SHIM6 WG E. Nordmark
Internet-Draft Sun Microsystems
Expires: March 31, 2006 September 27, 2005

Level 3 multihoming shim protocol
draft-ietf-shim6-proto-00.txt

Status of this Memo

By submitting this Internet-Draft, each author represents that any applicable patent or other IPR claims of which he or she is aware have been or will be disclosed, and any of which he or she becomes aware will be disclosed, in accordance with Section 6 of BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt.

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

This Internet-Draft will expire on March 31, 2006.

Copyright Notice

Copyright (C) The Internet Society (2005).

Abstract

The SHIM6 working group is exploring a layer 3 shim approach for providing locator agility below the transport protocols, so that multihoming can be provided for IPv6 with failover and load spreading properties, without assuming that a multihomed site will have a provider independent IPv6 address which is announced in the global IPv6 routing table. The hosts in a site which has multiple provider allocated IPv6 address prefixes, will use the shim6 protocol specified in this document to setup state with peer hosts, so that the state can later be used to failover to a different locator pair,

should the original one stop working.

This document picks a particular approach to such a protocol and tries to flush out a bunch of details, with the hope that the WG can better understand the details in this proposal as well as discovering and understanding alternative designs that might be better. Thus this proposal is my no means cast in stone as the direction; quite to the contrary it is a depth first exploration of the design space.

Table of Contents

1. Introduction

The SHIM6 working group, and the MULTI6 WG that preceded it, is exploring a layer 3 shim approach for providing locator agility below the transport protocols, so that multihoming can be provided for IPv6 with failover and load spreading properties, without assuming that a multihomed site will have a provider independent IPv6 address which is announced in the global IPv6 routing table. The hosts in a site which has multiple provider allocated IPv6 address prefixes, will use the shim6 protocol specified in this document to setup state with peer hosts, so that the state can later be used to failover to a different locator pair, should the original one stop working.

This document takes the outlines contained in [16] and [15] and expands to an actual proposed protocol.

We assume that redirection attacks are prevented using the mechanism specified in HBA [4].

The WG mailing list is discussing the scheme used for reachability detection [5]. The schemes that are being discussed are Context Unreachability Detection (CUD) or Force Bidirectional communication Detection (FBD). This document doesn't discuss the tradeoffs between the two, but it does suggest a set of keepalive and probe messages that are sufficient to handle both. Once the WG has decided which approach to take, we can remove the unneeded messages.

There is a related but slightly separate issue of how the hosts can find which of the locator pairs is working after a failure. This is discussed in [6]. We don't yet know how these details will be done, but this draft specifies a separate "Explore" message as a placeholder for such a protocol mechanism.

1.1 Placement of the shim

TBD: Copy material from [16].

2. Terminology

This document uses the terms MUST, SHOULD, RECOMMENDED, MAY, SHOULD NOT and MUST NOT defined in RFC 2119 [7]. The terms defined in RFC 2460 [1] are also used.

2.1 Definitions

This document introduces the following terms (taken from [16]):

upper layer protocol (ULP)
: A protocol layer immediately above IP. Examples are transport protocols such as TCP and UDP, control protocols such as ICMP, routing protocols such as OSPF, and internet or lower-layer protocols being "tunneled" over (i.e., encapsulated in) IP such as IPX, AppleTalk, or IP itself.

interface
: A node's attachment to a link.

address
: An IP layer name that contains both topological significance and acts as a unique identifier for an interface. 128 bits. This document only uses the "address" term in the case where it isn't specific whether it is a locator or a identifier.

locator
: An IP layer topological name for an interface or a set of interfaces. 128 bits. The locators are carried in the IP address fields as the packets traverse the network.

identifier
: An IP layer name for an IP layer endpoint (stack name in [18]). The transport endpoint name is a function of the transport protocol and would typically include the IP identifier plus a port number.
NOTE: This proposal does not specify any new form of IP layer identifier, but still separates the identifying and locating properties of the IP addresses.

upper-layer identifier (ULID)
: An IP locator which has been selected for communication with a peer to be used by the upper layer protocol. 128 bits. This is used for pseudo-header checksum computation and connection identification in the ULP. Different sets of

communication to a host (e.g., different connections) might use different ULIDs in order to enable load spreading.

Since the ULID is just one of the IP locators/ addresses of the node, there is no need for a separate name space and allocation mechanisms.

address field — The source and destination address fields in the IPv6 header. As IPv6 is currently specified this fields carry "addresses". If identifiers and locators are separated these fields will contain locators for packets on the wire.

FQDN — Fully Qualified Domain Name

Host-pair context — The state that the multihoming shim maintains for a particular peer. The peer is identified by one or more ULIDs.

2.2 Notational Conventions

A, B, and C are hosts. X is a potentially malicious host.

FQDN(A) is the domain name for A.

Ls(A) is the locator set for A, which consists of the locators L1(A), L2(A), ... Ln(A).

ULID(A) is an upper-layer ID for A. In this proposal, ULID(A) is always one member of A's locator set.

This document also makes use of internal conceptual variables to describe protocol behavior and external variables that an implementation must allow system administrators to change. The specific variable names, how their values change, and how their settings influence protocol behavior are provided to demonstrate protocol behavior. An implementation is not required to have them in the exact form described here, so long as its external behavior is consistent with that described in this document. See Section 7 for a description of the conceptual data structures.

3. Assumptions

The general approach of a level3 shim as well as this specific proposal makes the following assumptions:

- When there is ingress filtering in the ISPs, that the use of all {source, destination} locator pairs will cause the packets to exit using different ISPs so that all exit ISPs can be tried. Since there might be only one destination locator, when the peer supports shim6 but is not multihomed, this implies that the selection of the exit ISP should be related to the source address in the packets.

