Network Working Group
Internet-Draft
Expires: August 12, 2005

P. Jokela
Ericsson Research NomadicLab
R. Moskowitz
ICSAlabs, a Division of TruSecure Corporation
P. Nikander
Ericsson Research NomadicLab

February 11, 2005

Using ESP transport format with HIP
draft-jokela-hip-esp-00

Status of this Memo

Copyright Notice

Abstract

This memo specifies an Encapsulated Security Payload (ESP) based mechanism for transmission of user data packets, to be used with the Host Identity Protocol (HIP).

Table of Contents

1. Introduction

In the Host Identity Protocol Architecture, [8], hosts are identified with public keys. The Host Identity Protocol [5] base exchange allows any two HIP-supporting hosts to authenticate each other and to create a HIP association between themselves. During the base exchange, the hosts generate a piece of shared keying material using an authenticated Diffie-Hellman exchange.

The HIP base exchange specification [5] does not describe any transport formats or methods for user data, to be used during the actual communication; it only defines that it is mandatory to implement the Encapsulated Security Payload (ESP) [4] based transport format and method. This document specifies how ESP is used with HIP to carry actual user data.

To be more specific, this document specifies a set of HIP protocol extensions and their handling. Using these extensions, a pair of ESP Security Associations (SAs) is created between the hosts during the base exchange. The resulting ESP Security Associations use keys drawn from the keying material, KEYMAT, generated during the base exchange. After the HIP association and required ESP SAs have been established between the hosts, the user data communication is protected using ESP.

It should be noted that HIs, HITs, or LSIs are not carried explicitly in the headers of user data packets. Instead, the ESP Security Parameter Index (SPI) is used to indicate the right host context. The SPIs are selected during the HIP ESP setup exchange. For user data packets, the combination of ESP SPIs and IP addresses are used indirectly to identify the host context, thereby avoiding any additional explicit protocol headers.

2. Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC2119 [1].

3. Overview

3.1 Using ESP with HIP

The HIP base exchange is used to set up a HIP association between two hosts. The base exchange provides two-way host authentication and key material generation, but it does not provide any means for protecting data communication between the hosts. In this document we specify the use of ESP for protecting user data traffic after the HIP base exchange. Note that this use of ESP is intended only for host-to-host traffic; security gateways are not supported.

It should be noted that the HIP use of ESP differs somewhat from the standard IPsec use of ESP (see the rest of this document for the details). However, it is possible to support the HIP way of using ESP with a fully standards compliant IPsec implementation by adding the necessary header rewriting mechanisms below IPsec in the stack. As these mechanisms can be located at the network side of IPsec, such an implementation cannot add any integrity or confidentiality problems that would not be present in the implementation and configuration without the addition. However, explicit care must be taken to avoid introducing any new denial-of-service attacks.

To support ESP use, the HIP base exchange messages require some minor additions to the parameters transported. In the R1 packet, the responder adds the possible ESP transforms in a new ESP_TRANSFORM parameter before sending it to the Initiator. The Initiator gets the proposed transforms, selects one of those proposed transforms, and sets it in I2 packet in an ESP_TRANSFORM parameter. In this I2 packet, the Initiator also sends the SPI value that it wants to be used for ESP traffic flowing from the Responder to the Initiator. This information is carried using the new ESP_INFO parameter. When finalizing the ESP SA setup, the Responder sends its SPI value to the Initiator in the R2 packet.

The initial session keys are drawn from the generated keying material, KEYMAT, after the HIP keys have been drawn as specified in [5].

When the HIP association is removed, also the related ESP SAs MUST be removed.

An existing HIP-created ESP SA may need updating during the lifetime of HIP association. This documents specifies the rekeying of an existing HIP-created ESP SA, using the UPDATE message. The ESP_INFO parameter introduced above is also used for this purpose.

In the rest of this document, an unqualified mention of ESP SA is

implicitly understood to refer to a HIP-created ESP SA, unless otherwise explicitly mentioned.

3.2 Semantics of the Security Parameter Index (SPI)

SPIs are used in ESP to find the right Security Association for received packets. The ESP SPIs have added significance when used with HIP; they are a compressed representation of a pair of HITs. Thus, SPIs MAY be used by intermediary systems in providing services like address mapping. Note that since the SPI has significance at the receiver, only the < DST, SPI >, where DST is a destination IP address, uniquely identifies the receiver HIT at any given point of time. The same SPI value may be used by several hosts. A single < DST, SPI > value may denote different hosts and contexts at different points of time, depending on the host that is currently reachable at the DST.

Each host selects for itself the SPI it wants to see in packets received from its peer. This allows it to select different SPIs for different peers. The SPI selection SHOULD be random; the rules of Section 2.1 of the ESP specification [4] must be followed. A different SPI SHOULD be used for each HIP exchange with a particular host; this is to avoid a replay attack. Additionally, when a host rekeys, the SPI MUST be changed. Furthermore, if a host changes over to use a different IP address, it MAY change the SPI.

One method for SPI creation that meets the above criteria would be to concatenate the HIT with a 32-bit random or sequential number, hash this (using SHA1), and then use the high order 32 bits as the SPI.

