MN) and the Home Agent (HA).

The study describes IPsec in Part 2, describes the Internet Key Exchange version 2 (IKEv2) in Part 3, and the use of IPsec and IKEv2 in a Mobile IPv6 scenario in Part 4. Part 5 summarizes the study, Part 6 contains the references and it is followed by Part 7 the appendices, containing the abbreviations, and some details of the mandatory entries of the databases used in IPsec.

# 2 The Internet Security Protocol

## *2.1 Overview*

The Internet Security Protocol is defined in RFC 4301 by the Network Working Group of IETF . This document discusses the fundamental components of IPsec, and the processing of IPsec. Additional RFCs define the two security protocols, i.e., Authentication Header (AH) and Encapsulating Security Payload (ESP) , the default Security Association (SA) management protocol (IKEv2) , and the mandatory-to-implement cryptography algorithms for each protocol, such as .

### 2.1.1 Aim

The aim of IPsec is to create security services at the network layer, and to provide a security framework enabling the appropriate selection of security services for given traffic at the IP packets. Moreover, it remains open for the use of any cryptography algorithm to be able to keep up with the new results in cryptanalysis, the deprecation of old algorithms, and the employment of new ones. The list of mandatory-to-implement algorithms is periodically updated in separate RFCs.

### 2.1.2 General discussion of IPsec

IPsec is a very well designed framework providing a wide range of configuration possibilities on the level of cipher suites, security protocols, security services and security policies. Together with IKEv2, it enables easy deployment. However, IKEv2 is very complex, since it is open for many use scenarios with different security infrastructures in the background. IPsec provides high-level security, when it is well-configured, and it gives assurance for application developers, since they do not have to develop their own security protocols. However, the security level of a whole system can also be very low when using IPsec, since there are factors out of the scope of the IPsec security design, such as implementation quality, personal, physical, procedural, compromising emanations, and computer security. Furthermore, defects of OS in which IPsec is running, e.g. the bad quality of random sources, sloppy system management interfaces, are also affecting the overall security level.

The IPsec protocol defines a security framework that, on the other hand, is difficult to manage. Many people look on it as a too complex architecture, and in fact, applications do not support too much management of IPsec policies, i.e., they do not come out with default policy recommendations for IPsec. Probably, application developers often do not have the resources to give support for the IPsec, since, e.g., in different scenarios different policies are needed, and IPsec and network layer is not their main focus. Application providers can not be sure about whether IPsec is implemented in the hosts or not, that is why they tend to use security solutions at higher layers, such as TLS and SSL. Moreover, IPsec implementations are also “guilty”, since they should be ameliorated to provide more manageable management APIs for higher level applications, to support “cross-layer” reachability.

IPsec is probably a better solution to apply instead of TLS, when the aim is to protect system level protocols such as mobility signaling.

## *2.2 Security services in IPsec*

IPsec provides two choices of security services through two distinct security protocols: the Authentication Header (AH) and the Encapsulating Security Payload (ESP) protocol. The AH protocol provides support for connectionless integrity (or packet level, not session level integrity), data origin authentication, and protection against replays, but it does not support confidentiality. The ESP protocol supports confidentiality, connectionless integrity, anti-replay protection and optional data original authentication.

### 2.2.1 Transport and tunnel mode

Both, AH and ESP, support two modes of operation: the transport and the tunnel mode. The transport mode provides end-to-end protection between the communicating end-points by encrypting the packet payload. The tunnel mode encrypts the entire IP packet and encapsulates the encrypted original packet in the payload of a new IP packet. With tunnel mode, no part of the original packet is exposed to potential threats as the new packet is transmitted through intermediate nodes of the network.

In IPv6, in transport mode, the security protocol header appears after the base IP header and selected extension headers, but may appear after or before destination options. It must appear before next layer protocols. In the case of ESP, the transport mode SAs provide security only for the next layer protocols, not for the IP header or any extension before the ESP header. In the case of AH, the protection is extended to selected portions of the IP header and selected options (IPv6 Hop-by-Hop extension header, or IPv6 Destination extension headers). If AH is employed in tunnel mode, portions of the outer IP header as well as the whole inner packet is protected. With ESP the protection is afforded only to the tunneled packet, not to the outer header.

## 2.3 The IPsec processing model

IPsec may operate in a host, as a security gateway (SG) (e.g., in an IPsec enabled router), or as an independent device. The protection offered by IPsec is based on requirements defined by the Security Policy Database (SPD). All traffic crossing the IPsec boundary must be matched against the entries in SPD. A policy can have one of the three consequences:

- protect the packet using a specific SA pointed by the SPD entry,
- discard the packet, or
- bypass IPsec protection and let the packet to cross the IPsec boundary without any change.

The IPsec processing model is shown in Figure 1.

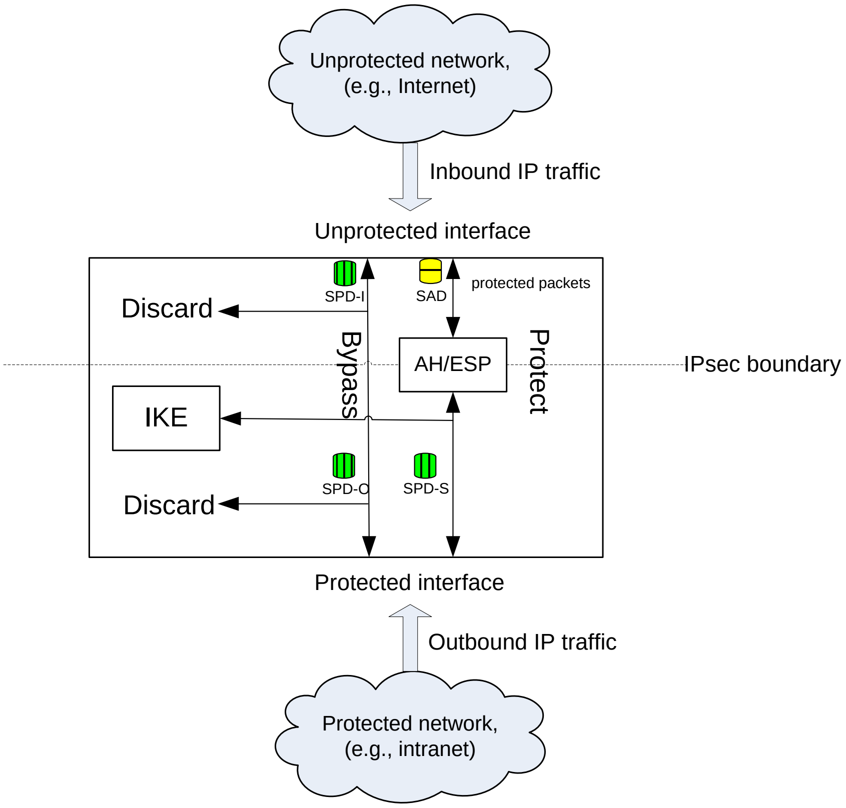


**Figure 1. Top level IPsec processing model**

Note that the traffic coming from the protected network is called outbound IP traffic, the traffic coming from the unprotected network is called inbound IP traffic.

IPsec creates boundary between unprotected and protected interfaces. The unprotected interface may be, e.g., an interface connected to the public Internet, the protected interface may be connected to a closed and safe intranet. When the IPsec operates on an end-host, the protected interface, i.e., the lower-part of Figure 1 is an “interface” to the upper–layer protocols.

## 2.3.1 The three databases of IPsec

The IPsec operation is built on three databases. In practice, these databases may be decorrelated to several instances. The databases are the Security Policy Database (SPD) that contains all the required security policies that the IPsec entity has to apply, the Security Association Database (SAD) that collects entries of each currently used Security Associations (SAs), and the Peer Authorization Database (PAD) that gives constraints on the security associations that the remote peers are allowed to negotiate, and how the remote peers should authenticate their identities.

The IPsec standard states that all the three databases are ordered databases (i.e., lists), and the look-up procedure is sequential, i.e., it stops when the first appropriate entry is found.

### 2.3.1.1 Security Policy Database (SPD)

The aim of the SPD is to determine the disposition of all inbound and outbound traffic. Traffic selectors point out which policy to apply for a given traffic type (i.e., protect, bypass or discard).

The SPD entries are logically split into three parts:

- SPD-O: discard and bypass policies for all outbound IP traffic
- SPD-S: protection policies for all outbound IP traffic
- SPD-I: discard and bypass policies for unprotected (plaintext) inbound IP traffic

One SPD-S entry often enables the creation of a range of SAs. It is due to the fact, that the traffic selector values in the policy entry may contain a range of values, and there is a flag called “populate from packet” (PFP), that indicates if the SA proposal should contain the traffic selector values given in the policy entry or the values populated from the packet.

The SPD can also contain named SPD entries, which can be matched based on the IKE identity of the remote peer. These policy entries enable the creation of SA pairs which contain always the actual IP address of the remote peer.

### 2.3.1.2 Security Association Database (SAD)

An SA is a management construct used to enforce policies. The SAD contains established (keyed) SAs. Thus the SAD is just a collection of currently applied, and instantiated policies. It can be maintained either manually or dynamically. In case of dynamic configuration, IKEv2 is the default SA management protocol. However, the role of IKEv2 is just the negation. The information to negotiate about the new SA pair is coming from the specific SPD-S entry of the peers, and constrained by the PAD entries of the remote peer.

#### 2.3.1.3 The use of SPD and SAD

The IPsec processing model, shown in Figure 1, presents on a high-level the purposes of SPD and SAD databases for outbound and inbound traffic.

The outbound IP traffic, going from unprotected to protected state, is matched firstly against SPD-S entries. SPD-S entries are the policies which will trigger the protection of the IP packet, by pointing out the appropriate SA entry in the SAD. If the matching SPD-S policy entry dictates the use of an SA entry that does not exist in the SAD, then a new SA pair must be created. In this case, the SPD triggers IKEv2 to negotiate the requested SA pair with the peer IPsec node. If no corresponding rule is found in SPD-S, then the packet is matched against SPD-O entries. The SPD-O entries trigger either the action discard or bypass of the packet.

For inbound IP traffic, the protected packets are matched against the Security Association Database (SAD). The matching of packet to the SA entries is based on the Security Pointer Index (SPI) in the AH/ESP header, and in some cases the destination address and the source address of the packet. If a corresponding SA entry is found, the inbound IP packet is processed (e.g., decrypted) using the parameters of the SA. If no matching SA entry is found for the packet then it is discarded at the unprotected side of the IPsec boundary.

For inbound IP traffic, the unprotected packets are matched against SPD-I entries, and there are either discarded or bypassed through the IPsec boundary.

#### 2.3.1.4 Peer Authorization Database (PAD)

The Peer Authorization Database (PAD) can be considered as the top coordinator. Its function is related with SA management and peer authentication. It contains information about the remote peers.

A PAD entry contains information about what authentication method can be used by a specific remote peer. Only successfully authenticated peers are able to negotiate SAs. The PAD controls, who can negotiate new SAs with the local peer. The PAD entries contain also restrictions about the SAs that a remote peer should negotiate with the local peer. Basically, a remote peer should not be able to negotiate SAs for traffic which is completely independent of itself. Practically, the PAD entries give constraints on the traffic selector values (TSi) that the remote peer can propose. The PAD entries also indicate that when negotiating a new SA pair, whether the policies with a specific peer should be looked up based on the identity (IKE IDi) of the peer, or based on the traffic selector values (TSi) proposed by the peer.

It is important to note that the original IPsec was developed to protect traffic between machines, and to use stationary SAs. For this purpose, it was enough to have the SAD in each peer, configured manually. The introduction of SPD makes it possible for an easier deployment of SAs, since the SAs can be created dynamically, when needed. Nowadays, there is a tendency where the service authorizing entity wants to authenticate not the devices but the users, i.e., the identities using the service. This is the case with IP based mobility services, since the Mobile Service Provider (MSP) wants to be sure that only the authorized subscriber uses the service, independently of the device. The Peer Authorization Database makes possible the definition of how to authenticate peers and to authenticate them based on identifiers which relate to contracted subscribers.

