HIP Working Group M. Komu
Internet-Draft HIIT
Intended status: Experimental T. Henderson
Expires: May 4, 2009 The Boeing Company
P. Matthews
(Unaffiliated)
H. Tschofenig
Nokia Siemens Networks
A. Keranen, Ed.
Ericsson Research Nomadiclab
October 31, 2008

Basic HIP Extensions for Traversal of Network Address Translators
draft-ietf-hip-nat-traversal-05.txt

Status of this Memo

Abstract

This document specifies extensions to the Host Identity Protocol (HIP) to facilitate Network Address Translator (NAT) traversal. The extensions are based on the use of the Interactive Connectivity Establishment (ICE) methodology to discover a working path between two end-hosts, and on standard techniques for encapsulating

Encapsulating Security Payload (ESP) packets within the User Datagram Protocol (UDP). This document also defines elements of procedure for NAT traversal, including the optional use of a HIP relay server. With these extensions HIP is able to work in environments that have NATs and provides a generic NAT traversal solution to higher-layer networking applications.

Table of Contents

1. Introduction

HIP [RFC5201] is defined as a protocol that runs directly over IPv4 or IPv6, and HIP coordinates the setup of ESP security associations [RFC5202] that are also specified to run over IPv4 or IPv6. This approach is known to have problems traversing NATs and other middleboxes [RFC5207]. This document defines HIP extensions for the traversal of both Network Address Translator (NAT) and Network Address and Port Translator (NAPT) middleboxes. The document generally uses the term NAT to refer to these types of middleboxes.

Currently deployed NAT devices do not operate consistently even though a recommended behavior is described in [RFC4787]. The HIP protocol extensions in this document make as few assumptions as possible about the behavior of the NAT devices so that NAT traversal will work even with legacy NAT devices. The purpose of these extensions is to allow two HIP-enabled hosts to communicate with each other even if one or both of the communicating hosts are in a network that is behind one or more NATs.

Using the extensions defined in this document, HIP end-hosts use techniques drawn from the Interactive Connectivity Establishment (ICE) methodology [I-D.ietf-mmusic-ice] to find operational paths for the HIP control protocol and for ESP encapsulated data traffic. The hosts test connectivity between different locators and try to discover a direct end-to-end path between them. However, with some legacy NATs, utilizing the shortest path between two end-hosts located behind NATs is not possible without relaying the traffic through a relay, such as a TURN server [RFC5128]. Because relaying traffic increases the roundtrip delay and consumes resources from the relay, with the extensions described in this document, hosts try to avoid using the TURN server whenever possible.

HIP has defined a Rendezvous Server [RFC5204] to allow for mobile HIP hosts to establish a stable point-of-contact in the Internet. This document defines extensions to the Rendezvous Server that solve the same problems but for both NATed and non-NATed networks. The extended Rendezvous Server, called a "HIP relay server," forwards all HIP control packets between an Initiator and Responder, allowing Responders to be located behind NATs. This behavior is in contrast to the HIP rendezvous service that forwards only the initial I1 packet of the base exchange, which is less likely to work in a NATed environment [RFC5128]. Therefore, when using relays to traverse NATs, HIP uses a HIP relay server for the control traffic and a TURN server for the data traffic.

The basis for the connectivity checks is ICE [I-D.ietf-mmusic-ice]. [I-D.ietf-mmusic-ice] describes ICE as follows:

> "The Interactive Connectivity Establishment (ICE) methodology is a technique for NAT traversal for UDP-based media streams (though ICE can be extended to handle other transport protocols, such as TCP) established by the offer/answer model. ICE is an extension to the offer/answer model, and works by including a multiplicity of IP addresses and ports in SDP offers and answers, which are then tested for connectivity by peer-to-peer connectivity checks. The IP addresses and ports included in the SDP and the connectivity checks are performed using the revised STUN specification [RFC5389], now renamed to Session Traversal Utilities for NAT."

The standard ICE [I-D.ietf-mmusic-ice] is specified with SIP in mind and it has some features that are not necessary or suitable as such for other protocols. [I-D.rosenberg-mmusic-ice-nonsip] gives instructions and recommendations on how ICE can be used for other protocols and this document follows those guidelines.

Two HIP hosts that implement this specification communicate their locators to each other in the HIP base exchange. The locators are then paired with the locators of the other endpoint and prioritized according to recommended and local policies. These locator pairs are then tested sequentially by both of the end hosts. The tests may result in multiple operational pairs but ICE procedures determine a single preferred address pair to be used for subsequent communication.

In summary, the extensions in this document define:

- UDP encapsulation of HIP packets
- UDP encapsulation of IPsec ESP packets
- registration extensions for HIP relay services
- how the ICE "offer" and "answer" are carried in the base exchange
- interaction with ICE connectivity check messages
- backwards compatibility issues with rendezvous servers
- a number of optimizations (such as when the ICE connectivity tests can be omitted)

2. Terminology

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].

This document borrows terminology from [RFC5201], [RFC5206], [RFC4423], [I-D.ietf-mmusic-ice], and [RFC5389]. Additionally, the following terms are used:

Rendezvous server:
A host that forwards I1 packets to the Responder.

HIP relay server:
A host that forwards all HIP control packets between the Initiator

and the Responder.

TURN server:
A server that forwards data traffic between two end-hosts as defined in [I-D.ietf-behave-turn].

Locator:
As defined in [RFC5206]: "A name that controls how the packet is routed through the network and demultiplexed by the end-host. It may include a concatenation of traditional network addresses such as an IPv6 address and end-to-end identifiers such as an ESP SPI. It may also include transport port numbers or IPv6 Flow Labels as demultiplexing context, or it may simply be a network address." It should noted that "address" is used in this document as a synonym for locator.

LOCATOR (written in capital letters):
Denotes a HIP control message parameter that bundles multiple locators together.

ICE offer:
The Initiator's LOCATOR parameter in a HIP I2 control message.

ICE answer:
The Responder's LOCATOR parameter in a HIP R2 control message.