The selected SPI is communicated to the peer in the third (I2) and fourth (R2) packets of the base HIP exchange. Changes in SPI are signaled with ESP_INFO parameters.

4. Details of using ESP with HIP

HIP does not negotiate any lifetimes. All ESP lifetimes are local policy. The only lifetimes a HIP implementation MUST support are sequence number rollover (for replay protection), and SHOULD support timing out inactive ESP SAs. An SA times out if no packets are received using that SA. The default timeout value is 15 minutes. Implementations MAY support lifetimes for the various ESP transforms.

4.1 A note on implementation options

It is possible to implement this specification in multiple different ways. As noted above, one possible way of implementing is to rewrite IP headers below IPsec. In such an implementation, IPsec is used as if it was processing IPv6 transport mode packets, with the IPv6 header containing HITs instead of IP addresses in the source and destionation address fields. In outgoing packets, after IPsec processing, the HITs are replaced with actual IP addresses, based on the HITs and the SPI. In incoming packets, before IPsec processing, the IP addresses are replaced with HITs, based on the SPI in the incoming packet. In such an implementation, all IPsec policies are based on HITs and the upper layers only see packets with HITs in the place of IP addresses. Consequently, support of HIP does not conflict with other use of IPsec as long as the SPI spaces are kept separate.

Another way for implementing is to use the proposed BEET mode (A Bound End-to-End mode for ESP) [10]. The BEET mode provides some features from both IPsec tunnel and transport modes. The HIP uses HITs as the "inner" addresses and IP addresses as "outer" addresses like IP addresses are used in the tunnel mode. Instead of tunneling packets between hosts, a conversion between inner and outer addresses is made at end-hosts and the inner address is never sent in the wire after the initial HIP negotiation. BEET provides IPsec transport mode syntax (no inner headers) with limited tunnel mode semantics (fixed logical inner addresses - the HITs - and changeable outer IP addresses).

Compared to the option of implementing the required address rewrites outside of IPsec, BEET has one implementation level benefit. The BEET-way of implementing the address rewriting keeps all the configuration information in one place, at the SADB. On the other hand, when address rewriting is implemented separately, the implementation must make sure that the information in the SADB and the separate address rewriting DB are kept in synchrony. As a result, the BEET mode based way of implementing is RECOMMENDED over the separate implementation.

4.2 ESP Security Associations

Each HIP association is linked to two ESP SAs, one for incoming packets and one for outgoing packets. The Initiator's incoming SA corresponds with the Responder's outgoing one, and vice versa. The initiator defines the SPI for the former association, as defined in Section 3.2. This SA is called SA-RI, and the corresponding SPI is called SPI-RI. Respectively, the Responder's incoming SA corresponds with the Initiator's outgoing SA and is called SA-IR, with the SPI being called SPI-IR.

The Initiator creates SA-RI as a part of R1 processing, before sending out the I2, as explained in Section 7.4. The keys are derived from KEYMAT, as defined in Section 8. The Responder creates SA-RI as a part of I2 processing, see Section 7.5.

The Responder creates SA-IR as a part of I2 processing, before sending out R2; see Section 7.5. The Initiator creates SA-IR when processing R2; see Section 7.6.

4.3 Updating ESP SAs and rekeying

After the initial HIP base exchange and SA establishment, both hosts are in the ESTABLISHED state. There are no longer Initiator and Responder roles and the association is symmetric. In this subsection, the party that initiates the rekey procedure is denoted with I' and the peer with R'.

I' initiates the rekeying process when needed (see Section 7.8). It creates an UPDATE packet with required information and sends it to the peer node. The old SAs are still in use, local policy permitting.

R', after receiving and processing the UPDATE (see Section 7.9), generates new SAs: SA-I'R' and SA-R'I'. It does not take the new outgoing SA into use, but uses still the old one, so there termporarily exists two SA pairs towards the same peer host. The SPI for the new outgoing SA, SPI-R'I', is picked from the received UPDATE packet. For the new incoming SA, R' generates the new SPI value, SPI-I'R', and includes it in the response UPDATE packet.

When I' receives a response UPDATE from R', it generates new SAs, as described in Section 7.9: SA-I'R' and SA-R'I'. It starts using the new outgoing SA immediately.

R' starts using the new outgoing SA when it receives traffic on the new incoming SA. After this, R' can remove the old SAs. Similarly, when the I' receives traffic from the new incoming SA, it can safely

remove the old SAs.

4.4 Security Association Management

An SA pair is indexed by the 2 SPIs and 2 HITs (both local and remote HITs since a system can have more than one HIT). An inactivity timer is RECOMMENDED for all SAs. If the state dictates the deletion of an SA, a timer is set to allow for any late arriving packets.

4.5 Security Parameter Index (SPI)

The SPIs in ESP provide a simple compression of the HIP data from all packets after the HIP exchange. This does require a per HIT-pair Security Association (and SPI), and a decrease of policy granularity over other Key Management Protocols like IKE.

When a host rekeys, it gets a new SPI from its partner.

4.6 Supported Transforms

All HIP implementations MUST support AES [3] and HMAC-SHA-1-96 [2]. If the Initiator does not support any of the transforms offered by the Responder it should abandon the negotiation and inform the peer with a NOTIFY message about a non-supported transform.

