HIP Working Group   M. Komu, Ed.
Internet-Draft   HIIT
Intended status: Standards Track   S. Schuetz
Expires: September 6, 2007   M. Stiemerling
                                                                  NEC
                                                            L. Eggert
                                                                Nokia
                                                             A. Pathak
                                                           IIT Kanpur
                                                         March 5, 2007

        HIP Extensions for the Traversal of Network Address Translators
                    draft-ietf-hip-nat-traversal-01

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 which MAY be behind NATs to contact HIP
   responders which MAY be behind another NAT.

Table of Contents

1. Introduction

   The Host Identity Protocol (HIP) describes a new communication
   mechanism for Internet hosts [RFC4423]. It introduces a new
   namespace and protocol layer between the network and transport layers
   that decouples the identifier and locator roles to support e.g.
   mobility and multihoming in the 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]. This document specifies how HIP can traverse
   through legacy NAT middleboxes that are not aware of HIP or ESP. The
   mechanisms defined in this document do not assume that the NAT
   middleboxes are reconfigured, as long as they allow UDP traffic.

   The use of HIP in NAT traversal has also some additional benefits
   provided by the new namespace. First, it is possible to address
   hosts behind a single NAT middlebox in a relatively simple way. The
   NAT middlebox translates the locators, but the Host Identifiers and
   ESP SPIs remain the same. Second, multiple services can share the
   same transport layer port number behind a single NAT. There is no
   multiplexing issue as long as these services have different Host
   Identifiers.

   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 located at a
   publicly reachable address in both cases. In the third case, both
   parties are located behind (different) NATs. This document describes

extensions for the first case in Section 3.3, for the second case in
Section 3.4 and in Section 3.5 for the third case.

The mechanisms described here also cover use of rendezvous server
from NATted environments. The rendezvous server MUST be used when
the responder is behind a NAT because otherwise successful NAT
traversal cannot be guaranteed. The rendezvous server MUST be
located in a publicly addressable location. Cascading of multiple
NAT enabled rendezvous servers is not possible, although there may be
other kind of rendezvous servers on the path. The NAT middleboxes
MUST support address independent mapping in the case where both hosts
are behind NAT devices. Otherwise, some other external relaying

mechanism MUST be used. Endpoint independent filtering is not
required in any of the cases. The NAT categories are defined in
[I-D.srisuresh-behave-p2p-state].

The mechanisms described in 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 for this document. The responder listens at a fixed
UDP port number for incoming HIP control packets. The port number
can be manually configured to the NAT to allow passing incoming
traffic directly to the host behind the NAT (port forwarding). The
benefit of such a configuration is that it does not require any
rendezvous server for the host behind the NAT. Although this
document does not prevent such configurations, it is out of scope
because of two drawbacks. First, it allows only a single responder
behind the NAT box. Second, manual configuration through several NAT
devices may be difficult or administratively prohibited.

The mobility and multihoming mechanisms of HIP [I-D.ietf-hip-mm],
allow HIP hosts to change network location during the lifetime of a
HIP 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 move to publicly addressable
locations.

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. Detecting NATs

In order to know whether to use the NAT traversal mechanisms, HIP
hosts need to detect the presence and type of NAT middleboxes along
the path to their peer hosts. This document does not describe any
new NAT detection mechanism but rather assumes that the NAT is
detected using some external mechanism. Hence, no special HIP
parameters are required in HIP control messages to detect NATs. The
NAT detection MUST occur prior to a base exchange, after node
movement and prior to sending UPDATE messages.

For example, STUN [RFC3489] offers a generic mechanism for detecting
both the presence and type of a NAT. In STUN, the host contacts a
STUN server that is always located at a publicly reachable address.
The STUN server replies back and provides information on the NAT
presence and type.

A limitation of STUN is that it cannot detect whether the responder
is behind the same NAT as the initiator. This can lead to an
unoptimal route through the public address of the NAT, especially in
combination the rendezvous extensions that are described later in
this document. In the worst case, the NAT may not be able to forward
the traffic unless it supports "hairpin translation" as described in
[I-D.srisuresh-behave-p2p-state].

To guarantee connectivity behind the same NAT, the initiator MUST
detect the hairpin support of the NAT as described in
[I-D.ietf-behave-nat-behavior-discovery]. If the NAT supports
hairpinning, the initiator uses the UDP encapsulation procedures
described in the following sections. If the NAT does not support
hairpinning, the initiator SHOULD broadcast a single I1 packet
without UDP encapsulation to the local network. The responder MUST
process the I1 according to [I-D.ietf-hip-base]. However, the
initiator MUST continue with the UDP encapsulation mechanisms
described in the following sections because the responder may
actually be located in a different network.

HIP-aware NATs are not in the scope of this document. In the future,
it may be possible to use some other protocol that is launched in

parallel with e.g. STUN to detect the presence of HIP aware NATs.
When the path between the initiator and responder consists of HIP
aware NATs, the extensions defined in this document SHOULD NOT be
used.

3. HIP Across NATs

The HIP base exchange as defined in [I-D.ietf-hip-base] works well in
public networks. However, this does not work with some legacy NATs
that are not able to multiplex HIP or ESP traffic. As a result, such
NATs just drop HIP control traffic and/or ESP data traffic. As a
solution for this, we propose UDP encapsulation of control and data
traffic using a specific scheme described in this document. The
scheme also allows hosts behind NATs to act as servers.

[RFC3948] describes UDP encapsulation of transport and tunnel mode
ESP packets. This document describes a similar mechanism for BEET
mode ESP packets [I-D.nikander-esp-beet-mode].

3.1. 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-alive.

3.1.1. Control Traffic

```
     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           |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                                                               |
    ~                   HIP Header and Parameters                   ~
    |                                                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
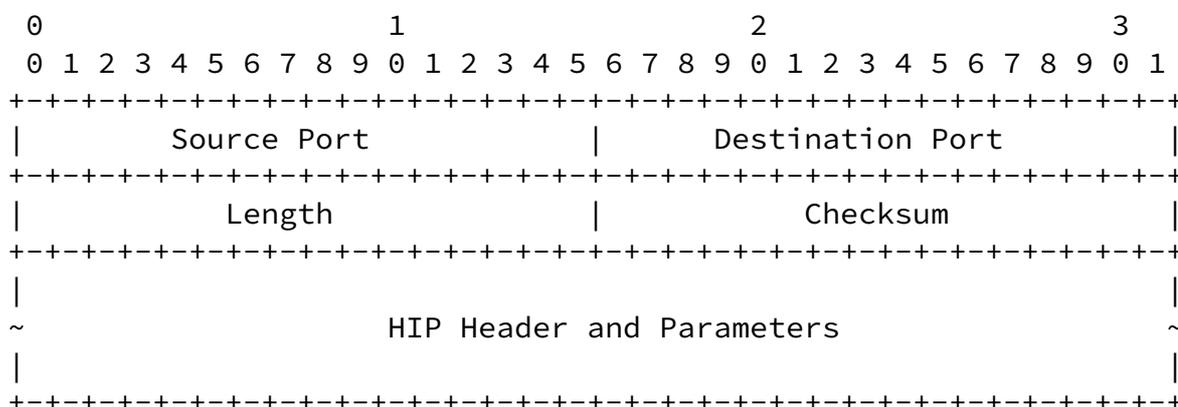
        Figure 1: Format for UDP-encapsulated HIP control traffic.

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]. The HIP header and parameter follow the conventions
[I-D.ietf-hip-base] with the exception that the HIP header checksum
MUST be zero. The HIP headers checksum is zero for two reasons.
First, the UDP header contains already a checksum. Second, the
checksum definition in [I-D.ietf-hip-base] includes the IP addresses
in the checksum calculation which is not applicable on HIP unaware
NAT devices.

3.1.2. Control Channel Keep-Alives

   The keep-alive for control channel are basically UDP encapsulated
   NOTIFY packets [I-D.ietf-hip-base]. The NOTIFY packets MAY contain
   HIP parameters. The NAT traversal mechanisms encapsulate these
   NOTIFY packets within the payload of UDP packets.

3.1.3. Data Traffic

```
     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            |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                                                               |
```

```
  ~                            ESP Header                             ~
  |                                                                   |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

        Figure 2: Format for UDP-encapsulated IPsec ESP BEET-mode traffic.