- Even without ingress filtering, there is the assumption that if the host tries all {source, destination} locator pairs, that it has done a good enough job of trying to find a working path to the peer. Since we want the protocol to provide benefits even if the peer has a single locator, this seems to imply that the choice of source locator needs to somehow affect the exit path from the site.

4. Protocol Overview

The shim6 protocol operates in several phases over time. The following sequence illustrates the concepts:

- An application on host A decides to contact B using some upper-layer protocol. This results in the ULP on A sending packets to B. We call this the initial contact. Assuming the IP addresses selected by Default Address Selection [9] work, then there is no action by the shim at this point in time. Any shim context establishment can be deferred until later.

- Some heuristic on A or B (or both) determine that it might make sense to make this communication robust against locator failures. For instance, this heuristic might be that more than 50 packets have been sent or received. This makes the shim initiate the 4-way context establishment exchange.

  As a result of this exchange, both A and B will know a list of locators for each other.

- Communication continues without any change for the ULP packets. In addition, there might be some messages exchanged between the shim sub-layers for (un)reachability detection.

- At some point in time something fails. Depending on the approach to reachability detection, there might be some advise from the ULP, or the shim (un)reachability detection might discover that there is a problem.

  At this point in time one or both ends of the communication need to explore the different alternate locator pairs until a working pair is found, and rehome to using that pair.

- Once a working alternative locator pair has been found, the shim will rewrite the packets on transmit, and tag the packets with a shim context tag, and send them on the wire. The receiver will use the <Source Locator, Destination Locator, Context Tag> to find the context state which will indicate which addresses to place in the IPv6 header before passing the packet up to the ULP. The result is that from the perspective of the ULP the packet passes unmodified end-to-end, even though the IP routing infrastructure sends the packet to a different locator.

- The shim (un)reachability detection will monitor the new locator pair as it monitored the original locator pair, so that subsequent failures can be detected.

o When the shim thinks that the context state is no longer used, it can garbage collect the state; there is no coordination necessary with the peer host before the state is removed. There is an error message defined to be able to signal when there is no context state, which can be used to detect and recover from both premature garbage collection, as well as complete state loss (crash and reboot) of a peer.

The data packets in shim6 are carried completely unmodified as long as the ULID pair is used as the locator pair. After a switch to a different locator pair the packets are "tagged" so that the receiver can always determine the context to which they belong. For commonly used IP protocols this is done by using a different value in the Flow Label field, that is, there is no additional header added to the packets. But for other IP protocol types there is an extra 8 byte header inserted, which carries the next header value.

4.1 Context Tags and Use of Flow Label

A context between to hosts is actually a context between two ULIDs. The context is identified by a pair of context tags. Each end gets to allocate a context tag, and once the context is established, the shim6 control messages contain the context tag that the receiver of the message allocated. Thus at a minimum the combination of <peer ULID, local ULID, local context> tag MUST uniquely identify one context.

In addition, the non-shim6 messages, which we call payload packets, will not contain the ULIDs after a failure. This introduces the requirement that the <peer locator, local locator, local context tag> MUST uniquely identify the context. Since the peer's set of locators might be dynamic the simplest form of unique allocation of the local context tag is to pick a number that is unique on the host. Hosts which serve multiple ULIDs using disjoint sets of locators can maintain the context tag allocation per such disjoint set.

As an optimization, to ensure that payload packets, even after the locators have been switched from being the original ones, do not require an extra shim extension header, the proposal uses the Flow Label field in the IPv6 header to carry the context tag for common upper layer protocols such as TCP and UDP. This works as follows:

o The allocation and usage of the flow label during the initial communication is unchanged. Thus the TCP, UDP, etc packets are sent with a flow label which is allocated according to [11].

o When the context is created, each endpoint picks an unused context tag based on the constraints above, which is also not used as a

flow label for the set of locators.

- o The context tag is used by the shim to refer to the shim state in the control messages (such as probes, and locator updates.

- o The payload traffic (TCP, UDP, etc.) continue to flow unchanged.

- o Should there be a need to switch to a different locator pair, then the TCP, UDP, etc packets will be sent using an alternate locator pair, and with a flow label that is the same as the (20 bit) context tag.

The mechanism for detecting a loss of context state at the peer that is currently proposed in this document assumes that the receiver can tell the packets that need locator rewriting, even after it has lost all state (e.g., due to a crash followed by a reboot). The next section specifies how this can be done.

Note that there is no need for a single context to have more than one context tag; whether the locator pair is <A1, B2>, <A1, B3> or <A2, B3> the same context tag is used in the flow label field. Only the payload packets using the original <A1, B1> locator pair use the flow label (which is different than the context tag).

Whether we overload the flow label field to carry the context tag or not, any protocol (such as RSVP or NSIS) which signals information about flows from the host stack to devices in the path, need to be made aware of the locator agility introduced by a layer 3 shim, so that the signaling can be performed for the locator pairs that are currently being used.

4.2 Protocol type overloading

The mechanism for detecting a loss of context state at the peer that is currently proposed in this document assumes that the receiver can tell the packets that need locator rewriting, even after it has lost all state (e.g., due to a crash followed by a reboot). There is an alternative to detection of lost state outlined in Section 18.

The idea is to steal a partial bit from the protocol type fields that are used in the Next Header values, so that the common upper layer protocols can be identified.

For example:

- o TCP has protocol 6; TCP using alternate locators has protocol P.

