HIP Working Group P. Nikander
Internet-Draft Ericsson Research Nomadic Lab
Expires: April 18, 2005 J. Laganier
LIP / Sun Microsystems
October 18, 2004

Host Identity Protocol (HIP) Domain Name System (DNS) Extensions
draft-ietf-hip-dns-00

Status of this Memo

Copyright Notice

Abstract

This document specifies two new resource records for the Domain Name System (DNS), and how to use them with the Host Identity Protocol (HIP). These records allow a HIP node to store in the DNS its Host Identity (its public key), Host Identity Tag (a truncated hash of its public key), and Rendezvous Servers (RVS).

Table of Contents

1. Introduction

This document specifies two new resource records (RRs) for the Domain Name System (DNS) [7], and how to use them with the Host Identity Protocol (HIP) [9]. These records allow a HIP node to store in the DNS its Host Identity (its public key), Host Identity Tag (a truncated hash of its public key), and Rendezvous Servers (RVS) [12].

The current Internet architecture defines two global namespaces: IP addresses and domain names. The Domain Name System provides a two way lookup between these two namespaces. The HIP architecture [10] defines a new third namespace, called the Host Identity Namespace. This namespace is composed of Host Identifiers (HI) of HIP nodes. The Host Identity Tag (HIT) is one representation of an HI. This representation is obtained by taking the output of a secure hash function applied to the HI, truncated to the IPv6 address size. HITs are supposed to be used in the place of IP addresses in some ULPs and applications.

The Host Identity Protocol [9] allows two HIP nodes to establish a pair of unidirectional IPsec Security Association. These SAs are bound to the HI instead of IP addresses. The proposed HIP multi-homing mechanisms [11] allow a node to dynamically change its set of underlying IP addresses while maintaining IPsec SA and transport layer session survivability. The HIP rendezvous extensions [12] proposal allows a HIP node to maintain HIP reachability while not relying on dynamic DNS updates to make its peers aware of its current location (the set of IP address(es)).

Although a HIP node can initiate HIP communication "opportunistically" (without a priori knowledge of its peer's HI), doing so exposes both endpoints to Man-in-the-Middle attacks on the HIP handshake. Hence, there is a desire to gain knowledge of peers' HI before applications and ULPs initiate communication.

Currently, most of the Internet applications that need to communicate with a remote host first translate a domain name (often obtained via user input) into one or more IP address(es). This step occurs prior to communication with the remote host, and relies on a DNS lookup.

With HIP, IP addresses are expected to be used mostly for on-the-wire communication between end hosts, while most ULPs and applications uses HIs or HITs instead (ICMP might be an example of an ULP not using them). Consequently, we need a means to translate a domain name into an HI. Using the DNS for this translation is pretty straightforward: We define a new HIPHI (HIP HI) resource record. Upon query by an application or ULP for a FQDN -> IP lookup, the resolver would then additionally perform an FQDN -> HI lookup, and

use it to construct the resulting HI -> IP mapping (which is internal to the HIP layer). The HIP layer uses the HI -> IP mapping to translate HIs and their local representations (HITs, IPv4 and IPv6-compatible LSIs) into IP addresses and vice versa.

This draft introduces the following new DNS Resource Records:

- HIPHI, for storing Host Identifiers and Host Identity Tags
- HIPRVS, for storing rendezvous server information

2. Conventions used in this document

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].

3. Use cases

In this section, we briefly introduce a number of use cases where the DNS is useful with the Host Identity Protocol.

With HIP, most application and ULPs are unaware of the IP addresses used to carry packets on the wire. Consequently, a HIP node could take advantage of having multiple IP addresses for fail-over, redundancy, mobility, or renumbering, in a manner which is transparent to most ULPs and applications (because they are bound to HIs, hence they are agnostic to these IP address changes).

In these situations, a node wishing to be reachable by reference to its FQDN should store the following informations in the DNS:

- A set of IP address(es).
- A Host Identity (HI) and/or Host Identity Tag (HIT).
- An IP address or DNS name of its Rendezvous Server(s) (RVS).

