HIP
Internet-Draft
Expires: September 7, 2006

P. Nikander
Ericsson
H. Tschofenig
Siemens
X. Fu
Univ. Goettingen
T. Henderson
The Boeing Company
J. Laganier
DoCoMo Euro-Labs
March 6, 2006

Preferred Alternatives for Tunnelling HIP (PATH)
draft-nikander-hip-path-01.txt

Status of this Memo

Copyright Notice

Abstract

With the extensions defined in this document Host Identity Protocol

(HIP) can traverse legacy Network Address Translators (NATs) and certain firewalls. The extension will be useful as part of the base exchange and with the HIP Registration Extension. By using a rendezvous server an additional entity inside the network is utilized, which not only allows but also supports more restrictive NATs to be traversed.

Table of Contents

1. Introduction

This document defines extensions and allows the Host Identity Protocol (HIP) to be used in an environment where legacy NATs or Firewalls are present. To support this functionality it is necessary to provide

- UDP encapsulation for HIP signaling messages
- UDP encapsulation for IPsec traffic

The problems of allowing IPsec protected traffic and the corresponding signaling protocol (IKEv1) to traverse a NA(P)T are well described in [5]. A proposal for UDP encapsulation of IPsec protected traffic is described in [6]. It is possible to design an optimized version of it for usage with HIP. This aspect is, however, outside the scope of this document.

This document tries to accomplish the following goals:

- Make HIP work through legacy NATs (and possibly through some firewalls)
- Make HIP hosts reachable behind NATs

HIP signaling consists of (for static and bootstrapping) base exchange [1], which establish a HIP association state, and (in particullar for mobilility and multi-homing scenarios) an Update packet containing a LOCATOR parameter [2] which allows a HIP host to notify a peer about alternate locator(s) at which it is reachable. A third party in the network infrastructure, the rendezvous server (RVS) is typically used to allow a HIP initiator to learn a responder's (present) locator before initializing HIP base exchange. The interaction of HIP hosts with the rendezvous server is described in [3]. Currently, two possible ESP transport formats are being defined for carrying HIP user data, namely the standard ESP [7] and

the BEET mode [8].

This document builds on the RVS concept to allow HIP signaling and ESP-mode data traffic to successfully traverse legacy NA(P)Ts (and if necessary, firewalls). There are two possible approaches to achieve this:

- First, it is possible to combine a HIP rendezvous server and a STUN server [9], TURN server [10] or NSIS NATFW [11] node. Here, the STUN, TURN or NSIS NATFW protocol is used to allow the PATH client to learn the public IP address (and port number) created at

the NAT. The client obviously needs to support the client part of the protocol as well.

- Alternatively, the HIP registration protocol can be extended to integrate a NAT detection check. This ensembles NAT traversal support in IKEv2 [12] and corresponding extensions for IKEv1 (see [5] and [13]).

This document employs the latter approach to avoid the complexity of integrating another protocol and additional message exchanges, while still taking some advantages of the former approach. In contrast, an integration with STUN or TURN may not bring better security features in the protocol exchange.

In the proposed approach, a new parameter UDP-REA (UDP encapsulated REAddress packet) is introduced to support NAT detection. When a HIP message needs to be sent from a host to RVS (for registration messages) or another HIP (HIP signaling and data traffic), with a UDP-REA it is now possible to detect the existence of NATs and thus retrieve or create only the preferred IP address and port number, instead of the private address and port number for the host. Thus, this approach is called "Preferred Alternatives for Tunnelling HIP", or PATH.

To allow the client to inform the PATH server about its public IP address and port in a secure fashion (where this is possible and appropriate), another parameter is also introduced: S-UDP-REA, a secure version of the UDP-REA parameter. By using this parameter a secure traversal of legacy NATs is supported. The S-UDP-REA

parameter information can be obtained for example by interacting with NSIS or MIDCOM. This provides better security properties, however the details of interaction with NSIS or MIDCOM are outside the scope of this document.

Please note that the goal of this document is different from that of [14], where middleboxes (such as NATs and firewalls) are assumed to be HIP-aware and participate in the HIP message exchange. As a result, the security properties of these protocols are different as well.

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 [4].

For an interaction between a HIP host with a rendezvous server, two communicating entities are also denoted as PATH client and PATH server in this document. The PATH server always resides on the same entity as the rendezvous server. A PATH client is a HIP-aware device which supports the extensions defined in this document in addition to the HIP Registration Extension [15]. The PATH client might be located behind a legacy NAT and initiates the protocol exchange with the PATH server. The PATH server interacts with the client in the way specified in this document.

Different types of NATs (e.g., full cone, restricted NAT) are being deployed today. [9] assigns these NAT boxes to certain categories based on their data traffic forwarding or blocking behaviors. The existence of different NAT types has an impact on the protocol.

3. Protocol Extensions

This section explains the necessary protocol extensions to support the above-mentioned functionality.

3.1. UDP Encapsulation of HIP

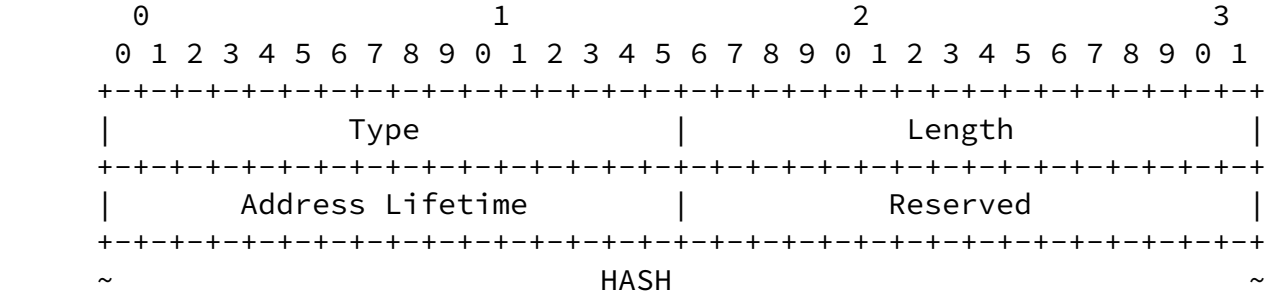
In order to deal with NA(P)Ts, it is necessary that the HIP signaling messages are UDP encapsulated. Moreover, the source port and the destination port MUST NOT be expected at a fixed port number. This aspect of NAT traversal is known from IPsec/IKE and also reflected in the design of IKEv2.

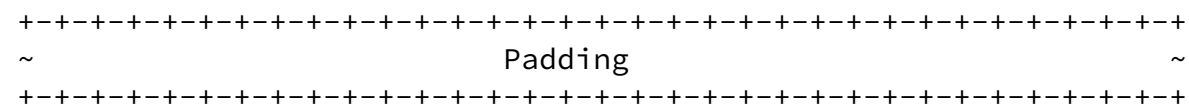
It is a policy issue whether to enable UDP encapsulation immediately when the first HIP base message is sent (i.e., the I1 message).