Transport address:
Transport layer port and the corresponding IPv4/v6 address.

Candidate:
A transport address that is a potential point of contact for receiving data.

Host candidate:
A candidate obtained by binding to a specific port from an IP address on the host.

Server reflexive candidate:
A translated transport address of a host as observed by a HIP relay server or a STUN/TURN server.

Peer reflexive candidate:
A translated transport address of a host as observed by its peer.

Relayed candidate:
A transport address that exists on a TURN server. Packets that arrive at this address are relayed towards the TURN client.

3. Overview of Operation

```
                                 +-------+
                                 | HIP   |
              +--------+         | Relay |         +--------+
              | TURN   |         +-------+         | STUN   |
              | Server |        /         \        | Server |
              +--------+       /           \       +--------+
                              /             \
                             /               \
                            /                 \
                           /  <- Signaling ->  \
                          /                     \
                    +-------+                +-------+
                    |  NAT  |                |  NAT  |
                    +-------+                +-------+
                     /                              \
                    /                                \
               +-------+                           +-------+
               | Init- |                           | Resp- |
               | iator |                           | onder |
               +-------+                           +-------+
```

Figure 1: Example network configuration

In an example configuration depicted in Figure 1, both Initiator and Responder are behind one or more NATs, and both private networks are connected to the public Internet. To be contacted from behind a NAT, the Responder must be registered with a HIP relay server reachable on the public Internet, and we assume as a starting point that the Initiator knows both the Responder's HIT and the address of one of

its relay servers (how the Initiator learns of the Responder's relay server is outside of the scope of this document, but may be through DNS or another name service).

The first steps are for both the Initiator and Responder to register with a relay server (need not be the same one) and gather a set of address candidates. Next, the HIP base exchange is carried out by encapsulating the HIP control packets in UDP datagrams and sending them through the Responder's relay server. As part of the base exchange, each HIP host learns of the peer's candidate addresses through the ICE offer/answer procedure embedded in the base exchange.

Once the base exchange is completed, HIP has established a working communication session (for signaling) via a relay server, but the hosts still work to find a better path, preferably without a relay, for the ESP data flow. For this, ICE connectivity checks are carried out until a working pair of addresses is discovered. At the end of the procedure, if successful, the hosts will have enabled a UDP-based flow that traverses both NATs, with the data flowing directly from NAT to NAT or via a TURN server. Further HIP signaling can be sent over the same address/port pair and is demultiplexed from data traffic via a marker in the payload. Finally, NAT keepalives will be sent as needed.

If either one of the hosts knows that it is not behind a NAT, hosts can negotiate during the base exchange a different mode of NAT traversal that does not use ICE connectivity checks, but only UDP encapsulation of HIP and ESP. Also, it is possible for the Initiator to simultaneously try a base exchange with and without UDP encapsulation. If a base exchange without UDP encapsulation succeeds, no ICE connectivity checks or UDP encapsulation of ESP are needed.

4. Protocol Description

This section describes the normative behavior of the protocol extension. Examples of packet exchanges are provided for illustration purposes.

4.1. Relay Registration

HIP rendezvous servers operate in non-NATed environments and their use is described in [RFC5204]. This section specifies a new middlebox extension, called the HIP relay server, for operating in NATed environments. A HIP relay server forwards all HIP control packets between the Initiator and the Responder.

End-hosts cannot use the HIP relay service for forwarding the ESP data plane. Instead, they use TURN servers [I-D.ietf-behave-turn] for that.

A HIP relay server MUST silently drop packets to a HIP relay client that has not previously registered with the HIP relay. The registration process follows the generic registration extensions defined in [RFC5203] and is illustrated in Figure 2.

```
      HIP                                                      HIP
      Relay                                                    Relay
      Client                                                   Server
        |   1. UDP(I1)                                           |
        +------------------------------------------------------->|
        |                                                        |
        |   2. UDP(R1(REG_INFO(RELAY_UDP_HIP)))                  |
        |<-------------------------------------------------------+
        |                                                        |
        |   3. UDP(I2(REG_REQ(RELAY_UDP_HIP)))                   |
        +------------------------------------------------------->|
        |                                                        |
        |   4. UDP(R2(REG_RES(RELAY_UDP_HIP), REG_FROM))         |
        |<-------------------------------------------------------+
```

Figure 2: Example Registration to a HIP Relay

In step 1, the relay client (Initiator) starts the registration procedure by sending an I1 packet over UDP. It is RECOMMENDED that the Initiator selects a random port number from the ephemeral port range 49152-65535 for initiating a base exchange. However, the allocated port MUST be maintained until all of the corresponding HIP Associations are closed. Alternatively, a host MAY also use a single fixed port for initiating all outgoing connections. The HIP relay server MUST listen to incoming connections at UDP port HIPPORT.

In step 2, the HIP relay server (Responder) lists the services that it supports in the R1 packet. The support for HIP-over-UDP relaying is denoted by the RELAY_UDP_HIP value.

In step 3, the Initiator selects the services it registers for and lists them in the REG_REQ parameter. The Initiator registers for HIP relay service by listing the RELAY_UDP_HIP value in the request parameter.

In step 4, the Responder concludes the registration procedure with an

R2 packet and acknowledges the registered services in the REG_RES parameter. The Responder denotes unsuccessful registrations (if any)

in the REG_FAILED parameter of R2. The Responder also includes a REG_FROM parameter that contains the transport address of the client as observed by the relay (Server Reflexive candidate). After the registration, the client sends NAT keepalives periodically to the relay to keep possible NAT bindings between the client and the relay alive.

4.2. ICE Candidate Gathering

If a host is going to use ICE, it needs to gather a set of address candidates. The candidate gathering SHOULD be done as defined in Section 4.1 of [I-D.ietf-mmusic-ice]. Candidates need to be gathered for only one media stream and component. Component ID 1 should be used for ICE processing, where needed. Initiator takes the role of the ICE controlling agent.