   Figure 2 shows how IPsec ESP BEET-mode packets are encapsulated
   within UDP. Again, a minimal UDP packet carries the ESP packet in
   its payload. The contents of the UDP source and destination ports
   are described in later sections. The UDP length and checksum field
   MUST be computed as described in [RFC0768].

3.1.4. FROM_NAT Parameter

```
        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            |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |                                                               |
       |                                                               |
       |                            Address                            |
       |                                                               |
       |                                                               |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
       |            UDP Port            |            Padding            |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
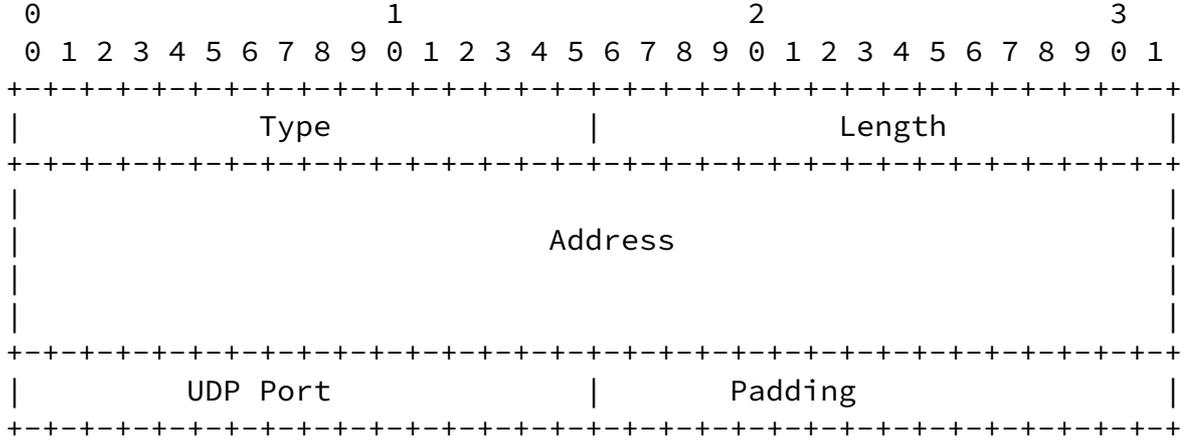
       Type   [TBD by IANA (63998 = $2^{16} - 2^{11} + 2^9 - 2$)]
       Length   18
       Address   An IPv6 address or an IPv4 address in IPv4-in-IPv6
                   format.
       UDP Port   A UDP port number

                   Figure 3: Format for the FROM_NAT Parameter

   Figure 3 shows FROM_NAT parameter. The use of this parameter is
   described in the following sections.

3.1.5. VIA_RVS_NAT Parameter

```
    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            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                                                               |
   |                                                               |
   |                           Address                             |
   |                                                               |
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |            UDP Port           |            Padding            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

   Type         [ TBD by IANA (64002 = 2^16 - 2^11 + 2^9 + 2) ]
   Length       16
   Address      An IPv6 address or an IPv4-in-IPv6 format IPv4 address
   UDP Port     A UDP port
```
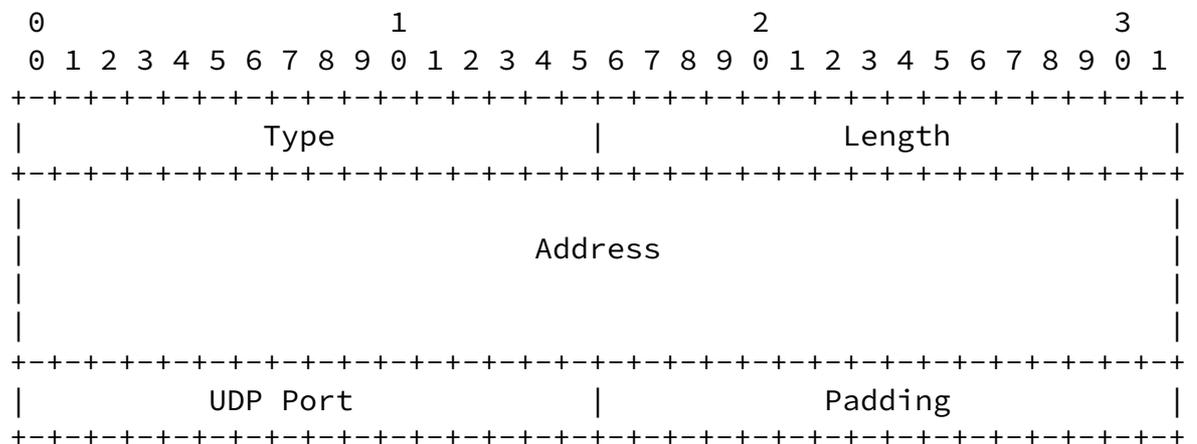
              Figure 4: Format for the VIA_RVS_NAT Parameter

   Figure 4 shows VIA_RVS_NAT parameter. The parameter is used for
   diagnostic purposes, similarly as VIA_RVS parameter in
   [I-D.ietf-hip-rvs]. The exact use of this parameter is described in
   later sections.

3.2. UDP Encapsulation/Decapsulation of IPsec BEET-Mode ESP

   [RFC3948] describes UDP encapsulation of the IPsec ESP transport and
   tunnel mode. This section describes the UDP encapsulation of the
   BEET mode.

3.2.1. UDP Encapsulation of IPsec BEET-Mode ESP

   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]. When 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
   security parameter indices (SPIs) is defined in [I-D.ietf-hip-esp].
   Additional SA parameters, such as IP addresses and UDP ports, MUST be
   defined according to this section. Two SAs MUST be defined on each
   host for one HIP association; one for outgoing data and another one
   for incoming data.

The BEET mode provides limited tunnel mode semantics without the
regular tunnel mode overhead. [I-D.nikander-esp-beet-mode] In the
BEET mode, transport-layer checksums in the payload data are 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].

Next, the resulting BEET-mode packet is UDP encapsulated. For this
purpose, a UDP header MUST be inserted between the IP and ESP header.
The source and destination ports are filled in. The UDP checksum
MUST be calculated based on the outer addresses (locators) of the
IPsec 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 5 illustrates the BEET-mode UDP encapsulation procedure for a
TCP packet.

```
   ORIGINAL TCP PACKET:
      +------------------------------------------+
      | inner IPv6 hdr | ext hdrs   |     |      |
      |   with HITs    | if present | TCP | Data |
      +------------------------------------------+

   PACKET AFTER BEET-MODE ESP PROCESSING:
      +----------------------------------------------------------+
      | inner IPv6 hdr | ESP | dest |     |      |   ESP   | ESP |
      |   with HITs    | hdr | opts.| TCP | Data | Trailer | ICV |
      +----------------------------------------------------------+
                            |<------- encryption -------->|
                      |<----------- integrity ----------->|

   FINAL PACKET AFTER BEET_MODE IP HEADER PROCESSING:
      +-------------------------------------------------------------+
      | outer IPv4 | UDP | ESP | dest |     |      |   ESP   | ESP |
      |    hdr     | hdr | hdr | opts.| TCP | Data | Trailer | ICV |
      +-------------------------------------------------------------+
                               |<------- encryption ------->|
                         |<----------- integrity ----------->|
```

        Figure 5: UDP Encapsulation of an IPsec BEET-mode ESP packet
                       containing a TCP segment.

3.2.2. UDP Decapsulation of IPsec BEET-Mode ESP

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

   The NAT traversal methods described in this section are based on
   connection reversal and UDP hole punching similar to
   [I-D.ietf-behave-nat-udp]. However, the methods in this section are
   adapted for HIP purposes, especially with the rendezvous server in
   mind.

3.3. Initiator Behind NAT

   This section discusses mechanisms to reach a HIP responder located in
   publicly addressable network by a HIP initiator that is located
   behind a NAT. The section describes also the case where the
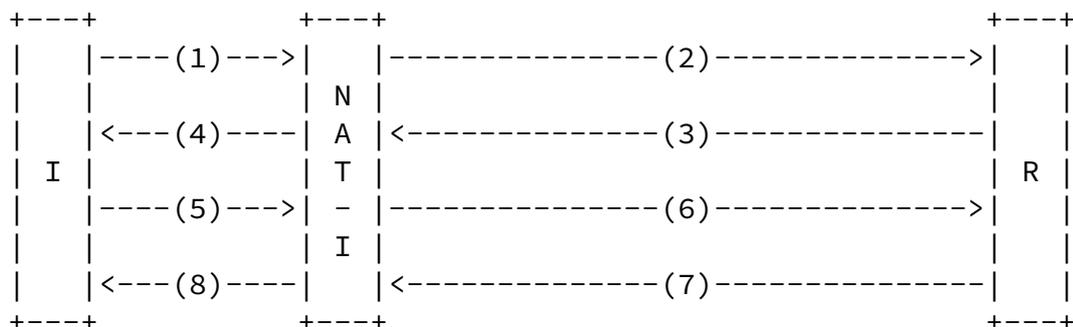   responder is using a rendezvous service.

   Table 1 lists some short-hand notations used in this section. For
   simplicity, the ports mangled by NAT are presented as example port
   numbers (11111, 22222, etc) instead of symbolic ones. In the
   examples, we assume that the NAT(s) timeout after the I1-R1 exchange
   over UDP because of e.g large RTT or high puzzle difficulty. In such
   a case, the NAT drops the related UDP port state and port numbers
   change for the I2-R2 exchange.

   +-----------------+---+
   | Notation | Explanation |
   +-----------------+---+
HIT-I	Initiator's HIT
HIT-R	Responder's HIT
IP-I	Initiator's IP address
IP-R	Responder's IP address
IP-RVS	IP address of the responder's rendezvous
	server

