HIP Working Group
Internet-Draft
Expires: August 31, 2006

V. Schmitt
NEC
A. Pathak
IIT Kanpur
M. Komu
HIIT
L. Eggert
M. Stiemerling
NEC

February 27, 2006

HIP Extensions for the Traversal of Network Address Translators
draft-schmitt-hip-nat-traversal-00

Status of this Memo

Copyright Notice

Abstract

This document specifies extensions to Host Identity Protocol (HIP) to support traversal of Network Address Translator (NAT) middleboxes. The traversal mechanism tunnels HIP control and data traffic over UDP and enables HIP initiators behind NATs to contact HIP responders in the global Internet. Future revisions of this document will describe mechanisms to contact HIP responders behind NATs.

Table of Contents

1. Introduction

The Host Identity Protocol (HIP) describes a new communication mechanism for Internet hosts [I-D.ietf-hip-arch]. It introduces a new namespace and protocol layer between the network and transport layers that decouples the identifier and locator roles that the IP address fulfill in the current Internet architecture.

The HIP protocol [I-D.ietf-hip-base] cannot operate across Network Address Translator (NAT) middleboxes, as described in [I-D.irtf-hiprg-nat]. Several different flavors of NATs exist [RFC2663]. This document describes HIP extensions for the traversal of both Network Address Translator (NAT) and Network Address and Port Translator (NAPT) middleboxes. It generally uses the term NAT to refer to both types of middleboxes, unless it needs to distinguish between the two types.

Three basic cases exist for NAT traversal. In the first case, only the initiator of a HIP base exchange is located behind a NAT. In the second case, only the responder of a HIP base exchange is located behind a NAT. The respective peer host is assumed to be in the global Internet in both cases. Complementary HIP extensions for these two cases should support a third case where both parties are located behind NATs. This document currently describes extensions for the first case only; future revisions will describe extensions for the second case and third case.

The mechanisms described this document are based on encapsulating both the control and data traffic in UDP in order to traverse NAT(s). The data traffic is assumed to be ESP. Other types of data traffic are out of scope.

Note that due to the mobility and multihoming mechanisms of HIP [I-D.ietf-hip-mm], HIP hosts may change network location during the lifetime of a HIP association. Either host of an established HIP association may move from the global Internet to behind a NAT and vice versa many times during the lifetime of the association. Consequently, hosts need to start using the proposed NAT traversal mechanisms after a mobility event relocates one or both peers behind a NAT. They may also stop using the proposed mechanisms if they both relocate to the public Internet.

Note that in order to determine whether to use the proposed NAT traversal mechanisms, HIP hosts need to detect the presence and type of NAT middleboxes between them. STUN [RFC3489] offers a generic mechanism for this purpose. This document therefore does not describe a NAT detection mechanism and assumes that existing mechanisms, such as STUN, are in use.

Finally, 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].

2. HIP Across NATs

HIP communication has two distinct phases. During the first phase, the "base exchange", HIP establishes synchronized state at the initiator and responder of the communication. During the second phase, the "data exchange", HIP uses this state to encrypt data traffic using IPsec ESP.

NAT traversal for HIP communication requires mechanisms for handling both the base exchange and the following IPsec ESP traffic. This document describes methods for encapsulating both types of traffic inside of UDP [RFC0768], similar to other NAT traversal mechanisms.

One unique aspect of the proposed solution is the use of UDP-encapsulated IPsec BEET mode [I-D.nikander-esp-beet-mode] instead of the more common UDP-encapsulated IPsec transport mode [RFC3948]. The main advantage of UDP-encapsulated IPsec BEET mode over UDP-encapsulated IPsec transport mode for HIP is the explicit definition of inner and outer addresses in BEET mode. The inner addresses of BEET associations are used for processing at all layers of the stack, whereas the outer addresses are only used to transmit the final packets on the wire. This explicit distinction between addresses matches the identifier/locator differentiation in HIP, i.e., HITs as identifiers are inner addresses and IP addresses as locators are outer addresses.

[RFC3948] describes UDP encapsulation of IPsec ESP transport and tunnel mode. This document only describes the changes required for UDP encapsulation of BEET mode.

A HIP implementation that supports the proposed NAT traversal extensions MUST listen on UDP ports 50500 and 54500 for arriving packets. When packets arrive on these two UDP ports, the implementation MUST accept them regardless of their source port. Port 50500 is used for UDP-encapsulated HIP base exchange and other control traffic, port 54500 is used for UDP-encapsulated ESP data traffic.

2.1. Packet Formats

This section defines the UDP-encapsulation packet format for HIP base exchange and control traffic, IPsec ESP BEET-mode traffic and NAT

keep-alives.

2.1.1. Control Traffic

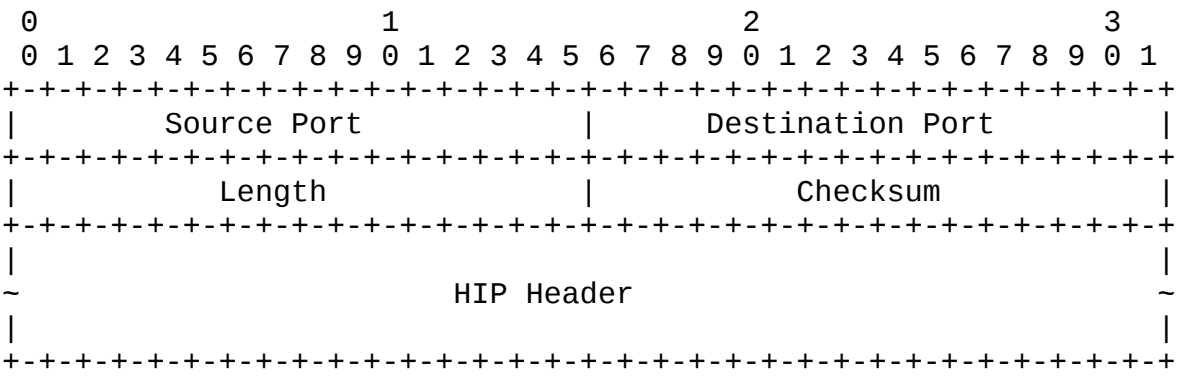

Figure 1: Format for UDP-encapsulated HIP control traffic.