For IPv4, the packet format is shown in Appendix E of [1]. The same specification states that UDP encapsulation is forbidden for IPv6 but might still be necessary, particularly for IPv4-IPv6 transition.

3.2. UDP-REA parameter

This section defines the UDP-REA parameter which will be used in the traversal of legacy NATs.





Type (2 bytes):
This parameter has the value of TBD.

Length (2 bytes)
Represents the length in octets,
excluding Type, Length and Padding.

Address Lifetime (2 bytes):
This field represents the address lifetime, in seconds.

HASH (variable):
This field of variable length contains the hash of
IP address and port information.

Padding (variable):
Padding information following the HASH value

The HASH is calculated as follows:

HASH = PRF(RANDOM | Source IP | Destination IP | Source Port | Destination Port)

Where, PRF is a hash algorithm negotiated through HIP_TRANSFORM (see Section 5.2.7 of [1]); the IP address is 4 octets for an IPv4 address and 16 octets for an IPv6 address; the port numbers are encoded in network byte-order. A RANDOM value is included to prevent pre-computation attacks. The puzzle mechanism could be used for this purpose.

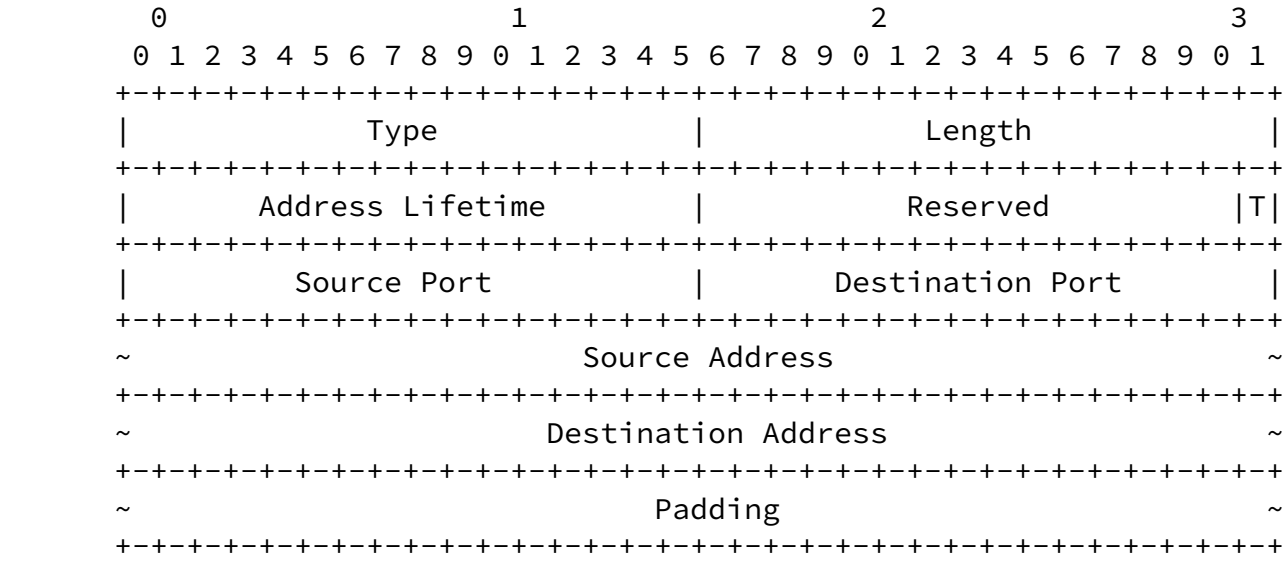
The UDP-REA parameter is zero-padded to 8 bytes. The length field contains the length of the payload without padding.

3.3. S-UDP-REA parameter

This section defines the S-UDP-REA parameter, the secure version of the UDP-REA parameter. An end host might be able to retrieve address

information securely using some protocols, such as MIDCOM or the NSIS

NATFW NSLP. These protocols enable the PATH client to create and retrieve a NAT binding in a secure fashion. This information is then communicated from the PATH client to the PATH server experiencing integrity protection, thus it is called secured UDP-REA (S-UDP-REA). Furthermore, when there is a stateful packet filter firewall along the path, S-UDP-REA may be used to allow UDP encapsulation. UDP-REA Section 3.2 would not be able to detect or act accordingly in such a situation.



Type (2 bytes):
   This parameter has the value of TBD.

Length (2 bytes)
   Represents the length in octets,
   excluding Type, Length and Padding.

Address Lifetime (2 bytes):
   This field represents the address lifetime, in seconds.

Type (T) Flag (1 bit):
   If this bit is set to 1 then the values in the Address
   fields are IPv6 addresses otherwise a IPv4 addresses.

Source Port (2 bytes):
   This field contains the source port.

Destination Port (2 bytes):
   This field contains the destination port.

 Source Address (4 or 16 bytes):
   This field contains either an IPv4 or an IPv6 address.

 Destination Address (4 or 16 bytes):
   This field contains either an IPv4 or an IPv6 address.

Padding (variable):
   Padding information following the HASH value.

4. Message Handling Rules

The PATH client attaches the UDP-REA payload to indicate support for legacy NAT traversal. Thereby it generates a hash value over the source IP address, source port, destination IP and destination port from the IP header of the HIP message. When the HIP message traverses a NAT along the path between the client and the server, its IP header will be modified. When the server receives the HIP message, it will compare the hash value carried in the HASH field of the UDP-REA parameter and the value computed on the IP address header information. If the two values do not match, then the server determines that someone along the path modified the IP header (and hopefully it is a NAT but not an adversary). The server will then use the information in the IP header to return a response to the client. If the two values are equal then it is assumed that no NAT is located along the path and UDP encapsulation is not necessary.

If a PATH client is able to obtain S-UDP-REA related information, the S-UDP-REA parameter in an integrity protected fashion instead of the plain UDP-REA should be used. This offers better security and additional capability of traversing firewalls.

This section provides further information on the message handling.

- Checksum and length field are provided in the UDP header and might not need to be repeated in the HIP header.

- HIP version (either normal or secured) is determined by the used destination port when sending the I1 packet.

- The digital signature and the keyed message digest is computed over the original payload. First, a "normal" HIP packet is constructed, then the HMAC and the digital signatures are computed. Afterwards the HIP packet is encapsulated into the UDP format.

- Short timeout (e.g., 200ms) after first packet and therefore encourage NAT-less operation.

o If preferred source address is in RFC 1918 address space, then I1 is UDP encapsulated.

5. Examples

5.1. HIP Initiator behind a NAT

This figure shows the usage of the UDP-REA parameter by the Initiator and the Responder to detect the presence of a NAT along the path. In this example, we assume the HIP Initiator is behind a NAT and the HIP Initiator initially starts with UDP encapsulation.