```
| IP-NAT-I         | Public IP of the NAT of the initiator      |
| IP-NAT-R         | Public IP of the NAT of the responder      |
| UDP(50500,11111) | UDP packet with source port 50500 and      |
|                  | destination port 11111                     |
| UDP(11111,22222) | Example port numbers mangled by a NAT      |
```

```
| UDP(44444,22222) | Port 44444 is used throughout the examples to |
|                  | denote the NAT mangled source port of I2 as   |
|                  | received by the rendezvous server during the  |
|                  | registration                                  |
+------------------+-----------------------------------------------+
```

                Table 1: Notations Used in This Section

3.3.1. NAT Traversal of HIP Control Traffic

   This section describes the details of enabling NAT traversal for HIP
   control traffic for the base exchange [I-D.ietf-hip-base] through UDP
   encapsulation for the case when the initiator of the association is
   located behind a NAT and the responder is located in a publicly
   addressable network. UDP-encapsulated HIP control traffic MUST use
   the packet formats described in Section 3.1. When sending UDP-
   encapsulated HIP control traffic, a HIP implementation MUST zero the
   HIP header checksum before calculating the UDP checksum. The
   receiver MUST only verify the correctness of the UDP checksum and
   MUST NOT verify the checksum of the HIP header.

   The initiator of a UDP-encapsulated HIP base exchange MUST use the
   UDP destination port 50500 for all control packets it sends. It is
   RECOMMENDED to use 50500 as the source port as well, but an
   implementation MAY use a (randomly selected) unoccupied source port.
   If it uses a random source port, it MUST listen for and accept
   arriving HIP control/ESP Data packets on this port until the
   corresponding HIP association is torn down. The random source port
   is RECOMMENDED to be in the range of the dynamic and private ports
   (49152-65535). Using a random source port, instead of a fixed one,
   enables to have multiple clients behind a NAT middlebox that supports
   only address translation but no port translation. This is referred

to as port overloading in [I-D.ietf-behave-nat-udp].

The responder of a UDP-encapsulated HIP base exchange MUST use 50500
as the source port for all UDP-encapsulated control packets it sends.
The source address for all the packets that the responder sends MUST
be the same as the IP address on which responder receives packets
from initiator. The responder MUST respond to any arriving UDP-
encapsulated control message using UDP encapsulation as well. Hosts
MUST process UDP-encapsulated base exchange messages equivalently to
non-encapsulated messages, i.e., according to [I-D.ietf-hip-base].

The remainder of this section clarifies this process through an
example which is illustrated in Figure 6. It shows an initiator with
the private address IP-I behind a NAT. The NAT has the public IP
address as NAT. The responder is in a publicly addressable location
IP-R.

```
     +---+                +---+                                  +---+
     |   |----(1)--->|   |--------------(2)------------->|   |
     |   |           | N |                              |   |
     |   |<---(4)----| A |<-------------(3)--------------|   |
     | I |           | T |                              | R |
     |   |----(5)--->| - |--------------(6)------------->|   |
     |   |           | I |                              |   |
     |   |<---(8)----|   |<-------------(7)--------------|   |
     +---+                +---+                                  +---+

     1. IP(IP-I, IP-R)      UDP(50500, 50500)  I1(HIT-I, HIT-R)
     2. IP(IP-NAT-I, IP-R)  UDP(11111, 50500)  I1(HIT-I, HIT-R)
     3. IP(IP-R, IP-NAT-I)  UDP(50500, 11111)  R1(HIT-R, HIT-I)
     4. IP(IP-R, IP-I)      UDP(50500, 50500)  R1(HIT-R, HIT-I)
     5. IP(IP-I, IP-R)      UDP(50500, 50500)  I2(HIT-I, HIT-R)
     6. IP(IP-NAT-I, IP-R)  UDP(22222, 50500)  I2(HIT-I, HIT-R)
     7. IP(IP-R, IP-NAT-I)  UDP(50500, 22222)  R2(HIT-R, HIT-I)
     8. IP(IP-R, IP-I)      UDP(50500, 50500)  R2(HIT-R, HIT-I)
```

Figure 6: Example of a UDP-encapsulated HIP base exchange (initiator
      behind a NAT, responder in a publicly addressable location).

Before beginning the base exchange, the initiator detects that it is
behind a NAT through some external mechanism, e.g. STUN. The
initiator starts 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 50500. It is addressed to IP-R on port 50500. The NAT in
Figure 6 forwards the I1 but substitutes the source address IP-I with
its own public address IP-NAT-I and the source UDP port 50500 with
11111.

When the responder in Figure 6 receives the UDP-encapsulated I1
packet on UDP port 50500, it decapsulates the packet and processes
the decapsulated packet according to [I-D.ietf-hip-base]. The
responder replies back with a UDP-encapsulated R1 using the addresses
and port information of I1. Thus, the R1 packet is destined to the
source IP address and UDP port of the I1, i.e., IP address IP-NAT-I
and port 11111. The NAT receives the I1 and substitutes the
destination of this packet with the initiator address (IP-I) and port
information (50500).

The initiator receives a UDP-encapsulated R1 packet from the
responder, decapsulates and 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 from IP-I port

50500 to IP-R port 50500. The NAT again substitutes the source
information. For illustration purposes, the NAT state times out and
it chooses a different source port (22222) for the I2 than for the I1
(11111).

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 the message, 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 sends
the R2 packet from IP-R port 50500 to IP-NAT-I port 22222. The NAT
again replaces the destination information in the R2 with IP-I port
50500

Usually, the I1-R1 and I2-R2 exchanges occur fast enough for the NAT
state to persist. This means that the NAT uses the same port for the

I1-R1 exchange to translate as the I2-R2 exchange. However, the host
MUST handle even 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 illustrated in Figure 6.

3.3.2. NAT Traversal of HIP Data Traffic

   This section describes the details of enabling NAT traversal of HIP
   data traffic. As described in Section 3, HIP data traffic is carried
   in UDP-encapsulated IPsec BEET-mode ESP packets.

3.3.2.1. IPsec BEET-Mode Security Associations

   The initiator MUST use UDP destination port 50500 for all UDP-
   encapsulated ESP packets it sends. It MAY also use port 50500 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.

   The responder of a UDP-encapsulated IPsec BEET-mode ESP exchange MUST
   use 50500 as the source port for all UDP-encapsulated ESP packets it
   sends. The destination port is the port from which the responder is
   receiving UDP encapsulated ESP data from the initiator.

   Both the initiator and the responder of a HIP association MUST define
   BEET mode with UDP encapsulation as the IPsec mode for the SA after a
   successful base exchange. The inner source address MUST be the local
   HIT used during base exchange and the inner destination address MUST
   be the HIT of the peer. The other parts of the SA are described in

Komu, et al.   Expires September 6, 2007   [Page 13]

   individual sections.

3.3.2.1.1. Security Associations at the Initiator

   The initiator of a UDP-encapsulated base exchange defines its
   outbound SA as shown in Table 2

   +--------------+---+
   | Field | Value |
   +--------------+---+
   | Outer src | The local IP address from which the base exchange |

```
| address       | packets were transmitted                          |
| Outer dst     | The peer IP address to which base exchange packets|
| address       | were transmitted                                  |
| UDP src port  | The port number as chosen for I2 packet in base   |
|               | exchange                                          |
| UDP dst port  | Port 50500                                        |
+---------------+---------------------------------------------------+
```

                    Table 2: Outbound SA at initiator

   The initiator of a UDP-encapsulated base exchange defines its inbound
   SA as shown in Table 3

```
+---------------+---------------------------------------------------+
| Field         | Value                                             |
+---------------+---------------------------------------------------+
| Outer src     | The peer IP address to which base exchange packets|
| address       | were transmitted                                  |
| Outer dst     | The local IP address from which the base exchange |
| address       | packets were transmitted                          |
| UDP src port  | Port 50500                                        |
| UDP dst port  | Initiator MUST use the UDP source port it uses in |
|               | the outbound SA here                              |
+---------------+---------------------------------------------------+
```

                    Table 3: Inbound SA at initiator

3.3.2.1.2. Security Associations at the Responder

   The responder of a UDP-encapsulated base exchange defines its
   outbound SA shown in Table 4.