In addition to AES, all implementations MUST implement the ESP NULL encryption and authentication algorithms. These algorithms are provided mainly for debugging purposes, and SHOULD NOT be used in production environments. The default configuration in implementations MUST be to reject NULL encryption or authentication.

4.7 Sequence Number

The Sequence Number field is MANDATORY when ESP is used with HIP. Anti-replay protection MUST be used in an ESP SA established with HIP. This means that each host MUST rekey before its sequence number reaches 2^32, or if extended sequence numbers are used, 2^64.

In some instances, a 32-bit sequence number is inadequate. In the ESP_TRANSFORM parameter, a peer MAY require that a 64-bit sequence numbers be used. In this case the higher 32 bits are NOT included in the ESP header, but are simply kept local to both peers. 64-bit sequence numbers must only be used for ciphers that will not be open to cryptanalysis as a result. AES is one such cipher.

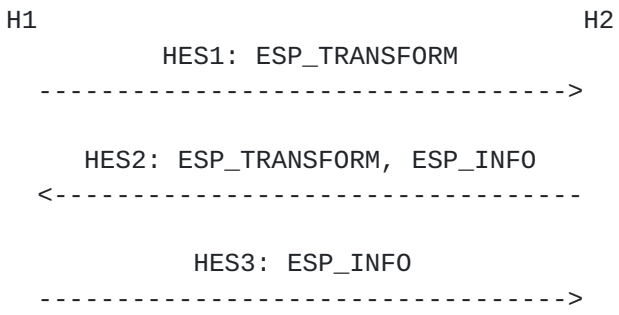
5. The protocol

In this section, the protocol for setting up an ESP association to be used with HIP association is described.

5.1 ESP in HIP

5.1.1 Setting up an ESP Security Association

Setting up an ESP Security Association between hosts using HIP consists of three conceptual messages passed between the hosts. The reader should note that these conceptual messages are not sent as separate messages but mapped onto other HIP messages; see below.



Setting up an ESP Security Association between HIP hosts requires three messages. During the set up, the hosts exchange information about the used protocols and other related information that is required during an ESP communication. As the messages are described in conceptual level, no actual HES packets are defined. In a typical implementation, the required parameters are included in R1, I2, and R2 messages. However, the messages can be transmitted also after the HIP assocation setup in UPDATE messages.

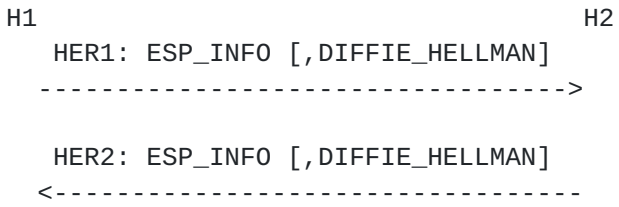
The HES1 message contains the ESP_TRANSFORM parameter, in which the sending host defines the possible ESP transforms it is willing to use for the ESP SA.

The HES2 message contains the response to a HES1 message. The sender must select one of the proposed ESP transforms from the HES1 packet and include the selected one in the ESP_TRANSFORM parameter in HES2 packet. In addition to the transform, the host includes the ESP_INFO parameter, containing the SPI value to be used by the peer host.

In the HES3 message, the ESP SA setup is finalized. The packet contains the SPI information required by the host H1 for the ESP SA.

5.1.2 Updating an existing ESP SA

The update process is accomplished using two messages. The messages are again conceptual. In a typical implementation the required parameters are sent in HIP UPDATE messages.



The host willing to update the ESP SA creates and sends a HER1 message. The message contains the ESP_INFO parameter, containing the old SPI value that was used, the new SPI value to be used, and the index value for the keying material, giving the point from where the next keys will be drawn. If new keying material must be generated, the HER1 message will contain also the DIFFIE_HELLMAN parameter, defined in [5].

The host receiving the HER1 message MUST reply with a HER2 message. In HER2, the host sends the ESP_INFO parameter containing the corresponding values: old SPI, new SPI, and the keying material index. If the incoming HER1 contained a DIFFIE_HELLMAN parameter, the HER2 MUST also contain a DIFFIE_HELLMAN parameter.

In a typical HIP implementation the required parameters are transmitted in UPDATE messages, as described in Section 6.3.

6. Parameter and packet formats

In this section, new and modified HIP parameters are presented, as well as modified HIP packets.

6.1 New parameters

Two new HIP parameters are defined for setting up ESP transport format associations in HIP communication and for rekeying existing ones. Also, the NOTIFY parameter, described in [5], has two new error parameters.