Figure 1 shows how HIP control packets are encapsulated within UDP. A minimal UDP packet carries a complete HIP packet in its payload. Contents of the UDP source and destination ports are described below. The UDP length and checksum field MUST be computed as described in [RFC0768].

2.1.2. Control Channel Keep-Alives

The keep-alives for control channel are basically UPDATE packets [I-D.ietf-hip-base]. The UPDATE packets MAY contain HIP parameters. The NAT traversal mechanisms encapsulate these UPDATE packets within the payload of UDP packets, as shown in Figure 2.

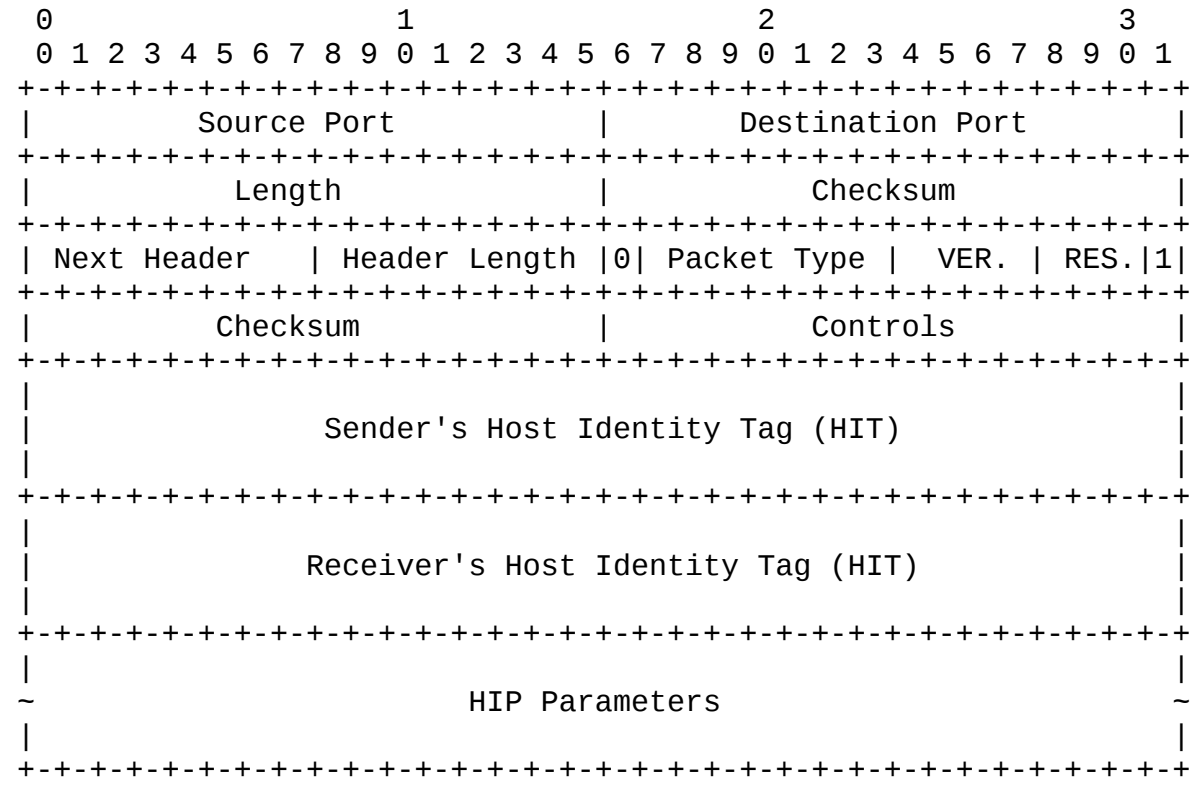

Figure 2: Format for UDP NAT control channel keep-alive packets.

Contents of the UDP source and destination ports are described below. The UDP length and checksum field MUST be computed as described in [RFC0768]. The HIP packet header fields are filled in as described in [I-D.ietf-hip-base], except for the HIP checksum, which is always set to zero. The type of the HIP header is 16 (UPDATE).