When a HIP node wants to initiate a communication with another HIP node, it first needs to perform a HIP base exchange to set-up a HIP association towards its peer. Although such an exchange can be initiated opportunistically, i.e., without a priori knowledge of the responder's HI, by doing so both nodes knowingly risk man-in-the-middle attacks on the HIP exchange. To prevent these attacks, it is recommended that the initiator first obtain the HI of the responder, and then initiate the exchange. This can be done, for example, through manual configuration or DNS lookups. Hence, a new HIPHI RR is introduced.

When a HIP node is frequently changing its IP address(es), the dynamic DNS update latency may prevent it from publishing its new IP address(es) in the DNS. For solving this problem, the HIP architecture introduces Rendezvous Servers (RVS). A HIP host uses a Rendezvous Server as a Rendezvous point, to maintain reachability with possible HIP initiators. Such a HIP node would publish in the DNS its RVS IP address or DNS name in a HIPRVS RR, while keeping its RVS up-to-date with its current set of IP addresses.

Then, when some other node wants to initiate a HIP exchange with such a responder, it retrieves the RVS IP address by looking up a HIPRVS RR at the FQDN of the responder, and sends an I1 to this IP address. The I1 will then be relayed by the RVS to the responder, which will then complete the HIP exchange, either directly or via the RVS [12].

Note that storing HIP RR information in the DNS at a FQDN which is assigned to a non-HIP node might have ill effects on its reachability by HIP nodes.

3.1 Simple static singly homed end-host

A HIP node having a single static network attachment, wishing to be reachable by reference to its FQDN, would store in the DNS, in addition to its IP address(es), its Host Identity (HI) in a HIPHI resource record.

3.2 Mobile end-host

A mobile HIP node wishing to be reachable by reference to its FQDN would store in the DNS, instead of its IP address(es), its HI in a HIPHI RR, and the IP address(es) of its Rendezvous Server(s) in HIPRVS resource record(s). The mobile HIP node also need to notify its Rendezvous Servers of any change in its set of IP address(es).

A host wanting to reach this mobile host would then send an I1 to one of its RVS. Following, the RVS will relay the I1 up to the mobile node, which will complete the HIP exchange.

3.3 Multi-homed Site or End-host

A HIP node with several distinct network attachments is multi-homed. A HIP node attached to a network with multiple ISPs is in a multi-homed site will possibly have multiple prefixes and addresses. Such HIP node might also be reachable via several distinct Rendezvous Servers. In addition to its set of IP address(es), a multi-homed end-host would store in the DNS its HI in a HIPHI RR, and possibly the IP address(es) of its RVS(s) in HIPRVS RRs.

4. Overview of using the DNS with HIP

4.1 Different types of HITs

There are _currently_ two types of HITs. HITs of the first type consists just of the least significant bits of the hash of the public key. HITs of the second type consist of a binary prefix Host Assigning Authority (HAA) field, and only the last bits come from a hash of the Host Identity. This latter format for HIT is recommended for 'well known' systems. It is possible to support a resolution mechanism for these names in directories like DNS.

Note that the format how HITs are stored in the DNS may be different form the format actually used in protocols, the HIP base exchange [9] included. This is because the DNS RR explicitly contains the HIT type and algorithm, while some protocols may prefer to use a prefix to indicate the HIT type. The implementations are expected to use the actual HI when comparing Host Identities.

4.1.1 Host Assigning Authority (HAA) field

The 64 bits of HAA supports two levels of delegation. The first is a registered assigning authority (RAA). The second is a registered identity (RI, commonly a company). The RAA is 24 bits with values assign sequentially by ICANN. The RI is 40 bits, also assigned sequentially but by the RAA.