The candidate gathering can be done at any time, but it needs to be done before sending an I2 or R2 if ICE is used for the connectivity checks. It is RECOMMENDED that all three types of candidates (host, server reflexive and relayed) are gathered to maximize probability of successful NAT traversal. However, if no TURN server is used, and the host has only a single local IP address to use, the host MAY use the local address as the only host candidate and the address from the REG_FROM parameter discovered during the relay registration as a server reflexive candidate. In this case, no further candidate gathering is needed.

4.3. NAT Traversal Mode Negotiation

This section describes the usage of a new non-critical parameter type. The presence of the parameter in a HIP base exchange means that the end-host supports NAT traversal extensions described in this document. As the parameter is non-critical, it can be ignored by an end-host which means that the host does not support or is not willing to use these extensions.

The NAT traversal mode parameter applies to a base exchange between end-hosts, but currently does not apply to a registration with a HIP relay server. The NAT traversal mode negotiation in base exchange is illustrated in Figure 3.

```
    Initiator                                               Responder
      | 1. UDP(I1)                                                  |
      +------------------------------------------------------------>|
      |                                                             |
      | 2. UDP(R1(.., NAT_TRAVERSAL_MODE(list of modes), ..))       |
      |<------------------------------------------------------------+
      |                                                             |
      | 3. UDP(I2(.., NAT_TRAVERSAL_MODE(selected mode), LOCATOR..)) |
      +------------------------------------------------------------>|
      |                                                             |
      | 4. UDP(R2(.., LOCATOR, ..))                                 |
      |<------------------------------------------------------------+
      |                   ....                                      |
```

Figure 3: Negotiation of NAT Traversal Mode

In step 1, the Initiator sends an I1 to the Responder. In step 2, the Responder responds with an R1. The R1 contains a list of NAT traversal modes the Responder supports in the NAT_TRAVERSAL_MODE parameter as shown in Table 1.

Type	Purpose
RESERVED	Reserved for future use
UDP-ENCAPSULATION	Use only UDP encapsulation of the HIP signaling traffic and ESP (no ICE connectivity checks)
ICE-STUN-UDP	UDP encapsulated control and data traffic with ICE-based connectivity checks using STUN messages

```
   +-------------------+-------------------------------------------+
```

Table 1: NAT Traversal Modes

In step 3, the Initiator sends an I2 that includes a NAT_TRAVERSAL_MODE parameter. It contains the mode selected by the Initiator from the list of modes offered by the Responder. The I2 also includes the locators of the Initiator in a LOCATOR parameter. The locator parameter in I2 is the "ICE offer".

In step 4, the Responder concludes the base exchange with an R2 packet. The Responder includes a LOCATOR parameter in the R2 packet. The locator parameter in R2 is the "ICE answer".

4.4. Connectivity Check Pacing Negotiation

As explained in [I-D.ietf-mmusic-ice], when a NAT traversal mode with connectivity checks is used, new transactions should not be started too fast to avoid congestion and overwhelming the NATs.

For this purpose, during the base exchange, hosts can negotiate a transaction pacing value, Ta, using a TRANSACTION_PACING parameter in I2 and R2 messages. The parameter contains the minimum time (expressed in milliseconds) the host would wait between two NAT traversal transactions, such as starting a new connectivity check or retrying a previous check. If a host does not include this parameter in the base exchange, a Ta value of 500ms MUST be used as that host's minimum value. The value that is used by both of the hosts is the higher out of the two offered values.

Hosts SHOULD NOT use values smaller than 20ms for the minimum Ta, since such values may not work well with some NATs, as explained in [I-D.ietf-mmusic-ice].

The minimum Ta value SHOULD be configurable. Guidelines for selecting a Ta value are given in Appendix A. Currently this feature applies only to the ICE-STUN-UDP NAT traversal mode.

4.5. Base Exchange via HIP Relay Server

This section describes how Initiator and Responder perform a base exchange through a HIP relay server. The NAT traversal mode negotiation (denoted as NAT_TM in the example) was described in the previous section and shall not be repeated here. If a relay receives an R1 or I2 packet without the NAT traversal mode parameter, it drops it and sends a NOTIFY error message to the sender of the R1/I2.

It is RECOMMENDED that the Initiator sends an I1 packet encapsulated in UDP when it is destined to an IPv4 address of the Responder. Respectively, the Responder MUST respond to such an I1 packet with an R1 packet over the transport layer and using the same transport protocol. The rest of the base exchange, I2 and R2, MUST also use the same transport protocol.

```
     I                           HIP relay                           R
     | 1. UDP(I1)                   |                                |
     +----------------------------->| 2. UDP(I1(RELAY_FROM))         |
     |                              +------------------------------->|
     |                              |                                |
     |                              | 3. UDP(R1(RELAY_TO, NAT_TM))   |
     | 4. UDP(R1(RELAY_TO),NAT_TM ) |<-------------------------------+
     |<-----------------------------+                                |
     |                              |                                |
     | 5. UDP(I2(LOCATOR),NAT_TM)   |                                |
     +----------------------------->|  6. UDP(I2(LOCATOR,RELAY_FROM),|
     |                              |       NAT_TM)                  |
     |                              +------------------------------->|
     |                              |                                |
     |                              | 7. UDP(R2(LOCATOR,RELAY_TO))   |
     | 8. UDP(R2(LOCATOR,RELAY_TO)) |<-------------------------------+
     |<-----------------------------+                                |
     |                              |                                |
```

Figure 4: Base Exchange via a HIP Relay Server

In step 1 of Figure 4, the Initiator sends an I1 packet over the transport layer to the HIT of the Responder (and IP address of the relay). The source address is one of the locators of the Initiator.

In step 2, the HIP relay server receives the I1 packet at port HIPPORT. If the destination HIT belongs to a registered Responder, the relay processes the packet. Otherwise, the relay MUST drop the packet silently. The relay appends a RELAY_FROM parameter to the I1 packet which contains the transport source address and port of the I1 as observed by the relay. The relay protects the I1 packet with RELAY_HMAC as described in [RFC5204], except that the parameter type is different (see Section 5.8). The relay changes the source and destination ports and IP addresses of the packet to match the values the Responder used when registering to the relay, i.e., the reverse of the R2 used in the registration. The relay MUST recalculate the transport checksum and forward the packet to the Responder.