### 2.3.2 Cryptographically strengthened, IP packet-level access control

IPsec provide access control, similarly to traditional gateways or routers. The policies can be created in a very fine granularity due to the wide range of traffic selectors. However, an IPsec gateway can provide higher-level security for access control than traditional gateways, since the access control is supported by cryptography mechanisms. If any of the confidentiality, integrity check, data origin authentication, anti-replay services leads to failure, when processing inbound protected IP traffic, the access will be prohibited.

Moreover, only authenticated peers can create SA entries in the SAD. Thus, before any processing, the inbound IP traffic will be discarded by the SPD-I policy, if there is no SA entry for the traffic. Authentication of peers is performed during the IKE initialization phase. Then, during the data session, the data origin authentication assures that the IP packet really comes from the previously authenticated peer.

### 2.3.3 IPsec processing

This part describes how IP packets are processed when they cross the IPsec boundary from the protected to the unprotected side and then from the unprotected to the protected side. The first part in this section presents the basic situation, when the appropriate SAD entries exist at both sides at the request time for processing. The second part shows the case when the SA entries do not exist at the either of the two entities. In this case the SPD-S policy of the sender entity triggers an IKEv2 negotiation to create the appropriate SA pair.

#### 2.3.3.1 Basic IPsec processing

Basically, IPsec operates in the following way. The sender entity, within that the IP packet crosses the IPsec boundary from the protected to the unprotected interface, looks into its SPD database. The outbound IP packet must be matched firstly against SPD-S policy entries. In this case the entry is found in the SPD-S cache, and it points out the security association to apply (SAout) in the SAD. Then the packet is processed by IPsec based on the SAout entry. The IPsec processing may include encryption, integrity check value computation, and the extension of the IP packet with ESP header/trailer or with AH header, and maybe an Authentication payload in case of ESP protocol. Finally the packet is forwarded through the unprotected interface to the receiver.

From the point of view of the receiver, the received IP packet belongs to inbound IP traffic. The receiver looks up its SAD for appropriate SAin entry. The look-up is based on the SPI in the ESP/AH header and maybe the destination IP address and/or source IP address. If the appropriate SAin is found, the entity un-protects the packet using the settings in SAin. However, if there is no appropriate SAin, the packet is dropped, and an audit log must be created since an attack could have been attempted.

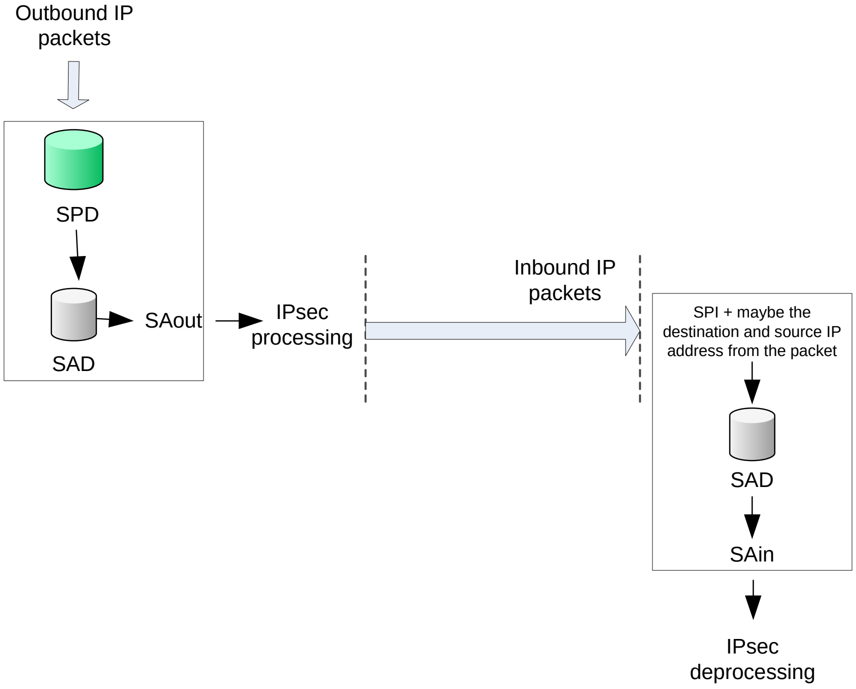


**Figure 2. Basic IPsec processing.**

### 2.3.3.2 Processing with IKE

This part presents the way of processing an outbound IP packet while it is crossing the IPsec boundary from protected to unprotected interface, but the appropriate SPD-S policy entry describes an SAout which does not exist in the SAD. If the SAout entry is not in the SAD, two alternative action can take place. If there is no SA management protocol, then the IP packet is dropped. Otherwise the supported SA management protocol is triggered to negotiate the new SAout, and to insert it into the SAD. The default SA management protocol is IKEv2. Typically, a successful negotiation results in the creation of an SAout entry at the initiator/responder and an SAin at the responder/initiator. These define one SA in each direction.

Figure 3 presents the main steps when IKEv2 SA negotiation is triggered before an outbound IP packet can be processed and transmitted. The figure only shows the SA negotiation part. After the presented procedure, the IPsec processing for the IP packet happens the same way as described for the basic IPsec processing (see Figure 2).

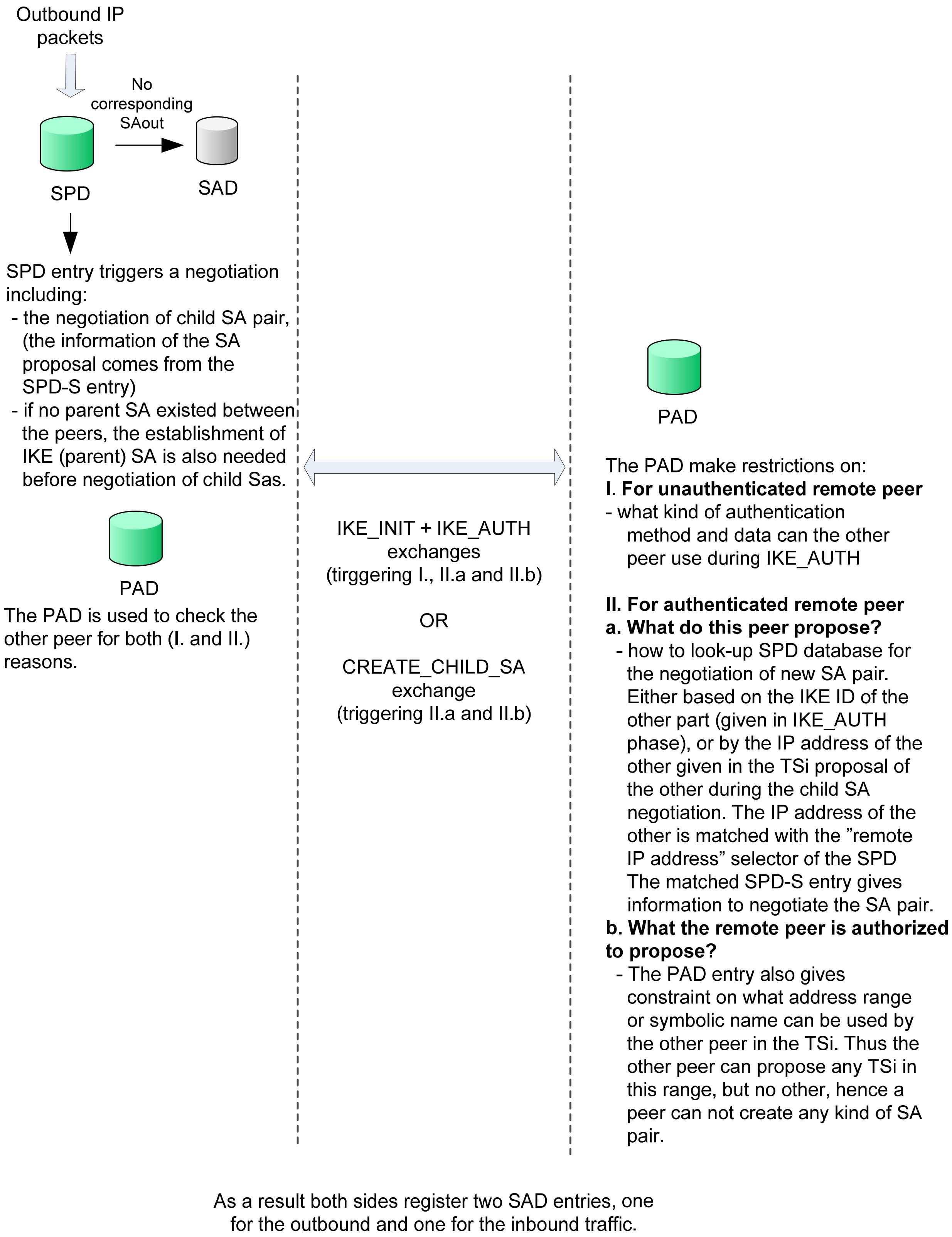
The IKEv2 negotiation, in Figure 3, includes the negotiation of a child SA pair, typically with the CREATE_CHILD_SA exchange of IKEv2. However, if the parent (IKE) SA does not exist, then an IKE initialization procedure anticipates the child SA creation. The IKE initialization includes an IKE_INIT message exchange for the creation of parent (IKE) SA pair, and the IKE_AUTH phase for the authentication of peers. In addition, the IKE_AUTH phase includes the creation of the child SA pair.

The peers use their PAD for three reasons:

I. In case of unauthenticated peers (not having established IKE SA pair), they authenticate each other in the IKE_AUTH exchange. The claimed identity and AUTH payload of each entity is verified using the PAD at the other side, hence the PAD contains information about which authentication method and data are the peers authorized to use.

II. a: The PAD is used to indicate, how to look-up the SPD, and find out what should the SA negotiation contain. This procedure is surely used at the responder of the negotiation, in order to find out how to restrict the SA proposal of the initiator to the needs of the responder. Perhaps this step is made by the initiator of the SA negotiation to look into its SPD-S database, what SA to propose.

II. b: After the peers have authenticated each other, and there exists a parent SA pair between them, the creation of child SA pair comes. The PAD is used to check the SA child constraints, i.e., what traffic selectors the other entity can propose as its local address and port range. The responder side surely makes this check, but it is not unambiguous from the IKEv2 standard , if the initiator of the SA negotiation needs also to check the others' SA description, since it was the initiator which made the initial proposal. Probably the initiator also makes this check. The standard has an implicit meaning that the initiator should check that whether the responder sends back a restricted set of the SA proposal of the initiator.



SPD entry triggers a negotiation including:

- the negotiation of child SA pair, (the information of the SA proposal comes from the SPD-S entry)
- if no parent SA existed between the peers, the establishment of IKE (parent) SA is also needed before negotiation of child Sas.

The PAD is used to check the other peer for both (**I**. and II.) reasons.

IKE_INIT + IKE_AUTH exchanges (tirggering I., II.a and II.b)

OR

CREATE_CHILD_SA exchange (triggering II.a and II.b)

The PAD make restrictions on:

**I. For unauthenticated remote peer**

- what kind of authentication method and data can the other peer use during IKE_AUTH

**II. For authenticated remote peer**

**a. What do this peer propose?**

- how to look-up SPD database for the negotiation of new SA pair. Either based on the IKE ID of the other part (given in IKE_AUTH phase), or by the IP address of the other given in the TSi proposal of the other during the child SA negotiation. The IP address of the other is matched with the ”remote IP address” selector of the SPD The matched SPD-S entry gives information to negotiate the SA pair.

**b. What the remote peer is authorized to propose?**

- The PAD entry also gives constraint on what address range or symbolic name can be used by the other peer in the TSi. Thus the other peer can propose any TSi in this range, but no other, hence a peer can not create any kind of SA pair.

As a result both sides register two SAD entries, one for the outbound and one for the inbound traffic.

**Figure 3. IPsec processing  when IKEv2 is triggered.**

### 2.3.3.3 Detailed description of the processing of outbound and inbound traffic

This part describes in details the processing of outbound and inbound traffic at the IPsec boundary. It shows that how and when SPD-I, SPD-O, SPD-S policies, and the SAD (i.e., the current instantiations of the policies) are used. Figure 4 and Figure 5 show the processing of outbound and inbound IP traffic, respectively.

The detailed description involves also the usage of SPD caches, which purpose is to speed up the SPD look-up procedure. The idea is to split into several parts the SPD and check only the part which may have policy entry for the specific case. Practically, an administrator may define the SPD as one entity. Then the policies need to be decorrelated into three logical parts, i.e., the SPD-S, SPD-O and SPD-I. They also may be decorrelated to more instances, if there are more than one unprotected or protected interfaces.