```
+---------------+---------------------------------------------------+
| Field         | Value                                             |
+---------------+---------------------------------------------------+
| Outer src     | The local IP address from which the base exchange |
```

```
| address      | packets were transmitted                           |
| Outer dst    | Peer IP address of the I2 packet received during   |
| address      | the base exchange                                  |
| UDP src      | Port 50500                                         |
| port         |                                                    |
| UDP dst      | Source UDP port of the I2 packet received from the |
| port         | initiator during base exchange                     |
+--------------+----------------------------------------------------+
```

                    Table 4: Outbound SA at Responder

   Similarly, the responder of a UDP-encapsulated base exchange defines
   its inbound SA as shown in Table 5

```
+--------------+----------------------------------------------------+
| Field        | Value                                              |
+--------------+----------------------------------------------------+
| Outer src    | Source IP address of the I2 packet received from   |
| address      | the initiator during base exchange                 |
| Outer dst    | The local IP address from which the base exchange  |
| address      | packets were transmitted                           |
| UDP src      | Source UDP port of the I2 packet received from the |
| port         | initiator during base exchange                     |
| UDP dst      | Port 50500                                         |
| port         |                                                    |
+--------------+----------------------------------------------------+
```

                    Table 5: Inbound SA at responder

3.3.3. Use of the Rendezvous Service when only the Initiator is Behind
        NAT

   The rendezvous extensions for HIP without NAT traversal have been
   defined in [I-D.ietf-hip-rvs]. This section addresses only the
   scenario where a NATted HIP node uses the rendezvous service to
   contact another HIP node in a publicly addressable network. Figure 7
   illustrates the mechanism described in this section.

   A rendezvous server MUST listen on UDP port 50500 for incoming UDP
   encapsulated I1 packets. However, in this specific case with only
   the initiator behind NAT, the rendezvous server MUST NOT relay the I1
   packets. Instead, the rendezvous server replies to the initiator
   with a NOTIFY message that includes the responder's locator in a
   VIA_RVS parameter. The rendezvous server can differentiate this

scenario from the others because the I1 arrives UDP encapsulated, but the responder has registered without UDP encapsulation.

Upon receiving the NOTIFY with the locators of the responder through the NAT, the initiator MUST send an I1 to the responder. However, it MUST continue retransmissions using the RVS location. This is mandatory because NOTIFY messages are not protected with signatures and can be forged by a rogue host.

When the initiator receives an R1 through the NAT, the responder verifies the integrity of the packet and replies with an I2. The responder should be aware that the I2 may arrive from a different port than the I1. In such a case, the responder should send the R2 to the source port of I2.
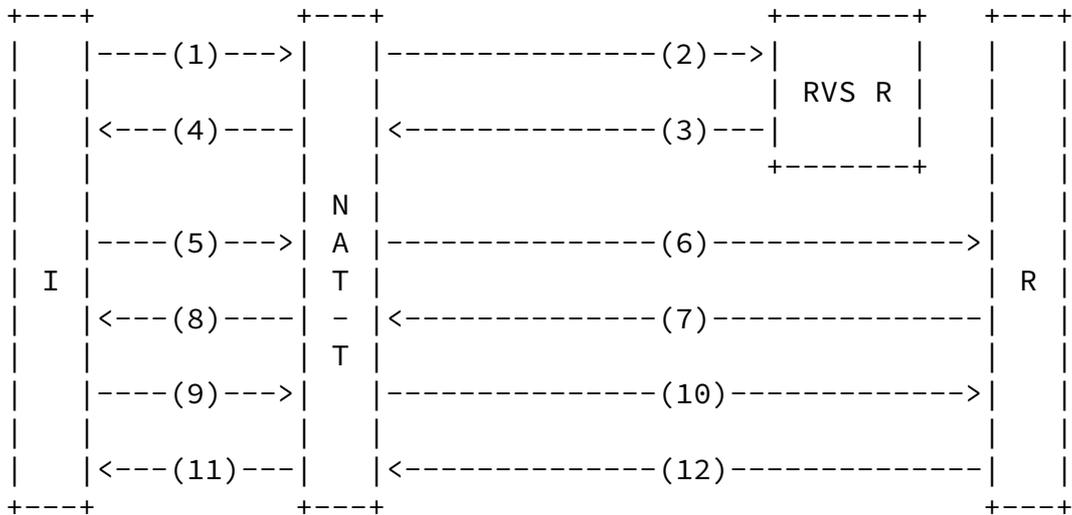
```
   +---+              +---+                     +-------+   +---+
   |   |----(1)--->|   |---------------(2)-->|       |   |   |
   |   |           |   |                    | RVS R |   |   |
   |   |<---(4)----|   |   |<-------------(3)---|       |   |   |
   |   |           |   |   |                    +-------+   |   |
   |   |           |   | N |                                |   |
   |   |----(5)--->| A |--------------(6)-------------->|   |
   | I |           | T |                                | R |
   |   |<---(8)----|   | - |<-------------(7)---------------|   |
   |   |           |   | T |                                |   |
   |   |----(9)--->|   |   |--------------(10)------------->|   |
   |   |           |   |   |                                |   |
   |   |<---(11)---|   |   |<-------------(12)-------------|   |
   +---+              +---+                                +---+
```