- o UDP has protocol 17; TCP using alternate locators has protocol P+1.
- o ICMPv6 has protocol 58; ICMPv6 using alternate locators has protocol P+2.
- o SCTP has protocol 132; SCTP using alternate locators has protocol P+3.
- o DCCP has protocol ??; DCCP using alternate locators has protocol P+4.
- o ESP has protocol 50; ESP using alternate locators has protocol P+5.
- o AH has protocol 51; AH using alternate locators has protocol P+6.
- o FRAG has protocol 44; FRAG using alternate locators has protocol P+7.

Thus with 7 or so additional protocol field values we can do a reasonable job of overloading the flow label field and get detection of lost context state.

4.3 Securing shim6

The mechanisms are secured using a combination of techniques:

- o The HBA technique [4] for validating the locators to prevent an attacker from redirecting the packet stream to somewhere else.
- o Requiring a Reachability Probe+Reply before a new locator is used as the destination, in order to prevent 3rd party flooding attacks.
- o The first message does not create any state on the responder. Essentially a 3-way exchange is required before the responder creates any state. This means that a state-based DoS attack (trying to use up all of memory on the responder) at least provides an IPv6 address that the attacker was using.
- o The context establishment messages use nonces to prevent replay attacks.

4.4 Overview of Shim Control Messages

The shim context establishment is accomplished using four messages;

I1, R1, I2, R2. Normally they are sent in that order from initiator and responder, respectively. Should both ends attempt to set up context state at the same time (for the same ULID pair), then their I1 messages might cross in flight, and result in an immediate R2 message. [The names of these messages are borrowed from HIP [17].]

There is a No Contex error message defined, when a control or payload packet arrives and there is no matching context state at the receiver. When such a message is received, it will result in the destruction of the shim context and a re-establishment.

The peers' lists of locators are normally exchanged as part of the context establishment exchange. But the set of locators might be dynamic. For this reason there is a Locator List Update message and acknowledgement.

Even though the list of locators is fixed, a host might determine that some preferences might have changed. For instance, it might determine that there is a locally visible failure that implies that some locator(s) are no longer usable. Currently this mechanism has a separate message pair (Rehome Request and acknowledgement), but perhaps this can be encoded using the Locator List Update message pair with a preference option and no change to the list of locators.

At least two approaches (CUD and FBD) have been discussed for the shim (un)reachability detection [5]. This document attempt to define messages for both cases; once the WG has picked an approach we can delete any unneeded messages.

The CUD approach uses a probe message and acknowledgement, which can be suppressed e.g. using positive advise from the ULP. This message pair also seems needed to verify that the host is indeed present at a new locator before the data stream is redirected to that locator, in order to prevent 3rd party DoS attacks.

The FBD approach uses a keepalive message, which is sent when a host has received packets from the peer, but the ULP has not given the host an opportunity to send any payload packet to the peer.

The above probe and keepalive messages assume we have an established host-pair context. However, communication might fail during the initial context (that is, when the application or transport protocol is trying to setup some communication). If we want the shim to be able to optimize discovering a working locator pair in that case, we need a mechanism to test the reachability of locators independent of some context. We define a locator pair test message and acknowledgement for this purpose, even though it isn't yet clear whether we need such a thing.

Finally, when the context is established and there is a failure there needs to be a way to explore the set of locator pairs to efficiently find a working pair. We define an explore message as a placeholder for some mechanism in this space [6].

5. Message Formats

The shim6 messages are all carried using a new IP protocol number TBD [to be assigned by IANA]. The shim6 messages have a common header, defined below, with some fixed fields, followed by type specific fields.

5.1 Common Shim6 header

The common part of the header has a next header and header extension length field which is consistent with the other IPv6 extension headers, even if the next header value is only used for data payload which is carried with a shim6 header on the front.

The shim6 headers must be a multiple of 8 octets, hence the minimum size is 8 octets.

The common message header is as follows:

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Next Header  |  Hdr Ext Len  |         Checksum              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |                                               |
+-+-+-+-+-+-+-+-+                                               |
|                    Type specific format                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Fields:

Next Header: 8-bit selector. Normally set to NO_NXT_HDR (59). Indicates the next header value for the shim6 payload messages.

Hdr Ext Len: 8-bit unsigned integer. Length of the shim6 header in 8-octet units, not including the first 8 octets.

Checksum: 16-bit unsigned integer. The checksum is the 16-bit one's complement of the one's complement sum of the entire shim6 header message starting with the shim6 next header field, and ending as indicated by the Hdr Ext Len. Thus when there is a payload following the shim6 header, the payload is NOT included in the shim6 checksum.

Type: 8-bit unsigned integer. Identifies the actual message from the table below.

Type Value	Message
1	I1 (first establishment message from the initiator)
2	R1 (first establishment message from the responder)
3	I2 (2nd establishment message from the initiator)
4	R2 (2nd establishment message from the responder)
5	No Context Error
6	Locator List Update
7	Locator List Update Acknowledgement
8	Rehome Request
9	Rehome Acknowledgement
10	Reachability Probe
11	Reachability Probe Reply
12	Payload
13	Keepalive
14	Locator Pair Test
15	Locator Pair Test Reply
16	Context Locator pair explore