In step 3, the Responder receives the I1 packet. The Responder processes it according to the rules in [RFC5201]. In addition, the Responder validates the RELAY_HMAC according to [RFC5204] and silently drops the packet if the validation fails. The Responder replies with an R1 packet to which it includes a RELAY_TO parameter. The RELAY_TO parameter MUST contain same information as the RELAY_FROM parameter, i.e., the Initiator's transport address, but the type of the parameter is different. The RELAY_TO parameter is not integrity protected by the signature of the R1 to allow pre-created R1 packets at the Responder.

In step 4, the relay receives the R1 packet. The relay drops the packet silently if the source HIT belongs to an unregistered host. The relay MAY verify the signature of the R1 packet and drop it if the signature is invalid. Otherwise, the relay rewrites the source address and port, and changes the destination address and port to match RELAY_TO information. Finally, the relay recalculates transport checksum and forwards the packet.

In step 5, the Initiator receives the R1 packet and processes it according to [RFC5201]. It replies with an I2 packet that uses the destination transport address of R1 as the source address and port. The I2 contains a LOCATOR parameter that lists all the ICE candidates (ICE offer) of the Initiator. The candidates are encoded using the format defined in Section 5.7. The I2 packet MUST also contain the NAT traversal mode parameter with ICE-STUN-UDP or some other selected mode.

In step 6, the relay receives the I2 packet. The relay appends a RELAY_FROM and a RELAY_HMAC to the I2 packet as explained in step 2.

In step 7, the Responder receives the I2 packet and processes it according to [RFC5201]. It replies with an R2 packet and includes a RELAY_TO parameter as explained in step 3. The R2 packet includes a LOCATOR parameter that lists all the ICE candidates (ICE answer) of the Responder. The RELAY_TO parameter is protected by the HMAC.

In step 8, the relay processes the R2 as described in step 4. The relay forwards the packet to the Initiator.

Hosts MAY include the address of their HIP relay server in the LOCATOR parameter in I2/R2. The traffic type of this address MUST be "HIP signaling" and it MUST NOT be used as an ICE candidate. This address MAY be used for HIP signaling also after the base exchange. If the HIP relay server locator is not included in I2/R2 LOCATOR parameters, it SHOULD NOT be used after the base exchange, but the HIP signaling SHOULD use the same path as the data traffic.

4.6. ICE Connectivity Checks

If a HIP relay server was used, the Responder completes the base exchange with the R2 packet through the relay. When the Initiator successfully receives and processes the R2, both hosts have transitioned to ESTABLISHED state. However, the destination address the Initiator and Responder used for delivering base exchange packets belonged to the HIP relay server. Therefore, the address of the relay MUST NOT be used for sending ESP traffic. Instead, if a NAT

traversal mode with ICE connectivity checks was selected, the Initiator and Responder MUST start the connectivity checks.

Creating the check list for the ICE connectivity checks should be performed as described in Section 5.7 of [I-D.ietf-mmusic-ice] bearing in mind that only one media stream and component is needed (so there will be only a single checklist and all candidates should have the same component ID value). The actual connectivity checks MUST be performed as described in Section 7 of [I-D.ietf-mmusic-ice]. Regular mode SHOULD be used for the candidate nomination. Section 5.2 defines the details of the STUN control packets. As a result of the ICE connectivity checks, ICE nominates a single transport address pair to be used if an operational address pair was found. The end-hosts MUST use this address pair for the ESP traffic.

The connectivity check messages MUST be paced by the value negotiated during the base exchange as described in Section 4.4. If neither one of the hosts announced a minimum pacing value, value of 500ms MUST be used.

For retransmissions, the RTO value should be calculated as follows:

```
   RTO = MAX (500ms, Ta * P)
```

In the RTO formula, Ta is the value used for the connectivity check pacing and P is the number of pairs in the checklist when the connectivity checks begin. This is identical to the formula in [I-D.ietf-mmusic-ice] if there is only one checklist.

4.7. NAT Keepalives

To prevent NAT states from expiring, communicating hosts send periodically keepalives to each other. HIP relay servers MAY refrain from sending keepalives if it's known that they are not behind a middlebox that requires keepalives. An end-host MUST send keepalives every 15 seconds to refresh the UDP port mapping at the NAT(s) when the control or data channel is idle. To implement failure tolerance, an end-host SHOULD have shorter keepalive period.

The keepalives are STUN Binding Indications if the hosts have agreed on ICE-STUN-UDP NAT traversal mode during the base exchange. Otherwise, HIP NOTIFY messages MAY be used. A HIP relay server MUST NOT forward the NOTIFY messages.

The communicating hosts MUST send keepalives to each other using the transport locators they agreed to use for data and signaling when they are in ESTABLISHED state. Also, the Initiator MUST send a NOTIFY message to the relay to keep the NAT states alive on the path between the Initiator and relay when the Initiator has not received any response to its I1 or I2 from the Responder in 15 seconds. The relay MUST NOT forward the NOTIFY messages.

4.8. Base Exchange without ICE Connectivity Checks

In certain network environments, the ICE connectivity checks can be omitted to reduce initial connection set up latency because base exchange acts as an implicit connectivity test itself. There are three assumptions about such as environments. First, the Responder should have a long-term, fixed locator in the network. Second, the Responder should not have a HIP relay server configured for itself. Third, the Initiator can reach the Responder by simply UDP encapsulating HIP and ESP packets to the host. Detecting and configuring this particular scenario is prone to administrative failure unless carefully planned.

In such a scenario, the Responder MAY include only the UDP-ENCAPSULATION NAT traversal mode in the R1 message. Likewise, if the Initiator knows that it can receive ESP and HIP signaling traffic by using simply UDP encapsulation, it can choose the UDP-ENCAPSULATION mode in the I2 message, if the Responder listed it in the supported modes. In both of these cases the locators from I2 and R2 packets will be used also for the UDP encapsulated ESP.