As IPv6 "global site-local" addresses were proposed in the IPv6 WG to replace IPv6 site-local address, it is questionable if HIP needs a kind of "global site-local" HAA, which would be generated by a given site by setting the RAA field to 0 while the RI field is filled by either random or EUI-48 bits.

4.2 Storing HI and HIT in DNS

Any conforming implementation may store Host Identifiers in a DNS HIPHI RDATA format. An implementation may also store a HIT along with its associated HI. If a particular form of an HI or HIT does not already have a specified RDATA format, a new RDATA-like format SHOULD be defined for the HI or HIT.

4.3 Storing HAA in DNS

Any conforming implementation may store a site's Host Assigning Authority in a DNS HIPHI RDATA format. A HAA MUST be stored similarly to a Type 2 HIT, while the least significant bits are set to 0. If a particular form of a HAA does not already have an associated HIT specified RDATA format, a new RDATA-like format SHOULD

be defined for the HIT/HAA.

4.4 Providing multiple IP addresses

With HIP, ULPs doesn't see which IP address is indeed used to carry packets on the wire. Consequently, a HIP node could take advantage of having multiple IP addresses for ULPs and applications fail over, redundancy, etc. This can be achieved either by storing multiple addresses in the DNS, while these addresses might be those of different IP interfaces or Rendezvous servers.

4.4.1 Storing Rendezvous Servers in the DNS

The HIP Rendezvous server (HIPRVS) resource record indicates an address (or a FQDN resolvable into an address) towards which a HIP I1 packet might be sent to trigger the establishment of an association with the entity named by this resource record.

An RVS receiving such an I1 would then forward it to the appropriate responder (the owner of the destination HIT in this I1). The responder will then complete the exchange with the initiator, possibly without ongoing help from the RVS.

Any conforming implementation may store Rendezvous Server's IP address(es) or DNS name in a DNS HIPRVS RDATA format. If a particular form of a RVS reference does not already have a specified RDATA format, a new RDATA-like format SHOULD be defined for the RVS.

4.5 Initiating connections based on DNS names

A Host Identity Protocol exchange SHOULD be initiated whenever the DNS lookup returns HIPHI resource records. Furthermore, if the DNS lookups also returns HIPRVS resource records, the addresses of these RVS SHOULD be put in the destination IP addresses list while initiating the afore mentioned HIP exchange. Since some hosts may choose not to have HIPHI information in DNS, hosts MAY implement support opportunistic HIP.

4.6 HI and HIT verification

Upon return of a HIPHI RR, a host MUST always calculate the HI-derivative HIT to be used in the HIP exchange, as specified in the HIP architecture [10], while the HIT possibly embedded along SHOULD only be used as an optimization (e.g., table lookup).

5. Storage Format

5.1 HIPHI RDATA format

The RDATA for a HIPHI RR consists of a HIT type, an algorithm type, a HIT, and a public key.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   HIT type    | HIT algorithm |  PK algorithm |               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+    HIT        |
~                                                               ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               /
/                          Public Key                           /
/                                                               /
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-|
```

5.1.1 HIT type format

The HIT type field indicates the Host Identity Tag (HIT) type and the implied HIT format.

The following values are defined:

```
0          No HIT is present.
1          A Type 1 HIT is present.
2          A Type 2 HIT is present.
3-6        Unassigned
7          A HAA is present.
```

5.1.2 HIT algorithm format

The HIT algorithm indicates the hash algorithm used to generate the Host Identity Tag (HIT) from the HI.

The following values are defined:

```
0          Reserved.
1          SHA1
2-255      Unassigned
```

5.1.3 PK algorithm type format

The algorithm type indicates the public key cryptographic algorithm

and the implied public key field format. This document reuse the values defined for the 'algorithm type' of the IPSECKEY RR [13] 'gateway type' field.

The presently defined values are given only informally:

0 No key is present.
1 A DSA key is present, in the format defined in RFC2536 [3].
2 A RSA key is present, in the format defined in RFC3110 [5].

5.1.4 HIT format

There's currently two types of HITs, and a single type of HAA. Both of them have a variable length and are stored within a single <character-string> holding the bits of the HITs or HAA:

- A *Type 1* HIT: least significant bits of the hash (e.g., SHA1) of the public key (Host Identity), which is possibly following in the HIPHI RR.
- A *Type 2* HIT: binary prefix (HAA) concatenated with a the least significant bits of the hash (e.g., SHA1) of the public key (Host Identity), which is possibly following in the HIPHI RR.
- A HAA: binary prefix (HAA) concatenated with 0, up to the associated HIT length.

5.1.5 Public key format

Both of the public key types defined in this document (RSA and DSA) reuse the public key formats defined for the IPSECKEY RR [13] (which in turns contains the algorithm-specific portion of the KEY RR RDATA, all of the KEY RR DATA after the first four octets, corresponding to the same portion of the KEY RR that must be specified by documents that define a DNSSEC algorithm).

In the future, if a new algorithm is to be used both by IPSECKEY RR and HIPHI RR, it would probably use the same public key encodings for both RRs. Unless specified otherwise, the HIPHI public key field would use the same public key format as the IPSECKEY RR RDATA for the corresponding algorithm.

The DSA key format is defined in RFC2536 [3].

The RSA key format is defined in RFC3110 [5].

5.2 HIPRVS RDATA format

The RDATA for a HIPRVS RR consists of a preference value, a Rendezvous server type and either one or more Rendezvous server

address, or one Rendezvous server FQDN.

```
       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
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
      |  preference   |     type      |                               |
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+      Rendezvous server        |
      ~                                                               ~
      +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

5.2.1 Preference format

This is an 8-bit preference order for this record. This used to specify the preference given to this RR amongst others at the same owner. Lower values are preferred. If there is a tie with some RRs, the server should return a set of RRs ordered in a load balancing manner (e.g., round robin).

5.2.2 Rendezvous server type format

The Rendezvous server type indicates the format of the information stored in the Rendezvous server field.

This document reuses the type values for the 'gateway type' field of the IPSECKEY RR [13]. The presently defined values are given only informally:

0 Reserved.
1 One or more 4-byte IPv4 address(es) in network byte order are present.
2 One or more 16-byte IPv6 address(es) in network byte order are present.
3 One or more variable length wire-encoded domain names as described in section 3.3 of RFC1035 [1]. The wire-encoded format is self-describing, so the length is implicit. The domain names MUST NOT be compressed.

5.2.3 Rendezvous server format

The Rendezvous server field indicates one or more address(es) (or one or more FQDN(s) resolvable into one or more address(es)) towards which a HIP I1 packet might be send in order to reach the entity named by this resource record.

This document reuses the format used for the 'gateway' field of the IPSECKEY RR [13], but allows to concatenate several IP (v4 or v6) addresses. The presently defined formats for the data portion of the

Rendezvous server field are given only informally:

- o One or more 32-bit IPv4 address(es) in network byte order.
- o One or more 128-bit IPv6 address(es) in network byte order.
- o One or more variable length wire-encoded domain names as described in section 3.3 of RFC1035 [1]. The wire-encoded format is self-describing, so the length is implicit. The domain names MUST NOT be compressed.

6. Transition mechanisms

During a transition period, instead of storing the HI or HIT in a HIPHI RR, the HIT MAY be stored in an AAAA RR. If a HIT is stored in an AAAA RR, it MUST be returned as the last item in the set of AAAA RRs returned to avoid as most as possible conflicts with non-HIP IPv6 nodes.

During a transition period, similarly to what may happen with HITs, the RVS's IP address might be stored in an A or AAAA RR instead of a HIPRVS RR. If a RVS IP address is stored in an A or AAAA RR, it MUST be returned as the last item in the set of returned RRs to avoid as most as possible conflicts with non-HIP IPv6 nodes.