Parameter	Type	Length	Data
ESP_INFO	1	12	Remote's old SPI, new SPI and other info
ESP_TRANSFORM	19	variable	ESP Encryption and Authentication Transform(s)

6.1.1 ESP_INFO

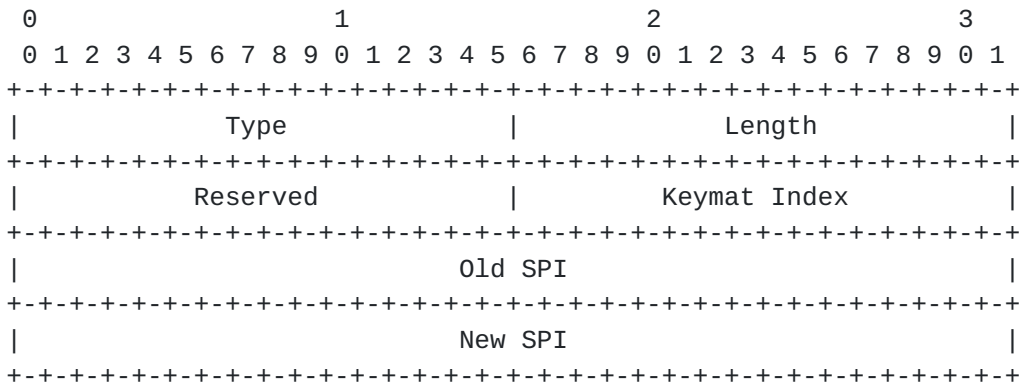
During establishment and updating an ESP SA, the SPI value of both hosts must be transmitted between the hosts. An additional information that is required when the hosts are drawing keys from the generated keying material is the index value from where the keys are retrieved. The ESP_INFO parameter is used to transmit this information between the hosts.

During the initial ESP SA setup, the hosts send the SPI value that they want the peer to use when sending ESP data to them. The value is set in the New SPI field of the ESP_INFO parameter. In the initial setup, there does not exist any old value for the SPI, thus the Old SPI value field is set to zero. The Old SPI field value may also be zero when additional SAs are set up between HIP hosts, e.g. in case of multihomed HIP hosts [6]. However, such use is beyond the scope of this specification.

The Keymat index value points to the place in keymat from where the keying material for the ESP SAs is drawn. The Keymat index value is zero only when the ESP_INFO is sent during a rekeying process and new keying material is generated.

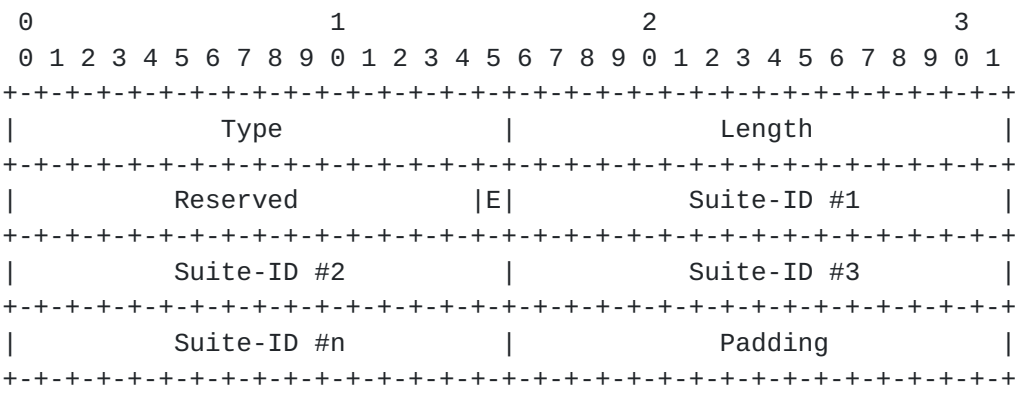
During the life of an SA established by HIP, one of the hosts may need to reset the Sequence Number to one (to prevent wrapping) and rekey. The reason for rekeying might be an approaching sequence number wrap in ESP, or a local policy on use of a key. Rekeying ends the current SAs and starts new ones on both peers.

During the rekeying process, the ESP_INFO parameter is used to transmit the changed SPI values and the keying material index.



Type	1
Length	12
Keymat Index	Index, in bytes, where to continue to draw ESP keys from KEYMAT. If the packet includes a new Diffie-Hellman key the field MUST be zero. Note that the length of this field limits the amount of keying material that can be drawn from KEYMAT. If that amount is exceeded, the packet MUST contain a new Diffie-Hellman key.
Old SPI	Old SPI for data sent to the source address of this packet. If this is an initial SA setup, the Old SPI value is zero.
New SPI	New SPI for data sent to the source address of this packet.

6.1.2 ESP_TRANSFORM



Type	19
Length	length in octets, excluding Type, Length, and padding
E	One if the ESP transform requires 64-bit sequence numbers (see Section 4.7
Reserved	zero when sent, ignored when received
Suite-ID	defines the ESP Suite to be used

The following Suite-IDs are defined ([7],[9]):

Suite-ID	Value
RESERVED	0
ESP-AES-CBC with HMAC-SHA1	1
ESP-3DES-CBC with HMAC-SHA1	2
ESP-3DES-CBC with HMAC-MD5	3
ESP-BLOWFISH-CBC with HMAC-SHA1	4
ESP-NULL with HMAC-SHA1	5
ESP-NULL with HMAC-MD5	6

There MUST NOT be more than six (6) ESP Suite-IDs in one ESP_TRANSFORM parameter. The limited number of Suite-IDs sets the maximum size of ESP_TRANSFORM parameter. The ESP_TRANSFORM MUST contain at least one of the mandatory Suite-IDs.