Table 1

5.2 I1 Message Format

The I1 message is the first message in the context establishment exchange.

```
     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
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |      59       |  Hdr Ext Len  |           Checksum            |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |       1       | Res |      Initiator Context Tag              |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                  Initiator Nonce                              |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                                                               |
    +                         Options                               +
    |                                                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Fields:

Next Header: NO_NXT_HDR (59).

Type: 1

Res: 4-bit field. Reserved for future use. Zero on transmit. MUST be ignored on receipt.

Initiator Context Tag: 20-bit field. The Context Tag the initiator has allocated for the context.

Initiator Nonce: 32-bit unsigned integer. A random number picked by the initiator which the responder will return in the R1 message.

The following options are allowed in the message:

ULID pair: TBD Do we need to carry the ULIDs, or assume they are the same as the address fields in the IPv6 header? Depends on how we handle failures during initial contact.

5.3 R1 Message Format

The R1 message is the second message in the context establishment exchange. The responder sends this in response to an I1 message, without creating any state specific to the initiator.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      59       |  Hdr Ext Len  |           Checksum            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       2       |                Reserved                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  Initiator Nonce                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  Responder Nonce                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                         Options                               +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Fields:

Next Header: NO_NXT_HDR (59).

Type: 2

Reserved: 24-bit field. Reserved for future use. Zero on transmit. MUST be ignored on receipt.

Initiator Nonce: 32-bit unsigned integer. Copied from the I1 message.

Responder Nonce: 32-bit unsigned integer. A number picked by the initiator which the initiator will return in the I2 message.

The following options are allowed in the message:

Responder Validator: Variable length mandatory option. Typically a hash generated by the responder, which the responder uses together with the Responder Nonce value to verify that an I2 message is indeed sent in response to a R1 message, and that the parameters in the I2 message are the same as those in the I1 message.

5.4 I2 Message Format

The I2 message is the third message in the context establishment exchange. The initiator sends this in response to a R1 message, after checking the Initiator Nonce, etc.

```
    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
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |      59       |  Hdr Ext Len  |         Checksum              |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |       3       | Res |      Initiator Context Tag              |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                  Initiator Nonce                              |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                  Responder Nonce                              |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                                                               |
   +                         Options                               +
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Fields:

Next Header: NO_NXT_HDR (59).

Type: 3

Res: 4-bit field. Reserved for future use. Zero on transmit. MUST be ignored on receipt.

Initiator Context Tag: 20-bit field. The Context Tag the initiator has allocated for the context

Initiator Nonce: 32-bit unsigned integer. A random number picked by the initiator which the responder will return in the R2 message.

Responder Nonce: 32-bit unsigned integer. Copied from the R1 message.

The following options are allowed in the message:

Responder Validator: Variable length mandatory option. Copied from the Validator in the R1 message.

ULID pair: TBD Do we need to carry the ULIDs, or assume they are the same as the address fields in the IPv6 header?

Locator list: Optionally sent when the initiator immediately wants to tell the responder its list of locators. When it is sent, the necessary HBA/CGA information for validating the locator list MUST also be included.

Locator Preferences: Optionally sent when the locators don't all have equal preference.

CGA Parameter Data Structure: Included when the locator list is included so the receiver can verify the locator list.

CGA Signature: Included when the some of the locators in the list use CGA (and not HBA) for validation.

5.5 R2 Message Format

The R2 message is the fourth message in the context establishment exchange. The responder sends this in response to an I2 message. The R2 message is also used when both hosts send I1 messages at the same time and the I1 messages cross in flight.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      59       |  Hdr Ext Len  |           Checksum            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       4       | Res |       Responder Context Tag             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   Initiator Nonce                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                          Options                              +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Fields:

Next Header: NO_NXT_HDR (59).

Type: 4

Res: 4-bit field. Reserved for future use. Zero on transmit. MUST be ignored on receipt.

Responder Context Tag: 20-bit field. The Context Tag the responder has allocated for the context

Initiator Nonce: 32-bit unsigned integer. Copied from the I2 message.

The following options are allowed in the message:

Locator List: Optionally sent when the responder immediately wants to tell the initiator its list of locators. When it is sent, the necessary HBA/CGA information for validating the locator list MUST also be included.

Locator Preferences: Optionally sent when the locators don't all have equal preference.

CGA Parameter Data Structure: Included when the locator list is included so the receiver can verify the locator list.

CGA Signature: Included when the some of the locators in the list use CGA (and not HBA) for validation.

5.6 No Context Error Message Format

Should a host receive a packet (payload packet or shim6 control message such a a locator update) and the host does not have any context state for the locators (in the IPv6 source and destination fields) and the context tag, then it will generate a No Context Error. The error includes the packet that was received, subject to the packet not exceeding 1280 octets.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      59       |  Hdr Ext Len  |           Checksum            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       5       |                    Reserved                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                          Options                              +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Fields:

Next Header: NO_NXT_HDR (59).

Type: 5

Reserved: 24-bit field. Reserved for future use. Zero on transmit. MUST be ignored on receipt.

The following options are allowed in the message:

Packet in Error: Variable length mandatory option containing the IPv6 packet that was in error, starting with the IPv6 header, and normally containing the full packet. If the resulting No Context Error message would exceed 1280 octets, the Packet In Error option will not include the full packet in error in order to limit the error to 1280 octets.

5.7 Locator List Update Message Format

The Locator List Update (LLU) Message contains a complete replacement of the senders locator list, and the options necessary for HBA/CGA to secure this. The basic sanity check that prevents off-path attackers from generating bogus updates is the context tag in the message.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      59       |  Hdr Ext Len  |          Checksum             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       6       | Res |      Initiator Context Tag              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Request Nonce                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                         Options                               +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Fields:

Next Header: NO_NXT_HDR (59).

Type: 6

Res: 4-bit field. Reserved for future use. Zero on transmit. MUST be ignored on receipt.

Initiator Context Tag: 20-bit field. The Context Tag the initiator has allocated for the context.

Request Nonce: 32-bit unsigned integer. A random number picked by the initiator which the responder will return in the acknowledgement message.

The following options are allowed in the message:

Locator List: The list of the senders (new) locators. The locators might be unchanged and only the preferences have changed.