When no ICE connectivity checks are used, locator exchange and return routability tests for mobility and multihoming are done as specified in [RFC5206] with the exception that UDP encapsulation is used.

4.9. Simultaneous Base Exchange with and without UDP Encapsulation

The Initiator MAY also try to simultaneously perform a base exchange with the Responder without UDP encapsulation. In such a case, the Initiator sends two I1 packets, one without and one with UDP encapsulation, to the Responder. The Initiator MAY wait for a while before sending the other I1. How long to wait and in which order to send the I1 packets can be decided based on local policy. For retransmissions, the procedure is repeated.

The I1 packet without UDP encapsulation may arrive directly at the Responder. When the recipient is the Responder, the procedures in [RFC5201] are followed for the rest of the base exchange. The Initiator may receive multiple R1 messages, with and without UDP encapsulation, from the Responder. However, after receiving a valid R1 and answering to it with an I2, further R1 messages that are not retransmits of the original R1 MUST be ignored.

The I1 packet without UDP encapsulation may also arrive at a HIP-

capable middlebox. When the middlebox is a HIP rendezvous server and the Responder has successfully registered to the rendezvous service, the middlebox follows rendezvous procedures in [RFC5204].

If the Initiator receives a NAT traversal mode parameter in R1 without UDP encapsulation, the Initiator MAY ignore this parameter and send an I2 without UDP encapsulation and without any selected NAT traversal mode. When the Responder receives the I2 without UDP encapsulation and without NAT traversal mode, it will assume that no NAT traversal mechanism is needed. The packet processing will be done as described in [RFC5201]. The Initiator MAY store the NAT traversal modes for future use e.g., to be used in case of mobility or multihoming event which causes NAT traversal to be taken in to use during the lifetime of the HIP association.

4.10. Sending Control Messages after the Base Exchange

After the base exchange, the end-hosts MAY send HIP control messages directly to each other using the transport address pair established for data channel without sending the control packets through the HIP relay server. When a host does not get acknowledgments, e.g., to an UPDATE or CLOSE message after a timeout based on local policies, the host SHOULD resend the packet through the relay, if it was listed in the LOCATOR parameter in the base exchange.

If control messages are sent through a HIP relay server, the sender MUST include a RELAY_TO parameter to them. Also the HIP relay server MUST add a RELAY_FROM parameter to the control messages it relays.

5. Packet Formats

The following subsections define the parameter and packet encodings for the HIP, ESP and ICE connectivity check packets. All values MUST be in network byte order.

5.1. HIP Control Packets

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |        Source Port            |       Destination Port        |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |           Length              |           Checksum            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                      32 bits of zeroes                        |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                                                               |
   ~                  HIP Header and Parameters                    ~
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 5: Format of UDP-encapsulated HIP Control Packets

HIP control packets are encapsulated in UDP packets as defined in Section 2.2 of [RFC3948], "rules for encapsulating IKE messages", except a different port number is used. Figure 5 illustrates the encapsulation. The UDP header is followed by 32 zero bits that can be used to differentiate HIP control packets from ESP packets. The HIP header and parameters follow the conventions of [RFC5201] with the exception that the HIP header checksum MUST be zero. The HIP header checksum is zero for two reasons. First, the UDP header contains already a checksum. Second, the checksum definition in [RFC5201] includes the IP addresses in the checksum calculation. The NATs unaware of HIP cannot recompute the HIP checksum after changing IP addresses.

A HIP relay server or a Responder without a relay MUST listen at UDP port HIPPORT for incoming UDP encapsulated HIP control packets.

5.2. Connectivity Checks

The connectivity checks are performed using STUN Binding Requests as defined in [I-D.ietf-mmusic-ice]. This section describes the details of the parameters in the STUN messages.

The Binding Requests MUST use STUN short term credentials with HITs of the Initiator and Responder as the username fragments. The username is formed from the username fragments as defined in Section 7.1.1.3 of [I-D.ietf-mmusic-ice] with the Initiator being the "offerer" and the Responder being the "answerer". The HITs are used as usernames by expressing them in IPv6 hexadecimal ASCII format [RFC1884], using lowercase letters, each 16 bit HIT fragment separated by a one byte colon (hex 0x3a). The leading zeroes MUST

NOT be omitted so that the username's size is fixed.

The STUN password is drawn from the DH keying material. Drawing of HIP keys is defined in [RFC5201] Section 6.5 and drawing of ESP keys in [RFC5202] Section 7. Correspondingly, the hosts MUST draw symmetric keys for STUN according to [RFC5201] Section 6.5. The hosts draw the STUN key after HIP keys, or after ESP keys if ESP transform was successfully negotiated in the base exchange. Both hosts draw a 128 bit key from the DH keying material, express that in hexadecimal ASCII format using only lowercase letters (resulting in 32 numbers or lowercase letters), and use that as both the local and peer password. [RFC5389] describes how hosts use the password for message integrity of STUN messages.

Both the username and password are expressed in ASCII hexadecimal format to prevent the need to run them through SASLPrep as defined in [RFC5389].

The connectivity checks MUST contain PRIORITY attribute. They MAY contain USE-CANDIDATE attribute as defined in Section 7.1.1.1 of [I-D.ietf-mmusic-ice].

The Initiator is always in the controller role during a base exchange. Hence, the ICE-CONTROLLED and ICE-CONTROLLING attributes are not needed and SHOULD NOT be used. When two hosts are initiating a connection to each other simultaneously, HIP state machine detects it and assigns the host with the larger HIT as the Responder as explained in Sections 4.4.2 and 6.7 in [RFC5201].