Mandatory implementations: ESP-AES-CBC with HMAC-SHA1 and ESP-NULL with HMAC-SHA1.

6.1.3 NOTIFY parameter

The HIP base specification defines a set of NOTIFY error types. The following error types are required for describing errors in ESP

Transform crypto suites during negotiation.

NOTIFY PARAMETER - ERROR TYPES	Value
NO_ESP_PROPOSAL_CHOSEN	18

None of the proposed ESP Transform crypto suites was acceptable.

INVALID_ESP_TRANSFORM_CHOSEN	19

The ESP Transform crypto suite does not correspond to one offered by the responder.

6.2 HIP ESP Setup protocol - HES

This section describes the HES protocol conceptual packets and how the parameters are located in the packets. The reader must understand that these are only conceptual packets and they are NOT protected in any way. In an implementation, the parameters MUST be included in other messages that are protected in an appropriate manner.

6.2.1 HES1

The ESP_TRANSFORM contains the ESP modes supported by the sender, in the order of preference. All implementations MUST support AES [3] with HMAC-SHA-1-96 [2].

In a typical implementation, the HES1 contents are included in the HIP R1 packet. The following figure shows the resulting R1 packet layout.

The HIP parameters for the R1 packet:

```
IP ( HIP ( [ R1_COUNTER, ]
           PUZZLE,
           DIFFIE_HELLMAN,
           HIP_TRANSFORM,
           ESP_TRANSFORM,
           HOST_ID,
           [ ECHO_REQUEST, ]
           HIP_SIGNATURE_2 )
           [, ECHO_REQUEST ])
```

6.2.2 HES2

The ESP_INFO contains the sender's SPI for this association as well as the keymat index from where the ESP SA keys will be drawn. The Old SPI value is set to zero.

The ESP_TRANSFORM contains the ESP mode selected by the sender of HES2. All implementations MUST support AES [3] with HMAC-SHA-1-96 [2].

In a typical implementation, the HES2 contents are included in the HIP I2 packet. The following figure shows the resulting I2 packet layout.

The HIP parameters for the I2 packet:

```
IP ( HIP ( ESP_INFO,
           [R1_COUNTER,]
           SOLUTION,
           DIFFIE_HELLMAN,
           HIP_TRANSFORM,
           ESP_TRANSFORM,
           ENCRYPTED { HOST_ID },
           [ ECHO_RESPONSE ,]
           HMAC,
           HIP_SIGNATURE
           [, ECHO_RESPONSE] ) )
```

6.2.3 HES3

The HES3 contains an ESP_INFO parameter, which has the SPI value of the sender of the HES3 for this association. The ESP_INFO has also the keymat index value telling the point from where the ESP SA keys are drawn.

In a typical implementation, the HES3 contents are included in the HIP R2 packet. The following figure shows the resulting R2 packet layout.

The HIP parameters for the R2 packet:

```
IP ( HIP ( ESP_INFO, HMAC_2, HIP_SIGNATURE ) )
```

6.3 HIP ESP Rekeying protocol - HER

Like in HES, this section describes the rekeying protocol with

conceptual packets. In an implementation, the information is inicluded in other packets, providing appropriate protection.

6.3.1 HER1

During ESP transport form usage, the HER1 packet is used for initiating rekeying. The HER1 packet MUST carry an ESP_INFO and MAY carry a DIFFIE_HELLMAN parameter.

Intermediate systems that use the SPI will have to inspect HIP packets for ones carrying HER1 information. The packet is signed for the benefit of the intermediate systems. Since intermediate systems may need the new SPI values, the contents cannot be encrypted.

In a typical implementation, the HER1 contents are sent in an UPDATE packet. The following figure shows the contents of a rekeying initialization UPDATE packet.

The HIP parameters for the UPDATE packet initiating rekeying:

IP (HIP (ESP_INFO, SEQ, [ACK, DIFFIE_HELLMAN] HMAC, HIP_SIGNATURE))

6.3.2 HER2

During ESP transport form usage, the HER2 packet is used for acking a HER1. The HER2 packet MUST carry an ESP_INFO and MAY carry a DIFFIE_HELLMAN parameter.

Intermediate systems that use the SPI will have to inspect HIP packets for packets carrying HER2 information. The packet is signed for the benefit of the intermediate systems. Since intermediate systems may need the new SPI values, the contents cannot be encrypted.

In a typical implementation, the HER2 contents are sent in an UPDATE packet. The following figure shows the contents of a rekeying acknowledgement UPDATE packet.

The HIP parameters for the UPDATE packet:

IP (HIP (ESP_INFO, ACK, [DIFFIE_HELLMAN,] HMAC, HIP_SIGNATURE))

6.4 ICMP messages

The ICMP message handling is mainly described in the HIP base specification [5]. In this section, we describe the actions related

to ESP security associations.

6.4.1 Unknown SPI

If a HIP implementation receives an ESP packet that has an unrecognized SPI number, it MAY respond (subject to rate limiting the responses) with an ICMP packet with type "Parameter Problem", the Pointer pointing to the the beginning of SPI field in the ESP header.