2.1.3. Data Traffic

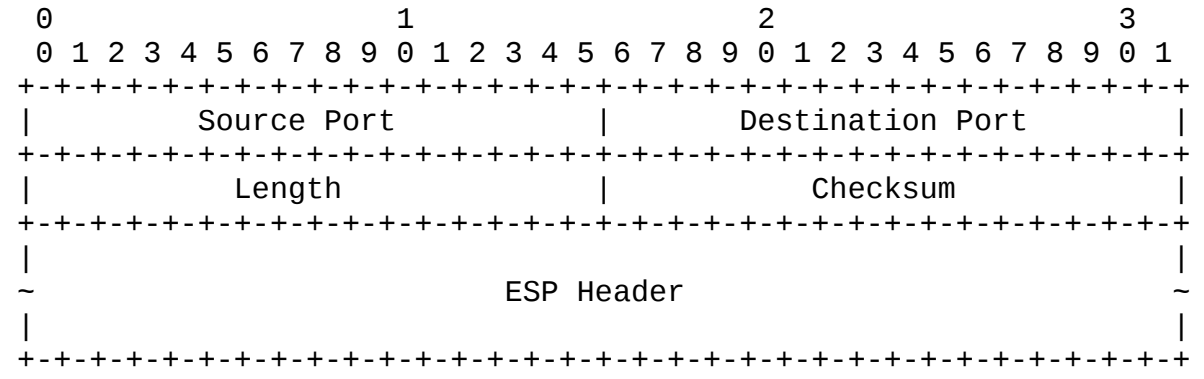

Figure 3: Format for UDP-encapsulated IPsec ESP BEET-mode traffic.

Figure 3 shows how IPsec ESP BEET-mode packets are encapsulated within UDP. Again, a minimal UDP packet carries the ESP packet in its payload. Contents of the UDP source and destination ports are described below. The UDP length and checksum field MUST be computed as described in [RFC0768].

2.1.4. Data Channel Keep-Alives

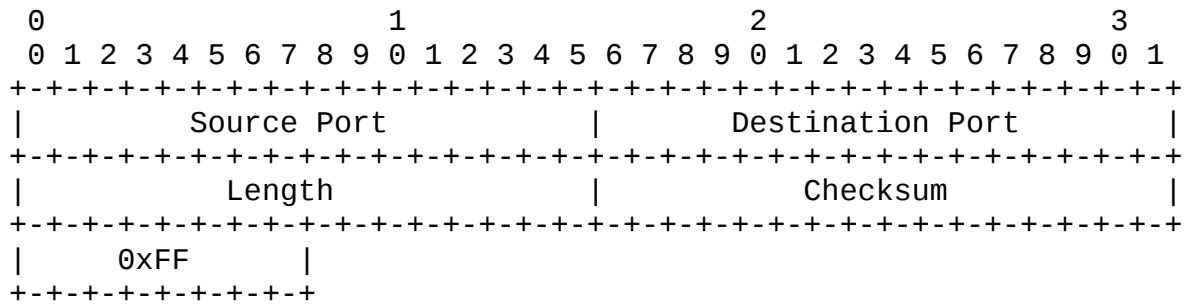

Figure 4: Format for UDP NAT data channel keep-alive packets.

The keep-alive packets to refresh NAT state for the ESP channel are minimal UDP packets that carry a dummy payload of a single octet, as illustrated in Figure 4. The single-octet payload value MUST be 0xFF. The format is the same as in [RFC3948]. Contents of the UDP source and destination ports are described below. The UDP length and checksum field MUST be computed as described in [RFC0768].

2.2. NAT Traversal of HIP Control Traffic

This section describes the details of enabling NAT traversal for HIP control traffic, such as the base exchange [I-D.ietf-hip-base], through UDP encapsulation. As pointed out in Section 1, this document currently assumes that only the initiator is located behind one or more NATs whereas the responder is in the global Internet.

UDP-encapsulated HIP control traffic MUST use the packet formats described in Section 2.1. When sending UDP-encapsulated HIP control traffic, a HIP implementation that supports the proposed NAT traversal extensions MUST zero the HIP header checksum before calculating the UDP checksum. Receivers MUST only verify the correctness of the UDP checksum and MUST NOT verify the checksum of the HIP header. The reason why HIP checksum is not used is that it is redundant and requires the use of inner addresses (extra complexity for NAT transformation).

The initiator of a UDP-encapsulated HIP base exchange MUST use the UDP destination port 50500 for all control packets it sends. It MAY use 50500 as the source port as well, or it MAY use a random,

unoccupied source port. If it uses a random source port, it MUST listen for and accept arriving HIP control packets on this port until the corresponding HIP association is torn down. A random source port MUST be in the range of the dynamic and private ports (49152-65535). Using a random source port instead of a fixed one makes it possible to have multiple clients behind a NAT middlebox that does only address translation but no port translation.

The responder of a UDP-encapsulated HIP base exchange MUST use 50500 as the source port for all UDP-encapsulated control packets it sends. As a source IP address, it MUST use the IP address that prior packets from the initiator of this HIP association arrived on. Similarly, it MUST use the source IP address and source UDP port of prior packets from the initiator of the respective HIP association as the destination IP address and destination UDP port. The responder MUST NOT respond to any arriving UDP-encapsulated control message with an unencapsulated reply.

Whether or not HIP control packets are UDP-encapsulated MUST NOT affect the HIP state machine. HIP implementations that implement the NAT traversal mechanisms generally MUST process all UDP-encapsulated control messages equivalently to unencapsulated control messages, i.e., according to [I-D.ietf-hip-base]. The single exception to this rule is that if the base exchange was UDP-encapsulated, implementations MUST establish a UDP-encapsulated BEET-mode ESP IPsec association (see Section 2.3) for subsequent data traffic instead of the regular transport-mode ESP association specified in [I-D.ietf-hip-esp].

The remainder of this section clarifies this process through an example, illustrated in Figure 5. It shows an initiator I with the private IP address IP(I) behind a NAT. The NAT has the private IP address IP(NAT1) and the public IP address IP(NAT2). The responder is located in the public Internet at the IP address IP(R).