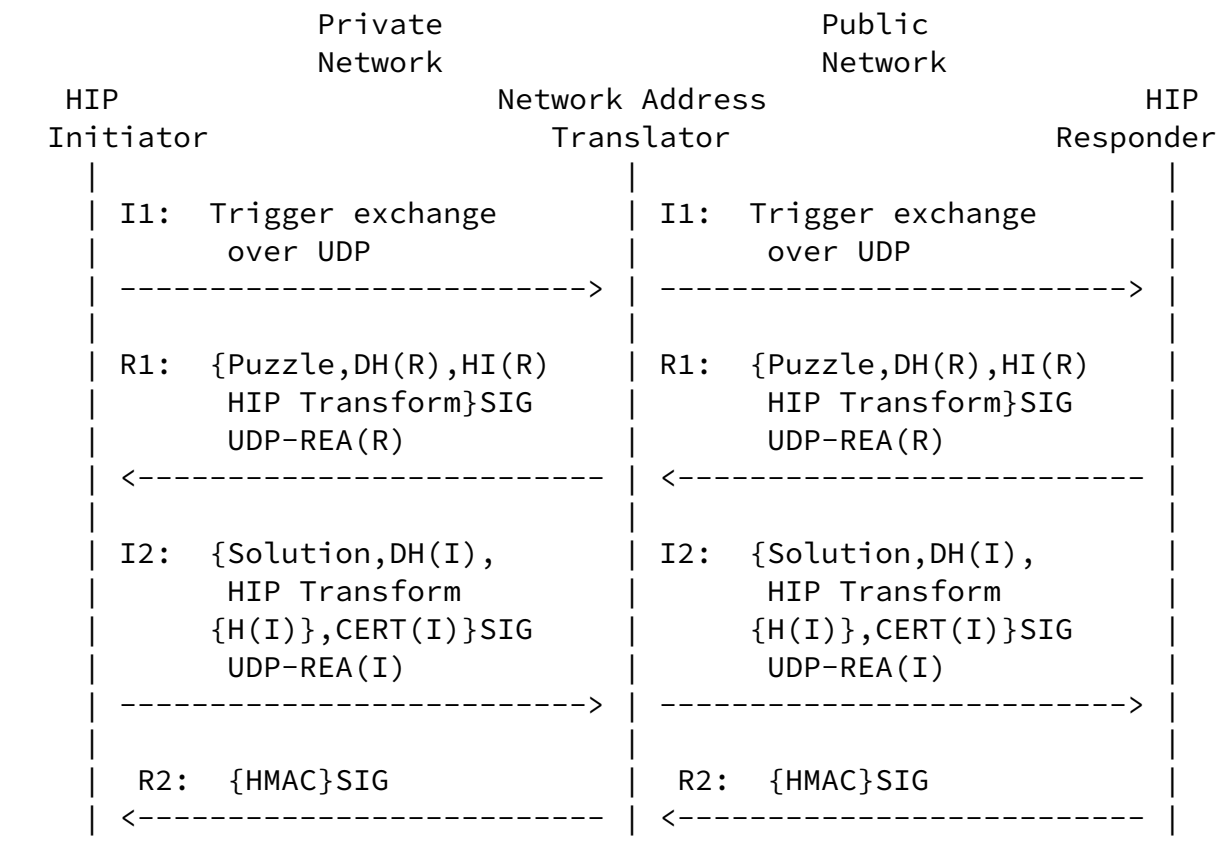


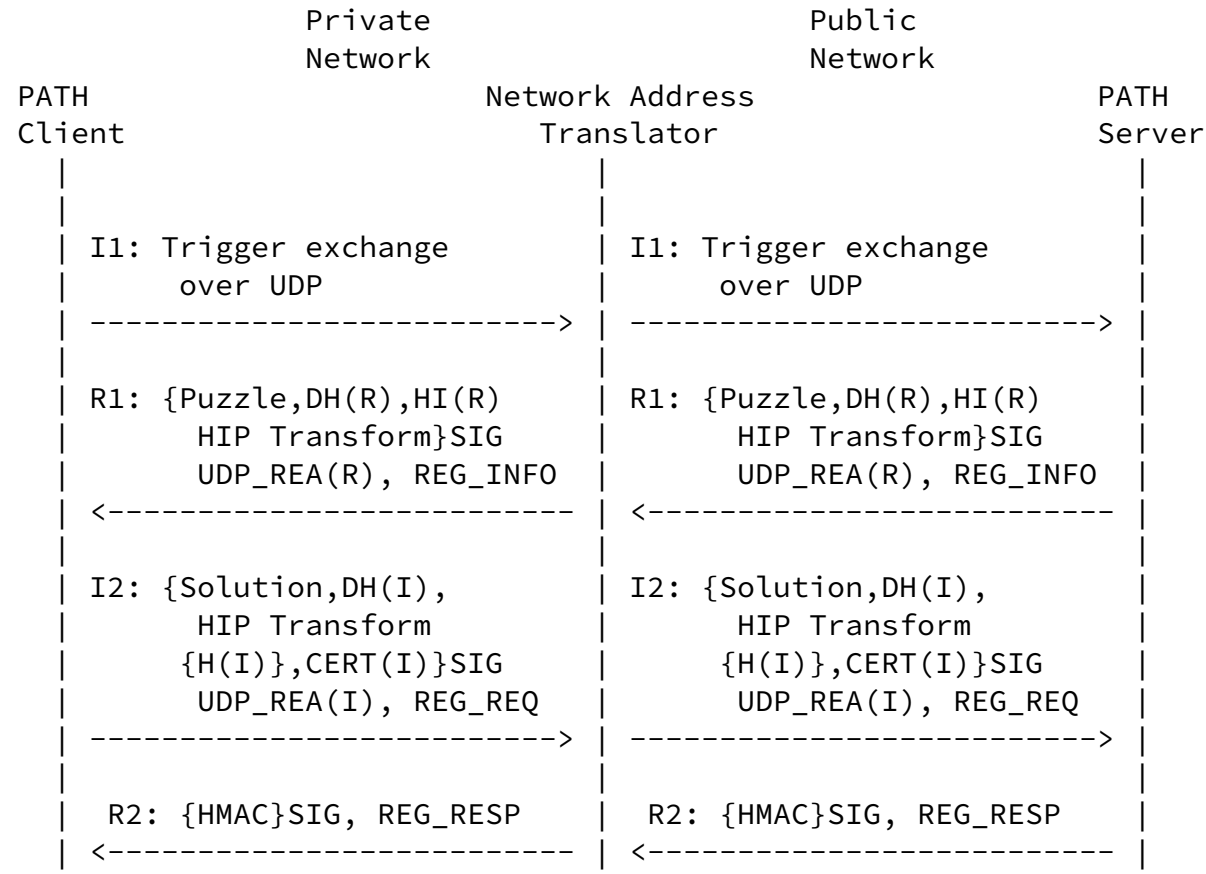
Figure 3: HIP Initiation behind a NAT: Message Flow

5.2. PATH Server Registration and Keep Alive

This section illustrates the message exchange for a PATH client registering with a PATH server, as introduced with [15]. After the protocol exchange is finalized, both peers are mutually authenticated and authorized by each other and a security association for HIP has been established.