7. Security Considerations

Though the security considerations of the HIP DNS extensions still need to be more investigated and documented, this section contains a description of the known threats involved with the usage of the HIP DNS extensions.

In a manner similar to the IPSECKEY RR [13], the HIP DNS Extensions allows to provision two HIP nodes with the public keying material (HI) of their peer. These HIs will be subsequently used in a key exchange between the peers. Hence, the HIP DNS Extensions introduce the same kind of threats that IPSECKEY does, plus threats caused by the possibility of using unpublished initiator and opportunistic mode in HIP.

A HIP node SHOULD obtain both the HIPHI and HIPRVS RRs from a trusted party trough a secure channel insuring proper data integrity of the RRs. This might be DNSSEC, or another secure channel to another directory lookup service.

In the absence of a proper secure channel, both parties are vulnerable to MitM and DoS attacks, and unrelated parties might be subject to DoS attacks as well. These threats are described in the following sections.

7.1 Attacker tampering with an unsecure HIPHI RR

The HIPHI RR contains public keying material in the form of the named peer's public key (the HI) and its secure hash (the HIT). Both of these are not sensitive to attacks where an adversary gains knowledge of them. However, an attacker that is able to mount an active attack on the DNS, i.e., tampers with this HIPHI RR (e.g., using DNS spoofing) is able to mount Man-in-the-Middle attacks on the cryptographic core of the eventual HIP exchange (responder's HIPHI and HIPRVS rewritten by the attacker).

7.2 Attacker tampering with an unsecure HIPRVS RR

The HIPRVS RR contains a destination IP address where the named peer is reachable by an I1 (HIP Rendezvous Extensions IPSECKEY RR [12]). Thus, an attacker able to tamper with this RRs is able to redirect I1 packets sent to the named peer to a chosen IP address, for DoS or MitM attacks. Note that this kind of attacks are not specific to HIP and exist independently to whether or not HIP and the HIPRVS RR are used. Such an attacker might tamper with A and AAAA RRs as well.

An attacker might obviously use these two attacks in conjunction: It will replace the responder's HI and RVS IP address by its owns in a

spoofed DNS packet sent to the initiator HI, then redirect all exchanged packets through him and mount a MitM on HIP. In this case HIP won't provide confidentiality nor initiator HI protection from eavesdroppers.

7.3 Opportunistic HIP

A HIP initiator may not be aware of its peer's HI, and/or its HIT (e.g., because the DNS does not contains HIP material, or the resolver isn't HIP-enabled), and attempt an opportunistic HIP exchange towards its known IP address, filling the responder HIT field with zeros in the I1 header. Such an initiator is vulnerable to a MitM attack because it can't validate the HI and HIT contained in a replied R1. Hence, an implementation MAY choose not to use opportunistic mode.

7.4 Anonymous Initiator

A HIP initiator may choose to use an unpublished HI, which is not stored in the DNS by means of a HIPHI RR. A responder associating with such an initiator knowingly risks a MitM attack because it cannot validate the initiator's HI. Hence, an implementation MAY choose not to use unpublished mode.

7.5 Hash and HITs Collisions

As many cryptographic algorithm, some secure hashes (e.g. SHA1, used by HIP to generate a HIT from an HI) eventually become insecure, because an exploit has been found in which an attacker with a reasonable computation power breaks one of the security features of the hash (e.g., its supposed collision resistance). This is why a HIP end-node implementation SHOULD NOT authenticate its HIP peers based solely on a HIT retrieved from DNS, but rather use both the HI and HIT.

8. IANA Considerations

IANA needs to allocate two new RR type code for HIPHI and HIPRVS from the standard RR type space.

IANA does not need to open a new registry for the HIPHI RR type for public key algorithms because the HIPHI RR reuse 'algorithms types' defined for the IPSECKEY RR [13]. The presently defined numbers are given here only informally:

```
0 is reserved
1 is RSA
2 is DSA
```

IANA needs to open a new registry for the HIPHI RR HIT type. Defined types are:

```
0          No HIT is present
1          A Type 1 HIT is present
2          A Type 2 HIT is present
3-6        Unassigned
7          A HAA is present
```

Adding new reservations requires IETF consensus RFC2434 [14].