```
    1.  IP(IP-I, IP-RVS)      UDP(50500, 50500)   I1(HIT-I, HIT-R)
    2.  IP(IP-NAT-I, IP-RVS)  UDP(11111, 50500)   I1(HIT-I, HIT-R)
    3.  IP(IP-RVS, IP-NAT-I)  UDP(50500, 11111)
                                NOTIFY(HIT-R, HIT-I, VIA_RVS(IP-R))
    4.  IP(IP-RVS, IP-I)      UDP(50500, 50500)
                                NOTIFY(HIT-R, HIT-I, VIA_RVS(IP-R))
    5.  IP(IP-I, IP-R)        UDP(50500, 50500)   I1(HIT-I, HIT-R)
    6.  IP(IP-NAT-I, IP-R)    UDP(22222, 50500)   I1(HIT-I, HIT-R)
    7.  IP(IP-R, IP-NAT-I)    UDP(50500, 22222)   R1(HIT-R, HIT-I)
    8.  IP(IP-R, IP-I)        UDP(50500, 50500)   R1(HIT-R, HIT-I)
    9.  IP(IP-I, IP-R)        UDP(50500, 50500)   I2(HIT-I, HIT-R)
   10.  IP(IP-NAT-I, IP-R)    UDP(33333, 50500)   I2(HIT-I, HIT-R)
   11.  IP(IP-R, IP-NAT-I)    UDP(50500, 33333)   R2(HIT-R, HIT-I)
   12.  IP(IP-R, IP-I)        UDP(50500, 50500)   R2(HIT-R, HIT-I)
```

   Figure 7: Example of a UDP-encapsulated HIP base exchange via RVS
   (initiator behind a NAT, responder and RVS on the public Internet).
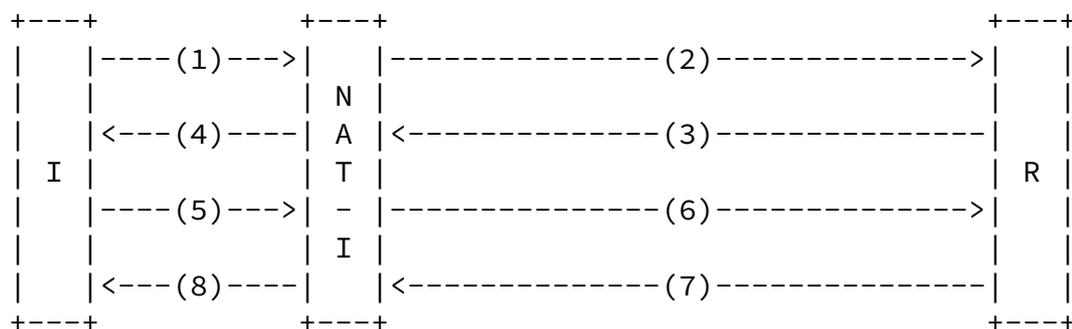
3.4. Responder Behind NAT

   This section discusses mechanisms to reach a HIP responder that is
   located behind a NAT. This section assumes that the initiator is
   located on publicly addressable network. The initiator contacts the
   responder through an RVS server.

3.4.1. Rendezvous Client Registration From Behind NAT

   The rendezvous client registration [I-D.ietf-hip-rvs] describes the
   case when rendezvous client is present in publicly addressable
   network. This section defines an extension to the rendezvous client
   registration for the case when the rendezvous client has detected
   that it is behind a NAT. The process in the NAT case is identical to
   the case without NAT, except that UDP encapsulation is used. The
   registration is illustrated in Figure 8.

   A node behind a NAT MUST first register to the RVS when it is going
   to act as a responder for some other nodes. The node (i.e.
   rendezvous client) performs a base exchange with the RVS over UDP as
   described in Section 3.3 by sending I1 UDP encapsulated and 50500 as
   destination port number. RVS sends REG_INFO parameter in R1 to which
   rendezvous client replies with REG_REQ paramter in I2. Both I1 and
   R1 are sent using UDP-encapsulation. If RVS grants service to the
   rendezvous client, it MUST store the source IP address and source
   port number of the I2 UDP packet that it had received from the
   rendezvous client during base exchange. The source IP address
   belongs to the NAT and the source port number is the NAT mangled
   port. RVS then replies with REG_RESP in R2 over UDP. If the
   registration process results in a successful REG_RESP, the rendezvous
   client MUST send NAT keepalives (Section 3.1.2) to keep the mapping
   in the NAT with the RVS open. The NAT keepalives sent from
   rendezvous client to the RVS MUST have the same source port as the I2
   packet.

   When the RVS receives an I1 packet from a HIP node to be relayed to
   the successfully registered rendezvous client behind NAT, RVS MUST
   relay the I1 over UDP with the destination port as the one stored
   during registration. The RVS also zeroes the HIP header checksum of
   the I1. This process is explained in Section 3.4.2.

```
      +---+            +---+                          +---+
      |   |----(1)--->|   |-------------(2)------------->|   |
      |   |           | N |                          |   |
      |   |<---(4)----| A |<-------------(3)--------------|   |
      | I |           | T |                          | R |
      |   |----(5)--->| - |-------------(6)------------->|   |
      |   |           | I |                          |   |
      |   |<---(8)----|   |<-------------(7)--------------|   |
      +---+            +---+                          +---+
```

        Initiator = Rendezvous client, Responder = Rendezvous server

        1. IP(IP-I, IP-R)   UDP(50500, 50500) I1(HIT-I, HIT-R)
        2. IP(IP-NAT-I, IP-R) UDP(33333, 50500) I1(HIT-I, HIT-R)
        3. IP(IP-R, IP-NAT-I) UDP(50500, 33333)
              R1(HIT-R, HIT-I, REG_INFO)
        4. IP(IP-R, IP-I)   UDP(50500, 50500)
              R1(HIT-R, HIT-I, REG_INFO)
        5. IP(IP-I, IP-R)   UDP(50500, 50500)
              I2(HIT-I, HIT-R, REG_REQ)
        6. IP(IP-NAT-I, IP-R) UDP(44444, 50500)
              I2(HIT-I, HIT-R, REG_REQ)
        7. IP(IP-R, IP-NAT-I) UDP(50500, 44444)
              R2(HIT-R, HIT-I, REG_RES)
        8. IP(IP-R, IP-I)   UDP(50500, 50500)
              R2(HIT-R, HIT-I, REG_RES)

                Figure 8: Rendezvous NAT Client Registration

3.4.2. NAT Traversal of HIP Control Traffic

   This section describes the details of enabling NAT traversal for base
   exchange packets [I-D.ietf-hip-base] through UDP encapsulation, for

the case when the HIP initiator is on publicly addressable network
and the HIP responder is behind NAT. The process is illustrated in
Figure 9.

Before the HIP base exchange starts, the responder of the HIP base
exchange MUST have completed a successful rendezvous client
registration using the scheme defined in Section 3.4.1.

The initiator of the HIP base exchange sends a plain I1 packet
(without UDP encapsulation) to the RVS as described in
[I-D.ietf-hip-rvs]. In this case, the rendezvous server detects that
the I1 is not UDP encapsulated, but the rendezvous client has
registered using UDP encapsulation.

To relay the I1 packet, RVS MUST zero the HIP header checksum from

the I1 packet. RVS MUST add a FROM parameter, as described in
[I-D.ietf-hip-rvs], which contains the IP address of HIP initiator.
The FROM parameter is integrity protected by a RVS_HMAC parameter as
described in [I-D.ietf-hip-rvs]. RVS replaces the destination IP
address in the IP header of the packet with IP that it had stored
during the rendezvous client registration (which is the IP address of
the outermost NAT behind which rendezvous client is located). It
MUST then encapsulate the I1 packet within UDP. The source port in
the UDP header MUST be 50500 and the destination port MUST be the
same as the source port number (44444) of the I2 packet which it had
stored during the registration process. RVS then recomputes the IP
header checksum and sends the packet.