When the PATH client starts to interact with the PATH server, the client can detect the presence of the legacy NAT along the path, by including UDP-REA parameter in the registration messages and making the computation in the client and server.

Figure 4 shows such a message exchange.



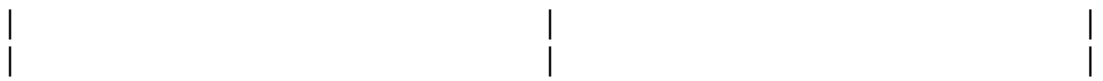


Figure 4: Registration Protocol Message Flow

Here, the HIP Registration messages are extended to not only carry REG_INFO (in the R1 message), REG_REQ (in the I2 message), REG_RESP (in the R2 message) and HIP_SIGNATURE, but also contain UDP_REQ for the detection of NATs and behaving properly.

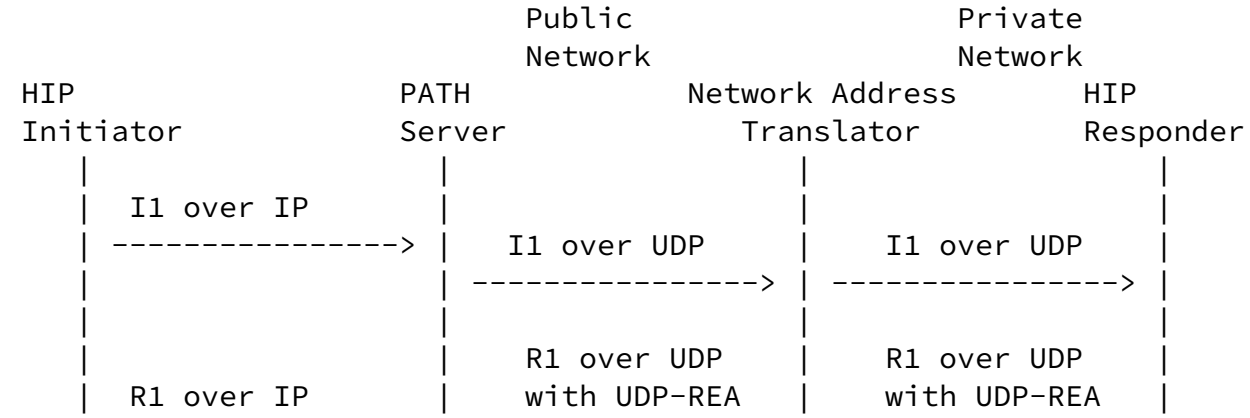
Note that this protocol exchange implicitly indicates that the PATH client will use the source IP address of the I1 and I2 messages as the preferred address when it needs to send out packets. The PATH server will use the source IP address of the incoming packet as the preferred address even though it was not authenticated (i.e., integrity protected). This extended HIP registration protocol, which comprises a 4-way message exchange including a return routability test, ensures that the PATH server can reach the PATH client and that the message has not been crafted by an off-path adversary.

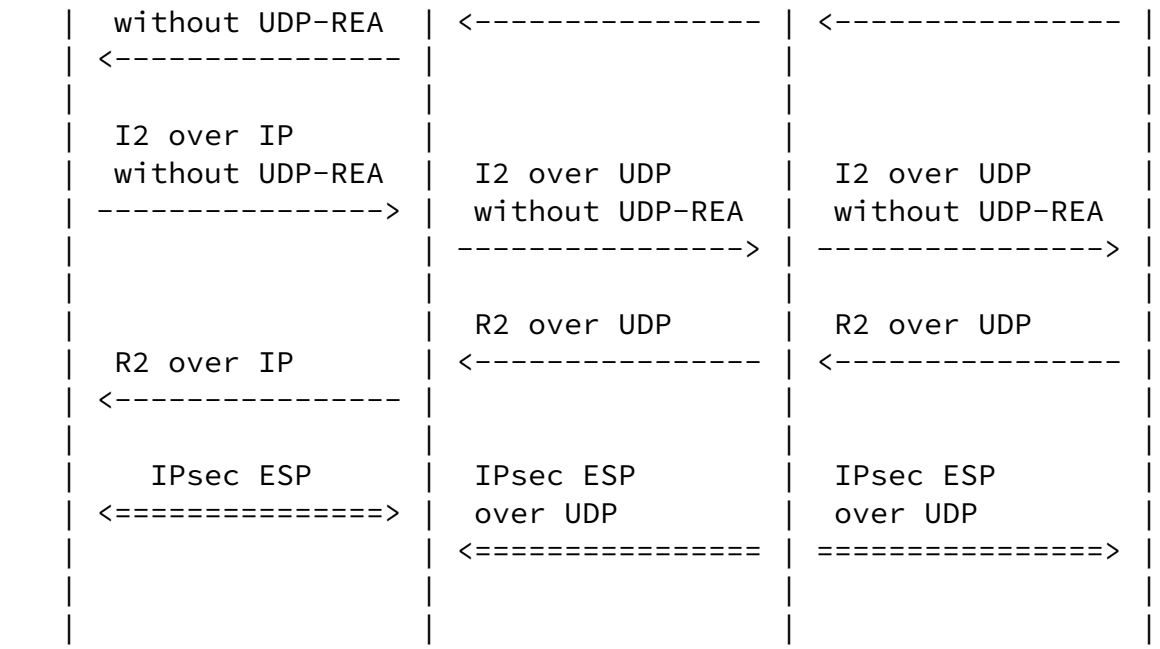
5.3. Message flow for data receiver behind a NAT

This section shows two approaches for a message flow where one HIP

node acting as the data receiver is behind a NAT. The registration with the PATH server is not shown in the figure. Figure 5 only shows the HIP base exchange between the HIP Initiator and the HIP Responder interacting with the PATH server. Figure 5 shows such a protocol exchange taken from [2].

Figure 5 shows that the HIP base exchange between the HIP Initiator and the PATH server does not use UDP encapsulation. UDP encapsulation for HIP signaling messages and for the IPsec data traffic is only enabled between the PATH server and the HIP Responder which is enabled with this extension to the HIP registration protocol. Note that IPsec data traffic will traverse the PATH server to experience UDP encapsulation. The main advantage of this approach is two-fold: (1) the HIP Initiator does not need to support the extension defined in this document and (2) traversal of more restrictive NATs can be supported when the PATH server also changes IP address information.