Locator Preferences: Optionally sent when the locators don't all have equal preference.

CGA Parameter Data Structure: Included so the receiver can verify the locator list.

CGA Signature: Included when the some of the locators in the list use CGA (and not HBA) for validation.

5.8 Locator List Update Acknowledgement Message Format

This message is sent in response to a LLU message.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      59       |  Hdr Ext Len  |           Checksum            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       7       | Res |      Responder Context Tag              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     Request Nonce                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                          Options                              +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Fields:

Next Header: NO_NXT_HDR (59).

Type: 7

Res: 4-bit field. Reserved for future use. Zero on transmit. MUST be ignored on receipt.

Initiator Context Tag: 20-bit field. The Context Tag the responder has allocated for the context.

Request Nonce: 32-bit unsigned integer. Copied from the LLU message.

The following options are allowed in the message:

TBD any options?:

5.9 Rehome Request Message Format

TBD Is there any use to have a separate Rehome pair of messages? The sender can indicates its new knowledge of one of its locators (such as it no longer working) using the LLU message. Would it be useful to be able to specify just failure or preference changes without listing the actual locators? This would require that the locator list is ordered so that a Rehome Request can refer to the locators by some short index.

Perhaps this functionality can be accomplished by sending a Locator Update message and only including new Locator Preferences, without including any Locator List option? If so, we don't need a separate message.

5.10 Rehome Acknowledgement Message Format

TBD: See above.

5.11 Reachability Probe Message Format

The Reachability Probe message is used to prevent 3rd party DoS attacks, and can also be used to verify whether a context is reachable at a given locator should that be needed for the general reachability detection mechanism (e.g., if we pick the CUD mechanism where one end sends probes and expects a reply).

Before a host uses a locator for the peer that is different than the ULID, it needs to verify that the peer is indeed present at that locator by sending a Context Verify and receiving an acknowledgement. This message includes the ULID pair as well as the context tag, so that the peer can indeed verify that it has that ULID and that the context tag is correct.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      59       |  Hdr Ext Len  |           Checksum            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      10       | Res |        Receiver Context Tag             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Request Nonce                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                         Options                               +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Fields:

Next Header: NO_NXT_HDR (59).

Type: 10

Res: 4-bit field. Reserved for future use. Zero on transmit. MUST be ignored on receipt.

Receiver Context Tag: 20-bit field. The Context Tag the peer has allocated for the context.

Request Nonce: 32-bit unsigned integer. A random number picked by the initiator which the responder will return in the acknowledgement message.

The following options are allowed in the message:

ULID pair: The ULID pair that is being probed.

5.12 Reachability Probe Reply Message Format

This is sent in response to a Reachability Probe message. Although, if the receiver of the RT does not have a matching context it will send a No Context Error message.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      59       |  Hdr Ext Len  |          Checksum             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      11       | Res |       Receiver Context Tag              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Request Nonce                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                         Options                               +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Fields:

Next Header: NO_NXT_HDR (59).

Type: 11

Res: 4-bit field. Reserved for future use. Zero on transmit. MUST be ignored on receipt.

Receiver Context Tag: 20-bit field. The Context Tag the peer has allocated for the context.

Request Nonce: 32-bit unsigned integer. Copied from the request message.

The following options are allowed in the message:

ULID pair: The ULID pair that is being probed. Copied from the Probe message.

5.13 Payload Message Format

The payload message is used for payload which do not have a designated "foo-inside-shim6" protocol type, as specified in Section 4.2.

Since the shim is placed between the IP endpoint sub-layer and the IP routing sub-layer in the host, the shim header will be placed before any endpoint extension headers (fragmentation headers, destination options header, AH, ESP), but after any routing related headers (hop-by-hop extensions header, routing header, a destinations options header which precedes a routing header). When tunneling is used, whether IP-in-IP tunneling or the special form of tunneling that

Mobile IPv6 uses (with Home Address Options and Routing header type 2), there is a choice whether the shim applies inside the tunnel or outside the tunnel, which effects the location of the shim6 header.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Next Header  |       0       |         Checksum              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      12       |                 Reserved                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Fields:

Next Header: The payload which follows this header.

Hdr Ext Len: 0 (since the header is 8 octets).

Checksum: The checksum of the 8 octets.

Type: 12

Reserved: Reserved for future use. Zero on transmit. MUST be ignored on receipt.

5.14 Keepalive Message Format

The keepalive message would be used if we decide to do the Force Bidirectional communication as a way to get verification that the locator pair continues to work. If we are not going to do FBD we probably will not need this message.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      59       |  Hdr Ext Len  |         Checksum              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      13       | Res |         Receiver Context Tag            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                           Options                             +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Fields:

Next Header: NO_NXT_HDR (59).

Type: 13

Res: 4-bit field. Reserved for future use. Zero on transmit. MUST be ignored on receipt.

Receiver Context Tag: 20-bit field. The Context Tag the peer has allocated for the context.

The following options are allowed in the message:

TBD any options?:

5.15 Locator Pair Test Message Format

The above Reachability Probe message probes a context. This message just probes a locator. If we are going to handle failure during initial contact using the shim, then the shim needs to be able to find out what locators are working (and that they correspond to a desirable ULID) without assuming there is a context setup, and without knowing the actual ULID. The latter is needed so that we can handle the case when the AAAA RRset contains any combination of multiple hosts and multiple IP addresses for a given host. Having the responder send back the ULID that corresponds to a particular locator allows the initiator to take the AAAA RRset and determine which IPv6 addresses therein are for different hosts.

Once we understand how the shim will be involved in locator failures during initial contact, then we can determine whether we need this mechanism, and whether it can be overloaded on the Probe Message (e.g., by making the Receiver Context tag optional in the Reachability Probe message).