5.3. Keepalives

The keepalives for HIP associations that are created with ICE are STUN Binding Indications, as defined in [RFC5389]. In contrast to the UDP encapsulated HIP header, the non-ESP-marker between the UDP header and the STUN header is excluded. Keepalives MUST contain the FINGERPRINT STUN attribute but SHOULD NOT contain any other STUN attributes and SHOULD NOT utilize any authentication mechanism. STUN messages are demultiplexed from ESP and HIP control messages using the STUN markers, such as the magic cookie value and the FINGERPRINT attribute.

Keepalives for HIP associations created without ICE are HIP control messages that have NOTIFY as the packet type. The NOTIFY messages do not contain any parameters.

5.4. NAT Traversal Mode Parameter

Format of the NAT_TRAVERSAL_MODE parameter is similar to the format of the ESP_TRANSFORM parameter in [RFC5202] and is shown in the Figure 6. This specification defines traversal mode identifiers UDP-ENCAPSULATION and ICE-STUN-UDP. The identifier RESERVED is reserved for future use. Future specifications may define more traversal modes.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |             Type              |             Length            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |           Reserved            |          Mode ID #1           |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |           Mode ID #2          |          Mode ID #3           |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |          Mode ID #n           |            Padding            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

```
   Type       [ TBD by IANA: 608 ]
   Length     length in octets, excluding Type, Length, and padding
   Reserved   zero when sent, ignored when received
   Mode ID    defines the NAT traversal mode to be used
```

The following NAT traversal mode IDs are defined:

ID	Value
RESERVED	0
UDP-ENCAPSULATION	1
ICE-STUN-UDP	2

Figure 6: Format of the NAT_TRAVERSAL_MODE parameter

The sender of a NAT_TRAVERSAL_MODE parameter MUST make sure that there are no more than six (6) Mode IDs in one NAT_TRAVERSAL_MODE parameter. The limited number of Mode IDs sets the maximum size of the NAT_TRAVERSAL_MODE parameter.

5.5. Connectivity Check Transaction Pacing Parameter

The TRANSACTION_PACING parameter shown in Figure 7 contains only the connectivity check pacing value, expressed in milliseconds, as 32 bit unsigned integer.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |             Type              |            Length             |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                            Min Ta                             |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

```
   Type     [ TBD by IANA: 610 ]
   Length   length in octets, excluding Type and Length
   Min Ta   the minimum connectivity check transaction pacing
            value the host would use
```

Figure 7: Format of the TRANSACTION_PACING parameter

5.6. Relay and Registration Parameters

Format of the REG_FROM, RELAY_FROM and RELAY_TO parameters is shown in Figure 8. All parameters are identical except for the type. REG_FROM is the only parameter covered with the signature.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|             Type              |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|             Port              |    Protocol   |     Reserved  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                            Address                            |
|                                                               |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

Type       [ TBD by IANA:
           REG_FROM:   950
           RELAY_FROM: 63998 (2^16 - 2^11 + 2^9 - 2)
           RELAY_TO:   64002 (2^16 - 2^11 + 2^9 + 2) ]
Length     20
Port       transport port number; zero when plain IP is used
Protocol   IANA assigned, Internet Protocol number.
           17 for UDP, 0 for plain IP.
Reserved   reserved for future use; zero when sent, ignored
           when received
Address    an IPv6 address or an IPv4 address in "IPv4-Mapped
           IPv6 address" format
```

Figure 8: Format of the REG_FROM, RELAY_FROM and RELAY_TO parameters

REG_FROM contains the transport address and protocol where the HIP relay server sees the registration coming from. RELAY_FROM contains the address where the relayed packet was received from by the relay server and the protocol that was used. The RELAY_TO contains same information about the address where a packet should be forwarded to.

5.7. LOCATOR Parameter

The generic LOCATOR parameter format is the same as in [RFC5206]. However, presenting ICE candidates requires a new locator type. The generic and NAT traversal specific locator parameters are illustrated in Figure 9.

```
      0                   1                   2                   3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |             Type              |            Length             |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     | Traffic Type  |  Locator Type | Locator Length|  Reserved   |P|
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |                       Locator Lifetime                        |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |                            Locator                            |
     |                                                               |
     |                                                               |
     |                                                               |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     .                                                               .
     .                                                               .
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     | Traffic Type  |  Loc Type = 2 | Locator Length|  Reserved   |P|
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |                       Locator Lifetime                        |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |     Transport Port            |  Transp. Proto|     Kind      |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |                           Priority                            |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |                              SPI                              |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |                            Locator                            |
     |                                                               |
     |                                                               |
     |                                                               |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 9: LOCATOR parameter

The individual fields in the LOCATOR parameter are described in Table 2.

Field	Value(s)	Purpose
Type	193	Parameter type
Length	Variable	Length in octets, excluding Type and Length fields and padding
Traffic Type	0-2	Is the locator for HIP signaling (1), for ESP (2), or for both (0)
Locator Type	2	"Transport address" locator type
Locator Length	7	Length of the fields after Locator Lifetime in 4-octet units
Reserved	0	Reserved for future extensions
Preferred (P) bit	0	Not used for transport address locators; MUST be ignored by the receiver.
Locator Lifetime	Variable	Locator lifetime in seconds
Transport Port	Variable	Transport layer port number
Transport Protocol	Variable	IANA Assigned, transport layer Internet Protocol number. Currently only UDP (17) is supported.
Kind	Variable	0 for host, 1 for server reflexive, 2 for peer reflexive or 3 for relayed address
Priority	Variable	Locator's priority as described in [I-D.ietf-mmusic-ice]
SPI	Variable	SPI value which the host expects to see in incoming ESP packets that use this locator
Locator	Variable	IPv6 address or an "IPv4-Mapped IPv6 address" format IPv4 address [RFC3513]

Table 2: Fields of the LOCATOR parameter

5.8. RELAY_HMAC Parameter

The RELAY_HMAC parameter value has the TLV type 65520 (2^16 - 2^5 + 2^4). It has the same semantics as RVS_HMAC [RFC5204].

5.9. Registration Types

The REG_INFO, REG_REQ, REG_RESP and REG_FAILED parameters contain values for HIP relay server registration. The value for RELAY_UDP_HIP is 2.

