HIP Extensions for the Traversal of Network Address Translators

draft-schmitt-hip-nat-00

Miika Komu, Vivien Schmitt, Abhinav Pathak, Lars Eggert and Martin Stiemerling
Table of Contents

- Motivation and goals
- Packet formats and UDP port numbers
- Base exchange, mobility and multihoming
- Firewall configuration
- Issue list
Motivation and Goals

• To create a very practical, implementable and deployable draft to support legacy NAT traversal
  – On-going implementation work at NEC and HIIT
• Primary goal: initiator behind NAT
• Secondary goal: responder behind NAT
• Non-goals: firewall + NAT combinations
• Support both base exchange and mobility extensions
• NAT detection using external protocols (no modifications to the base exchange or UPDATE)
  – Benefit: future compatibility with NSIS
  – Drawback: requires a third host and incurs some extra latency
HIP Control Channel Header Format

```
| 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 with zero checksum |
```
ESP Data Channel Formats: ESP-in-UDP and ESP Channel Keepalive [RFC3948]

```
+-----------+-----------+-----------+-----------+
|           |           |           |           |
| Source Port| Destination Port|
+-----------+-----------+-----------+-----------+
|           |           |           |           |
| Length    | Checksum  |
|           |           |           |           |
|           |           |           |           |
|           |           |           |           |
| ESP Header|           |           |           |
```

```
+-----------+-----------+-----------+-----------+
|           |           |           |           |
| Source Port| Destination Port|
+-----------+-----------+-----------+-----------+
|           |           |           |           |
| Length    | Checksum  |
|           |           |           |           |
|           |           |           |           |
| 0xFF      |           |
+-----------+-----------+-----------+-----------+
```
# Base Exchange over UDP

<table>
<thead>
<tr>
<th>Private Network</th>
<th>Public Internet</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP (I)</td>
<td>IP (NAT1)</td>
</tr>
<tr>
<td>IP (R) : P(50500)</td>
<td>IP (NAT2) : P(NAT-A)</td>
</tr>
<tr>
<td>IP (I) : P(I)</td>
<td>IP (NAT2) : P(NAT-B)</td>
</tr>
<tr>
<td>IP (R) : P(50500)</td>
<td>IP (R) : P(50500)</td>
</tr>
</tbody>
</table>

**Message Flow:**

1. **I1:** IP (I) : P(I) -> IP (R) : P(50500)
2. **R1:** IP (R) : P(50500) -> IP (I) : P(I)
3. **I2:** IP (I) : P(I) -> IP (R) : P(50500)
4. **R2:** IP (R) : P(50500) -> IP (I) : P(I)
About the UDP Ports

- Separate UDP ports for receiving packets:
  - Control: 50500, Data: 54500

- UDP ports for sending packets:
  - Same as above or random from the range of 49152-65535

- NAT transforms IP address or ports

- I1-R1 and I2-R2 can arrive on different ports
  - Timeouts
    - Responder is stateless and does not create any (port related) state until I2
Mobility

• Use scenarios:
  1. Host moves behind a NAT into a public network
  2. Host moves within the same NAT
  3. Host moves behind a different NAT
  4. Host moves from a public network behind a NAT

• Detect the presence of NAT before handover and start/stop using UDP encapsulation accordingly

• Hosts must check the HIP control message integrity to protect against reflected packets (with forged ports)
Multihoming

- More complicated than simple mobility
  - For example, one interface can be in public network and another behind a NAT
  - Host should trigger NAT detection simultaneously using multiple interfaces
  - Asymmetric routes
  - UDP tunnels should be distinguished by SPI rather than HIT pairs
- Not (yet) handled in detail in the draft
Firewall Configuration

- Firewall processing can occur before or/and after NAT
- Required firewall policies (firewall before NAT):
  - Source ports 50500, 54500 and 49152 – 65535
  - Destination ports 50500 and 54500
- Further restrictions with “UDP connection tracking”
  - Keepalive interval must be smaller than firewall timeout value
- Firewalls are not in the main focus of the draft
Issue 1: Use Same Port Numbers as IKE