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      59       |  Hdr Ext Len  |          Checksum             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      14       |                  Reserved                     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Request Nonce                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                    Target ULID                                +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                         Options                               +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Fields:

Next Header: NO_NXT_HDR (59).

Type: 14

Reserved: 24-bit field. Reserved for future use. Zero on transmit. MUST be ignored on receipt.

Request Nonce: 32-bit unsigned integer. A random number picked by the sender which the target will return in the reply message.

Target ULID: 128-bit IPv6 address.

The following options are allowed in the message:

TBD any options?:

5.16 Locator Pair Test Reply Message Format

If a host receives a Locator Pair Test message, and the Target ULID is one of its IP addresses, then it will send this reply.

TBD: If ULID doesn't match, does it just ignore the test message? Or send some error?

TBD: Should the responder instead return its ULID, so that it is easier for the sender to determine which of the IPv6 addresses from

the DNS correspond to different hosts vs. different locators for the same host?

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      59       |  Hdr Ext Len  |           Checksum            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      15       |                   Reserved                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Request Nonce                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                    Target ULID                                +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                         Options                               +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Fields:

Next Header: NO_NXT_HDR (59).

Type: 15

Reserved: 24-bit field. Reserved for future use. Zero on transmit. MUST be ignored on receipt.

Request Nonce: 32-bit unsigned integer. Copied from the test message.

Target ULID: 128-bit IPv6 address. Copied from the test message. TBD: Or should the host be able to fill this in to make it easier for the peer to determine which locators refer to the same host?

The following options are allowed in the message:

TBD any options?:

5.17 Context Locator Pair Explore Message Format

This is a placeholder for the protocol mechanism outlined in [6]. The idea behind that mechanism is to be able to handle the case when one locator pair works in from A to B, and another locator pair works

from B to A, but there is no locator pair which works in both directions. The protocol mechanism is that as A is sending explore packets to B, B will observe which locator pairs it has received from and report that back in explore packets it is sending to A.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      59       |  Hdr Ext Len  |         Checksum              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      16       | Res |       Receiver Context Tag              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                   Sequence Number                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                         Options                               +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Fields:

Next Header: NO_NXT_HDR (59).

Type: 16

Res: 4-bit field. Reserved for future use. Zero on transmit. MUST be ignored on receipt.

Receiver Context Tag: 20-bit field. The Context Tag the peer has allocated for the context.

Sequence Number: 32-bit unsigned integer. Used to determine which packets have been received by the peer.

The following options are allowed in the message:

Explorer Results: Indication of what Explorer messages the sender has recently received from the peer.

6. Option Formats

The options follow the same layout as in RFC 2461 [2]. Thus they all are a multiple of 8 octets.

```
 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     |              ...              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
~                              ...                              ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Fields:

Type: 8-bit identifier of the type of option. The options defined in this document are below.

Length: 8-bit unsigned integer. The length of the option (including the type and length fields) in units of 8 octets. The value 0 is invalid. Nodes MUST silently discard an ND packet that contains an option with length zero.

Option Name	Type
Validator	1
Locator List	2
Locator Preferences	3
CGA Parameter Data Structure	4
CGA Signature	5
ULID Pair	6
Packet In Error	7
Explorer Results	8

Table 2

6.1 Validator Option Format

The responder can choose exactly what input uses to compute the validator, and what one-way function (MD5, SHA1) it uses, as long as the reponder can verify that the validator it receives back in the I2 packet is indeed one that 1) it computed, 2) it computed for the particular context, and 3) that it isn't a replayed I2 message.

One way for the responder to do this is to maintain a single secret (S) and a running counter for the Responder Nonce. For each I1 message, the responder can then increase the counter, use the counter value as the responder nonce, and use the following information as input to the one-way function:

- o The the secret S
- o That Responder Nonce
- o The Initiator Context Tag from the I1 message
- o The ULIDs from the I1 message
- o The locators from the I1 message (strictly only needed if they are different from the ULIDs)

```
 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 = 1    |    Length     |                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                               |
~                           Validator                           ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Fields:

Validator: Variable length content whose interpretation is local to the responder.

6.2 Locator List Option Format

The Locator List Option is used to carry all the locators of the sender. Note that the order of the locators is important, since the Locator Preferences and the Explorer packet refers to the locators by using the index in the list.

TBD: Do we need this when all the locators are contained in the PDS?

```
 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 = 2    |    Length     |                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                               |
|                           Reserveds                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
~                            Locators                           ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Fields:

Reserved: 48-bit field. Reserved for future use. Zero on transmit. MUST be ignored on receipt.

Locators: A variable number of 128-bit locators. The number of locators present can be determined by the option length field.

6.3 Locator Preferences Option Format

The Locator Preferences option can have some flags to indicate whether or not a locator is known to work. In addition, the sender can include a notion of preferences. It might make sense to define "preferences" as a combination of priority and weight the same way that DNS SRV records has such information. The priority would provide a way to rank the locators, and within a given priority, the weight would provide a way to do some load sharing. See [8] for how SRV defines the interaction of priority and weight.

As of this draft we define the preferences to include three 8-bit fields: a priority, a weight, and 8-bits of flags. The intent is that the TBD flags can carry information such as "this locator is not working", and "this locator is temporary". The latter allows making the distinction between more stable addresses and less stable addresses when shim6 is combined with IP mobility, when we might have more stable home locators, and less stable care-of-locators.

The locators are not included in the preference list. Instead, the first element refers to locator that was in the first element in the Locator List option. This assumes that the Locator List option is stable. See Section 17.