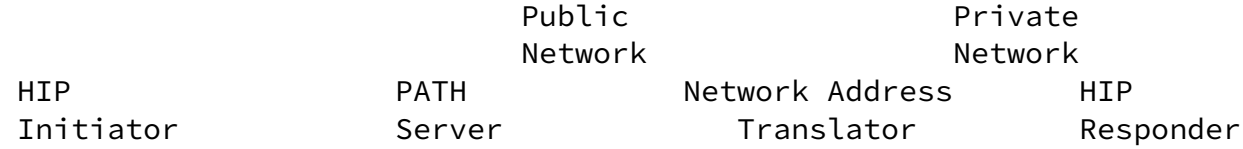

Public
Network
Private
Network
HIP
Initiator
PATH
Server
Network Address
Translator
HIP
Responder
I1 over IP
I1 over UDP
I1 over UDP
R1 over UDP
with UDP-REA
R1 over UDP
with UDP-REA
R1 over IP

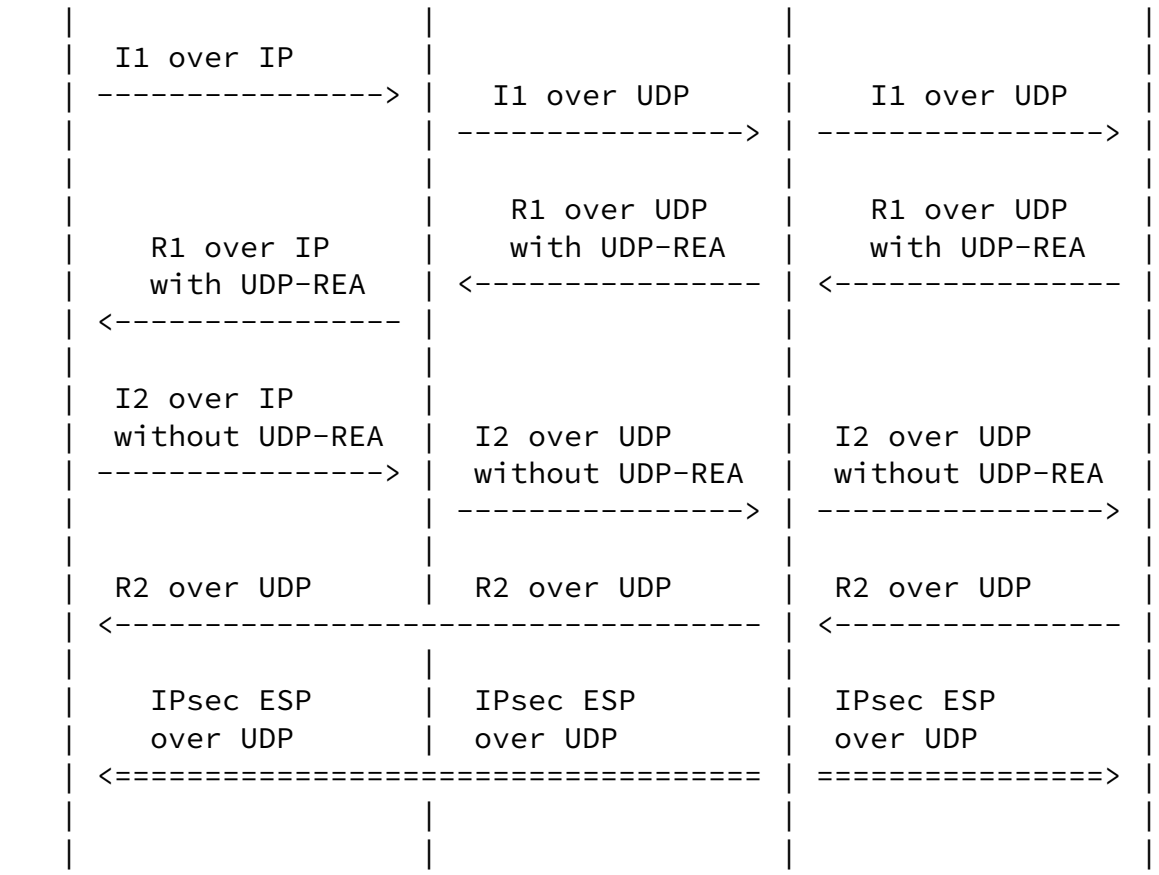


Legend:
  -->: HIP signaling messages
  ==>: Data traffic

Figure 5: Establishing contact (1/3)

Figure 6 modifies the message flow described in Figure 5 whereby R2 is already sent from the HIP Responder to the HIP Initiator directly. The responder thereby creates the necessary NAT binding at the NAT to potentially allow IPsec protected traffic from the initiator towards the responder to traverse the NAT. IPsec protected data traffic is sent only directly between the HIP Initiator and the HIP Responder.





Legend:
  -->: HIP signaling messages
  ==>: Data traffic

Figure 6: Establishing contact (2/3)

Sending the IPsec protected data traffic via the PATH server is useful if a NAT is very restrictive. This method also addresses privacy and denial of service issues as raised in the rendezvous server discussion. As symmetrical NATs are rare and an additional proxy host should be avoided, the second approach is recommended as the default method. The selection of the approach is a policy decision.

5.4. Mobility and multihoming message flow

After the PATH client has registered itself to the PATH server, as described in Figure 4, the PATH client might roam within a network or roam outside a network. Whenever the PATH client obtains a new IP address (either due to mobility, IP address reconfiguration or

switching of interfaces), a UPDATE (containing UDP-REA) message will be sent towards the PATH server to update the stored IP address information. Note that the initial registration procedure might be executed without a NAT along the path. Hence, the messages may be carried over IP and do not require UDP encapsulation. When the PATH client roams to a new network, UDP encapsulation should be used to detect the presence of a NAT. Hence, it is required to have the capability to enable UDP encapsulation for the HIP base exchange (and for the IPsec protected data traffic) in addition to the registration messages.

Figure 7 shows such a protocol exchange which ensembles the work in [2] but applies it in the PATH/RVS registration. Note UPP_REA is used for NAT traversal.