IANA needs to open a new registry for the HIPHI RR HIT algorithm type. Defined types are:

```
0          Reserved
1          SHA1
2-255      Unassigned
```

Adding new reservations requires IETF consensus RFC2434 [14].

IANA does not need to open a new registry for the HIPRVS RR Rendezvous server type because the HIPHI RR reuse the 'gateway types' defined for the IPSECKEY RR [13]. The presently defined numbers are given here only informally:

```
0 is reserved
1 is IPv4
2 is IPv6
3 is a wire-encoded uncompressed domain name
```

9. Acknowledgments

Some parts of this draft stem from [9]. This work is heavily influenced by [13], which serves as a model for this document.

The authors would like to thanks the following people, who have provided thoughtful and helpful discussions and/or suggestions, that have improved this document: Rob Austein, Hannu Flinck, Tom Henderson, Miika Komu, Andrew McGregor, Erik Nordmark, and Gabriel Montenegro.

10. References

10.1 Normative references

[1] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, November 1987.

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

[3] Eastlake, D., "DSA KEYs and SIGs in the Domain Name System (DNS)", RFC 2536, March 1999.

[4] Crawford, M., "Binary Labels in the Domain Name System", RFC 2673, August 1999.

[5] Eastlake, D., "RSA/SHA-1 SIGs and RSA KEYs in the Domain Name System (DNS)", RFC 3110, May 2001.

[6] Bush, R., Durand, A., Fink, B., Gudmundsson, O. and T. Hain, "Representing Internet Protocol version 6 (IPv6) Addresses in the Domain Name System (DNS)", RFC 3363, August 2002.

[7] Klensin, J., "Role of the Domain Name System (DNS)", RFC 3467, February 2003.

[8] Thomson, S., Huitema, C., Ksinant, V. and M. Souissi, "DNS Extensions to Support IP Version 6", RFC 3596, October 2003.

[9] Moskowitz, R., Nikander, P. and P. Jokela, "Host Identity Protocol", draft-ietf-hip-base-01 (work in progress), October 2004.

[10] Moskowitz, R. and P. Nikander, "Host Identity Protocol Architecture", draft-ietf-hip-arch-00 (work in progress), October 2004.

[11] Nikander, P., "End-Host Mobility and Multi-Homing with Host Identity Protocol", draft-ietf-hip-mm-00 (work in progress), October 2004.

[12] Laganier, J. and L. Eggert, "Host Identity Protocol (HIP) Rendezvous Extensions", draft-ietf-hip-rvs-00 (work in progress), October 2004.

[13] Richardson, M., "A method for storing IPsec keying material in DNS", draft-ietf-ipseckey-rr-10 (work in progress), April 2004.

10.2 Informative references

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

[15] Rescorla, E. and B. Korver, "Guidelines for Writing RFC Text on Security Considerations", BCP 72, RFC 3552, July 2003.

Authors' Addresses

Pekka Nikander
Ericsson Research Nomadic Lab
JORVAS FIN-02420
FINLAND

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

Julien Laganier
LIP (CNRS-INRIA-ENSL-UCBL) & Sun Labs (Sun Microsystems)
180, Avenue de l'Europe
Saint Ismier CEDEX 38334
France

Phone: +33 476 188 815
EMail: ju@sun.com

Appendix A. Using multiple HIs with multiple IPs

The RRs defined in this document are "flat", in the sense that the IP addresses and HIs are associated to an FQDN on an equality basis. In the case where an FQDN is resolved into multiple HIs (HIPHI RRs) and IP addresses (A, AAAA or HIPRVS RRs), the requester cannot associate an IP address with a specific HI, nor the opposite.