```
 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 = 3    |    Length     |     Pri[1]    |  Weight[1]    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Flags[1]    |    Pri[2]     |  Weight[2]    |   Flags[2]    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
~                              ...                              ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Fields:

Pri[i]: 8-bit unsigned integer. The Priority associated with the i'th locator in the Locator List option that is in use.

Weight[i]: 8-bit unsigned integer. The Weight associated with the i'th locator in the Locator List option that is in use.

Flags[i]: 8-bit unsigned integer. The flags associated with the i'th locator in the Locator List option that is in use.

The set of flags is TBD: Assume there will be two initially: BROKEN and TEMPORARY.

6.4 CGA Parameter Data Structure Option Format

This option contains the CGA parameter data structure (hereafter called the PDS). When HBA is used to validate the locators, the PDS contains the HBA multiprefix extension. When CGA is used to validate the locators, in addition to the CGA PDS, the signature will need to be included as a CGA Signature option.

```
 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 = 4    |    Length     |                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                               |
~                   CGA Parameter Data Structure                ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Fields:

CGA Parameter Data Structure: Variable length content. Content
               defined in [4].

6.5 CGA Signature Option Format

When CGA is used for validation of one or more of the locators in the PDS, then the message in question will need to contain this option.

```
 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 = 5    |    Length     |                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                               |
~                           CGA Signature                       ~
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Fields:

CGA Signature: Variable length content. Content defined in [4].

6.6 ULID Pair Option Format

It isn't clear whether we need this option. It depends whether we want to be able to setup a context for a ULID pair when that ULID pair can't be used to communicate. Thus the IPv6 addresses in the context establishment would not be the ULIDs.

```
 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 = 6    |    Length     |                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                               |
|                           Reserveds                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                         Sender ULID                           +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                        Receiver ULID                          +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Fields:

```
   Reserved:      48-bit field.  Reserved for future use.  Zero on
                  transmit.  MUST be ignored on receipt.

   Sender ULID:   A 128-bit IPv6 address.

   Receiver ULID: A 128-bit IPv6 address.
```

6.7 Packet In Error Option Format

```
    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 = 7    |    Length     |                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                               |
   |                           Reserveds                           |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   ~           IPv6 header, shim6/TCP/UDP header, etc              ~
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Fields:

```
   Reserved:      48-bit field.  Reserved for future use.  Zero on
                  transmit.  MUST be ignored on receipt.

   Packet:        A variable length field which contains the packet in
                  error starting with the IPv6 header.
```