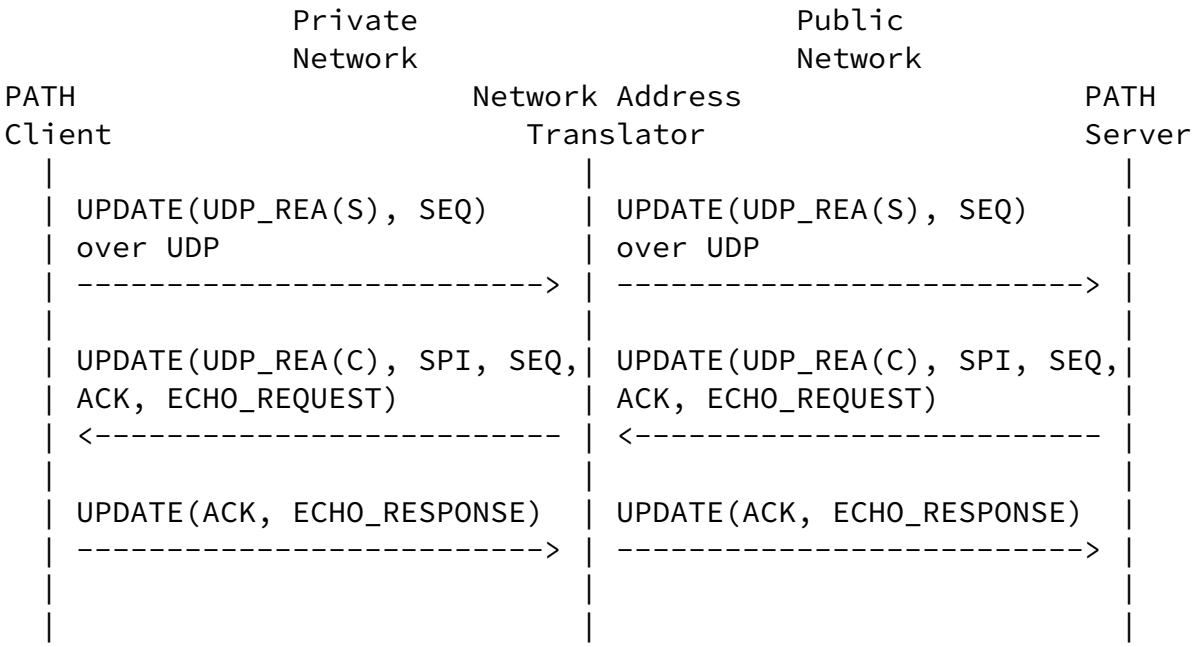


Figure 7: Mobility Message Flow

Further issues with mobility and multihoming are being investigated and will be detailed in next versions of the document.

6. New Requirements for IPsec

The text in this section focuses on Dynamic UDP Encapsulation for IPsec. By dynamic UDP encapsulation we mean UDP encapsulation per security association. Before describing the approach we describe some of the scenarios where dynamic UDP encapsulation is needed.

6.1. Association with server inside/outside NAT

The association of a client with a server outside NAT should have UDP encapsulation on while an association with a server within the same NAT should normal HIP association without any UDP encapsulation. This identification is done during base exchange. Dynamic UDP encapsulation based on security association could achieve this.

Figure 8 shows difference in association between different servers.

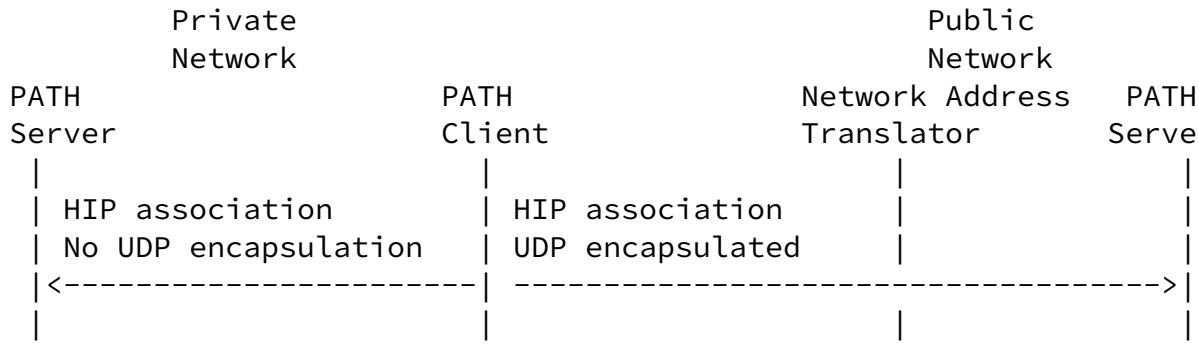


Figure 8: Dynamic NAT Associations

6.2. Mobility Scenarios

If a HIP client moves from behind the NAT to outside it then it would not need any more UDP encapsulation as it can have an HIP association without any UDP encapsulation. So when the client moves out of NAT it should reset all the NAT variables that are in security associations.

To achieve dynamic UDP encapsulation for Legacy NAT traversal we need to define it per Security Association basis. The SA would contain an indicator which would indicate whether or not the particular HIP

association needs UDP encapsulation or not. By default this indicator would be off. During base exchange if NAT is detected on the way then this indicator should be turned on. The UDP encapsulation in the kernel should also be based on this variable.

7. Security Considerations

Currently this text in this section focuses on the attacks between the PATH client and the PATH server since they differ from the description of threats provided in the past about NAT traversal for mobility protocols. The latter one have been investigated in context of IKE, IKEv2 and various other protocols and will be summarized in a future version of the document.

Attacks on the interaction between the PATH client and the PATH server can be classified as denial of service and might be launched against the PATH server itself, against third parties or against the PATH client.

PATH servers create state through the HIP registration protocol. A number of counter-measures are built-in into HIP registration protocol. A PATH server might use the client-puzzle mechanism to prevent a certain degree of DoS attacks. Additionally, it might be reasonable to limit the number of registrations at a PATH server itself. Since the PATH server needs to be discovered somehow it needs to be ensured that some security mechanisms are provided for this procedure. For example, if the PATH server is discovered using DNS SRV records then an attacker can compromise the DNS, it can inject fake records which map a domain name to the IP address of a PATH server run by the attacker. This will allow it to inject fake responses to launch a number of the attacks. This discovery procedure might, however, be part of the HIP Registration protocol. A detailed discussion about the security properties of the HIP registration protocol is outside the scope of this document. Even though the base HIP registration protocol is outside the scope of this document some of its security properties are highly relevant and applicable for this discussion. This document extends the capabilities of the registration protocol that might raise security

concerns. This section mostly focuses on the security properties of the UDP-REA parameter and it's semantic.

7.1. Third Party Bombing

Threat:

Third party bombing is also of concern when legacy NAT traversal mechanisms are in place. These attacks have been discovered in the context of Mobile IP and a threat description can be found in [16]. The main problem described in [16] is caused by the missing integrity protection of the IP address communicated from the PATH client to the PATH server. The PATH client cannot protect the IP address (without relying on additional protocol) since a NA(P)T is supposed to change the header's IP address (source, possibly

destination IP address and transport protocol identifiers). Instead of using the protected IP address inside the signaling message the PATH server is supposed to use IP header information. An adversary might provide change the IP header address to point to the intended target. Data sent to the PATH server will be send to the target rather than to the true IP address of the client.

Countermeasures:

To prevent third party bombing, the address provided by the PATH client via the IP header needs to be verified using a return-routability check. This check might either be provided as part of the base exchange (which involves two roundtrips) or as part of the REA message exchange which also provides mechanisms to execute such a test. This return-routability test MUST be performed in order to ensure that this and other attacks can be thwarted. A third party entity cannot respond to any of these HIP messages due to the cryptographic properties of the HIP base protocol and the multi-homing and mobility extensions.

7.2. Black hole

Threat:

This attack again exploits the ability for an adversary to act as a NAT and to modify the IP address information in the header.

This information will then be used by the PATH server to sent traffic towards the indicated address. If this address is not used by any entity (and particularly by the legitimate PATH client) then the traffic will be dropped. This attack is a denial of service attack.

Countermeasures:

This threat can be avoided using the same counter measures as third party bombing.

7.3. Man-in-the-middle attack

Threat:

This attack again requires the adversary to modify the IP header of the HIP registration protocol messages exchanged between the PATH server and the PATH client. Instead of pointing to a black hole or to a third party the adversary provides his address. This allows the adversary to eavesdrop the data traffic. However, in order to launch the attack, the adversary must have already been able to observe packets from the PATH client to the PATH server.