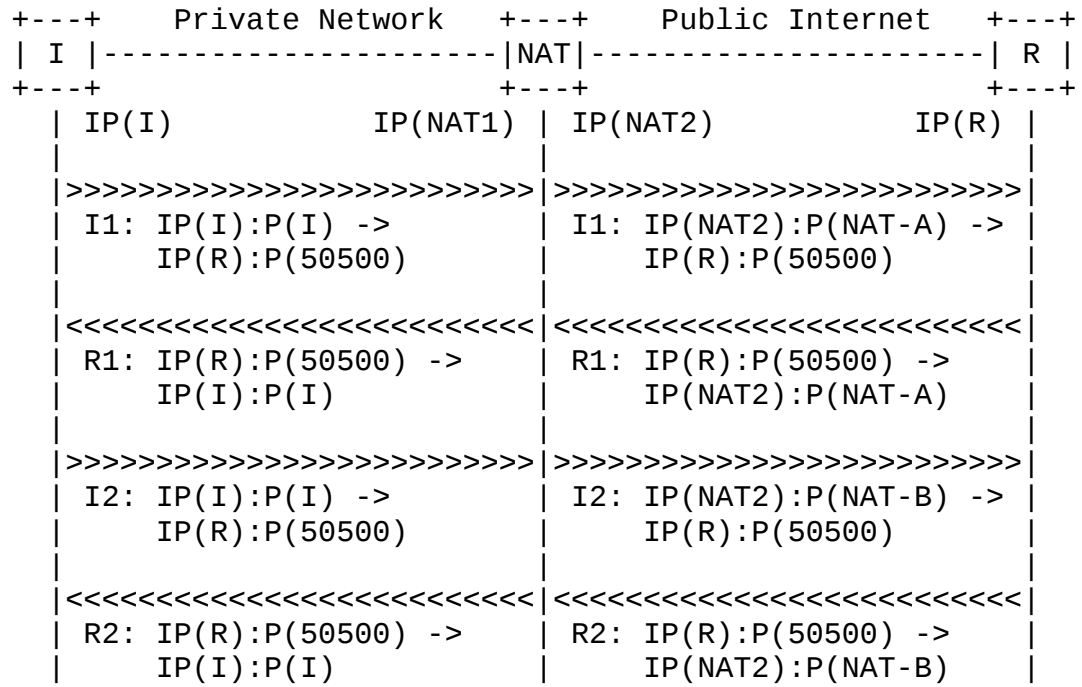

Figure 5: Example of a UDP-encapsulated HIP base exchange (initiator behind a NAT, responder on the public Internet).

The initiator I begins the base exchange by sending a UDP-encapsulated I1 packet to the responder. According to the rules specified above, the source IP address of this I1 packet is IP(I) and its source UDP port is P(I). It is addressed to IP(R) on port P(50500). The NAT in Figure 5 forwards the I1 but substitutes the source IP(I) with its own public IP address IP(NAT2) and substitutes the source UDP port P(I) with P(NAT-A), which will usually be different from P(I).

When the responder R in Figure 5 receives the UDP-encapsulated I1 packet on the UDP port 50500, it processes it according to [I-D.ietf-hip-base]. According to the rules specified above, if it replies with an R1 packet, the R1 uses the destination IP address and UDP port from the previous I1 packet as the source IP address and UDP port, i.e., IP(R) and P(50500). Thus R1 packet is destined to the source IP address and UDP port of the I1, i.e., IP(NAT2) and P(NAT-A). The NAT substitutes the destination of this packet, replacing IP(NAT2):P(NAT-A) with IP(I):P(I).

When the initiator receives a UDP-encapsulated R1 packet from the responder, it processes it according to [I-D.ietf-hip-base]. When it responds with a UDP-encapsulated I2 packet, it uses the same IP source and destination addresses and UDP source and destination ports that it used for sending the corresponding I1 packet, i.e., the

packet is addressed as IP(I):P(I) -> IP(R):P(50500). The NAT again substitutes the source information, replacing it with IP(NAT2): P(NAT-B).

When a responder receives a UDP-encapsulated I2 packet destined to UDP port 50500, it MUST use the UDP source port contained in this packet for further HIP communications with the initiator. It then processes the I2 packet according to [I-D.ietf-hip-base]. When it responds with an R2 message, it UDP-encapsulates it, using the UDP source port of the I2 packet as the destination UDP port, and sends it to the source IP address of the I2 packet, i.e., it addresses the R2 packet as IP(R):P(50500) -> IP(NAT2):P(NAT-B). The NAT again replaces the destination information in the R2 with IP(I):P(I).

Usually, the I1-R1 and I2-R2 exchanges occur fast enough for the NAT state to not time out. This means that the NAT uses the state established during the I1-R1 exchange to translate the I2-R2 exchange, i.e., the ports P(NAT-A) and P(NAT-B) will be identical. Note that the mechanism can handle the case where the NAT state times out between the two exchanges and the I1 and I2 arrive from different UDP source ports and/or IP addresses, as shown in Figure 5. It is important as the responder may be busy and also because the responder can remain stateless until receives an I2.

2.3. NAT Traversal of HIP Data Traffic

This section describes the details of enabling NAT traversal of HIP data traffic. As described in Section 2, HIP data traffic is carried in UDP-encapsulated IPsec BEET-mode ESP packets. Section 2.3.1 and Section 2.3.2 describe the UDP encapsulation and decapsulation procedure for IPsec BEET-mode ESP for HIP. Section 2.3.3 describes how hosts configure BEET-mode security associations for HIP communication as part of a UDP-encapsulated base exchange.