The caching mechanism is the same for each part. At the first use of a policy entry, if the entry is not in the cache, then it is inserted into the cache. Next time, the entry can be read out from the cache, and there is no need to read from the SPD stored in the RAM or the file system.

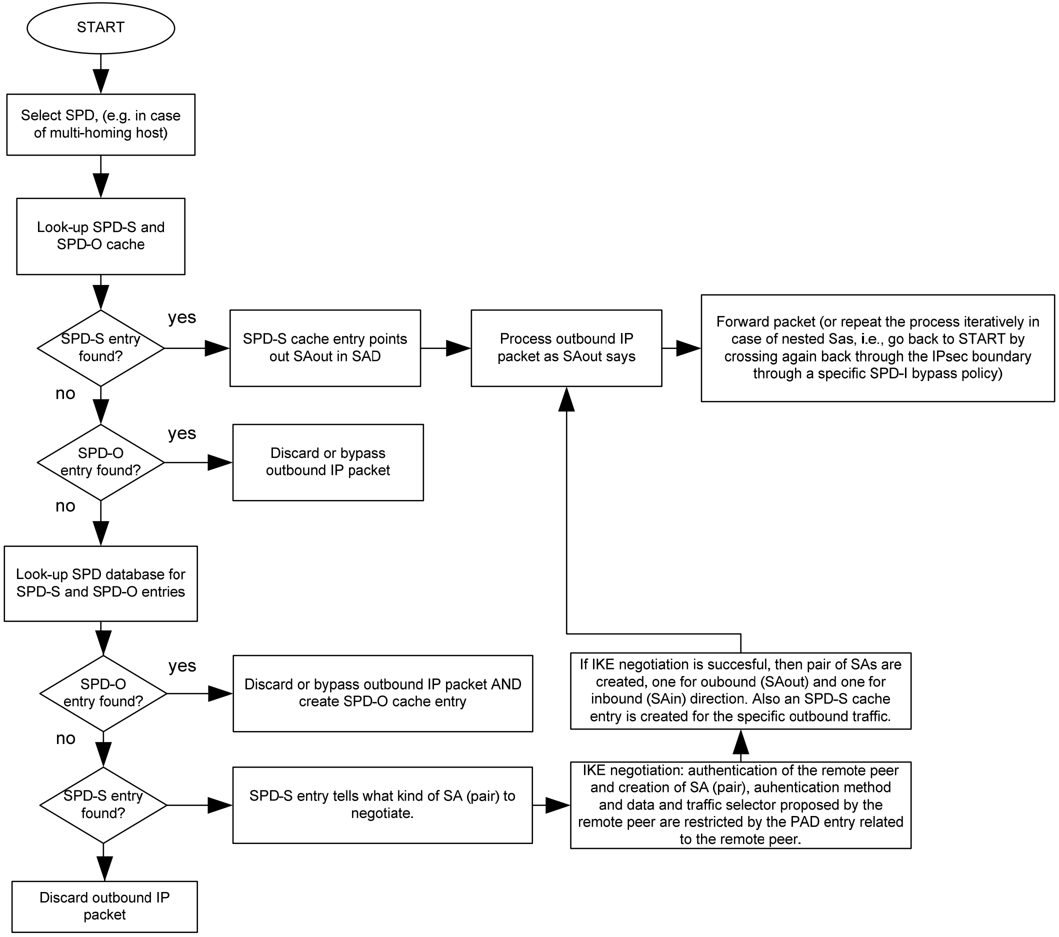

START
Select SPD, (e.g. in case of multi-homing host)
Look-up SPD-S and SPD-O cache
SPD-S entry found?
yes
no
SPD-S cache entry points out SAout in SAD
Process outbound IP packet as SAout says
Forward packet (or repeat the process iteratively in case of nested Sas, i.e., go back to START by crossing again back through the IPsec boundary through a specific SPD-I bypass policy)
SPD-O entry found?
yes
no
Discard or bypass outbound IP packet
Look-up SPD database for SPD-S and SPD-O entries
SPD-O entry found?
yes
no
Discard or bypass outbound IP packet AND create SPD-O cache entry
If IKE negotiation is succesful, then pair of SAs are created, one for oubound (SAout) and one for inbound (SAin) direction. Also an SPD-S cache entry is created for the specific outbound traffic.
SPD-S entry found?
SPD-S entry tells what kind of SA (pair) to negotiate.
IKE negotiation: authentication of the remote peer and creation of SA (pair), auhentication method and data and traffic selector proposed by the remote peer are restricted by the PAD entry related to the remote peer.
Discard outbound IP packet

**Figure 4. Processing of outbound IP traffic.**

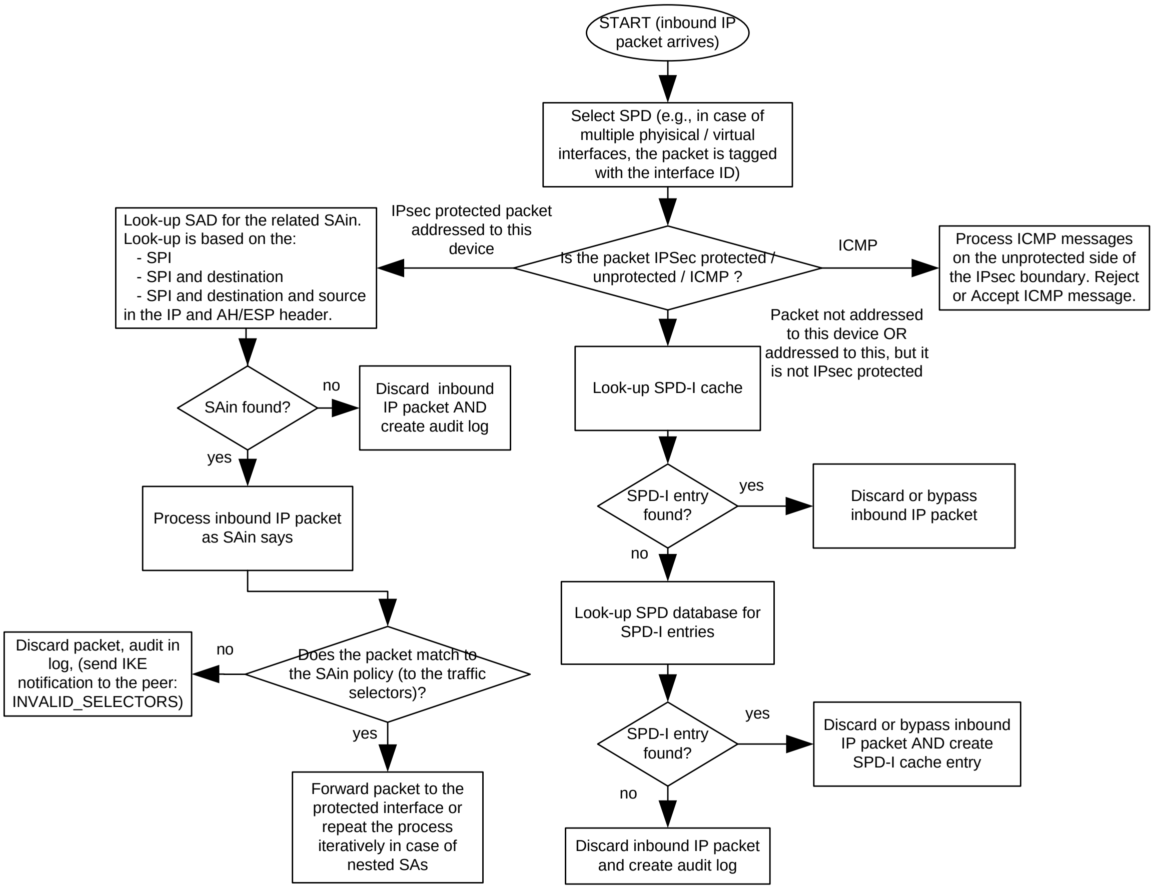

START (inbound IP packet arrives)
Select SPD (e.g., in case of multiple phyisical / virtual interfaces, the packet is tagged with the interface ID)
Is the packet IPSec protected / unprotected / ICMP ?
IPsec protected packet addressed to this device
ICMP
Process ICMP messages on the unprotected side of the IPsec boundary. Reject or Accept ICMP message.
Look-up SAD for the related SAin. Look-up is based on the:
- SPI
- SPI and destination
- SPI and destination and source in the IP and AH/ESP header.
Packet not addressed to this device OR addressed to this, but it is not IPsec protected
SAin found?
no
Discard inbound IP packet AND create audit log
yes
Look-up SPD-I cache
Process inbound IP packet as SAin says
SPD-I entry found?
yes
Discard or bypass inbound IP packet
no
Look-up SPD database for SPD-I entries
Discard packet, audit in log, (send IKE notification to the peer: INVALID_SELECTORS)
no
Does the packet match to the SAin policy (to the traffic selectors)?
yes
Forward packet to the protected interface or repeat the process iteratively in case of nested SAs
SPD-I entry found?
yes
Discard or bypass inbound IP packet AND create SPD-I cache entry
no
Discard inbound IP packet and create audit log

**Figure 5. Processing inbound IP traffic.**

## 2.4 Granularity of policies

The traffic selectors in the SPD entries enable to define security policies at different granularity levels. The security policies can define the same security policy for all the traffic that needs to be processed by IPsec. In this case, the security policy should be adapted to the aggregated security requirements of the protected traffic. The definition of security policies with coarse granularity is advantageous because of less SA management overhead, but disadvantageous since there may be traffic classes which are overprotected in some sense, increasing the total performance cost. Moreover, from security point of view, it is always better if we use a secret only in the required cases, and do not abuse it. In case of encryption, the usage of encryption key for all traffic may raise the chance of a known plaintext attack related to the case when only the traffic which needs confidentiality is encrypted. Thus, both from security point of view and performance point of view, it is better to apply more fine security policies. The negotiation cost of SAs is higher in case of finer granularity.

## *2.5 Supported cryptography algorithms*

RFC 4305 defines the current mandatory-to-implement algorithms for ESP and AH protocol. The algorithms are just enumerated in the followings. Their importance is also given with the following priority order: MUST > SHOULD > MAY > SHOULD NOT > MUST NOT. The +/- sign indicates the expected tendency of the importance of the algorithm.

### 2.5.1 ESP

The ESP protocol can provide confidentiality and data origin authentication service with the following encryption and authentication algorithms. Authentication algorithms also provide connectionless integrity service, i.e., protect integrity on per-packet base.

Encryption algorithms:

- NULL (MUST)
- TripleDES-CBC (MUST -)
- AES-CBC with 128 bit keys (SHOULD+)
- AES-CTR (SHOULD)
- DES-CBC (SHOULD NOT)

Authentication algorithms:

- HMAC-SHA1-96 (MUST)
- NULL (MUST)
- AES-XCBC-MAC-96 (SHOULD+)
- HMAC-MD5-96 (MAY)

Combined algorithms give both authentication and confidentiality service, thus in the case of choosing a combined algorithm, we do not need to select encryption and integrity algorithm. The only combined algorithm mentioned for ESP is:

- AES-CCM (SHOULD+)

### 2.5.2 AH

The AH protocol can provide data origin authentication and integrity service for almost the whole IP packet. The mandatory-to-implement authentication algorithms are the same as for ESP:

- HMAC-SHA1-96 (MUST)
- AES-XCBC-MAC-96 (SHOULD+)
- HMAC-MD5-96 (MAY)

### 2.5.3 IPsec User Interface suites

In order to support the communication of system administrators, RFC 4308 defines two possible User Interface (UI) suites for IPsec. UI suite contains a specific selection of cryptographic algorithms for the configuration of IPsec. The two suites are referred as

VPN-A and VPN-B, and they mean the following security configuration for IPsec and IKE or IKEv2.