In most cases (such as when the attack is launched from an access network), this means that the attacker could already observe packets sent to the client.

Countermeasures:

It is possible that an adversary modifies the IP address information in such a way that it will receive the all traffic for a particular PATH client. Therefore, it is necessary for the adversary to be along the path to mount the initial attack. This will allow the adversary to eavesdrop both the HIP message exchange and the subsequent data traffic. However, the HIP exchange is a cryptographic protocol which is resistant against these types of attack. The data traffic is IPsec protected and therefore the adversary will gain very little profit from this attack. To make things worse for the adversary, if the PATH client roams and uses the HIP registration protocol or the REA message to update state at the PATH server the adversary needs to be located somewhere along the path where it can observe this

exchange and to modify it. As a consequence, this attack is not particular useful for the adversary.

The S-UDP-REA parameter does not suffer from the same threats as the UDP-REA parameter since it aims to provide a secure mechanism for the PATH server and the PATH client to communicate addressing information. Still, the PATH server might want to authorize the parameters provided by the PATH client by either executing a return-routability check or by using other techniques (e.g., authorization certificates) to ensure that the PATH client is indeed reachable at the indicated addresses. A malicious PATH client might add wrong addressing information to redirect traffic to a black hole or a third party. This threat has a different degree than the previously discussed threats in the sense that the PATH server will most likely know the identity of the PATH client, if we assume that only authenticated and authorized clients are allowed to use the PATH server. If the PATH server is able to detect the malicious behavior it can act accordingly.

Finally, it is necessary to add a remark on the usage of NAT/Firewall signaling protocols in relationship with the S-UDP-REA parameter usage. If the PATH client uses these protocols in an insecure or inadequate way then the envisioned security of the S-UDP-REA parameter is seriously affected. A discussion of the security properties of various NAT/Firewall signaling protocols is outside the scope of the document (in the same way as these protocols are outside the scope of this document).

8. IANA Considerations

This document extends the HIP registration protocol by defining a new parameter (the UDP-REA and the S-UDP-REA parameter). These parameters need IANA registration:

TBD:

Changes to the PATH protocol are made through a standards track revision of this specification. This document does not create new IANA registries.

9. IAB Considerations

The IAB has studied the problem of "Unilateral Self Address Fixing (UNSAF)", which is the general process by which a client attempts to determine its address in another realm on the other side of a NAT

through a collaborative protocol reflection mechanism (RFC 3424 [17]). PATH is an example of a protocol that performs this type of function. The IAB has mandated that any protocols developed for this purpose document a specific set of considerations. This section meets those requirements.

The text in this section heavily borrows from [9].

9.1. Problem Definition

From RFC 3424 [17], any UNSAF proposal must provide:

> Precise definition of a specific, limited-scope problem that is to be solved with the UNSAF proposal. A short term fix should not be generalized to solve other problems; this is why "short term fixes usually aren't".

The specific problem being solved by PATH is to provide a means for a PATH client to detect the presence of one or more NATs between it and a PATH server. The purpose of such detection is to determine the need for UDP encapsulation by the PATH server (i.e., rendezvous server).

PATH affect both UDP encapsulation of data traffic (which is IPsec protected) and HIP signaling messages.

9.2. Exit Strategy

From [17], any UNSAF proposal must provide:

> Description of an exit strategy/transition plan. The better short term fixes are the ones that will naturally see less and less use as the appropriate technology is deployed.

PATH comes with its own built in exit strategy. This strategy is the detection operation that is performed as a precursor to the actual UNSAF address-fixing operation. The discovery operation, described in Section 3.2, attempts to discover the existence of, and type of, any NATS between the client and the PATH server. PATH does not aim to detect the type of NAT (due to known deficiencies) and the discovery of the existence of NAT is itself quite robust. As NATs

are phased out through the deployment of IPv6, the discovery operation will return immediately with the result that there is no NAT, and no further operations are required. Indeed, the discovery operation itself can be used to help motivate deployment of IPv6; if a user detects a NAT between themselves and the public Internet, they can call up their access provider and complain about it.

PATH can also help to facilitate the introduction of MIDCOM or NSIS. As MIDCOM or NSIS-capable NATs are deployed, HIP end hosts will, instead of using UDP-REA, first allocate an address binding using MIDCOM or NSIS and use S-UDP-REA. However, it is a well-known limitation of MIDCOM that it only works when the agent knows the middleboxes through which its traffic will flow. This issue is fixed with the path-coupled approach followed in NSIS. Once bindings have been allocated from those middleboxes, a PATH detection procedure can validate that there are no additional middleboxes on the path from the PATH server to the PATH client. If this is the case, the HIP end host can continue operation using the address bindings allocated from MIDCOM or NSIS. If it is not the case, PATH provides a mechanism for self-address fixing through the remaining MIDCOM or NSIS-unaware middleboxes. Thus, PATH provides a way to help transition to full MIDCOM or NSIS-aware networks.

9.3. Brittleness Introduced by PATH

From [17], any UNSAF proposal must provide:

> Discussion of specific issues that may render systems more "brittle". For example, approaches that involve using data at multiple network layers create more dependencies, increase debugging challenges, and make it harder to transition.

PATH has its own limitations in several ways:

> [EDITOR's NOTE: Depending on the signaling flow and the involvement of the PATH server some behavior is assumed by NATs. There could be other types of NATs that are deployed that would not work well with some of the proposed signaling message flows. For some of the message flows the binding acquisition usage of PATH does not work for all NAT types. It will work for any application running across full cone NATs only. For restricted cone and port restricted cone NAT, it may work for some cases. For symmetric NATs, the binding acquisition will not yield a usable address (in case that not all the signaling messages and the entire data traffic is routed through the PATH server). The tight dependency on the specific type of NAT may limit the

protocol application scenarios.]

PATH assumes that the server exists on the public Internet. If the server is located in another private address realm, the HIP end host may or may not be able to use the established state at the PATH server. This heavily depends on the protocol interaction between the other HIP end host and possibly other PATH servers than are cascaded.

The bindings allocated from the NAT need to be continuously refreshed. Since the timeouts for these bindings is implementation specific, the refresh interval cannot easily be determined. When the binding is not being actively used to receive traffic, but to wait for an incoming message, the binding refresh will needlessly consume network bandwidth.

The use of the PATH server as an additional network element introduces another point of potential security attack. These attacks are largely prevented by the security measures provided the HIP registration protocol, but not entirely.

The use of the PATH server as an additional network element introduces another point of failure. If the client cannot locate a PATH server, or if the server should be unavailable due to failure, no interaction can be performed.

The use of PATH to enable UDP encapsulation for IPsec protected data traffic and for HIP messages introduces an additional bandwidth consumption which might be problematic in certain wireless networks. The modified packet forwarding through the PATH server, which might be necessary to ensure traversal of certain NAT types, might represent a non-optimal route and may increase latency for some applications (depending on the location of the PATH server).

9.4. Requirements for a Long Term Solution

From [17], any UNSAF proposal must provide:

Identify requirements for longer term, sound technical solutions - contribute to the process of finding the right longer term

solution.