5.10. ESP Data Packets

[RFC3948] describes UDP encapsulation of the IPsec ESP transport and tunnel mode. On the wire, the HIP ESP packets do not differ from the transport mode ESP and thus the encapsulation of the HIP ESP packets is same as the UDP encapsulation transport mode ESP. However, the (semantic) difference to BEET mode ESP packets used by HIP is that IP header is not used in BEET integrity protection calculation.

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 [RFC5202]. When two peers perform a UDP-encapsulated base exchange, they MUST define a pair of IPsec SAs that produces UDP-encapsulated ESP data traffic.

The management of encryption/authentication protocols and SPIs is defined in [RFC5202]. The UDP encapsulation format and processing of HIP ESP traffic is described in Section 6.1 of [RFC5202].

6. Security Considerations

6.1. Privacy Considerations

The locators are in plain text format in favor of inspection at HIP-aware middleboxes in the future. The current draft does not specify encrypted versions of LOCATORs even though it could be beneficial for privacy reasons.

It is possible that an Initiator or Responder may not want to reveal all of its locators to its peer. For example, a host may not want to reveal the internal topology of the private address realm and it discards host addresses. Such behavior creates non-optimal paths when the hosts are located behind the same NAT. Especially, this could be a problem with a legacy NAT that does not support routing from the private address realm back to itself through the outer address of the NAT. This scenario is referred to as the hairpin problem [RFC5128]. With such a legacy NAT, the only option left would be to use a relayed transport address from a TURN server.

As a consequence, a host may support locator-based privacy by leaving out the reflexive candidates. However, the trade-off in using only host candidates can produce suboptimal paths that can congest the TURN server.

The use of HIP relay servers or TURN relays can be also useful for protection against Denial-of-Service attacks. If a Responder reveals only its HIP relay server addresses and Relayed candidates to

Initiators, the Initiators can only attack the relays. That does not prevent the Responder from initiating new outgoing connections if a path around the relay exists.

6.2. Opportunistic Mode

A HIP relay server should have one address per relay client when a HIP relay is serving more than one relay clients and supports opportunistic mode. Otherwise, it cannot be guaranteed that the HIP relay server can deliver the I1 packet to the intended recipient.

6.3. Base Exchange Replay Protection for HIP Relay Server

In certain scenarios, it is possible that an attacker, or two attackers, can replay an earlier base exchange through a HIP relay server by masquerading as the original Initiator and Responder. The attack does not require the attacker(s) to compromise the private key(s) of the attacked host(s). However, for this attack to succeed, the Responder has to be disconnected from the HIP relay server.

The relay can protect itself against replay attacks by involving in the base exchange by introducing nonces that the end-hosts (Initiator and Responder) have to sign. One way to do this is to add ECHO_REQUEST_M parameters to the R1 and I2 messages as described in [I-D.heer-hip-middle-auth] and drop the I2 or R2 messages if the corresponding ECHO_RESPONSE_M parameters are not present.

6.4. Demuxing Different HIP Associations

Section 5.1 of [RFC3948] describes a security issue for the UDP encapsulation in the 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 two hosts, because security associations are based on the same inner IP addresses.

This issue does not exist with the UDP encapsulation of HIP ESP transport format because the Responder uses HITs to distinguish between different Initiators.

7. IANA Considerations

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

This draft currently uses a UDP port in the "Dynamic and/or Private Port" and HIPPORT. Upon publication of this document, IANA is requested to register a UDP port and the RFC editor is requested to

change all occurrences of port HIPPORT to the port IANA has registered. The HIPPORT number 50500 should be used for initial experimentation.

This document updates the IANA Registry for HIP Parameter Types by assigning new HIP Parameter Type values for the new HIP Parameters: RELAY_FROM, RELAY_TO and REG_FROM (defined in Section 5.6), RELAY_HMAC (defined in Section 5.8), TRANSACTION_PACING (defined in Section 5.5), and NAT_TRAVERSAL_MODE (defined in Section 5.4).

8. Contributors

This draft is a product of a design team which also included Marcelo Bagnulo and Jan Melen who both have made major contributions to this document.

9. Acknowledgments

Thanks for Jonathan Rosenberg and the rest of the MMUSIC WG folks for the excellent work on ICE. In addition, the authors would like to thank Andrei Gurtov, Simon Schuetz, Martin Stiemerling, Lars Eggert, Vivien Schmitt, Abhinav Pathak for their contributions and Tobias Heer, Teemu Koponen, Juhana Mattila, Jeffrey M. Ahrenholz, Kristian Slavov, Janne Lindqvist, Pekka Nikander, Lauri Silvennoinen, Jukka Ylitalo, Juha Heinanen, Joakim Koskela, Samu Varjonen, Dan Wing and Jani Hautakorpi for their comments on this document.

Miika Komu is working in the Networking Research group at Helsinki Institute for Information Technology (HIIT). The InfraHIP project was funded by Tekes, Telia-Sonera, Elisa, Nokia, the Finnish Defence Forces, and Ericsson and Birdstep.

10. References

10.1. Normative References

[I-D.ietf-behave-turn]
Rosenberg, J., Mahy, R., and P. Matthews, "Traversal Using Relays around NAT (TURN): Relay Extensions to Session Traversal Utilities for NAT (STUN)", draft-ietf-behave-turn-11 (work in progress), October 2008.

[I-D.ietf-mmusic-ice]
Rosenberg, J., "Interactive Connectivity Establishment

(ICE): A Protocol for Network Address Translator (NAT) Traversal for Offer/Answer Protocols", draft-ietf-mmusic-ice-19 (work in progress), October 2007.

[RFC1884] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 1884, December 1995.

[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.

[RFC3513] Hinden, R. and S. Deering, "Internet Protocol Version 6 (IPv6) Addressing Architecture", RFC 3513, April 2003.

[RFC4423] Moskowitz, R. and P. Nikander, "Host Identity Protocol (HIP) Architecture", RFC 4423, May 2006.