```
                         +-------+
                         |       |
                  +----->|  RVS  +-----+              +----+
 +---+            |      |       |     |              |    |              +---+
 |   |---(1)---+         +-------+     +----(2)--->|   |---(3)--->|   |
 |   |                                            | N |          |   |
 |   |<-----------------(5)-------------------|  A  |<--(4)----|   |
 | I |                                            | T |          | R |
 |   |-----------------(6)------------------->|  -  |---(7)--->|   |
 |   |                                            | R |          |   |
 |   |<-----------------(9)-------------------|   |<--(8)----|   |
 +---+                                            +----+          +---+
```
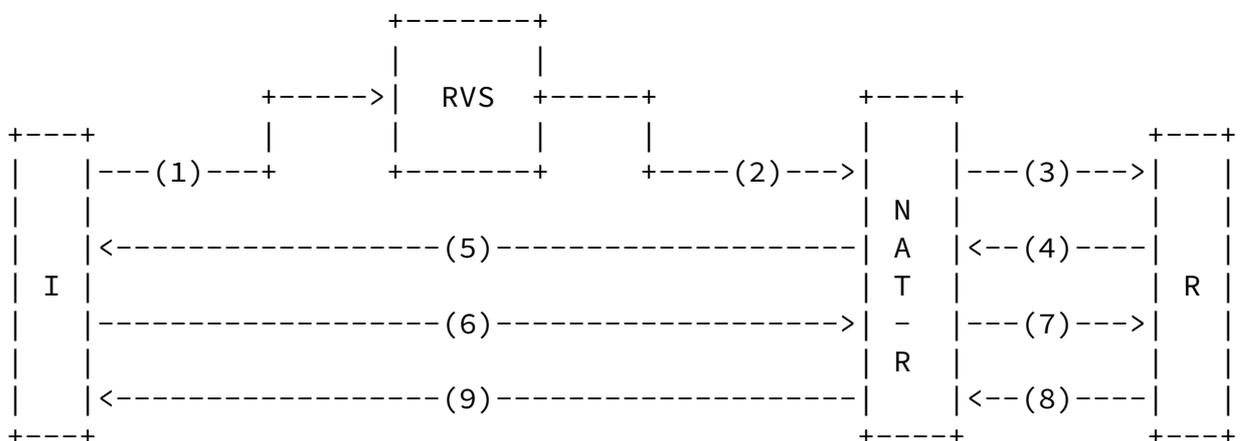
```
    1. IP(IP-I, IP-RVS)     I1(HIT-I, HIT-R)
    2. IP(IP-RVS, IP-NAT-R) UDP(50500, 44444)
         I1(HIT-I, HIT-R, FROM:IP-I, RVS_HMAC)
    3. IP(IP-RVS, IP-R)     UDP(50500, 50500)
         I1(HIT-I, HIT-R, FROM:IP-I, RVS_HMAC)
    4. IP(IP-R, IP-I)
         UDP(50500, 50500) R1(HIT-R, HIT-I, VIA_RVS_NAT(IP-FVS, 50500))
    5. IP(IP-NAT-R, IP-I)
         UDP(44444,   50500) R1(HIT-R, HIT-I, VIA_RVS_NAT(IP-FVS, 50500)
    6. IP(IP-I, IP-NAT-R)   UDP(50500, 44444)   I2(HIT-I, HIT-R)
    7. IP(IP-I, IP-R)       UDP(50500, 50500) I2(HIT-I, HIT-R)
    8. IP(IP-R, IP-I)       UDP(50500, 50500) R2(HIT-R, HIT-I)
    9. IP(IP-NAT-R, IP-I)   UDP(44444, 50500) R2(HIT-R, HIT-I)
```

       Figure 9: UDP-encapsulated HIP base exchange (initiator on public
                    Internet, responder behind a NAT).

   The relayed I1 packet travels from RVS to the NAT. The NAT changes
   the destination IP address of the UDP encapsulated I1 packet, and the
   destination port number in the UDP header. The responder accepts the
   packet from the RVS and processes it according to [I-D.ietf-hip-rvs].
   The resulting R1 must be encapsulated within UDP. The responder MAY

   append a VIA_RVS_NAT parameter to the message, which contains the IP
   address of the rendezvous and the port the rendezvous server used for
   relaying the I1. The RECOMMENDED source port is 50500 and the
   destination port number MUST be 50500. The destination address in
   the IP header MUST be the same as the one specified in the FROM
   parameter of the relayed I1 packet.

   The initiator MUST listen on port 50500 and it receives the UDP
   encapsulated R1. After verifying the HIP packet, it concludes that
   the responder is behind a NAT because the packet was UDP
   encapsulated. The initiator processes the R1 control packet
   according to [I-D.ietf-hip-base] and replies using I2 that is UDP
   encapsulated. The addresses and ports are derived from the received
   R1.

   The NAT translates and forwards the UDP encapsulated I2 packet to the
   responder. The resulting R2 packet is also UDP encapsulated using
   the address and port information from the received I2 packet.

3.4.3. NAT Traversal of HIP Data Traffic

   After a successful base exchange, both of the HIP nodes have
   communicated all the necessary information to establish UDP-
   encapsulated BEET mode Security Associations. The following section
   describes inbound and outbound security associations at initiator and
   responder.

3.4.3.1. Security Associations at the Initiator

   The initiator of a base exchange defines its outbound SA as shown in
   Table 6

   +-------------+---+
   | Field | Value |
   +-------------+---+
Outer src	The local IP address from which the base exchange
address	packets were transmitted
Outer dst	The peer IP address from which R2 packet was
address	received during base exchange
UDP src port	Port 50500
UDP dst port	Source port of incoming R2 packet during base
	exchange
   +-------------+---+

                   Table 6: Outbound SA at initiator

   The initiator of a base exchange defines its inbound SA as shown in
   Table 7

   +-------------+---+
   | Field | Value |
   +-------------+---+
Outer src	The peer IP address from which R2 packet was
address	received during base exchange
Outer dst	The local IP address from which the base exchange
address	packets were transmitted
UDP src port	Source port of incoming R2 packet during base
	exchange
UDP dst port	Port 50500
   +-------------+---+

Table 7: Inbound SA at initiator

3.4.3.2. Security Associations at the Responder

   The responder of a UDP-encapsulated base exchange defines its
   outbound SA shown in Table 8.

   +-------------+--+
   | Field | Value |
   +-------------+--+
Outer src	The local IP address from which the base exchange
address	packets were transmitted
Outer dst	The peer IP as that used during base exchange
address	
UDP src port	The as source port chosen during base exchange
UDP dst port	Port 50500
   +-------------+--+

                    Table 8: Outbound SA at Responder

   Similarly, the responder of a UDP-encapsulated base exchange defines
   its inbound SA as shown in Table 9

   +-------------+--+
   | Field | Value |
   +-------------+--+
Outer src	Source peer IP address as used in base exchange
address	
Outer dst	The local IP address from which the base exchange
address	packets were transmitted
UDP src port	Port 50500
UDP dst port	The as source port chosen during base exchange
   +-------------+--+

                    Table 9: Inbound SA at responder

3.5. Both Hosts Behind NAT

   This section describes the details of enabling NAT traversal for HIP
   control and ESP data traffic, such as the base exchange
   [I-D.ietf-hip-base], through UDP encapsulation, for the case when the

HIP initiator and the HIP responder are both behind two separate
NATs. The limitation of this approach is that the NAT middlebox MUST
support endpoint independent mapping
[I-D.srisuresh-behave-p2p-state].

The registration and rendezvous relay are handled similarly as
described in Section 3.3.3 and Section 3.4.1. Now that both hosts
are behind NATs, both the initiator (Section 3.3) and responder
(Section 3.4) mechanisms are combined here. There is one exception
though; the initiator does not retransmit an I1 but rather a NOTIFY
message.

3.5.1. NAT Traversal of HIP Control Traffic

Before an initiator can start the base exchange, the responder MUST
have completed a successful rendezvous client registration with its
RVS using the mechanism described in Section 3.4.1. The initiator of
the HIP base exchange starts the base exchange by sending a UDP
encapsulated I1 packet to RVS. The UDP packet MUST have destination
port number 50500 and the initiator is RECOMMENDED to use 50500 as
source port number. RVS MUST listen on UDP port 50500. RVS MUST
accept the packet as described in Section 3.3.3. As there has been a
successful rendezvous client registration between the responder and
the RVS as described in Section 3.4.1, the RVS knows the port number
to be used to communicate with the responder through the NAT. RVS
MUST add a FROM_NAT parameter to the I1 packet. The FROM_NAT
parameter contains the source address of the I1 packet, which is
effectively the address of the outermost NAT of the initiator. The
RVS copies the source port of the UDP encapsulated I1 packet into the
port number field of the FROM_NAT parameter. The FROM_NAT parameter
is integrity protected by an RVS_HMAC as described in
[I-D.ietf-hip-rvs]. It MUST replace the destination IP address of
the I1 packet by the one it had stored earlier during rendezvous
client registration. It MUST replace source IP address of I1 packet
with its own address. UDP source port of the relayed I1 packet MUST
be 50500 and destination port MUST be the same as one it had stored
during the client rendezvous registration. It MUST recompute the IP
header checksum.

Upon receiving the VIA_RVS_NAT parameter, the initiator sends NOTIFY
message without any contents to the responder, which responder MUST
ignore. This punches a hole to the NAT of the initiator.

The responder receives the I1 relayed by the RVS. The responder acts
as described in [Section 3.4.2](#) by replying with an R1. The R1 punches
a hole to the responder's NAT for the initiator. The R1 makes it to
the initiator because the initiator already punched a hole in its own
NAT with the empty NOTIFY message for the responder.

The initiator and responder complete the rest of the base exchange
with I2 and R2. The NAT state may timeout in case the R1 cookie was
relatively large or in case the RTT is large. For this reason, the
initiator MUST refresh the state of the NATs by resending empty
NOTIFY messages until it receives an R2.