**VPN-A:**

- IPsec configuration:
  - Protocol: ESP
  - ESP Encryption: TripleDES in CBC mode
  - ESP integrity: HMAC-SHA1-96

- IKE and IKEv2 configuration:
  - Encryption (in IKE SA): TripleDES in CBC mode
  - Pseudo-random function (PSF): HMAC-SHA1
  - Integrity (in IKE SA): HMAC-SHA1-96
  - Diffie-Hellman group: 1024-bit long modular exponential (given in for IKEv2)

**VPN-B:**

- IPsec configuration:
  - Protocol: ESP
  - ESP Encryption: AES with 128-bit keys in CBC mode
  - ESP integrity: AES-XCBC-MAC-96

- IKE and IKEv2 configuration:
  - Encryption (in IKE SA): AES with 128-bit keys in CBC mode
  - Pseudo-random function (PSF): AES-XCBC-PRF-128
  - Integrity (in IKE SA): AES-XCBC-MAC-96
  - Diffie-Hellman group: 1048-bit long modular exponential (given in for in IKEv2)

# 3 The Internet Key Exchange Protocol version 2

## *3.1 Overview*

The Internet Key Exchange version 2 (IKEv2) protocol is the default SA management protocol for IPsec. It is specified in RFC 4306 . Moreover, “IKEv2 Clarifications and Implementation Guidelines” were published in RFC 4718 , in October 2006, in order to clarify unambiguous issues.

### 3.1.1 Problem Statement

#### 3.1.1.1 Automatic SA generation

The basic need for IKEv2 is rising from the problem that the manual configuration of SAs is often not enough due to deployment reasons in large scenarios, and due to the need of SA refreshment, after the expiration of the lifetime of the SAs. IKE solves the negotiation for this purpose. However, it is the PAD that has to define, by default, what kind of SAs a remote peer can negotiate with the local peer.

#### 3.1.1.2 Authentication of the peers

Moreover, the authentication of peers also needs to be solved. Basically, IPsec gives data origin authentication service for IP packets, by assuring a proof that the packet came from a given IP source address. The authenticity (the property of the message to be authentic and not modified) is verified through the invocation of security policies at the receiver entity, i.e., the matching of the source address in the packet to the remote IP traffic selector, and the call for the authenticity check procedure using a key and message authentication algorithm (HMAC) stored in the corresponding SAin entry.

It is not enough to prove that the packet came from a given IP address. The receiver may want to bind the IP address to an authenticated remote peer identity. The identity may also be a user identity, and not a device. This requirement rises from the needs of some service providers, which want to track the activities of users, and not the devices. Therefore, the IPsec standard enables the use of named SPD policies, resulting in that security policies may also be defined in relation with real users, not only device IP addresses. The problem to solve outside of IPsec is to assure that the claimed identity communicating from a given address is not faked, thus IKE has to solve peer authentication. After a successful authentication, the SPD is looked-up based on the IKE identity (or other identity type, e.g. EAP-identity or locally defined identity) of the remote peer, and the secret for the authentication assures the origin authenticity of each packet.

**Problem of static SAs**

IPsec can create SAs which are logically bound to user identities, after the invocation of named SPD policies. By the way the generated SAD entries will never contain the identity value in the remote IP address selector, since at the generation of the SA entries IPsec replaces the identity of the peers with their current IP addresses. However, until the peers remain at the same IP address, the method assures trustfulness.

### 3.1.2 Basic idea

The basic idea with IKE and IKEv2 are essentially the same. Both solve the SA negotiation problem. They build-up a long-term security channel, i.e. an IKE SA pair, providing confidentiality and integrity services for the further IKE negotiations. We call the IKE SA the parent SA. Then, this security channel is used to negotiate the child SAs that the peers will use for the protection of their outbound and inbound traffic.

### 3.1.2.1 SA generation and rekeying

The SAs needs to be renegotiated in some periods, since they have a lifetime attribute, moreover, specific events can trigger their deletion (e.g., the over-roll of the sequence number counter for anti-replay protection). Firstly, the peer creates a new SA pair, negotiating it through the parent SA pair (see IKE_CREATE_CHILD_SA phase in Figure 6). For each child SA pair, it is an option to negotiate a new master key (using Diffie-Hellman ephemeral key exchange). After that the new child SA pair is created, the peer deletes the old SA pair from its database. After deletion, it might notify the other peer to delete the old SAD entries.

Parent SA pairs can also be recreated with IKE_CREATE_CHILD_SA phase, through the old parent SA pair. A new master secret must be negotiated using the Diffie-Hellman ephemeral key exchange. Once the new parent SA pair is created, the old pair must be deleted.

### 3.1.2.2 Authentication of the peers

A peer authentication phase (see IKE_AUTH in Figure 6 on page 7) follows always the initialization of the parent SA pair (see IKE_SA_INIT in Figure 6 on page 7). The peers send their authentication data through the security channel characterized by the parent SA pair.

IKEv2 does not enable the peers to negotiate the authentication algorithm that they prefer to use. Each peer uses its preferred authentication method and authentication data, they can use different methods (see Part 3.2). Consequently, the authentication data sent to the other peer contains also information on the used authentication method and type. However, some checking for the authorization of the peers is needed. The receiver peer can check, based on its PAD, whether the right authentication method was used by the remote peer. Furthermore, the PAD may also contain information for the verification of the authentication data.

IKEv2 makes possible that the peers request from each other and send to other some authentication related information using the CERT and CERTREQ message (see Figure 6). The CERTREQ may be used, if the PAD does not contain the additional information to verify the remote peer’s authentication data (e.g., request for certificate type). The CERT may be used to send information supporting the verification of the authenticity of the identity of the sender peer (e.g., a given certificate).

As a conclusion, the IKEv2 and IPsec standards do not describe, where to store authentication related information. The IPsec standard says, that the PAD already contains the credentials for the remote peers. The IKEv2 standards recommend, that if it is possible, send the credential information together with the authentication data (e.g., certificate-based signature).

### 3.1.3 Mechanism

Figure 6 shows the message exchange types in IKEv2. IKE_SA_INIT and IKE_AUTH message exchanges are made when the peers need to establish an IKE (parent) SA pair. The IKE_CREATE_CHILD_SA exchange is made when any of the peers want to create a child SA pair with the other peer. The INFORMATIONAL message exchange can be made as an extension of these exchanges or in separate messages. Any of the peers can be initiator. The IKE_SA_INIT and IKE_AUTH phases are always made together in the described order.

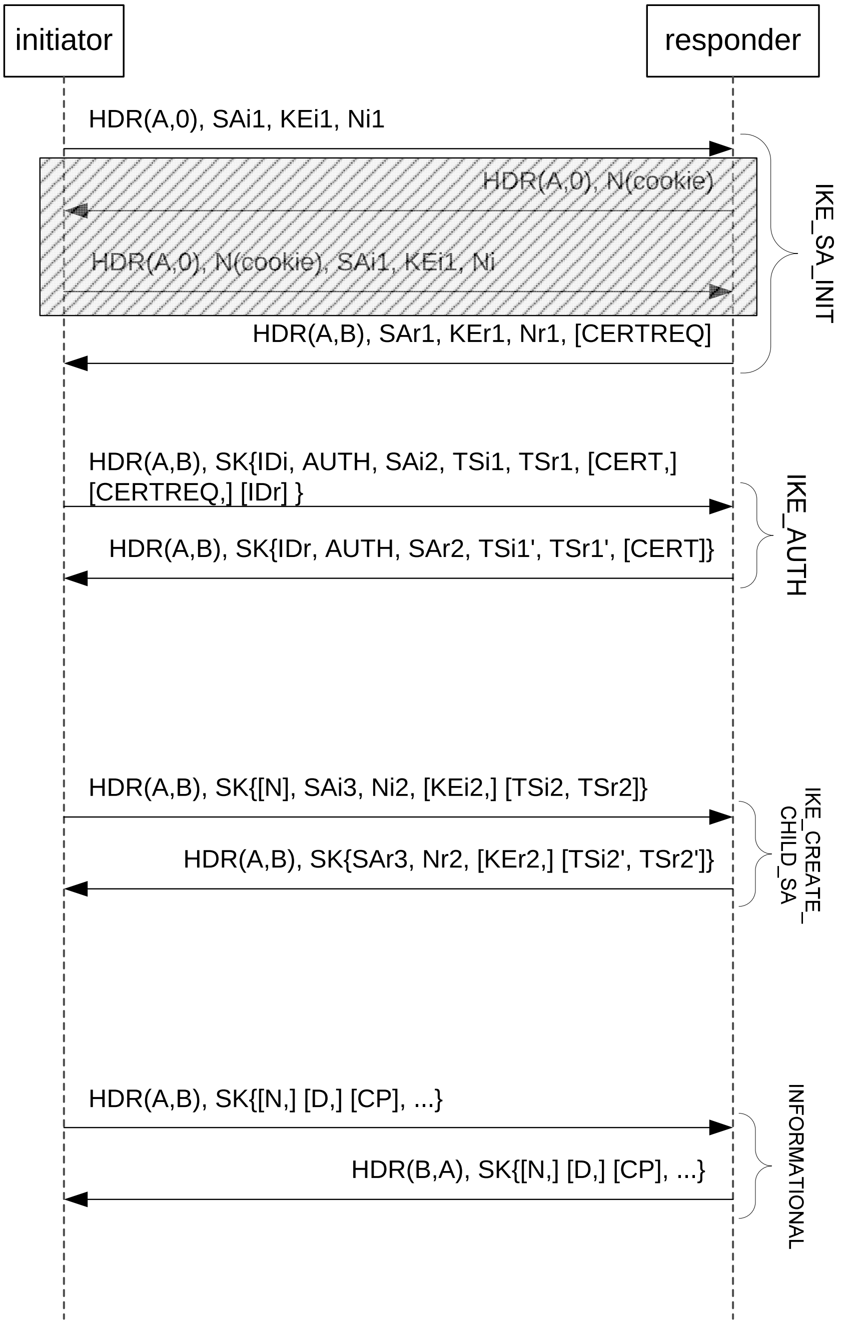

initiator
responder
HDR(A,0), SAi1, KEi1, Ni1
HDR(A,0), N(cookie)
HDR(A,0), N(cookie), SAi1, KEi1, Ni
HDR(A,B), SAr1, KEr1, Nr1, [CERTREQ]
IKE_SA_INIT
HDR(A,B), SK{IDi, AUTH, SAi2, TSi1, TSr1, [CERT,] [CERTREQ,] [IDr] }
HDR(A,B), SK{IDr, AUTH, SAr2, TSi1', TSr1', [CERT]}
IKE_AUTH
HDR(A,B), SK{[N], SAi3, Ni2, [KEi2,] [TSi2, TSr2]}
HDR(A,B), SK{SAr3, Nr2, [KEr2,] [TSi2', TSr2']}
IKE_CREATE_ CHILD_SA
HDR(A,B), SK{[N,] [D,] [CP], ...}
HDR(B,A), SK{[N,] [D,] [CP], ...}
INFORMATIONAL

**Figure 6. Message exchange types in IKEv2.**

The content of the messages are described in Table 1.

**Table 1. Description of IKE message contents**

| HDR | IKE Header | Description |
|---|---|---|
| SK{} | Encrypted payload | It represents the parent SA which provides confidentiality and integrity service |
| SA | Security Association proposal | SAi1 and SAr1 contain information on the negotiation of parent SA pair; SAi2, SAr2, SAi3 and SAr3 represent the negotiation of child SA pairs |
| KE | Key Exchange | It contains the public Diffie-Hellman value of a peer. The peers derive the master secret key based on the ephemeral Diffie-Hellman algorithm . |
| Ni, Nr | Nonce | Contain random values for freshness, used in secret calculations and calculation of the Authentication data in AUTH. |
| IDi | Identification of the initiator | Contains the identity of the initiator. |
| IDr | Identification of the responder | Contains the identity of the responder. |
| AUTH | Authentication | Contains an authentication payload, e.g., a digital signature, of the sender |
| CERT | Certificate | Contains supplementary authentication data to authenticate the sender, e.g. a certificate chain. It can also be used in case of non certificate based authentication methods to send a credential of the sender. |
| CERTREQ | Certificate request | It contains request for credentials of the receiver. |
| EAP | Extensible Authentication Protocol | In case of EAP-based peer authentication, these contain the EAP Request and Response messages. |
| TSi | Traffic Selector of the initiator | The traffic selector values for the proposed SA at the initiator's side (local IP address, protocol ID, local port) |
| TSr | Traffic Selector of the responder | The traffic selector values for the proposed SA at the responder's side (remote IP address, protocol ID, remote port) |
| CP | Configuration | Notification information, can contain request or replies for network information, e.g., NetBios Name Server (NBNS) address, DNS server address, internal sub-network address, etc. These messages can be used for the bootstrapping of a remote peer. |
| N | Notify | It contains an error or a status message for the other peer. |
| D | Delete | Notifies the receiver about the deletion of an SA. The other has to delete also the corresponding SAD entry. |

## 3.1.4 IKE initialization

The IKE initialization has two main purposes. Firstly, it creates the parent SA pair (IKE SA pair). Secondly it authenticates the peers. Moreover, the IKE_AUTH phase also negotiates a child SA pair.