Our experience with PATH has led to the following requirements for a long term solution to the NAT problem:

Requests for bindings and control of other resources in a NAT need to be explicit. Much of the brittleness in PATH derives from its guessing at the parameters of the NAT, rather than telling the NAT what parameters to use.

Control needs to be "in-band". There are far too many scenarios in which the client will not know about the location of middleboxes ahead of time. Instead, control of such boxes needs to occur in-band, traveling along the same path as the data will itself travel. This guarantees that the right set of middleboxes are controlled. NSIS exactly provides a solution for this purpose. Third-party controls are best handled using the MIDCOM framework.

Control needs to be limited. Users will need to communicate through NATs which are outside of their administrative control. In order for providers to be willing to deploy NATs which can be controlled by users in different domains, the scope of such controls needs to be extremely limited - typically, allocating a binding to reach the address where the control packets are coming from.

Simplicity is paramount. The control protocol will need to be implement in very simple clients. The servers will need to support extremely high loads. The protocol will need to be extremely robust, being the precursor to a host of application protocols. As such, simplicity is key.

9.5. Issues with Existing NAPT Boxes

From [17], any UNSAF proposal must provide:

Discussion of the impact of the noted practical issues with existing, deployed NA(P)Ts and experience reports.

Several of the practical issues with PATH involve future proofing - breaking the protocol when new NAT types get deployed. Fortunately, this is not an issue at the current time, since most of the deployed NATs are of the types assumed by PATH. The primary usage PATH has been found in the area of VoIP, to facilitate allocation of addresses for receiving RTP [12] traffic. In that application, the periodic keepalives are provided by the RTP traffic itself. However, several practical problems arise for RTP. First, RTP assumes that RTCP traffic is on a port one higher than the RTP traffic. This pairing property cannot be guaranteed through NATs that are not directly controllable. As a result, RTCP traffic may not be properly received. Protocol extensions to SDP have been proposed which

mitigate this by allowing the client to signal a different port for RTCP [18]. However, there will be interoperability problems for some time.

For VoIP, silence suppression can cause a gap in the transmission of RTP packets. This could result in the loss of a binding in the middle of a call, if that silence period exceeds the binding timeout. This can be mitigated by sending occasional silence packets to keep the binding alive. However, the result is additional brittleness; proper operation depends on the silence suppression algorithm in use, the usage of a comfort noise codec, the duration of the silence period, and the binding lifetime in the NAT.

9.6. In Closing

Some of the limitations of PATH are not design flaws. Due to the properties of HIP, PATH is fairly secure and robust form of legacy NAT traversal compared to other approach such as STUN. Some limitations are, however, related to the lack of standardized behaviors and controls in NATs. The result of this lack of standardization has been a proliferation of devices whose behavior is highly unpredictable, extremely variable, and uncontrollable. PATH does the best it can in such a hostile environment. Ultimately, the solution is to make the environment less hostile, and to introduce controls and standardized behaviors into NAT. However, until such time as that happens, PATH provides a good short term solution given the terrible conditions under which it is forced to operate. PATH also offers a long-term solution if NATs are NSIS or MIDCOM aware.

The main benefit is increased secure and a less brittle protocol operation since the NAT (or even firewalls) can be controlled and should then behave according to respective middlebox signaling protocol. Ultimately, NAT boxes might be HIP aware.

10. Acknowledgements

The authors would like to thank Aarthi Nagarajan, Abhinav Pathak, and Murugaraj Shanmugam for their helpful feedbacks on this document.

The authors would like to specially thank Lars Eggert for his contribution to previous versions of the draft.

11. Open Issues

Open issues can be found here:
http://www.tschofenig.com:8080/hip-nat/

12. References

12.1. Normative References

[1] Moskowitz, R., "Host Identity Protocol", draft-ietf-hip-base-05 (work in progress), March 2006.

[2] Nikander, P., "End-Host Mobility and Multihoming with the Host Identity Protocol", draft-ietf-hip-mm-03 (work in progress), March 2006.

[3] Laganier, J. and L. Eggert, "Host Identity Protocol (HIP) Rendezvous Extension", draft-ietf-hip-rvs-04 (work in progress), October 2005.

[4] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", March 1997.

12.2. Informative References

[5] Aboba, B. and W. Dixon, "IPsec-Network Address Translation (NAT) Compatibility Requirements", RFC 3715, March 2004.

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

[7] Kent, S., "IP Encapsulating Security Payload (ESP)", RFC 4303, December 2005.

[8] Melen, J. and P. Nikander, "A Bound End-to-End Tunnel (BEET) mode for ESP", draft-nikander-esp-beet-mode-05 (work in progress), February 2006.

[9] 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.

[10] Rosenberg, J., "Traversal Using Relay NAT (TURN)", draft-rosenberg-midcom-turn-08 (work in progress), September 2005.

[11] Stiemerling, M., "NAT/Firewall NSIS Signaling Layer Protocol (NSLP)", draft-ietf-nsis-nslp-natfw-09 (work in progress), February 2006.

[12] Kaufman, C., "Internet Key Exchange (IKEv2) Protocol", RFC 4306, December 2005.

[13] Kivinen, T., Swander, B., Huttunen, A., and V. Volpe, "Negotiation of NAT-Traversal in the IKE", RFC 3947, January 2005.

[14] Tschofenig, H. and M. Shanmugam, "Traversing HIP-aware NATs and Firewalls: Problem Statement and Requirements", draft-tschofenig-hiprg-hip-natfw-traversal-03 (work in progress), October 2005.

[15] Laganier, J., "Host Identity Protocol (HIP) Registration Extension", draft-ietf-hip-registration-01 (work in progress), December 2005.

[16] Dupont, F., "A note about 3rd party bombing in Mobile IPv6", draft-dupont-mipv6-3bombing-03 (work in progress), December 2005.

[17] Daigle, L. and IAB, "IAB Considerations for UNilateral Self-Address Fixing (UNSAF) Across Network Address Translation", RFC 3424, November 2002.

Authors' Addresses

Pekka Nikander
Ericsson Research Nomadic Lab
Hirsalantie 11
Turku FIN FIN-02420 JORVAS
Finland

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

Hannes Tschofenig
Siemens
Otto-Hahn-Ring 6
Munich, Bavaria 81739
Germany

Email: Hannes.Tschofenig@siemens.com
URI: http://www.tschofenig.com

Xiaoming Fu
University of Goettingen
Institute for Informatics
Lotzestr. 16-18
Goettingen 37083
Germany

Email: fu@cs.uni-goettingen.de

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

Email: thomas.r.henderson@boeing.com

Julien Laganier
DoCoMo Communications Laboratories Europe GmbH
DoCoMo Communications Laboratories Europe GmbH
Munich 80687
Germany

Phone: +49 89 56824 231
Email: julien.ietf@laposte.net
URI: http://www.docomolab-euro.com/

Intellectual Property Statement

Disclaimer of Validity

OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Copyright Statement

Acknowledgment

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