7. Packet processing

Packet processing is mainly defined in the HIP base specification [5]. This section describes the changes and new requirements for packet handling when the ESP transport format is used.

7.1 Processing outgoing application data

Outgoing application data handling is specified in the HIP base specification [5]. When ESP transport format is used, and there is an active HIP session for the given < source, destination > HIT pair, the outgoing datagram is protected using the ESP security association. In a typical implementation, this will result in a transport mode ESP packet to be sent.

1. Detect the proper ESP SA using the HITs in the packet header or other information associated with the packet
2. Process the packet normally, as if the SA was a transport mode SA.
3. Ensure that the outgoing ESP protected packet has proper IP addresses in its IP header, e.g., by replacing HITs left by the ESP processing. Note that this placement of proper IP addresses MAY also be performed at some other point in the stack, e.g., before ESP processing.

7.2 Processing incoming application data

Incoming HIP user data packets arrive as ESP protected packets. In the usual case the receiving host has a corresponding ESP security association, identified by the SPI and destination IP address in the packet. However, if the host has crashed or otherwise lost its HIP state, it may not have such an SA.

The basic incoming data handling is specified in the HIP base specification. Additional steps are required when ESP is used for protecting the data traffic. The following steps define the conceptual processing rules for incoming ESP protected datagrams targeted to an ESP security association created with HIP.

1. Detect the proper ESP SA using the SPI. If the resulting SA is a non-HIP ESP SA, process the packet according to standard IPsec rules. If there are no SAs identified with the SPI, the host MAY send an ICMP packet as defined in Section 6.4. How to handle lost state is an implementation issue.
2. The IP addresses in the datagram are replaced with the HITs associated with the SPI. Note that this IP-address-to-HIT conversion step MAY also be performed at some other point in the stack, e.g., after ESP processing.
3. If a proper HIP ESP SA is found, the packet is processed normally by ESP, as if the packet were a transport mode packet. The

packet may be dropped by ESP, as usual. In a typical implementation, the result of successful ESP decryption and verification is a datagram with the associated HITs as source and destination.

4. The datagram is delivered to the upper layer. Demultiplexing the datagram to the right upper layer socket is based on the HITs (or LSIs).

7.3 HMAC and SIGNATURE calculation and verification

The new HIP parameters described in this document, ESP_INFO and ESP_TRANSFORM, must be protected using HMAC and signature calculations. In a typical implementation, they are included in R1, I2, R2, and UPDATE packet HMAC and SIGNATURE calculations as described in [5].

7.4 Processing incoming conceptual HES1 packets

The incoming HES1 packet contains the ESP_TRANSFORM parameter. The receiving host select one of the ESP transform from the presented values. If no suitable value is found, the negotiation is terminated. The selected values are subsequently used when generating and using encryption keys, and when sending the HES2. If the proposed alternatives are not acceptable to the system, it may abandon the ESP SA establishment negotiation. A typical implementation where HES1 is piggybacked in the R1 message, and the proposed alternatives are not acceptable to the system, the receiver of an R1 may resend the I1 message within the retry bounds.

After selecting the ESP transform, the system prepares and creates an incoming ESP security association. It may also prepare a security association for outgoing traffic, but since it does not have the correct SPI value yet, it cannot activate it.

7.5 Processing incoming conceptual HES2 packets

The following steps are required to process the incoming HES2 packets.

- The ESP_TRANSFORM parameter is verified and it MUST contain a single value in the parameter and it MUST match one of the values offered in the HES1 packet.
- The ESP_INFO New SPI field is parsed to obtain the SPI that will be used for the Security Association outbound from the Responder and inbound to the Initiator. For this initial ESP SA establishment, the Old SPI value MUST be zero. The keymat index field contains the point from where the keying material for this ESP SA will be drawn.

- o The system prepares and creates both incoming and outgoing ESP security associations.
- o Upon successful processing of an HES2, the possible old Security Associations (as left over from an earlier incarnation of the HIP association) are dropped and the new ones are installed, and an HES3 is sent. Possible ongoing rekeying attempts are dropped.

7.6 Processing incoming HES3 packets

Before the ESP SA can be finalized, the ESP_INFO New SPI field is parsed to obtain the SPI that will be used for the ESP Security Association inbound to the sender of HES3. The system uses this SPI to create or activate the outgoing ESP security association used for sending packets to the peer.

7.7 Dropping HIP associations

When the system drops a HIP association, as described in the HIP base specification, the associated ESP SAs MUST also be dropped.

7.8 Initiating rekeying

A system may initiate the rekey procedure at any time. It MUST initiate a rekey if its incoming ESP sequence counter is about to overflow. The system MUST NOT replace its keying material until the rekeying packet exchange successfully completes. Optionally, depending on policy, a system may include a new Diffie-Hellman key for use in new KEYMAT generation. New KEYMAT generation occurs prior to drawing the new keys.