[RFC5201] Moskowitz, R., Nikander, P., Jokela, P., and T. Henderson, "Host Identity Protocol", RFC 5201, April 2008.

[RFC5202] Jokela, P., Moskowitz, R., and P. Nikander, "Using the Encapsulating Security Payload (ESP) Transport Format with the Host Identity Protocol (HIP)", RFC 5202, April 2008.

[RFC5203] Laganier, J., Koponen, T., and L. Eggert, "Host Identity Protocol (HIP) Registration Extension", RFC 5203, April 2008.

[RFC5204] Laganier, J. and L. Eggert, "Host Identity Protocol (HIP) Rendezvous Extension", RFC 5204, April 2008.

[RFC5206] Nikander, P., Henderson, T., Vogt, C., and J. Arkko, "End-Host Mobility and Multihoming with the Host Identity Protocol", RFC 5206, April 2008.

[RFC5389] Rosenberg, J., Mahy, R., Matthews, P., and D. Wing, "Session Traversal Utilities for NAT (STUN)", RFC 5389, October 2008.

10.2. Informative References

[I-D.heer-hip-middle-auth]
Heer, T., Wehrle, K., and M. Komu, "End-Host Authentication for HIP Middleboxes", draft-heer-hip-middle-auth-01 (work in progress),

July 2008.

[I-D.rosenberg-mmusic-ice-nonsip]
Rosenberg, J., "Guidelines for Usage of Interactive Connectivity Establishment (ICE) by non Session Initiation Protocol (SIP) Protocols", draft-rosenberg-mmusic-ice-nonsip-01 (work in progress), July 2008.

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

[RFC4787] Audet, F. and C. Jennings, "Network Address Translation

(NAT) Behavioral Requirements for Unicast UDP", BCP 127, RFC 4787, January 2007.

[RFC5128] Srisuresh, P., Ford, B., and D. Kegel, "State of Peer-to-Peer (P2P) Communication across Network Address Translators (NATs)", RFC 5128, March 2008.

[RFC5207] Stiemerling, M., Quittek, J., and L. Eggert, "NAT and Firewall Traversal Issues of Host Identity Protocol (HIP) Communication", RFC 5207, April 2008.

Appendix A. Selecting a Value for Check Pacing

Selecting a suitable value for the connectivity check transaction pacing is essential for the performance of connectivity check-based NAT traversal. The value should not be too small so that the checks do not cause congestion in the network or overwhelm the NATs. On the other hand, too high pacing value makes the checks last for a long time and thus increase the connection setup delay.

The Ta value may be configured by the user in environments where the network characteristics are known beforehand. However, if the characteristics are not know, it is recommended that the value is adjusted dynamically. In this case it's recommended that the hosts estimate the RTT between them and set the minimum Ta value so that only two connectivity check messages are sent on every RTT.

One way to estimate the RTT is to use the time it takes for the HIP relay server registration exchange to complete; this would give an estimate on the registering host's access link's RTT. Also the I1/R1 exchange could be used for estimating the RTT, but since the R1 can be cached in the network, or the relaying service can increase the delay notably, it is not recommended.

Appendix B. IPv4-IPv6 Interoperability

Currently relay client and server do not have to run any ICE connectivity tests as described in Section 4.8. However, it could be useful for IPv4-IPv6 interoperability when the HIP relay server actually includes both the NAT traversal mode parameter and multiple locators in R2. The interoperability benefit is that the relay could support IPv4-based Initiators and IPv6-based Responders by converting the network headers and recalculating UDP checksums.

Such an approach is underspecified in this document currently. It is not yet recommended because it may consume resources at the relay and requires also similar conversion support at the TURN relay for data packets.

Appendix C. Base Exchange through a Rendezvous Server

When the Initiator looks up the information of the Responder from DNS, it's possible that it discovers an RVS record [RFC5204]. In this case, if the Initiator uses NAT traversal methods described in this document, it uses its own HIP relay server to forward HIP traffic to the Rendezvous server. The Initiator will send the I1 message using its HIP relay server which will then forward it to the RVS server of the Responder. In this case, the value of the protocol field in the RELAY_TO parameter MUST be IP since RVS does not support UDP encapsulated base exchange packets. The Responder will send the R1 packet directly to the Initiator's HIP relay server and the following I2 and R2 packets are also sent directly using the relay.

In case the Initiator is not able to distinguish which records are RVS address records and which are Responder's address records (e.g., if the DNS server did not support HIP extensions), the Initiator SHOULD first try to contact the Responder directly, without using a HIP relay server. If none of the addresses is reachable, it MAY try out them using its own HIP relay server as described above.

Appendix D. Document Revision History

To be removed upon publication

Revision	Comments
draft-ietf-nat-traversal-00	Initial version.
draft-ietf-nat-traversal-01	Draft based on RVS.
draft-ietf-nat-traversal-02	Draft based on Relay proxies and ICE concepts.
draft-ietf-nat-traversal-03	Draft based on STUN/ICE formats.
draft-ietf-nat-traversal-04	Issues 25-27,29-36
draft-ietf-nat-traversal-05	Issues 28,40-43,47,49,51

Authors' Addresses

Miika Komu
Helsinki Institute for Information Technology
Metsanneidonkuja 4
Espoo
Finland

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

Thomas Henderson
The Boeing Company
P.O. Box 3707
Seattle, WA
USA

Email: thomas.r.henderson@boeing.com

Philip Matthews
(Unaffiliated)

Email: philip_matthews@magma.ca

Hannes Tschofenig
Nokia Siemens Networks
Linnoitustie 6
Espoo 02600
Finland

Phone: +358 (50) 4871445
Email: Hannes.Tschofenig@gmx.net
URI: http://www.tschofenig.com

Ari Keranen (editor)
Ericsson Research Nomadiclab
Hirsalantie 11
02420 Jorvas
Finland

Phone: +358 9 2991
Email: ari.keranen@ericsson.com

Full Copyright Statement

Intellectual Property