[I-D.nikander-esp-beet-mode] defines two types of addresses for IPsec BEET mode: inner and outer addresses. Because this document focuses on traversal of legacy IPv4 NATs, the outer BEET-mode address family MUST be IPv4. For HIP communication, the 128-bit Host Identity Tags (HIT) of the the local host and its peer MUST be used as inner BEET-mode addresses. Consequently, the inner BEET-mode address family in the IPsec stack MUST be IPv6. Note that legacy applications MAY use IPv4 local scope identifiers (LSIs) translated to the corresponding HITs by the HIP layer. Such usage is however implementation specific and out of scope of this document.

2.3.1. UDP Encapsulation of IPsec BEET-Mode ESP

An IPv6 packet targeted for UDP BEET-mode encapsulation MUST contain HITs as source and destination IP address. Any present transport-layer checksums in the payload data are consequently based on the HITs. The packet MUST then undergo BEET-mode ESP cryptographic processing as defined in Section 5.3 of [I-D.nikander-esp-beet-mode].

The resulting BEET-mode packet is then UDP encapsulated. For this purpose, a UDP header MUST be inserted between the existing IPv6 and ESP headers. The UDP checksum MUST be calculated based on an IPv4 pseudo-header that contains the outer addresses defined in the corresponding security association (see Section 2.3.3). The source and destination ports MUST also be set according to the security association. The other fields of the UDP header are computed as described in [RFC0768].

The resulting UDP packet MUST then undergo BEET IP header processing as defined in Section 5.4 of [I-D.nikander-esp-beet-mode].

Figure 6 illustrates the BEET-mode UDP encapsulation procedure for a TCP packet.

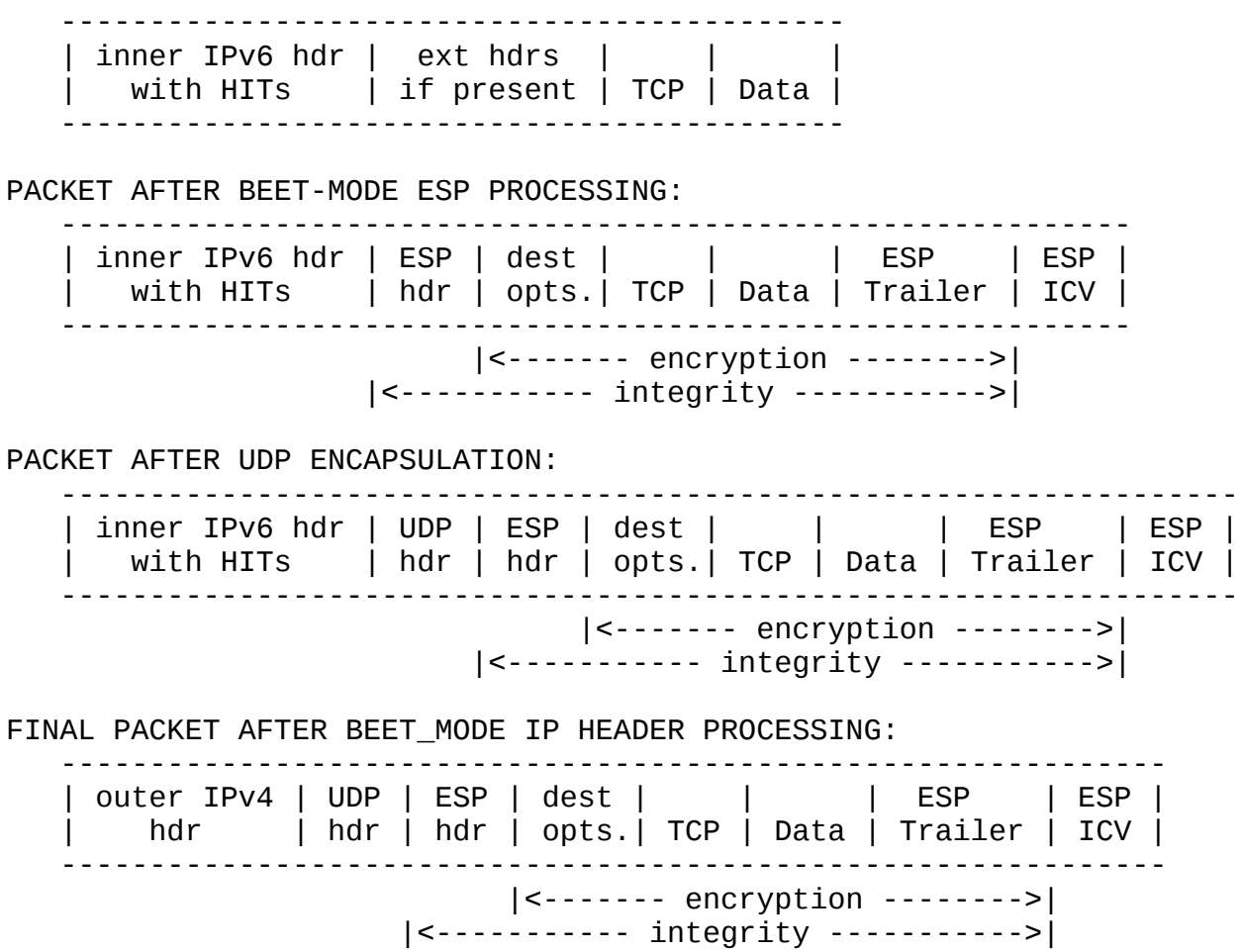

Figure 6: UDP Encapsulation of an IPsec BEET-mode ESP packet containing a TCP segment.