```
   +---+       +----+        +-------+        +----+        +---+
   |   +--(1)-->|     +---(2)-->+       |      |    |        |   |   |
   |   |        |     |         | RVS-R |      |    |        |   |   |
   |   |        |     |<--(3a)--+    +---(3b)---->|  |        |   |   |
   |   |        |     |       +-------+          |    |        |   |   |
   |   |<-(4a)--+ N  |                          | N  +--(4b)->|   |
   |   |        | A  |                          | A  |        |   |
   | I +--(5a)->| T  |                          | T  |<-(5b)--+ R |
   |   |        | -  |<--(6b)----------------(6a)->|  -  |        |   |
   |   |<-(7b)--+ I  |                          | R  +--(7a)->|   |
   |   |        |    |                          |    |        |   |
   |   +--(8)-->|    +--------------(9)----------->|    +--(10)->|   |
   |   |        |    |                          |    |        |   |
   |   |<-(13)--+    |<------------(12)-----------+    |<-(11)--+   |
   +---+       +----+                           +----+        +---+
```
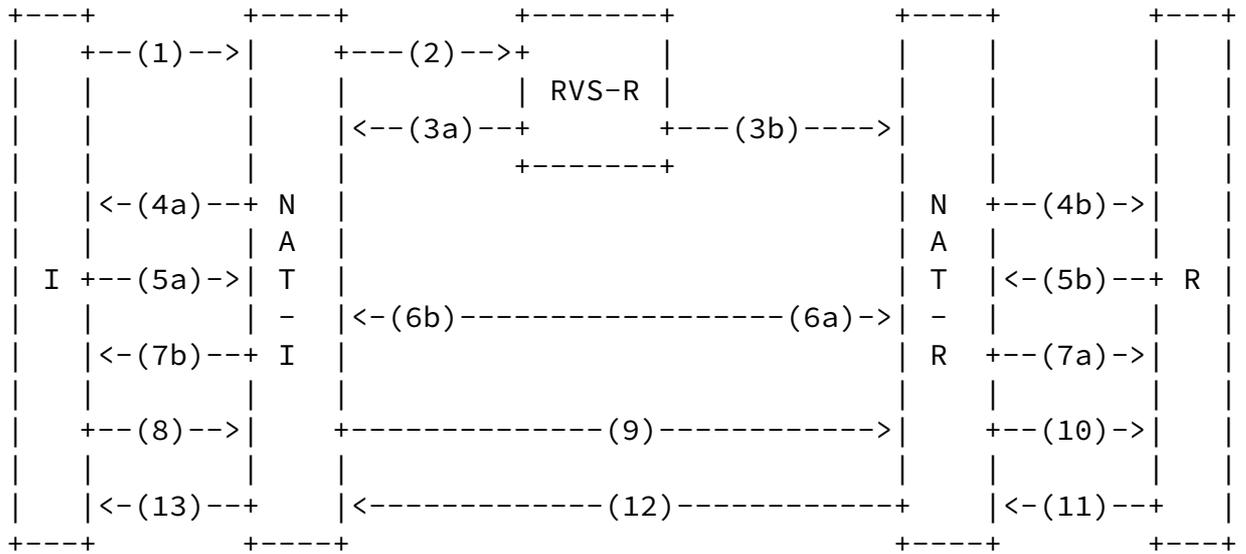
```
    1.   IP(IP-I, IP-RVS)        UDP(50500, 50500)  I1(HIT-I, HIT-R)
    2.   IP(IP-NAT-I, IP-RVS)    UDP(11111, 50500)  I1(HIT-I, HIT-R)

    3a.  IP(IP-RVS, IP-NAT-I)    UDP(50500, 11111)
           NOTIFY(HIT-R, HIT-I, VIA_RVS_NAT(IP-NAT-R, 44444)
    3b.  IP(IP-RVS, IP-NAT-R)    UDP(50500, 44444)
           I1(HIT-I, HIT-R, FROM_NAT:[IP-NAT-I,11111], RVS_HMAC)

    4a.  IP(IP-RVS-R, IP-I)      UDP(50500, 50500)
           NOTIFY(HIT-R, HIT-I, VIA_RVS_NAT(IP-NAT-R, 44444)
    4b.  IP(IP-RVS, IP-R)        UDP(50500, 50500)
           I1(HIT-I, HIT-R, FROM_NAT:[NAT-I,11111], RVS_HMAC)

    5a.  IP(IP-I, IP-NAT-R)      UDP(50500, 44444) NOTIFY(HIT-I, HIT-R)
    5b.  IP(IP-R, IP-NAT-I)      UDP(50500, 11111)
           R1(HIT-R, HIT-I, VIA_RVS_NAT(IP-FVS, 50500))

    6a.  IP(IP-NAT-I, IP-NAT-R) UDP(11111, 44444) NOTIFY(HIT-I, HIT-R)
    6b.  IP(IP-NAT-R, IP-NAT-I) UDP(44444, 11111)
           R1(HIT-R, HIT-I, VIA_RVS_NAT(IP-FVS, 50500))
    7a.  IP(IP-NAT-I, IP-NAT-R) UDP(11111, 50500) NOTIFY(HIT-I, HIT-R)
    7b.  IP(IP-NAT-R, IP-NAT-I) UDP(44444, 50500)
           R1(HIT-R, HIT-I, VIA_RVS_NAT(IP-FVS, 50500))
    8-10. I2(HIT-I, HIT-R), details similarly as in the cases before
    11-13 R2(HIT-R, HIT-I), details similarly as in the cases before
```

Figure 10: UDP-encapsulated HIP base exchange (initiator and
                responder behind a NAT, RVS on public IP).

    The UDP hole punching is applicable only in the case when the NAT
    devices on the path support address independent mapping
    [I-D.srisuresh-behave-p2p-state]. After the initiator has received a

    VIA_RVS_NAT parameter and has been in I1_SENT state for a policy
    specific period, the initiator MAY transition to E-FAILED state.
    Alternatively, it is RECOMMENED to switch to an external relay based
    protocol mechanism.

3.5.2. NAT Traversal of HIP Data Traffic

    After a successful base exchange, both the HIP nodes have all the
    parameters with them to establish UDP BEET mode Security Association.
    The following section describes inbound and outbound security
    associations at initiator and responder.

3.5.2.1. Security Associations at the Initiator

    The initiator of a base exchange defines its outbound SA as shown in
    Table 10

Field	Value
Outer src address	The local IP address from which the base exchange packets were transmitted
Outer dst address	The peer IP address from which R2 packet was received during base exchange
UDP src port	The as the port number chosen to send I2 during base exchange
UDP dst port	Source port of incoming R2 packet during base exchange

                    Table 10: Outbound SA at initiator

    The initiator of a base exchange defines its inbound SA as shown in
    Table 11

```
+-------------+--------------------------------------------------+
| Field       | Value                                            |
+-------------+--------------------------------------------------+
| Outer src   | The peer IP address from which R2 packet was     |
| address     | received during base exchange                    |
| Outer dst   | The local IP address from which the base exchange|
| address     | packets were transmitted                         |
| UDP src port| Source port of incoming R2 packet during base    |
|             | exchange                                         |
| UDP dst port| The as the port number chosen to send I2 during  |
|             | base exchange                                    |
+-------------+--------------------------------------------------+
```

Table 11: Inbound SA at initiator

3.5.2.2. Security Associations at the Responder

   The responder of a UDP-encapsulated base exchange defines its
   outbound SA shown in Table 12.

```
+-------------+--------------------------------------------------+
| Field       | Value                                            |
+-------------+--------------------------------------------------+
| Outer src   | The local IP address from which the base exchange|
| address     | packets were transmitted                         |
| Outer dst   | The peer IP as that used during base exchange    |
| address     |                                                  |
| UDP src port| The as source port chosen send R2 during base    |
|             | exchange                                         |
| UDP dst port| The as source port number of I2 packet during base|
|             | exchange                                         |
+-------------+--------------------------------------------------+
```

Table 12: Outbound SA at Responder

   Similarly, the responder of a UDP-encapsulated base exchange defines
   its inbound SA as shown in Table 13

```
+-------------+--------------------------------------------------+
| Field       | Value                                            |
+-------------+--------------------------------------------------+
```

```
| Outer src     | Source peer IP address as used in base exchange    |
| address       |                                                    |
| Outer dst     | The local IP address from which the base exchange  |
| address       | packets were transmitted                           |
| UDP src port  | The as source Port received from I2 during base    |
|               | exchange                                           |
| UDP dst port  | The as source port used to send R2 during base     |
|               | exchange                                           |
+---------------+----------------------------------------------------+
```

                   Table 13: Inbound SA at responder

3.6. 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.

   Typically, only outgoing traffic refreshes the NAT port state for
   security reasons. Consequently, both hosts SHOULD send periodic
   keep-alives for the UDP channel of all their established HIP
   associations if the channel has been idle for a specific period of
   time.

   For the UDP channel, keep-alives MUST be UDP-encapsulated HIP NOTIFY
   packets as defined in Section 3.1.2. 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
   SHOULD be 20 seconds. The peer host of the HIP association MUST
   discard the keep-alives.

3.7. HIP Mobility

   After a successful base exchange, a mobile node can change its
   network location using the mechanisms defined in [I-D.ietf-hip-mm].
   This section describes such mobility mechanisms in the presence of

NATs. However, the double jump scenario, where both peers move
simultaneously, is excluded.

The mobile node can change its location as described in Table 14.