- Share same control/data UDP port numbers as IKE (RFC3947 and 3948)
- Benefit: no extra firewall configuration for firewalls that already allow UDP encapsulated IKE and ESP traffic
- Drawback: requires software modifications in hosts that have both IKE and HIP installations
- Solution: based on feedback from mobike authors, use different port numbers
Issue 2: Random Source Port at Initiator

- Allow the initiator behind NAT to use a random source UDP source port

- Benefits:
  - Basic-NAT devices with only address translation may be supported better because the port varies
  - Multiple UDP tunnels between the same peers are possible

- (Drawback: firewall rules need to based on destination port, not source)

- Originally thought that this would create problems with UDP hole punching but this is not true

- Result: this is useful as an option, so it will stay in the draft
Issue 3: Server behind NAT 1/2

- Allow the responder to be located behind a NAT. The initiator may or may not be located behind a NAT.
- Benefit: nice for P2P applications
- Drawback: additional complexity
- Solution: a NAT rendezvous/relay is anyway required
  - Primary method: assume P2P friendly NAT and avoid triangular routing by UDP hole punching
  - Fallback method: triangular routing using TURN
Issue 3: Server behind NAT 2/2

- Design alternatives
  - ICE (overkill?)
  - Subset of ICE: UDP hole punching + TURN

- Editorial open issues
  - Describe in this or separate draft?
  - WG or RG?
  - In any case, client and server case should be 100 % compatible with each other
Issue 4: NAT and Rendezvous Server

- Problem: NAT drops the R1 if responder is using RVS
- Solution: RVS relays also the R1
Issue 5: LOCATOR and NAT 1/2

- What kind of addresses to use in LOCATOR parameters upon handovers?

- Alternative 1: use the private addresses
  - Works because UDP encapsulation overrides outer addresses both for HIP and ESP packets
  - Benefit: transparent to implement
  - Drawback: privacy problems
Issue 5: LOCATOR and NAT 2/2

• Alternative 2: detect and use the public addresses of NAT
  – Benefit: no privacy problems
  – Drawback: increases the complexity of the mobility implementation

• Alternative 3: filter out the private LOCATORs and just send UPDATE to punch a hole in the NAT
  – Simple to implement
  – The LOCATORs of alternatives 1 and 2 are not very useful anyway?
Issue 6: Inner Address as IPv4

- The draft does not describe the case where inner addresses are IPv4
- Solution: reduce the details of packet en/decapsulation procedures (see issue 7) and take no standpoint to the inner addresses
Issue 7: Editorial Notes

- Server behind NAT: this or other draft
- Generalize and reduce the details of packet en/decapsulation (replace with references)
- Other misc comments
Issue 8: Mobility and Data Channel Reactivation

• Use case:
  – Host moves behind a NAT
  – The control channel punched through the NAT using UPDATE
  – The data channel is punched through the NAT with ESP keepalive

• Problem:
  – The server host in the public network does not know which SA the keepalive is related to
  – As a result, the server cannot learn the new port numbers

• Solution: use the same UDP port for control and data
Issue 9: Hairpin Translation

- Both hosts are behind the same NAT
- STUN server is used for detecting NAT
- Problem: unless NAT supports hairpin translation, the hosts may communicate inefficiently through the NAT instead of directly with each other
- Solution: when the presence of NAT is detected, send first control packets without UDP encapsulation and only then with UDP encapsulation
Questions?

Miika Komu <miika.komu@hiit.fi>
Vivien Schmitt <schmitt@netlab.nec.de>
Abhivav Pathak <abpathak@cse.iitk.ac.in>
Lars Eggert <lars.eggert@netlab.nec.de>
Martin Stiemerling <stiemerling@netlab.nec.de>