2.3.2. UDP Decapsulation of IPsec BEET-Mode ESP

An incoming UDP-encapsulated IPsec BEET-mode ESP packet is decapsulated as follows. First, the packet MUST be verified as defined in Section 5.6 of [I-D.nikander-esp-beet-mode]. The packet MUST be dropped if the UDP checksum is invalid. Otherwise, the ESP data contained in the payload of the UDP packet MUST be decrypted as described in Section 5.6 of [I-D.nikander-esp-beet-mode].

Next, the IPv4 header containing the outer addresses MUST be discarded. The UDP header MUST also be discarded. Then, a new IPv6 header MUST be constructed, which includes the HITs as IP addresses. This header MUST be prepended to the decrypted data as specified in Section 5.6 of [I-D.nikander-esp-beet-mode].

2.3.3. IPsec BEET-Mode Security Associations

During the HIP base exchange, the two peers exchange parameters that enable them to define a pair of IPsec ESP security associations (SAs), as described in [I-D.ietf-hip-esp]. As mentioned in Section 2.2, if two peers perform a UDP-encapsulated base exchange, they MUST define a pair of IPsec SAs that result in UDP-encapsulated BEET-mode ESP data traffic.

The management of encryption and authentication protocols and of security parameter indices (SPIs) occurs as defined in [I-D.ietf-hip-esp]. Additional SA parameters, such as IP addresses and UDP ports, MUST be defined according to the following specification. Two SAs MUST be defined on each host for one HIP association; one for outgoing data and another one for incoming data.

The initiator of the base exchange MUST also initiate the IPsec BEET-mode ESP exchange, by either sending a UDP-encapsulated ESP data packet or a keep-alive packet (see Section 2.1.4) if it has no data to send. This MUST occur immediately after the base exchange has succeeded, after the initiator has defined its two SAs for the HIP association. The responder can only set up its SAs after receiving and verifying the first packet from the initiator, because it needs to learn the outer IP address and UDP port used by the NAT.

The initiator MUST use UDP destination port 54500 for all UDP-encapsulated ESP packets it sends. It MAY also use port 54500 as source port or it MAY use a random source port. If it uses a random source port, it MUST listen for and accept arriving UDP-encapsulated ESP packets on this port until the corresponding HIP association is torn down. A random source port MUST be in the range of the dynamic and private ports (49152-65535).

The responder of a UDP-encapsulated IPsec BEET-mode ESP exchange MUST use 54500 as the source port for all UDP-encapsulated ESP packets it sends. It MUST use the source UDP port of prior UDP-encapsulated ESP packets from the initiator of the respective HIP association as the destination port of the UDP tunnel.

2.3.3.1. Security Associations at the Initiator

The initiator of a UDP-encapsulated base exchange MUST define its outbound SA as follows:

IPsec ESP mode:
   BEET mode with UDP encapsulation

Inner source address:
the same local HIT used during the base exchange

Inner destination address:
the same responder HIT used during the base exchange

Outer source address:
the same local IP address from which the base exchange packets were transmitted

Outer destination address:
the same peer IP address to which base exchange packets were transmitted

UDP source port:
the initiator MAY use source port 54500 or it MAY use a random port in the range of 49152-65535

UDP destination port:
port 54500

Similarly, the initiator of a UDP-encapsulated base exchange MUST define its inbound SA as follows:

IPsec ESP mode:
BEET mode with UDP encapsulation

Inner source address:
the same responder HIT used during the base exchange

Inner destination address:
the same local HIT used during the base exchange

Outer source address:
the same peer IP address to which base exchange packets were transmitted

Outer destination address:
the same local IP address from which the base exchange packets were transmitted

UDP source port:
port 54500

UDP destination port:
the initiator MUST use the UDP source port it uses in the outbound SA as the UDP destination port here

2.3.3.2. Security Associations at the Responder

The responder of a UDP-encapsulated base exchange MUST define its SAs after the first UDP-encapsulated ESP packet from the initiator that has been received has been correctly processed, using the SPI defined during the base exchange. When it sets up its SAs, it MUST use the source IP address and source UDP port of that packet as outer destination address and destination UDP port. Note that NATs may modify this address and port and they usually will not match the values used at the initiator.

The responder of a UDP-encapsulated base exchange MUST define its outbound SA as follows:

IPsec ESP mode:
BEET mode with UDP encapsulation

Inner source address:
the same local HIT used during the base exchange

Inner destination address:
HIT of the initiator used during the base exchange

Outer source address:
the same local IP address from which the base exchange packets were transmitted

Outer destination address:
source IP address of the first correctly processed UDP-encapsulated ESP packet received from the initiator

UDP source port:
port 54500

UDP destination port:
source UDP port of the first correctly processed UDP-encapsulated ESP packet received from the initiator

Similarly, the initiator of a UDP-encapsulated base exchange MUST define its inbound SA as follows:

IPsec ESP mode:
BEET mode with UDP encapsulation

Inner source address:
HIT of the initiator used during the base exchange

Inner destination address:
the same local HIT used during the base exchange

Outer source address:
source IP address of the first correctly processed UDP-encapsulated ESP packet received from the initiator

Outer destination address:
the same local IP address from which the base exchange packets were transmitted

UDP source port:
source UDP port of the first correctly processed UDP-encapsulated ESP packet received from the initiator

UDP destination port:
port 54500

2.4. NAT Keep-Alives

Typically, NATs cache an established binding and time it out if they have not used it to relay traffic for a given period of time. This timeout is different for different NAT implementations. The BEHAVE working group is discussing recommendations for standardized timeout values.