### 3.1.4.1 IKE_SA_INIT

The purpose of this phase is the derivation of the master secret key with the ephemeral Diffie-Hellman key exchange method. The KE payloads contain the public Diffie-Hellman values of the peers. The IKE_SA_INIT message exchange results in the calculation of master secret key, and the calculation of further shared secret keys. The nonce values (Ni and Nr) are also used in these calculations

The resulted keys are:

- one integrity key for each direction
- one encryption key for each direction

These are used with the parent SA pair.

Additional keys are also derived:

- one authentication key for each direction, used in the calculation of authentication data in the IKE_AUTH phase.
- one shared secret used later, at the creation of child SA pairs

The peers agree on several algorithms, using the SAi1 (security association proposal of the initiator) and SAr1 (the choice of the responder from the proposal):

- A pseudo-random function (PRF) that is used to derive the previously described keys
- The encryption and integrity algorithm used in the parent SA pair.
- The Diffie-Hellman group index. The KEi1, containing the public Diffie-Hellman value of the initiator, calculated using the indexed group, is sent at the same time when the Diffie-Hellman group proposal. Thus it is possible, that the responder do not support the Diffie-Hellman group that was used. In this case the responder sends back an error notification and an indication which Diffie-Hellman group to use. After that, the IKE_SA_INIT exchange is repeated.

At the IKE_SA_INIT phase the responder may imply cookie-based authorization, in order to prevent resource exhaustion attacks on the responder, and IP source address spoofing of the initiator. In this case, the responder accepts the first IKE_SA_INIT message only if it contains a cookie calculated previously by the responder. If the initiator does not send the cookie, then the responder sends back a notification message with a fresh cookie. Then the initiator must include the cookie in form of a notification payload in the next IKE_SA_INIT request.

#### 3.1.4.2 IKE_AUTH

The purpose of this phase is the authentication of the claimed identities of the peers. The main authentication methods supported by IKE are discussed in Section 3.2. The PAD of each peer contains authentication information of the remote peers. The AUTH message contains the authentication method that the sender has chosen for itself, and the authentication data. The CERT message may contain supplementary information for the verification at the other side. The CERTREQ may request the other peer to provide supplementary authentication data, or may contain authentication preferences, such as preferred certificate encoding type or Certificate Authority. The [IDr] message can be used in case of multi-user responder host, to point out which identity should authenticate.

Moreover, the IKE_AUTH phase also creates a child SA pair. It contains the SA proposal (SAi2) of the initiator and the choice of the responder (SAr2). The traffic selectors (TSi, TSr) are also negotiated. The negotiation of child SA pair is involved in the authentication phase probably, because the IKE initialization and authentication is always triggered by a given SPD-S policy entry at the initiator in order to create a new SA pair.

### 3.1.5 IKE create child SA

This phase can be initiated by either IKE peer, for one of the two purposes

- Create or refresh an IKE (or parent) SA pair
- Create or refresh a child SA pair

In case of the negotiation of a parent SA pair, the traffic selectors (TSi, TSr) are not included in the negotiation, since there are not needed. However, a new ephemeral Diffie-Hellman key exchange must be involved (Kei, KEr), to create new master secret and derive new shared secrets. In case of child SA creation, the initiator may trigger the creation of a new master key and the derivation of shared secret keys; moreover, they must negotiate the traffic selector values (TSi, TSr)

## *3.2 Authentication methods*

Basically, IKEv2 supports three authentication methods:

- Pre-shared secret based authentication :
    - Raw RSA key based
- Certificate-based authentication :
    - X.509 certificate based signature
    - Hash and URL of X.509 certificate
    - Hash and URL of X.509 bundle
- EAP-method based authentication :
    - Certificate-based
    - Password-based
    - Symmetric key based

The pre-shared secret based authentication is preferable in small-scale environments. The certificate-based authentication methods are recommended in large-scale environments, mainly for server side authentication. The EAP-based methods are preferred by service providers who have already an established AAA infrastructure with EAP-server functions . The AAA server might substitute some functionalities of the PAD, since AAA servers act as remote peer authentication servers and authorizers.

### 3.2.1 EAP-based authentication methods

Mobile service providers aim to authenticate the users instead of their devices, and they intend to move the role of authentication from the HA to an AAA server. IKEv2 supports carrying of EAP-methods that adapts well to these requirements. It carries authentication methods that may require the interaction of the trusted user where the user may be authenticated based on the identifier of the selected authentication method (e.g., the EAP identifier for EAP-MD5 or certificate information in case of certificate-based EAP-TLS). Another difference of the mobile scenario compared to traditional static topologies is that while in a traditional scenario the main role of IKEv2 is to authenticate the peers once (at least once per day ), and then negotiate and rekey more often the SAs (at least every 8 hours for IPsec SAs ), now the IKEv2 reauthentication of the users becomes more important for the providers. RFC~4478 recommends an IKEv2 reauthentication lifetime between five minutes and one day, a very large interval. It depends on the selected EAP method, which value to choose.

Figure 7 presents the whole IKEv2 reauthentication message flow when applying EAP-TLS peer authentication with Diameter authentication server. The first two blocks, i.e., IKE_INIT and IKE_AUTH, are responsible for the key agreement and derivation of session keys, authentication of the IKEv2 peers (MN and HA) and the AAA server, and the negotiation of one IPsec SA pair. The third block may be present if another IPsec SA pair must be created based on the selected MIPv6 protection policy. The fourth block, i.e., INFORMATIONAL, performs the deletion of old IPsec and IKEv2 SA pairs when the new SAs have been established after an IKEv2 reauthentication.

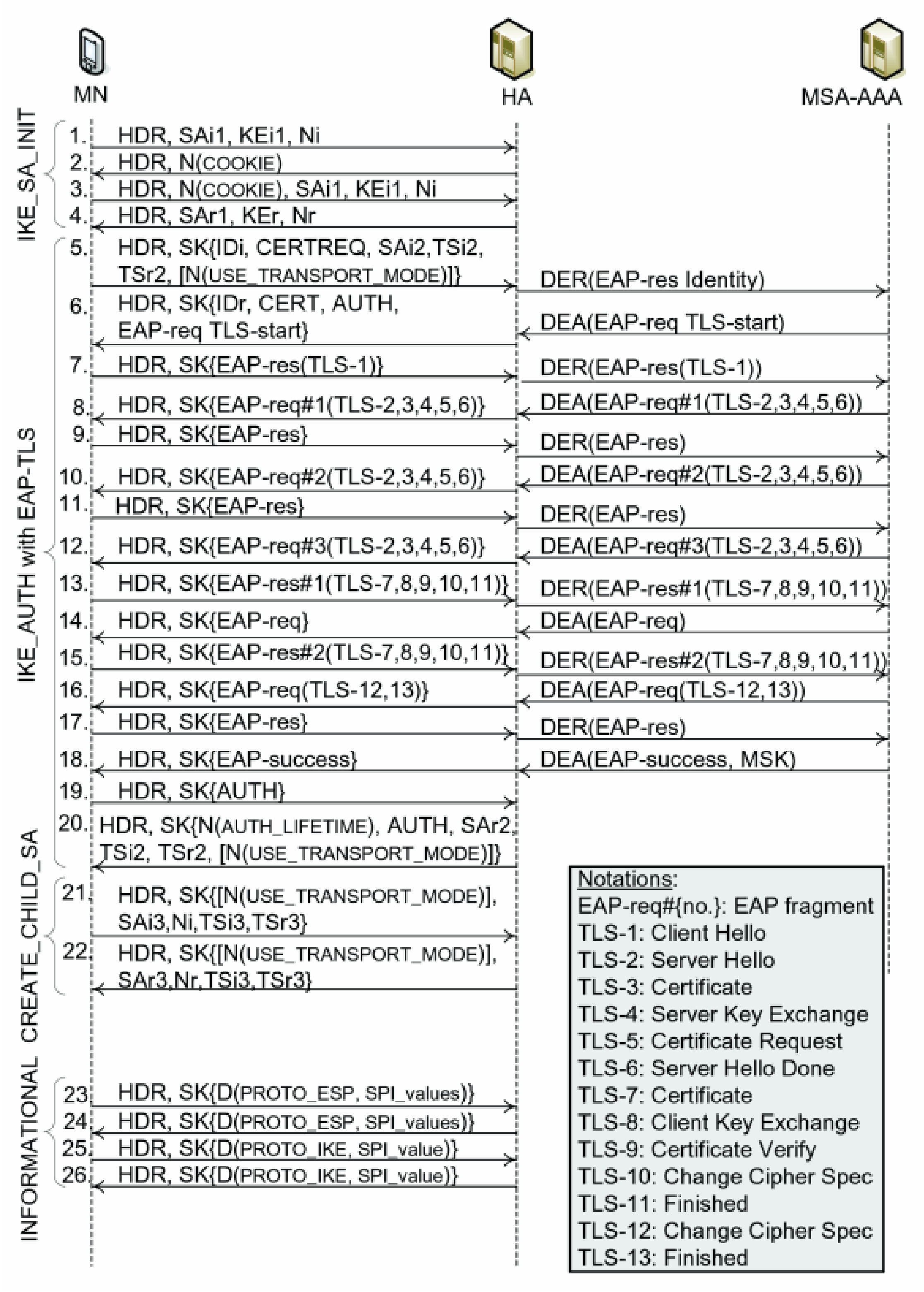

MN
HA
MSA-AAA
IKE_SA_INIT
IKE_AUTH with EAP-TLS
CREATE_CHILD_SA
INFORMATIONAL
1. HDR, SAi1, KEi1, Ni
2. HDR, N(COOKIE)
3. HDR, N(COOKIE), SAi1, KEi1, Ni
4. HDR, SAr1, KEr, Nr
5. HDR, SK{IDi, CERTREQ, SAi2,TSi2, TSr2, [N(USE_TRANSPORT_MODE)]}
DER(EAP-res Identity)
6. HDR, SK{IDr, CERT, AUTH, EAP-req TLS-start}
DEA(EAP-req TLS-start)
7. HDR, SK{EAP-res(TLS-1)}
DER(EAP-res(TLS-1))
8. HDR, SK{EAP-req#1(TLS-2,3,4,5,6)}
DEA(EAP-req#1(TLS-2,3,4,5,6))
9. HDR, SK{EAP-res}
DER(EAP-res)
10. HDR, SK{EAP-req#2(TLS-2,3,4,5,6)}
DEA(EAP-req#2(TLS-2,3,4,5,6))
11. HDR, SK{EAP-res}
DER(EAP-res)
12. HDR, SK{EAP-req#3(TLS-2,3,4,5,6)}
DEA(EAP-req#3(TLS-2,3,4,5,6))
13. HDR, SK{EAP-res#1(TLS-7,8,9,10,11)}
DER(EAP-res#1(TLS-7,8,9,10,11))
14. HDR, SK{EAP-req}
DEA(EAP-req)
15. HDR, SK{EAP-res#2(TLS-7,8,9,10,11)}
DER(EAP-res#2(TLS-7,8,9,10,11))
16. HDR, SK{EAP-req(TLS-12,13)}
DEA(EAP-req(TLS-12,13))
17. HDR, SK{EAP-res}
DER(EAP-res)
18. HDR, SK{EAP-success}
DEA(EAP-success, MSK)
19. HDR, SK{AUTH}
20. HDR, SK{N(AUTH_LIFETIME), AUTH, SAr2, TSi2, TSr2, [N(USE_TRANSPORT_MODE)]}
21. HDR, SK{[N(USE_TRANSPORT_MODE)], SAi3,Ni,TSi3,TSr3}
22. HDR, SK{[N(USE_TRANSPORT_MODE)], SAr3,Nr,TSi3,TSr3}
23. HDR, SK{D(PROTO_ESP, SPI_values)}
24. HDR, SK{D(PROTO_ESP, SPI_values)}
25. HDR, SK{D(PROTO_IKE, SPI_value)}
26. HDR, SK{D(PROTO_IKE, SPI_value)}
Notations:
EAP-req#{no.}: EAP fragment
TLS-1: Client Hello
TLS-2: Server Hello
TLS-3: Certificate
TLS-4: Server Key Exchange
TLS-5: Certificate Request
TLS-6: Server Hello Done
TLS-7: Certificate
TLS-8: Client Key Exchange
TLS-9: Certificate Verify
TLS-10: Change Cipher Spec
TLS-11: Finished
TLS-12: Change Cipher Spec
TLS-13: Finished

**Figure 7. IKEv2 with EAP-TLS peer authentication using Diameter authentication server and certificate-based authentication of the HA.**

## *3.3 Supported algorithms*

RFC 4307 recommends the following mandatory-to-implement algorithms for IKEv2.