In the conceptual state machine, a system will rekey when it has sent an ESP_INFO parameter to the peer and has received both an ACK of the relevant HER1 message and its peer's ESP_INFO parameter. To complete and outstanding rekeying request, both parameters must be received. It may be that the ACK and the ESP_INFO arrive in different UPDATE messages. This is always true if a system does not initiate rekeying but responds to a rekeying request from the peer, but may also occur if two systems initiate a rekey nearly simultaneously. In such a case, if the system has an outstanding rekeying request, it saves the one parameter and waits for the other before completing rekeying.

The following steps define the processing rules for initiating a rekey:

1. The system decides whether to continue to use the existing KEYMAT or to generate new KEYMAT. In the latter case, the system MUST generate a new Diffie-Hellman public key.
2. The system creates a HER1 packet, which contains the ESP_INFO parameter and an optional DIFFIE_HELLMAN parameter. If the HER1

contains the DIFFIE_HELLMAN parameter, the Keymat Index in the ESP_INFO parameter MUST be zero. If the HER1 does not contain DIFFIE_HELLMAN, the ESP_INFO Keymat Index MUST be greater or equal to the index of the next byte to be drawn from the current KEYMAT.

3. The system sends the HER1 packet, typically as an UPDATE packet with a SEQ parameter.
4. For reliability, the underlying UPDATE retransmission mechanism SHOULD be used.
5. The system MUST NOT delete its existing SAs, but continue using them if its policy still allows. The rekeying procedure SHOULD be initiated early enough to make sure that the SA replay counters do not overflow.
6. In case a protocol error occurs and the peer system acknowledges the HER1 but does not itself send a ESP_INFO, the system may not finalize the outstanding rekeying request. To guard against this, a system MAY re-initiate the rekeying procedure after some time waiting for the peer to respond, or it MAY decide to abort the ESP SA after waiting for an implementation-dependent time. The system MUST NOT keep an oustanding rekeying request for for an indefinite time.

To simplify the state machine, a host MUST NOT generate new HER1s while it has an outstanding rekeying request, unless it is restarting the rekeying process.

7.9 Processing conceptual HER1 packets

When a system receives a conceptual HER1 packet, it must be processed if the following conditions hold:

1. A corresponding HIP association must exist. This is usually ensured by the underlying UPDATE mechanism.
2. The state of the HIP association is ESTABLISHED.

If the above conditions hold, the following steps define the conceptual processing rules for handling the received HER1 packet:

1. If the received HER1 contains a DIFFIE_HELLMAN parameter, the received Keymat Index MUST be zero. If this test fails, the packet SHOULD be dropped and the system SHOULD log an error message.
2. If there is no outstanding rekeying request, the packet processing continues as specified in Section 7.9.1.
3. If there is an outstanding rekeying request, the packet processing continues as specified in Section 7.9.2.

7.9.1 Processing HER1 packet: no outstanding rekeying request

The following steps define the conceptual processing rules for

handling a received HER1 packet:

1. The system consults its policy to see if it needs to generate a new Diffie-Hellman key, and generates a new key if needed. The system records any newly generated or received Diffie-Hellman keys, for use in KEYMAT generation upon leaving the REKEYING state.
2. If the system generated new Diffie-Hellman key in the previous step, or it received a DIFFIE_HELLMAN parameter, it sets ESP_INFO Keymat Index to zero. Otherwise, the ESP_INFO Keymat Index MUST be greater or equal to the index of the next byte to be drawn from the current KEYMAT. In this case, it is RECOMMENDED that the host use the Keymat Index requested by the peer in the received ESP_INFO.
3. The system creates a HER2 packet, which contains an ESP_INFO parameter and the optional DIFFIE_HELLMAN parameter.
4. The system sends the HER2 packet and transitions to the REKEYING state. The system stores any received ESP_INFO and DIFFIE_HELLMAN parameters. At this point, it only needs to receive an acknowledgement for the sent HER2 to finish rekeying. In a usual case, the acknowledgement is handled by the underlying UPDATE mechanism.

7.9.2 **Processing HER1 packet:** outstanding rekeying request exists

The following steps define the conceptual processing rules for handling a received HER1 packet:

1. The system consults its policy to see if it has generated a new Diffie-Hellman key previously when it sent out the HER1. The system records any newly received Diffie-Hellman keys, for use in KEYMAT generation before finalizing the rekeying process.
2. If the system has generated new Diffie-Hellman key previously, or it received a DIFFIE_HELLMAN parameter, it sets ESP_INFO Keymat Index to zero. Otherwise, the ESP_INFO Keymat Index MUST be greater or equal to the index of the next byte to be drawn from the current KEYMAT. If neither of the hosts have included DIFFIE_HELLMAN parameter and the Keymat Index requested by the peer in the received ESP_INFO has a greater value than the Keymat Index the system has sent out previously it is RECOMMENDED that the system will use the greater value received from the peer.
3. The system creates a HER2 packet, which contains an ESP_INFO parameter and the optional DIFFIE_HELLMAN parameter if it was previously generated.
4. The system sends the HER2 packet. The system stores any received ESP_INFO and DIFFIE_HELLMAN parameters. At this point, it only needs to receive an acknowledgement for the sent HER2 to finish rekeying. In a usual case, the acknowledgement is handled by the underlying UPDATE mechanism.