To prevent NAT bindings that support the traversal of UDP-encapsulated HIP traffic from timing out during times when there is no control or data traffic, HIP hosts SHOULD send periodic keep-alive messages, similar to [RFC3948].

Typically, NATs only act on keep-alives received from hosts in the private network they connect to the public Internet, for security reasons. Consequently, those hosts SHOULD send periodic keep-alives for the control and data channels of all their established HIP associations if the respective channel has been idle for a specific period of time. Keep-alive intervals for control and data channels SHOULD be separately configurable parameters of a HIP association.

For the control channel, keep-alives MUST be UDP-encapsulated HIP UPDATE packets as defined in Section 2.1.2. The packets MUST use the same source and destination ports and IP addresses as the corresponding UDP tunnel. The default keep-alive interval for control channels MUST be 20 seconds.

For the data channel, keep-alives MUST be UDP packets defined in Section 2.1.4. The packets MUST use the same source and destination ports and IP addresses as the corresponding UDP tunnel. The receiver

of a UDP keep-alive packet MUST discard it. The default keep-alive interval for data channels MUST be 20 seconds.

2.5. HIP Mobility

This draft assumes that the initiator and responder do not change their network location during base exchanges. After a base exchange has succeeded, either host can change its network location using the mechanisms defined in [I-D.ietf-hip-mm]. This document currently only describes the case where the host behind a NAT changes its location. The case when the host in the public Internet changes its location is currently not described. This section only discusses mobility events of single SA pairs between the peers; it does not currently describe mobility events of multiple SA pairs and multihoming. In addition, all privacy issues are out of scope even though it is possible that the host behind the NAT reveals its private address during a mobility event.

When a host behind a NAT changes its location, it SHOULD detect the presence of NATs along the new paths to its peers using some external mechanism before sending any UPDATE messages. Alternatively, it MAY use some heuristics to conclude that it is behind a NAT rather than incur the latency of running NAT detection first.

If the host does not detect a NAT along a new path to a peer, it MAY start to use unencapsulated HIP traffic for the association with that peer. The host sends a regular, unencapsulated UPDATE packet to its peer to announce its new location. When the peer receives and verifies the UPDATE, it detects that no UDP header is present. Consequently, the peer MUST check the HIP header checksum and drop the packet if the validation fails. In addition, the peer MUST not use UDP encapsulation for communication with the host any longer, i.e., it must set up regular ESP SAs to the host. The host MUST do the same. The peer MUST also send further HIP control packets without UDP encapsulation.

However, when the host does detect a NAT along the new path to a peer, it MUST continue to use UDP encapsulation both for HIP and ESP packets, using the same ports it used along the old path. The host MUST send a UDP-encapsulated UPDATE packet to the peer. In addition, it MUST also send an ESP keep-alive to reactivate the UDP encapsulation of ESP traffic. The UDP source ports for both control and data channel MUST remain the same and the destination ports MUST be the ports 50500 for HIP and 54500 for ESP, respectively.

Upon receiving a HIP UPDATE packet from a peer, a host first validates the UDP checksum and then actual HIP packet. If either validation fails, the host MUST drop the packet. After successful

validation, it MUST check if the source port in the UDP packet has changed. If there are no changes, nothing needs to be done for the UDP tunnel. However, if the source port has changed, this indicates that the peer has moved behind a NAT. In this case, the receiver MUST stop using the earlier UDP tunnel. It MUST open a new UDP tunnel for HIP control packets based on the addresses and ports contained in the received UDP packet and MUST start using it for further communication with the peer.

An additional issue can occur due to NAT timeouts. Because an UPDATE exchange consist of at least three HIP control packets, it is possible that UPDATE packets can arrive from unexpected UDP ports at the host in the public Internet. As a consequence, the receiving host MUST start using the new UDP port for further communication with the sender, after successful HIP packet validation.

It is important that UPDATE packet validation MUST occur before tearing down a UDP tunnel. In the worst case, an attacker could cause a DoS attack by sending a replayed UPDATE packet with incorrected UDP source port information.

2.6. HIP Multihoming

Multiple security associations can exists between the same hosts. They may be connected through several paths, some of which may include a NAT and others may not. Implementations that support multihoming MUST support concurrent HIP associations between the same host pair in a way that allows some of them to use UDP encapsulation while others use basic HIP. Implementations MAY distinguish HIP associations based on the SPI instead of a HIT pair for this purpose.

2.7. Firewall Traversal

When the initiator or the responder of a HIP association is behind a firewall, additional issues arise.

When the initiator is behind a firewall, the NAT traversal mechanisms described in Section 2 depend on the ability to initiate communication via UDP to destination ports 50500 and 54500 from arbitrary source ports and to receive UDP response traffic from those ports to the chosen source port.

Most firewall implementations support "UDP connection tracking", i.e., after a host behind a firewall has initiated a UDP communication to the public Internet, the firewall relays UDP response traffic in the return direction. If no such return traffic arrives for a specific period of time, the firewall stops relaying the given IP address and port pair. The mechanisms described in

Section 2 already enable traversal of such firewalls, if the keep-alive interval used is less than the refresh interval of the firewall.

If the initiator is behind a firewall that does not support "UDP connection tracking", the NAT traversal mechanisms described in Section 2 can still be supported, if the firewall allows permanently inbound UDP traffic from ports 50500 and 54500 and destined to arbitrary source IP addresses and UDP ports.