### 3.3.1 Encryption in parent (IKE) SAs

- 3DES-CBC(MUST-)
- NULL (MAY)
- AES-128-CBC (SHOULD+)
- AES-CTR (SHOLUD)

### 3.3.2 Integrity in parent (IKE) SAs

- NONE
- HMAC-MD5-96 (MAY)
- HMAC-SHA1-96 (MUST)
- AUTH-AES-XCBC-96 (SHOULD+)

### 3.3.3 Pseudo-random function to generate shared secrets

- HMAC-MD5(MAY)
- HMAC-SHA1(MUST)
- AES128-CBC (SHOULD+)

### 3.3.4 Diffie-Hellman group value to generate master secret

- 1024 MODP Group (MUST-) given in
- 2048 MODP Group (SHOULD+) given in

# 4 Securing MIPv6 signaling with IPsec

Mobile IPv6 (MIPv6) is a signaling protocol which enables the movement of end-hosts in wireless networks. This part discusses how the protection of MIPv6 signaling traffic between the Mobile Node (MN) and Home Agent (HA) is recommended when IPsec and IKEv2 is used.

Mobile IPv6 and its extensions were described in details in a previous study with the title "Security threats in systems supporting IPv6 mobility and state-of-the art security solutions" . That study motivated the use of IPsec protection for the signaling traffic between the MNs and HAs. Moreover, other recommendations still exist to protect different parts of Mobile IPv6 signaling. It is interesting to discuss the protection of MIPv6 with IPsec and IKEv2, since this solution is referred to use also in case of the extensions of MIPv6, e.g., the Network Mobility protocol.

## *4.1 Overview of MIPv6 signaling*

In MIPv6, the following signaling messages are exchanged between the HA and the MN:

- Binding Update from the MN to the HA, and Binding Acknowledgment from the HA to the MN, as part of the binding update procedure, at the home registration. These messages are sent in the Mobility Header (MH).
- Home Test Init from the MN to the HA, and Home Test from the HA to the MN, as part of the return routability mechanism in the route optimization, at the correspondent registration. These messages are sent in the Mobility Header (MH).

These two message exchanges are executed at each movement, when the Care-of Address of the MN changes. First the home registration is executed, and then, if the MN is not in the home network, the correspondent registration is made.

Moreover, two other types of signaling messages may be transfered, but the transfer of these ones is not related to the movement of the MN. These signaling messages are the following ones:

- ICMPv6 Home Agent Disvovery Request and Reply message (for HA discovery), and the ICMPv6 Mobile Prefix Solicitation and Advertisement message (for home network prefix information). These messages are needed for the home network prefix information transfer to the MN and the HA address discovery.
- Signaling messages of multicast group membership protocols, and stateful address auto-configuration protocols (such as DHCPv6, signaling the change of home address of the MN).

## *4.2 Problem statement*

Protection of MIPv6 is needed to countermeasure potential threats related to multihoming and mobility. The main threat is the insertion of false Binding Cache Entries into the binding cache of the HA. This can lead to several attacks, such as:

- Basic address stealing
- Man-in-the-middle attacks, by impersonating two MNs, leading to secrecy and privacy violations
- Denial of Service attacks, blocking the communication with the MN
- Flooding of other nodes on the network
- Redirection attacks

For additional information on threats in different types of networks and services, especially for mobility and multihoming related threats, see .

## 4.3 Solution

In order to prevent potential threats related to MIPv6 signaling, one solution is to use IPsec. This is possible due to the fact, that the MN and the HA (or AAA) can have preliminary a registration phase. They have an established trust relationship, and the mobility service provider could distribute credentials to the users.

The signaling and control messages transmitted between the MN and HA require different protection types, since they are, or the mechanism that they are taking part is, sensitive in different ways for potential threats.

The Binding Updates and Acknowledgement messages require a non-NULL authentication algorithm and ESP in transport mode, in order to provide:

- Data origin authentication
- Connectionless integrity
- Optional anti-replay protection, which is fully protected if IKEv2 is used.
- Correct ordering: achieved by sequence numbers in BU and BA, and message authentication

The same is true for ICMPv6 prefix discovery messages.

The Home Test Init and Home Test messages require non-NULL authentication algorithm and non-NULL encryption algorithm, with ESP protocol in tunnel mode. The security requirements of these messages are the highest, because they need:

- Data origin authentication
- Connectionless integrity
- Confidentiality
- Optional anti-replay protection, which is fully protected if IKEv2 is used.
- Correct ordering: achieved by sequence numbers in BU and BA, and message authentication

The same requirements are needed for the signaling messages of multicast group protocols or stateful automatic address configuration protocols.

### 4.3.1 Policies

In order to provide appropriate protection levels, the MNs and HAs should have the security policy database entries, as described in this part. Part 4.3.1.1 describes the policies for basic IPsec processing without dynamic SA management, as described in Part 2.3.3.1. Part 4.3.1.2 describes the policies which trigger dynamic SA pair creation, as described in Part 2.3.3.2. In this case, the IKEv2 is applied for SA negotiation. The presented policies provide the finest granularity for security policies, since each signaling type gets its appropriate protection. Note, that this granularity level of policies is applicable only if the IPsec implementation supports fine traffic selectors .

The policies are in an ordered list, they can follow the order as it is in the description. The first matching policy entry for a packet is applied for the protection or “unprotection” (i.e. authenticity verification and/or decryption).

#### 4.3.1.1 Static configuration of SPD policies and SAD entries

In case of a static configuration, we have stationary SAD entries which should have an expiration time adapted to the manual refreshment rate of the system administrators. The SPD-S entries point out the SA to use.

When an SA entry has the attribute “OUT” it serves for protection of outbound traffic. When an SA entry has the attribute IN, then it serves to “unprotect” inbound traffic.

**Binding Updates (BU) and Acknowledgements (BAck)**

```
mobile node SPD-S:
  - IF source = home_address_1 & destination = home_agent_1 &
       proto = MH & local_mh_type =BU & remote_mh_type =
       BAck
    Then use SA SA1 (OUT) and SA2 (IN)

mobile node SAD:
  - SA1(OUT, spi_a, home_agent_1, ESP, TRANSPORT):
    source = home_address_1 & destination = home_agent_1 &
    proto = MH & mh_type = BU
  - SA2(IN, spi_b, home_address_1, ESP, TRANSPORT):
    source = home_agent_1 & destination = home_address_1 &
    proto = MH & mh_type = BAck

home agent SPD-S:
  - IF source = home_agent_1 & destination = home_address_1 &
       proto = MH & local_mh_type = BAck & remote_mh_type
       = BU
    Then use SA SA2 (OUT) and SA1 (IN)

home agent SAD:
  - SA2(OUT, spi_b, home_address_1, ESP, TRANSPORT):
    source = home_agent_1 & destination = home_address_1 &
    proto = MH & mh_type = BAck
  - SA1(IN, spi_a, home_agent_1, ESP, TRANSPORT):
    source = home_address_1 & destination = home_agent_1 &
    proto = MH & mh_type = BU
```

## Return Routabililty Messages

```
HoTi          Home Test Init
HoT           Home Test

        mobile node SPD-S:
          - IF source = home_address_1 & destination = any &
            proto = MH & local_mh_type = HoTi & remote_mh_type = HoT
            Then use SA SA3 (OUT) and SA4 (IN)

        mobile node SAD:
          - SA3(OUT, spi_c, home_agent_1, ESP, TUNNEL):
            source = home_address_1 & destination = any & proto = MH &
            mh_type = HoTi
          - SA4(IN, spi_d, care_of_address_1, ESP, TUNNEL):
            source = any & destination = home_address_1 & proto = MH &
            mh_type = HoT

        home agent SPD-S:
          - IF destination = home_address_1 & source = any &
            proto = MH & local_mh_type = HoT & remote_mh_type =
            HoTi
            Then use SA SA4 (OUT) and SA3 (IN)

        home agent SAD:
          - SA4(OUT, spi_d, care_of_address_1, ESP, TUNNEL):
            source = any & destination = home_address_1 & proto = MH &
            mh_type = HoT
          - SA3(IN, spi_c, home_agent_1, ESP, TUNNEL):
            source = home_address_1 & destination = any & proto = MH &
            mh_type = HoTi
```

## Mobile Prefix Discovery Messages

```
        mobile node SPD-S:
          - IF source = home_address_1 & destination = home_agent_1 &
               proto = ICMPv6 & local_icmp6_type = MPS &
               remote_icmp6_type = MPA
            Then use SA SA5 (OUT) and SA6 (IN)

        mobile node SAD:
          - SA5(OUT, spi_e, home_agent_1, ESP, TRANSPORT):
            source = home_address_1 & destination = home_agent_1 &
            proto = ICMPv6 & icmp6_type = MPS
          - SA6(IN, spi_f, home_address_1, ESP, TRANSPORT):
            source = home_agent_1 & destination = home_address_1 &
            proto = ICMPv6 & icmp6_type = MPA

        home agent SPD-S:
          - IF source = home_agent_1 & destination = home_address_1 &
               proto = ICMPv6 & local_icmp6_type = MPA &
               remote_icmp6_type = MPS
            Then use SA SA6 (OUT) and SA5 (IN)

        home agent SAD:
          - SA6(OUT, spi_f, home_address_1, ESP, TRANSPORT):
            source = home_agent_1 & destination = home_address_1 &
            proto = ICMPv6 & icmp6_type = MPA
          - SA5(IN, spi_e, home_agent_1, ESP, TRANSPORT):
            source = home_address_1 & destination = home_agent_1 &
          proto = ICMPv6 & icmp6_type = MPS
```

### Payload Packets

```
     mobile node SPD OUT:
       - IF interface = IPv6 tunnel to home_agent_1 &
            source = home_address_1 & destination = any &
            proto = X
         THEN USE SA SA7

     mobile node SPD IN:
       - IF interface = IPv6 tunnel from home_agent_1 &
            source = any & destination = home_address_1 &
            proto = X
         THEN USE SA SA8

     mobile node SAD:
       - SA7(OUT, spi_g, home_agent_1, ESP, TUNNEL):
         source = home_address_1 & destination = any & proto = X
       - SA8(IN, spi_h, care_of_address_1, ESP, TUNNEL):
         source = any & destination = home_address_1 & proto = X

     home agent SPD OUT:
       - IF interface = IPv6 tunnel to home_address_1 &
            source = any & destination = home_address_1 &
            proto = X
         THEN USE SA SA8

     home agent SPD IN:
       - IF interface = IPv6 tunnel from home_address_1 &
            source = home_address_1 & destination = any &
            proto = X
         THEN USE SA SA7

     home agent SAD:
       - SA8(OUT, spi_h, care_of_address_1, ESP, TUNNEL):
         source = any & destination = home_address_1 & proto = X
       - SA7(IN, spi_g, home_agent_1, ESP, TUNNEL):
         source = home_address_1 & destination = any & proto = X
```

## 4.3.1.2 Security policies for the dynamic configuration of SAs

This part presents the SPD-S policies when the system administrators prefer the dynamic SA configuration. This makes possible automatic rekeying of SAs.