7.10 Processing HER2 packets

When a system receives an HER2 packet, it must be processed if the following conditions hold:

1. A corresponding HIP association must exist. This is usually ensured by the underlying UPDATE mechanism.
2. The state of the HIP association is ESTABLISHED and there is an outstanding rekeying request.

If the above conditions hold, the following steps define the conceptual processing rules for handling the received HER2 packet:

1. If the received HER2 contains a DIFFIE_HELLMAN parameter, the received Keymat Index MUST be zero. If this test fails, the packet SHOULD be dropped and the system SHOULD log an error message.
2. If the HER2 packet contains the ESP_INFO parameter, the system finishes the rekeying procedure as described in Section 7.11.

7.11 Finalizing rekeying

A system leaves the REKEYING state, when it has received the corresponding acknowledgement packet from the peer. The following steps are taken:

1. If any of the received HER messages contains a new Diffie-Hellman key, the system has a new Diffie-Hellman key from initiating rekey, or both, the system generates new KEYMAT. If there is only one new Diffie-Hellman key, the existing old key is used as the other key.
2. If the system generated new KEYMAT in the previous step, it sets Keymat Index to zero, independent on whether the received HER1 included a Diffie-Hellman key or not. If the system did not generate new KEYMAT, it uses the lowest Keymat Index of the two ESP_INFO parameters.
3. The system draws keys for new incoming and outgoing ESP SAs, starting from the Keymat Index, and prepares new incoming and outgoing ESP SAs. The SPI for the outgoing SA is the new SPI value received in an ESP_INFO parameter. The SPI for the incoming SA was generated when the ESP_INFO was sent to the peer. The order of the keys retrieved from the KEYMAT during rekeying process is similar to that described in Section 8. Note, that only IPsec ESP keys are retrieved during rekeying process, not the HIP keys.
4. The system cancels any timers protecting the HER.
5. The system starts to send to the new outgoing SA and prepares to start receiving data on the new incoming SA.

7.12 Processing NOTIFY packets

The processing of NOTIFY packets is described in the HIP base specification.

8. Keying material

The keying material is generates as described in the HIP base specification. During the base exchange, the initial keys are drawn from the generated material. After the HIP association keys have been drawn, the ESP keys are drawn in the following order:

SA-gl ESP encryption key for HOST_g's outgoing traffic
SA-gl ESP authentication key for HOST_g's outgoing traffic
SA-lg ESP encryption key for HOST_l's outgoing traffic
SA-lg ESP authentication key for HOST_l's outgoing traffic

The four HIP keys are only drawn from KEYMAT during a HIP I1->R2 exchange. Subsequent rekeys using UPDATE will only draw the four ESP keys from KEYMAT. Section 7.9 describes the rules for reusing or regenerating KEYMAT based on the rekeying.

The number of bits drawn for a given algorithm is the "natural" size of the keys. For the mandatory algorithms, the following sizes apply:

AES 128 bits
SHA-1 160 bits
NULL 0 bits

9. Security Considerations

To be written.

10. References

10.1 Normative references

[1] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[2] Madson, C. and R. Glenn, "The Use of HMAC-SHA-1-96 within ESP and AH", RFC 2404, November 1998.

[3] Pereira, R. and R. Adams, "The ESP CBC-Mode Cipher Algorithms", RFC 2451, November 1998.

[4] Kent, S., "IP Encapsulating Security Payload (ESP)", draft-ietf-ipsec-esp-v3-05 (work in progress), April 2003.

[5] Moskowitz, R., "Host Identity Protocol", draft-ietf-hip-base-00 (work in progress), June 2004.

[6] Nikander, P., "End-Host Mobility and Multi-Homing with Host Identity Protocol", draft-ietf-hip-mm-00 (work in progress), October 2004.

[7] Kaufman, C., "Internet Key Exchange (IKEv2) Protocol", draft-ietf-ipsec-ikev2-07 (work in progress), April 2003.

[8] Moskowitz, R., "Host Identity Protocol Architecture", draft-ietf-hip-arch-01 (work in progress), December 2004.

10.2 Informative references

[9] Bellovin, S. and W. Aiello, "Just Fast Keying (JFK)", draft-ietf-ipsec-jfk-04 (work in progress), July 2002.

[10] Nikander, P., "A Bound End-to-End Tunnel (BEET) mode for ESP", draft-nikander-esp-beet-mode-00 (expired) (work in progress), Oct 2003.

Authors' Addresses

Petri Jokela
Ericsson Research NomadicLab
JORVAS FIN-02420
FINLAND

Phone: +358 9 299 1
EMail: petri.jokela@nomadiclab.com

Robert Moskowitz
ICSAlabs, a Division of TruSecure Corporation
1000 Bent Creek Blvd, Suite 200
Mechanicsburg, PA
USA

EMail: rgm@icsalabs.com

Pekka Nikander
Ericsson Research NomadicLab
JORVAS FIN-02420
FINLAND

Phone: +358 9 299 1
EMail: pekka.nikander@nomadiclab.com

Intellectual Property Statement

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at http://www.ietf.org/ipr.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

Disclaimer of Validity

Copyright Statement

Acknowledgment

Funding for the RFC Editor function is currently provided by the Internet Society.