When the responder is behind a firewall, the NAT traversal mechanisms described in Section 2 depend on the ability to receive UDP traffic on ports 50500 and 54500 from arbitrary source IP addresses and ports.

The NAT traversal mechanisms described in Section 2 require that the firewall - stateful or not - allow inbound UDP traffic to ports 50500 and 54500 and allow outbound UDP traffic to arbitrary UDP ports.

3. Security Considerations

Section 5.1 of [RFC3948] describes a security issue for the UDP encapsulation of standard IP tunnel mode when two hosts behind different NATs have the same private IP address and initiate communication to the same responder in the public Internet. The responder cannot distinguish between the two hosts, because security associations are based on the same inner IP addresses.

This issue does not exist with the UDP encapsulation of IPsec BEET mode as described in Section 2, because the responder use the HITs to distinguish between different communication instances.

4. IANA Considerations

This section is to be interpreted according to [RFC2434].

This draft currently uses two UDP ports in the "Dynamic and/or Private Port" range, i.e., 50500 and 54500. Upon publication of this document, IANA is requested to register two UDP ports and the RFC editor is requested to change all occurrences of ports 50500 and 54500 to the two ports IANA has registered.

5. Acknowledgements

The authors would like to thank Tobias Heer, Teemu Koponen, Juhana

Mattila, Jeffrey M. Ahrenholz, Thomas Henderson, Kristian Slavov, Janne Lindqvist and Pekka Nikander for their comments on this document.

[I-D.nikander-hip-path] presented some initial ideas for NAT traversal of HIP communication. This document describes significantly different mechanisms that, among other differences, use external NAT discovery and do not require encapsulation servers.

Lars Eggert and Martin Stiemerling are partly funded by Ambient Networks, a research project supported by the European Commission under its Sixth Framework Program. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Ambient Networks project or the European Commission.

Miika Komu is working for InfraHIP research group at Helsinki Institute for Information Technology (HIIT). The InfraHIP project is funded by Tekes, Elisa, Nokia, The Finnish Defence Forces and Ericsson.

6. References

6.1. Normative References

[I-D.ietf-hip-arch]
Moskowitz, R. and P. Nikander, "Host Identity Protocol Architecture", draft-ietf-hip-arch-03 (work in progress), August 2005.

[I-D.ietf-hip-base]
Moskowitz, R., "Host Identity Protocol", draft-ietf-hip-base-04 (work in progress), October 2005.

[I-D.ietf-hip-esp]
Jokela, P., "Using ESP transport format with HIP", draft-ietf-hip-esp-01 (work in progress), October 2005.

[I-D.ietf-hip-mm]
Nikander, P., "End-Host Mobility and Multihoming with the Host Identity Protocol", draft-ietf-hip-mm-02 (work in progress), July 2005.

[I-D.nikander-esp-beet-mode]
Melen, J. and P. Nikander, "A Bound End-to-End Tunnel (BEET) mode for ESP", draft-nikander-esp-beet-mode-04

(work in progress), November 2005.

[RFC0768] Postel, J., "User Datagram Protocol", STD 6, RFC 768, August 1980.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2434] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 2434, October 1998.

6.2. Informative References

[I-D.irtf-hiprg-nat]
Stiemerling, M., "Middlebox Traversal Issues of Host Identity Protocol (HIP) Communication", draft-irtf-hiprg-nat-01 (work in progress), January 2006.

[I-D.nikander-hip-path]
Nikander, P., "Preferred Alternatives for Tunnelling HIP (PATH)", draft-nikander-hip-path-00 (work in progress), February 2005.

[RFC2663] Srisuresh, P. and M. Holdrege, "IP Network Address Translator (NAT) Terminology and Considerations", RFC 2663, August 1999.

[RFC3489] Rosenberg, J., Weinberger, J., Huitema, C., and R. Mahy, "STUN - Simple Traversal of User Datagram Protocol (UDP) Through Network Address Translators (NATs)", RFC 3489, March 2003.

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

Appendix A. Document Revision History

To be removed upon publication

Revision	Comments
00	Initial version.

Authors' Addresses

Vivien Schmitt
NEC Network Laboratories
Kurfuerstenanlage 36
Heidelberg 69115
Germany

Phone: +49 6221 90511 0
Fax: +49 6221 90511 55
Email: schmitt@netlab.nec.de
URI: http://www.netlab.nec.de/

Abhinav Pathak
IIT Kanpur
B204, Hall - 1, IIT Kanpur
Kanpur 208016
India

Phone: +91 9336 20 1002
Email: abhinav.pathak@hiit.fi
URI: http://www.iitk.ac.in/

Miika Komu
Helsinki Institute for Information Technology
Tammasaarenkatu 3
Helsinki
Finland

Phone: +358503841531
Fax: +35896949768
Email: miika@iki.fi
URI: http://www.hiit.fi/

Lars Eggert
NEC Network Laboratories
Kurfuerstenanlage 36
Heidelberg 69115
Germany

Phone: +49 6221 90511 43
Fax: +49 6221 90511 55
Email: lars.eggert@netlab.nec.de
URI: http://www.netlab.nec.de/

Martin Stiemerling
NEC Network Laboratories
Kurfuerstenanlage 36
Heidelberg 69115
Germany

Phone: +49 6221 90511 13
Fax: +49 6221 90511 55
Email: stiemerling@netlab.nec.de
URI: http://www.netlab.nec.de/

Full Copyright Statement

Intellectual Property

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.

Acknowledgment

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