Considering the following DNS-IP load balancing model: Multiple initiators are querying a DNS server with A or AAAA RRs at a given FQDN. The DNS server replies with a round-robin ordered set of IP addresses, causing each initiator to connect to a different address (the first address of the set they received from the DNS). This model can be extended to HIP by having the DNS returning a round-robin ordered set of HIs and IP addresses. But then the problem is that the initiator would need to map each of these HIs to a subset of the returned set of IP addresses. Hence, perhaps there is a need for having a "hierarchical" model for these RRs, which will allows to tie an HI to a specific subset of IP addresses, as illustrated in the figure below:

```
                                    FQDN
                                      |
                                  +---+---+
                                  |       |
                                  V       V
        FQDN                   HI1,HI2   HI3
         |                        |       |
 +---+---+---+---+---+          +-+-+     |
 |   |   |   |   |   |          |   |     |
 V   V   V   V   V   V          V   V     V
IP1 IP2 IP3 HI1 HI2 HI3        IP1 IP2   IP3
```

'Flat' DNS model Vs. 'Hierarchical' HI model

However, as HIs and Type 1 HITs are not yet resolvable using the DNS, implementing such a model would certainly prove to be difficult. The use of Distributed Hash Tables (DHTs) might help to resolve HIs, but at this point the whole story isn't known. In the absence of HI resolvability, there is two solutions: index IP addresses and HIs/HITs used by HIP with a common key (e.g., the IP address, the HIT, a 8-bit int, etc.), or use a per-HI DNS name, pointed to by the FQDN global to the set of HIs, and pointing to the HIs, and IP addresses associated with this particular set of HIs. to map to specific HIs, in a manner similar to what is done with NS RRs.

In the first solution (indexing), each HIPHI, HIPRVS, and HIPLOC (a

new to-be-defined RR carrying the IP address of a HIP node, to be used by HIP instead of A and AAAA RRs, if present) would contain an additional HI index field allowing to link an HI with a subset of IP addresses and vice versa. This solution is neither space-efficient, nor it is architecturally clean.

In the second solution (parallel DNS names and bindings), the PTR RR is used to alias the name of a group of node into multiple FQDNs, which are then bound to set of HIs and IP addresses, as shown in the figure below. These additional FQDNs are kind of HIP sub-FQDNs; an easy way to generate them is to suffix, or prefix the unqualified name with a sufficient number of bits of the HIT to prevent collisions local to a FQDN (e.g., foo.bar.com might haves multiple HIP sub-FQDNs: foo_2fa6.bar.com, foo_8cc4.bar.com, etc.).

```
                                    FQDN
                                      |
                                  +---+---+
                                  |       |
                                  V       V
   FQDN_1<--FQDN-->FQDN_2      HI1,HI2   HI3
     |               |            |       |
 +---+---+---+     +-+-+        +-+-+     |
 |   |   |   |     |   |        |   |     |
 V   V   V   V     V   V        V   V     V
HI1 HI2 IP1 IP2   HI3 IP3      IP1 IP2   IP3
```

The 'Hierarchical' HIP model fitting in a 'Flat' DNS model

The current plan is to use the second solution unless HIP WG members express desire to have the first solution implemented.

Appendix B. Document Revision History

Revision	Comments
00	Compared to draft-nikander-hip-dns-00: Merge multihomed site and end-host use cases. Remove HAA related text not required for Type 2 HIT definition. Remove IPv6 LSIs definitions. Replace fixed length and algorithm Type 1 and Type 2 HITs by variable length, type and algorithm HITs. Remove 'Policy Considerations' section. Fill-in 'Security Considerations' section. Allow for several IP addresses in the same HIPRVS RR. Reuse the type values and IANA registries of IPSECKEY RR. Add Annex discussing alternatives for storing multiple parallels FQDN-to-HI and HI-to-IP at a single FQDN. Minor fixes to figures and their descriptive text. Update references.

Intellectual Property Statement

Disclaimer of Validity

Copyright Statement

Acknowledgment

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