### Binding Updates and Acknowledgements

```
       mobile node SPD-S:
         - IF source = home_address_1 & destination = home_agent_1 &
              proto = MH & local_mh_type = BU & remote_mh_type = BAck
           Then use SA ESP transport mode
                    IDi = user_1, IDr = home_agent_1,
                    TSi = home_address_1, MH, BU
                    TSr = home_agent_1, MH, BAck

       home agent SPD-S:
         - IF source = home_agent_1 & destination = home_address_1 &
```

```
        proto = MH & local_mh_type = BAck & remote_mh_type = BU
     Then use SA ESP transport mode
                 IDi = home_agent_1, IDr = user_1
                 TSi = home_agent_1, MH, BAck
                 TSr = home_address_1, MH, BU
```

## Return Routabililty Messages

```
mobile node SPD-S:
  - IF source = home_address_1 & destination = any &
        proto = MH & local_mh_type = HoTi &
        remote_mh_type = HoT
    Then use SA ESP tunnel mode
                 IDi = user_1, IDr = home_agent_1,
                 TSi = home_address_1, MH, HoTi
                 TSr = any, MH, HoT
                 outer src addr = care_of_address_1,
                 outer dst addr = home_agent_1,
                 inner src addr = home_address_1

home agent SPD-S:
  - IF source = any & destination = home_address_1 &
        proto = MH & local_mh_type = HoT &
        remote_mh_type = HoTi
    Then use SA ESP tunnel mode
                 IDi = home_agent_1, IDr = user_1
                 TSi = any, MH, HoT
                 TSr = home_address_1, MH, HoTi
                 outer src addr = home_agent_1,
                 outer dst addr = care_of_address_1,
                 inner dst addr = home_address_1
```

## Mobile Prefix Discovery Messages

```
mobile node SPD-S:
  - IF source = home_address_1 & destination = home_agent_1 &
        proto = ICMPv6 & local_mh_type = MPS &
        remote_mh_type = MPA
    Then use SA ESP transport mode
                 IDi = user_1, IDr = home_agent_1,
                 TSi = home_address_1, ICMPv6, MPS
                 TSr = home_agent_1, ICMPv6, MPA

 home agent SPD-S:
  - IF source = home_agent_1 & destination = home_address_1 &
        proto = ICMPv6 & local_mh_type = MPA &
        remote_mh_type = MPS
    Then use SA ESP transport mode
                 IDi = home_agent_1, IDr = user_1
                 TSi = home_agent_1, ICMPv6, MPA
                 TSr = home_address_1, ICMPv6, MPS
```

## Payload Packets

```
mobile node SPD-S:
  - IF interface = IPv6 tunnel to home_agent_1 & proto = X
    Then use SA ESP tunnel mode
                     IDi = user_1, IDr = home_agent_1,
                     TSi = home_address_1, X, port
                     TSr = any, X, port
                     outer src addr = care_of_address_1
                     outer dst addr = home_agent_1,
                     inner src addr = home_address_1

home agent SPD-S:
  - IF interface = IPv6 tunnel to home_address_1 & proto = X
    Then use SA ESP tunnel mode
                     IDi = home_agent_1, IDr = user_1,
                     TSi = any, X, port
                     TSr = home_address_1, X, port
                     outer src addr = home_agent_1,
                     outer dst addr = care_of_address_1,
                     inner dst addr = home_address_1
```

## 4.3.2 Interaction between MIPv6 and IPsec

IPsec and IKEv2 make the assumption that a peer have the same IPv6 address during the existence of SAs. Hence, after two peers authenticated each other and established an IKE SA pair, they can negotiate a huge number of child SAs without authenticating each other's identity.

### 4.3.2.1 Problem statement

The problem is that solely IPsec module can not track the IP address changes of local and remote IPsec entities.

Practically,  states that it is enough to establish the parent SA pair once in a day, and to regenerate child SA pairs in every 8 hours to keep the security at a reasonably high level. However, if the end-points of the SAs change their location, we need to negotiate new IKE (parent) SA pair and new child SA pairs. The more frequently the address changes, the higher performance cost the system suffers by the new SA negotiations. If this is not solved, then an IKE initialization process should be done before each binding update process, from the old location. Furthermore, depending on the granularity level of the policies, some new child SA pairs should also be negotiated, before home registration. This would lead to very high handover delays.

### 4.3.2.2 Basic idea

We would like to avoid repeated renegotiations of SAs, since these SAs were still secure, just became useless because their end-point could not be updated. The basic idea is to update the current SA and SPD entries at both sides by just changing the related local or remote IP address that has changed. This is will be only relevant in case of changing tunnel end IP addresses for IPsec (child) SA pairs.

### 4.3.2.3 Solution

The recommended solution for this problem is described in . MIPv6 applies two different methods to solve the mobility of child SAs. One is related with the SA pairs using ESP in transport mode, the other is related to SA pairs using ESP in tunnel mode. Furthermore, the referred standards also recommend a solution for the end-point change of parent (or IKE) SA pairs.

#### *4.3.2.3.1 Solution for the mobility of IKE SA pairs*

When a MN changes its care-of address (CoA), the parent SA pair between the MN and HA must be updated, since the parent SA becomes useless. Two possible solutions are described for the update of IKE SA pair:

- The first solution is for the case when no interaction between the MIPv6 and IPsec is supported. In this case, after each movement, the MN initiates the negotiation of an IKE SA pair between the new CoA and the HA address. Moreover, it creates the new child SA pairs for the protection of signaling, at least one pair for the protection of Binding Update and Acknowledgment messages. Then the Binding Update can update its location at the HA. During all these processes, the MN is not reachable by others.
- The second solution is, when both the HA and the MN support the change of the IKE SA entries, i.e., the update of the old CoA to the new CoA in the traffic selector part of the parent SAin and SAout entries. In this case, after the acquisition of the new CoA the binding update procedure can start immediately through the old child SA pair. The HA will use the HoA of the MN at the look-up of SAD, or if new child SA pair is needed for the binding update procedure, then the MN will use also its HoA to look-up the SPD-S policy, so the new CoA is not used at this phase. This is described in part 4.3.2.3.2.). If the binding update is successful, then the HA and the MN change the IKE SAin and IKE SAout entries by updating the address fields of the MN to the new CoA.

#### *4.3.2.3.2 Solution for transport mode child SAs*

The address change does not cause problem, because the MIPv6 module assures that the SAD and SPD look-up can be always based on the home address (HoA) of the MN.

The SPD-S and SAD policy entries contain the HoA of the MN at the local or remote IP address traffic selectors, and not the CoA. Consequently, if an inbound IP packet arrives to the HA or MN, and it contains a Home address destination option (in case of a packet sent from the MN) or a Type 2 Routing header (in case of a packet sent to the MN), then the MIPv6 module reads out the HoA from either of these packet headers, and substitutes the CoA in the IP header with the HoA. Then the MIPv6 module passes the packet to the IPsec module. In case of outbound packets, when the packet triggers the creation of a new SA pair, if the packet contains a Home address destination option field, that field must be used for the SPD look-up.

#### *4.3.2.3.3 Solution for tunnel mode child SAs – API needed from MIPv6 to IPsec module*

In case of tunnel mode protections, the policy look-up and SAD look-up can still be based on the HoA and the HA address, if MIPv6 supports the address CoA substitution in the source or destination field of the IP header. In tunnel mode, however, the original IP packet is extended with an outer IP header, which contains the addresses of the endpoints of the IPsec tunnel. If the end-point changes location, this has to be updated in the current SAs. The old CoA of the MN must be updated to the new CoA. This process can be made after a successful binding update, when the new BCE registration finished, and before the route optimization procedure. The procedure needs an API between the MIPv6 and the IPsec module.

Additionally, when a mobile node turns back to the home network, and de-registers the BCE, the security policy entries for tunnel mode between the MN and HA must be inactivated through the use of the API between the IPsec and MIPv6 module.

### 4.3.3 Movements and dynamic keying

The following part presents the interaction between MIPv6 and IPsec module by describing a chain of movement processes. It is interesting to see, that if the MN or the HA do not have the capability to update the IKE (parent) SA endpoints, then after each movements they must establish a new IKE SA pair containing the new CoA of the node. Then this is followed by the negotiation of the child SA pairs to protect signaling. That means, that in case of no mobility support the whole IKEv2 reauthentication should be done. The binding update process can be made only after this.

The remaining part of part 4.3.3 is cited from RFC 3776 and describes the dynamic keying in case of having or not having mobility support. It describes the process with IKEv1, but this is the same for IKEv2. In IKEv1, Phase 1 is for generating the parent or IKE SA pair. Phase 2 is to generate child SAs.

“In this section we describe the sequence of events that relate to movement with IKE-based security associations. In the initial state, the mobile node is not registered in any location and has no security associations with the home agent. Depending on whether the peers will be able to move IKE endpoints to new care-of addresses, the actions taken in Step 9 and 10 are different.

**Step 1.** Mobile node with the home address A moves to care-of address B.
**Step 2.** Mobile node runs IKE from care-of address B to the home agent, establishing a phase 1. The home agent can only act as the responder before it knows the current location of the mobile node.
**Step 3.** Protected by this phase 1, mobile node establishes a pair of security associations for protecting Mobility Header traffic to and from the home address A.
**Step 4**. Mobile node sends a Binding Update and receives a Binding Acknowledgement using the security associations created in Step 3.
**Step 5.** Mobile node establishes a pair of security associations for protecting return routability packets. These security associations are in tunnel mode and their endpoint in the mobile node side is care-of address B. For the purposes of our example, this step uses the phase 1 connection established in Step 2. Multiple phase 1 connections are also possible.
**Step 6.** The mobile node uses the security associations created in Step 5 to run return routability.

**Step 7.** The mobile node moves to a new location and adopts a new care-of address C.
**Step 8**. Mobile node sends a Binding Update and receives a Binding Acknowledgement using the security associations created in Step 3. The home agent ensures that the next packets sent using the security associations created in Step 5 will have the new care-of address as their destination address, as if the outer header destination address in the security association had changed.

**Step 9.** If the mobile node and the HA have the capability to change the IKE endpoints, they change the address to C. If they do not have the capability, both nodes remove their phase 1 connections created on top of the care-of address B and will establish a new IKE phase 1 on top of the care-of address C. This capability to change the IKE phase 1 end points is indicated through setting the Key Management Mobility Capability (K) flag [7] in the Binding Update and Binding Acknowledgement messages.
**Step 10.** If a new IKE phase 1 connection was setup after movement, the MN will not be able to receive any notifications delivered on top of the old IKE phase 1 security association. Notifications delivered on top of the new security association are received and processed normally. If the mobile node and HA were able to update the IKE endpoints, they can continue using the same IKE phase 1 connection. "

The same process is made in case of IKEv2. This is described in .

"If the mobile node moves and its care-of address changes, the IKEv2 SA might not be valid. The mobile node establishes the IKE SA with the home agent using its primary care-of address. The IKE SA endpoints are updated on the home agent when it receives the Binding Update from the mobile node's new care-of address and on the mobile node when it sends the Binding Update to the home agent or when it receives the Binding acknowledgement sent by the home agent. This capability to change IKE endpoints is indicated through setting the Key Management Capability (K) flag in the Binding Update and Binding Acknowledgement messages. If the mobile node or the home agent does not support this capability, and has no other means to update the addresses, then an IKEv2 exchange MUST be initiated to re-establish a new IKE SA."

## *4.4 Bootstrapping question*

The mip6 Working Group of IETF currently works on several standards and drafts, such as , , and , which cover the bootstrapping phase of the MN in different use scenarios. Basically, these documents deal with the problem of how to transfer the initialization data from the home network to the MN, when it is booted up. The solution of this problem enables an easier deployment of MIPv6. Moreover, the referred documents discuss how to establish the IKE SA pair with the HA, and which authentication method to use.

RFC 4640 discusses the problems for bootstrapping the MN. In the basic MIPv6 standard there is an implicit requirement that the MN must be provisioned with enough information that will permit to the MN to perform the first home registration. The configuration parameters needed by a MN are typically:

- Its home address (HoA)
- The HA address, which is maintainable through dynamic home agent assignment
- The MN and the HA may want to share some cryptographic material, and generate keys for child SAs
- Automatic SPD and PAD management for MNs and HAs

RFC 4640 and the related drafts suggest the application of existing AAA infrastructure, in order to authenticate users based on the credentials and secrets that are stored in the AAA server. The EAP-based authentication methods, supported by IKEv2, seem to be the preferred way to authenticate the users of the MNs. The AAA infrastructure can be used for the authentication of the user identities based on credentials stored in the AAA, authorization of protocol operations, and account and credit controls. Moreover, an AAA server may also provide the required configuration parameters for the bootstrapping phase of the MN. The AAA servers may take off the peer authorization task from the HAs (since HAs originally have the Peer Authorization Database, but the HA might not be the convenient location to perform authorization tasks).