```
+----+-------------------------+-------------------------------+
| No | From network            | To network                    |
+----+-------------------------+-------------------------------+
| 1  | Behind NAT              | Publicly Addressable Network  |
| 2  | Publicly Addressable    | Behind NAT                    |
|    | Network                 |                               |
| 3  | Behind NAT-A            | Stays behind NAT-A, but       |
|    |                         | different IP                  |
| 4  | Behind NAT-A            | Behind NAT-B                  |
| 5  | Publicly Addressable    | Publicly Addressable Network  |
|    | Network                 |                               |
+----+-------------------------+-------------------------------+
```

Table 14: End host mobility scenarios

The corresponding peer node can be located as follows Table 15

```
+----+-----------------------------------------+
| No | Peer Node network                       |
+----+-----------------------------------------+
| A  | Publicly Addressable Network With RVS    |
| B  | Publicly Addressable Network Without RVS |
| C  | Behind NAT With RVS                      |
| D  | Behind NAT Without RVS                   |
+----+-----------------------------------------+
```

Table 15: Peer host Network Scenarios

The NAT traversal mechanisms may not work when the corresponding node
is behind a NAT without RVS (case D), except when the mobile node
stays behind the same cone NAT (case 3D).

When a mobile node changes its location, it SHOULD detect the
presence of NATs along the new paths to its corresponding nodes using
some external mechanism before sending any UPDATE messages. If no
NAT was detected in such a case, it SHOULD send an UPDATE to its
corresponding nodes without UDP encapsulation.

The mobile node MUST send the UPDATE packet through the corresponding
node's RVS if it uses one, in addition to sending it to the
corresponding node directly. The mobile node encapsulates the UPDATE
packet within UDP only when it is behind a NAT. The corresponding
node MUST reply using UDP when the packet was encapsulated within
UDP, or without UDP when the UDP header was not present in the UPDATE
packet.

The rendezvous server relays the UPDATE similarly to I1. The
rendezvous server MUST add FROM parameter when it gets an UPDATE
packet without UDP encapsulation, or a FROM_NAT parameter when the
UPDATE packet it receives is UDP encapsulated and MUST in both cases
protect the packet with a HMAC parameter. Upon replying to the
UPDATE, the corresponding node MUST add a VIA_RVS (or VIA_RVS_NAT)
parameter to the reply.

The mobile node MUST leave out the NATted locators from the LOCATOR
parameter. This MUST be done before applying HMAC and SIGNATURE to
an R1, I2 or UPDATE packet. Thus, the LOCATOR parameter consists
only of the type and length fields when the mobile node has only
NATted addresses. When the mobile node has e.g. a single IPv6
address and one NATted address, the LOCATOR parameter consists of
single locator. The UDP header along with its port number conveys
the NATted locator to the peer.

3.8. 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 are not UDP encapsulated.

3.9. 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 3 depend on the ability to initiate
   communication via UDP to destination port 50500 from arbitrary source
   ports and to receive UDP response traffic from that port 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 3 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 3 can still be supported, if the firewall allows permanently
   inbound UDP traffic from port 50500 and destined to arbitrary source
   IP addresses and UDP ports.

   When the responder is behind a firewall, the NAT traversal mechanisms
   described in Section 3 depend on the ability to receive UDP traffic
   on port 50500 from arbitrary source IP addresses and ports.

   The NAT traversal mechanisms described in Section 3 require that the
   firewall – stateful or not – allow inbound UDP traffic to port 50500
   and allow outbound UDP traffic to arbitrary UDP ports. If necessary
   for firewall traversal, ports reserved for IKE MAY be used for
   initiating new connections, but the implementation MUST be able to
   listen for UDP packets from port 50500.

4. Security Considerations

4.1. A Difference to RFC3948

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 3, because the responder use the HITs to
distinguish between different communication instances.

4.2. Rendezvous and Responder Privacy

The rendezvous usage in this draft has been designed to follow the
RVS specification [I-D.ietf-hip-rvs] when the NAT supports end-point
independent filtering. However, as NAT networking presents some
additional challenges, it is not possible to follow the RVS design
exactly. Particularly, the mechanisms described in Figure 7 and
Section 3.5.1 require that the rendezvous server replies back to the
initiator with a message which includes the address and port of the
responder NAT. Another design choice would have been to relay also
the R1 (and I2 in case of both hosts behind NAT) through the
rendezvous server to delay the exposure of the responder NAT address
and port related information for additional DoS protection. However,
this choice was not selected to reduce round trip time. As a
consequence, the rendezvous client must accept the risk of lowered
privacy protection when it registers to the RVS over UDP as defined
in Figure 8.

5. 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" range, i.e., 50500. 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 50500 to the port IANA has registered.

6. Acknowledgements

The authors would like to thank Vivien Schmitt for his contributions
to previous versions of this draft. In addition, the authors would

7. References

7.1. Normative References

[I-D.ietf-hip-base]
           Moskowitz, R., "Host Identity Protocol",
           draft-ietf-hip-base-06 (work in progress), June 2006.

[I-D.ietf-hip-esp]
           Jokela, P., "Using ESP transport format with HIP",
           draft-ietf-hip-esp-04 (work in progress), October 2006.

[I-D.ietf-hip-mm]
           Nikander, P., "End-Host Mobility and Multihoming with the
           Host Identity Protocol", draft-ietf-hip-mm-04 (work in
           progress), June 2006.

[I-D.ietf-hip-rvs]
           Laganier, J. and L. Eggert, "Host Identity Protocol (HIP)

Rendezvous Extension", draft-ietf-hip-rvs-05 (work in
progress), June 2006.

[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-06
(work in progress), August 2006.

[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.

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

7.2. Informative References

[I-D.ietf-behave-nat-behavior-discovery]
            MacDonald, D. and B. Lowekamp, "NAT Behavior Discovery
            Using STUN", draft-ietf-behave-nat-behavior-discovery-00
            (work in progress), February 2007.

[I-D.ietf-behave-nat-udp]
            Audet, F. and C. Jennings, "NAT Behavioral Requirements
            for Unicast UDP", draft-ietf-behave-nat-udp-08 (work in
            progress), October 2006.

[I-D.irtf-hiprg-nat]
            Stiemerling, M., "NAT and Firewall Traversal Issues of
            Host Identity Protocol (HIP) Communication",
            draft-irtf-hiprg-nat-03 (work in progress), June 2006.

[I-D.nikander-hip-path]
            Nikander, P., "Preferred Alternatives for Tunnelling HIP
            (PATH)", draft-nikander-hip-path-01 (work in progress),

March 2006.

[I-D.srisuresh-behave-p2p-state]
           Srisuresh, P., "State of Peer-to-Peer(P2P) Communication
           Across Network Address Translators(NATs)",
           draft-srisuresh-behave-p2p-state-04 (work in progress),
           September 2006.

[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 |
   +-----------+--+
schmitt-00	Initial version.
ietf-00	Officially adopted as WG item. Solved issues
	1-9,11,12
   +-----------+--+

Authors' Addresses

   Miika Komu (editor)
   Helsinki Institute for Information Technology
   Tammasaarenkatu 3
   Helsinki

          Finland

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

          Simon Schuetz
          NEC Network Laboratories
          Kurfuerstenanlage 36
          Heidelberg 69115
          Germany

          Phone: +49 6221 4342 165
          Fax: +49 6221 4342 155
          Email: simon.schuetz@netlab.nec.de
          URI: http://www.netlab.nec.de/

          Martin Stiemerling
          NEC Network Laboratories
          Kurfuerstenanlage 36
          Heidelberg 69115
          Germany

          Phone: +49 6221 4342 113
          Fax: +49 6221 4342 155
          Email: stiemerling@netlab.nec.de
          URI: http://www.netlab.nec.de/

          Lars Eggert
          Nokia Research Center
          P.O. Box 407
          Nokia Group 00045
          Finland

          Phone: +358 50 48 24461
          Email: lars.eggert@nokia.com
          URI: http://research.nokia.com/people/lars_eggert/

     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/

Full Copyright Statement

Intellectual Property

Acknowledgment