The bootstrapping problem is discussed in two basic scenarios:

- the integrated scenario , and
- the split scenario .

In the integrated scenario, the authorization service and mobility service is provided by the same administrative domain, while in the split scenario they establish two separate administrative domains. Split scenario is more complex since it needs more serious security considerations between the HAs and AAA servers.

# 5 Summary

IPsec and IKEv2 provide configurability at three levels. Firstly, IKEv2 defines three main authentication types: pre-shared key, certificate-based, and EAP-based. Secondly, IPsec enables the definition of different granularity levels of the security policies that may have different security levels, and induce different performance costs. Thirdly, the SA configuration is also tunable because a wide variety of algorithm selections, both for IKEv2 and IPsec.

When we use IPsec for the protection of MIPv6 mobility service, it operates in the same way as for any kind of IP traffic protection. The security associations expire and need rekeying periodically. IKEv2 is the default SA management protocol for IPsec.

MIPv6 primarily has influence on the definition of security policies, by the specific nature and information sensitivity of different signaling messages and processes. The definition of the granularity of security policies is up to the system designers or administrators.

The stationary nature of SAs causes that the MIPv6 module needs interaction with the IPsec module at each movement of the MN. This interaction is needed at both sides, in the HA and the MN. Otherwise, if either of the entities does not solve this, then at each movement, the MN may trigger the whole IKEv2 initialization process before the binding update process can start. Normally, we have IKE SA refreshment processes and mobility signaling processes should run independently and in parallel.

# 6 References

- Z. Faigl, “Security threats in systems supporting IPv6 mobility and state-of-the art security solutions”, BME, June 2007, URL: http://www.ist-anemone.org/publications .
- S. Kent, K. Seo, “Security Architecture for the Internet Protocol”, RFC 4301, December 2005.
- S. Kent, “IP Encapsulating Security Payload (ESP)”, RFC 4303, December 2005.

- S. Kent, "IP Authentication Header ", RFC 4302, December 2005.
- C. Kaufman, "Internet Key Exchange (IKEv2) Protocol", RFC 4306, December 2005.
- P. Eronen, P. Hoffman, "IKEv2 Clarifications and Implementation Guidelines", RFC 4718, October 2006.
- H. Tschofenig, D. Kroeselberg, A. Pashalidis, Y. Ohba, F. Bersani, "EAP IKEv2 Method", draft-tschoefenig-eap-ikev2-12.txt, October 23, 2006.
- P. Hoffman, "Cryptographic Suites for IPsec", RFC 4308, December 2005.
- J. Schiller, "Cryptographic Algorithms for Use in the Internet Key Exchange Version 2 (IKEv2)", RFC 4307, December 2005.
- D. Eastlake 3rd, "Cryptographic Algorithm Implementation Requirements for Encapsulating Security Payload (ESP) and Authentication Header (AH)", RFC 4305, December 2005.
- R. Housley, "Using Advanced Encryption Standard (AES) CCM Mode with IPsec Encapsulating Security Payload (ESP)", RFC 4309, December 2005.
- EAP-IKEv2 Project, URL: http://eap-ikev2.ccns.pl, November 27, 2006.
- strongSwan: URL: http://www.strongswan.org/home.htm, November 27, 2006.
- S. Sugimoto, F. Dupont, M. Nakamura, "PF_KEY Extension as an Interface between Mobile IPv6 and IPsec/IKE", draft-sugimoto-mip6-pfkey-migrate-03.txt, September 19, 2006.
- J. Arkko, V. Devarapalli, F. Dupont, "Using IPsec to Protect Mobile IPv6 Signaling Between Mobile Nodes and Home Agents", RFC 3776, June 2004.
- V. Devarapalli, F. Dupont, "Mobile IPv6 Operation with IKEv2 and the revised IPsec Architecture", draft-ietf-mip6-ikev2-ipsec-07.txt, October 27, 2006.
- A. Patel, G. Giaretta "Problem Statement for Bootstrapping Mobile IPv6 (MIPv6)", RFC 4640, September 2006.
- G. Giaretta, I. Guardini, E. Demaria, J. Bournelle, R. Lopez, "AAA Goals for Mobile IPv6", draft-ietf-mip6-aaa-ha-goals-03, September 12, 2006.
- K. Chowdhury, A. Yegin, "MIP6-bootstrapping via DHCPv6 for the Integrated Scenario", draft-ietf-mip6-bootstrapping-integrated-dhc-01.txt, June 9, 2006.
- G. Giaretta, J. Kempf, V. Devarapalli, "Mobile IPv6 bootstrapping in split scenario", draft-ietf-mip6-bootstrapping-split-03.txt, October 20, 2006.
- E. Rescorla, "Diffie-Hellman Key Agreement Method", RFC 2631, June 1999.
- WPA supplicant, URL: http://hostap.epitest.fi/wpa_supplicant/, 12 December 2006.
- FreeRadius, URL: http://www.freeradius.org/, 12 December, 2006.
- D. Johnson, C. Perkins, J. Arkko, "Mobility Support in IPv6, " RFC 3775, June 2004.
- V. Devarapalli, R. Wakikawa, A. Petrescu, P. Thubert, "Network Mobility (NEMO) Basic Support Protocol, " RFC 3963, January 2005.
- Y. Nir, "RFC 4478: Repeated Authentication in Internet Key Exchange (IKEv2) Protocol," April 2006.

# 7 Appendices

## 7.1 Abbreviations

| | |
|---|---|
| AH | Authentication Header |
| CN | Correspondent Node |
| ESP | Encapsulating Security Payload |
| HA | Home Agent |
| IKEv2 | Internet Key Exchange Protocol version 2 |
| IPsec | Internet Security Protocol |
| MIPv6 | Mobile IPv6 (Mobility support for the IPv6 protocol) |
| MN | Mobile Node |
| PAD | Peer Authorization Database |
| SA | Security Association |
| SAD | Security Association Database |
| SPD | Security Policy Database |

## 7.2 The contents of SPD entries

This part summarizes the contents of a security policy entry in SPD. The detailed description of the entries can be found in .

- One to N traffic selectors: these selectors specify the traffic on which to apply the specific policy. The selectors can contain a specific value or a range of values, and are as follows:
  - Local Address
  - Remote Address
  - Next layer protocol (TCP, UDP, SCTP, ICMP, MH, etc)
  - Local port / ICMP message type / MIPv6 signaling message type
  - Remote port / ICMP message code
- PFP flags: The “populate from packet” (PFP) flags influence the negotiation of SAs, i.e., which traffic selector will a peer propose for a given SA. If the “populate from packet” (PFP) flag is true for a given traffic selector, then the TS proposal will include a value populated from the related field of the packet.. If the PFP flag is false, then the negotiated traffic selector value will contain the value or range written in the traffic selector attribute of the SPD entry.
- Name: an optional value, it is used to identify SPD entries. If the name value is used, then the entry is a “named SPD entry”. In fact, the look-up of the policies can be performed in two fundamental ways, dictated by PAD:
  - The first is the matching some of the fields of the IP packet to the traffic selectors in the policies. This happens during the data transfer phase of outbound IP packets. Policy look-up based on traffic selectors can also happen, when creating SAs with IKE negotiation, the proposed traffic

selector of the peer (TSi sent by the peer) may be used to look-up the security policies, and find matching entries, then negotiate the SA pair based on the entry values.

  - The second fundamental way is the use of named SPD entries for policy look-up. Two possible cases are mentioned in the IPsec standard for the use of named policy entries:
    The named policy entry is used by the responder during the IKE negotiation. The responder peer look-up its SPD based on the IKE IDi field of the initiator peer. During the negotiation of the child SA pair, however, the IP address of the initiator will be used in the instantiated SA pairs. This is advantageous when a remote peer may change always its IP addresses. The ID of the responder may be one of the four types:
      - Fully qualified user name string (email or RFC 822 address)
      - Fully qualified DNS name (FQDN)
      - X.500 Distinguished name (DER ASN1 DN ID)
      - Byte string

    Named SPD policies may be used by an initiator, when it is a multi-user host, and different users have different policies. This is interesting at the SA pair creation phase. The SPD is looked up based on the identity of the user, such as Unix UID, Windows Security ID, user name, account name etc. But the instantiated SA pair will contain the IP address of the user, i.e., the local address in the outbound SAD entry and the remote address in the inbound SAD entry will be the IP address of the user. Also the SPD-S cache entry will contain the IP address at the local address traffic selector.
- Processing:
  - Discard
  - Bypass
  - Protect. If protect, then there exists the next entry attributes:
- IPsec mode: tunnel/transport
- If tunnel mode, then the local and remote outer IP header addresses
- Use extended sequence number or not? Provides anti-replay protection, i.e., some weak integrity of ordering
- IPsec protocol: AH/ESP
- Algorithms for the protocol: integrity algorithm for AH; encryption and integrity algorithm or combined algorithm for ESP. List of possible settings in a decreasing priority order.
- Stateful fragment checking (in case of tunnel mode): this is related to fragmentation of packets. For more details on fragmentation, see the IPsec standard .
- Bypass DF bit (in case of tunnel mode)
- Bypass DSCP (in case of tunnel mode)

One SPD entry may result in the creation of a range of SAs. This is due to the possible uses of the selector range values and the “populate from packet” flags.

## 7.3 The contents of SAD entries

This part summarizes the contents of a security association (SA) entry in the Security Association Database (SAD). The detailed description of the entries can be found in .

- Security Parameters Index (SPI): The SPI, and in some cases the destination and source address in an inbound IP packet index the SA entry in the SAD.
- Security protocol: AH or ESP.
- Anti-replay protection:
  - Sequence number counter
  - Sequence counter overflow
  - Anti-replay window
- Security protocol configuration:
  - AH:
    - authentication key, algorithm
  - ESP:
    - encryption algorithm, key, mode, initialization vector
    - integrity algorithm, key etc.
    - combined mode algorithm, keys etc.
- IPsec protocol mode: tunnel or transport
- Lifetime: the standard recommends soft and hard lifetimes. In case of reaching the soft lifetime, a rekeying notification is triggered. In case of reaching the hard lifetime, the host deletes the SA and sends a delete message (an IKE notification) to delete it at the remote peer.
  - Timer based
  - Byte counter based
- Permitted DSCP values (QoS protection)
- Path MTU
- Bypass DSCP value, or map from inner header to outer IP header (in case of tunnel mode)
- Outer IP header, the source and destination IP address (in case of tunnel mode)
- Stateful fragmentation checking flag
- Bypass DF bit True/false

The lifetime of the SAs determine the refreshment rates of parent and child SAs. This has effect on the performance cost of IKE negotiations.

## 7.4 The contents of PAD entries

This part summarizes the contents of a peer authorization entry in the Peer Authorization Database (PAD). The detailed description of the entries can be found in .

- Peer ID or group: it is the index to find the appropriate entry based on the IKE ID field of the remote peer.
  - DNS name: it can be one specific name or a sub-tree

  - X.500 Distinguished name: it can be complete or sub-tree
  - RFC 822 email address: it can be complete or partially qualified
  - IPv4/IPv6 address range: it can be complete or range
  - Key ID: it must exactly match. Used with pre-shared key authentication
- Authentication method and supplementary authentication information
  - Authentication protocol: IKE/IKEv2/KINK
  - Authentication method and supplementary data for the authentication. In case of certificate-based authentication, the method is, e.g., X.509 signature, the data is the trust anchor point via the certificate of the peer is verifiable. In case of pre-hared key authentication the additional data is the pre-shared key.
- SA constraints: The PAD specifies the TSi addresses or symbolic names that the remote peer is authorized to represent when (child) SA are negotiated.
- SPD look-up method, when creating new child SAs
  - Based on the TSi of the remote peer
  - Based on the IKE IDi of the remote peer.