6.8 Explorer Results Option Format

TBD: This needs to indicate which explorer packets (sequence numbers, source and destination locators) that have been recently received, in order to detect which locator pairs work when there is no locator pair which works in both directions. When indicating locators it makes sense to use the offset in the Locator List (that was carries in the LLU option), since this takes less space than including the locators themselves.
p

```
    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 = 8    |    Length     |              TBD              |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   ~                              ...                              ~
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Fields:

TBD:

7. Conceptual Model of a Host

This section describes a conceptual model of one possible data structure organization that hosts will maintain for the purposes of shim6. The described organization is provided to facilitate the explanation of how the shim6 protocol should behave. This document does not mandate that implementations adhere to this model as long as their external behavior is consistent with that described in this document.

7.1 Conceptual Data Structures

The key conceptual data structure for the shim6 protocol is the host pair context. This is a data structures which contains the following information:

- The peer ULID; ULID(peer)
- The local ULID; ULID(local)
- The list of peer locators, with their preferences; Ls(peer)
- For each peer locator, a bit whether it has been validated using HBA, and a bit whether the locator has been probed to verify that the ULID is present at that location.
- The preferred peer locator - used as destination; Lp(peer)
- The set of local locators and the preferences; Ls(local)
- The preferred local locator - used as source; Lp(local)
- The context tag used to transmit packets - allocated by the peer; CT(peer)
- The context to expect in received packets - allocated by the local host; CT(local)
- Reachability state for the locator pairs.
- During pair exploration, information about the explore packets that have been sent and received.

The receiver finds the context by looking it up using <Source Locator, Destination Locator, CT(local)>, where the context tag is in the Flow Label field for ULP payload packets, and in the shim headers for control messages. The sender needs to be able to find the context state when a packet is passed down from the ULP. In that

case the lookup key is the pair of ULIDs.

8. Establishing Host Pair Contexts

TBD

8.1 Sending I1 messages

8.2 Receiving I1 messages

8.3 Receiving R1 messages

8.4 Retransmitting I1 messages

8.5 Receiving I2 messages

8.6 Retransmitting I2 messages

8.7 Concurrent context establishment

9. No Such Content Errors

TBD

10. Handling ICMP Error Messages

The routers in the path as well as the destination might generate various ICMP error messages, such as host unreachable, packet too big, and payload type unknown. It is critical that these packets make it back up to the ULPs so that they can take appropriate action.

When the ULP packets are sent unmodified, that is, while the initial locators=ULIDs are working, this introduces no new concerns; an implementation's existing mechanism for delivering these errors to the ULP will work. But when the shim on the transmitting side replaces the ULIDs in the IP address fields with some other locators, then an ICMP error coming back will have a "packet in error" which is not a packet that the ULP sent. Thus the implementation will have to apply the reverse mapping to the "packet in error" before passing the ICMP error up to the ULP.

This mapping is different than when receiving ULP packets from the peer, because in that case the packets contain CT(local). But the ICMP errors have a "packet in error" with CT(peer) since they were intended to be received by the peer. In any case, since the <Source Locator, Destination Locator, CT(peer)> has to be unique when received by the peer, the local host should also only be able to find one context that matches this tuple.

Should the ULP packet have been conveyed using the protocol type encoding (Section 4.2), then that encoding must be undone for the packet in error before it is delivered to the ULP. If the ULP packet had been encapsulated in a shim6 payload message, then this extension header must be removed. The result needs to be that the ULP receives an ICMP error where the contained "packet in error" looks as if the shim did not exist.

11. Taredown of the Host Pair Context

Each host can unilaterally decide when to tare down a host-pair context. It is RECOMMENDED that hosts not tare down the context when they know that there is some upper layer protocol that might use the context. For example, an implementation might know this is there is an open socket which is connected to the ULID(peer). However, there might be cases when the knowledge is not readily available to the shim layer, for instance for UDP applications which not not connect their sockets, or any application which retains some higher level state across (TCP) connections and UDP packets.

Thus it is RECOMMENDED that implementations minimize premature taredown by observing the amount of traffic that is sent and received using the context, and only after it appears quiescent, tare down the state.

12. Updating the Locator Pairs

TBD

13. Various Probe Mechanisms

TBD

14. Rehoming to a Different Locator Pair

TBD

15. Payload Packets before a Switch

When there is no context state for the ULID pair on the sender, there is no effect on how ULP packets are sent. If the host is using some heuristic for determining when to perform a deferred context establishment, then the host might need to do some accounting (count the number of packets sent and received) even before there is a host-pair context. This need to count packets might also appear on the receive side, depending on what heuristics the implementation has chosen.

If there is a host-pair context for the ULID pair, then the sender needs to verify whether context uses the ULIDs as locators, that is, whether Lp(peer) == ULID(peer) and Lp(local) == ULID(local).

If this is the case, then packets will be sent unmodified by the shim. If it is not the case, then the logic in Section 16 will need to be used.

There will also be some maintenance activity relating to (un)reachability detection, whether packets are sent with the original locators or not. The details of this is out of scope for this document and will be covered is follow-ons to [5].

16. Payload Packets after a Switch

When sending packets, if there is a host-pair context for the ULID pair, and the ULID pair is no longer used as the locator pair, then the sender needs to transfer the packet. The transformation depends on the payload type, since some protocol values can be carried without adding a shim6 extension header, and others need an 8-octet header.

Before the payload dependent transformation, the IP address fields are replaced. The IPv6 source address field is set to Lp(local) and the destination address field is set to Lp(peer). NOTE that this MUST NOT cause any recalculation of the ULP checksums, since the ULP checksums are carried end-to-end and the ULP pseudo-header contains the ULIDs which are preserved end-to-end.

The sender skips any "routing sub-layer extension headers", thus it skips any hop-by-hop extension header, any routing header, and any destination options header that is followed by a routing header. The (extension) header that follows after that is viewed as the ULP header.

If the ULP header is of a type listed in Section 4.2, then it is replaced by the "foo-in-shim6 value for that protocol type. And in this case, the context tag CT(peer) is placed in the flow label field in the IPv6 header. Then the packet can be passed to the IP routing sub-layer.

If the ULP header type is not listed in that section, then a shim6 Payload extension header is inserted in the packet before the ULP header. In this case the context tag CT(peer) is also placed in the flow label field, and the packet is passed down to the routing sub-layer. TBD: We could use the Reserved field in the payload message instead of using flow label in this case.

The receiver parses the (extension) headers in order. Should it find a shim6 extension header it will look at the type field in that header. If the type is Payload message, then the packet must be passed to the shim6 payload handling for rewriting. If the receiver finds one of the eight additional payload type (for "foo-inside-shim6"), then it treats this analogous to the case of a shim6 payload extension header.

In both cases the receiver extracts the context tag from the IPv6 flow label field, and uses this together with the IPv6 source and destination address fields to find a host-pair context. If no context is found, the receiver SHOULD generate a No Such Context error message (see Section 9).

With the context in hand, the receiver can now replace the IP address fields with the ULIDs kept in the context. Finally, the traces of the shim are removed from the packet; any payload extension header is removed, and the next header value in the preceding header is set to be the actual protocol number for the payload. Then the packet can be passed to the ULP.

17. Open Issues

The following open issues are known:

- Is there need for keeping the list of locators private between the two communicating endpoints? We can potentially accomplish that when using CGA but not with HBA, but it comes at the cost of doing some public key encryption and decryption operations as part of the context establishment.

- Forking the context state. On the mailing list we've discussed the need to fork the context state, so that different ULP streams can be sent using different locator pairs. No protocol extensions are needed if any forking is done independently by each endpoint. But if we want A to be able to tell B that certain traffic (a 5-tuple?) should be forked, then we need a way to convey this in the shim6 protocol. The hard part would be defining what selectors can be specified for the filter which determines which traffic uses which of the forks. So the question is whether we really need signaling for forking, or whether it is sufficient to allow each endpoint to do its own selection of which locator pair it is using for which traffic.

- If we allow forking, it seems like the mechanism for reachability detection, whether it is CUD or FBD, must be applied separately for each locator pair that is in use. Without forking a single locator pair will be in use for each host-pair context, hence things would be simpler.

- Having the Locator List option contain all the prefixes implies extra bytes when the locators are also in the CGA Parameter Data Structure option. To optimize this will still need to provide an ordered list, so that the Locator Preferences can refer to the locators by "index". (The Explore Results option might need to refer to them by index as well.)

- The index to a locator might get out of synch between the two ends if messages with a new Locator List option is lost. It might make sense to include a "generation" or "locator list version" number in the Locator List option so that the Locator Preference (and Explorer Result) options can refer to a particular version of the list.

- The specified mechanism (of relying on No Such Context errors) doesn't always detect the loss of the context on the peer when the original ULID=locators are used. See Section 18 for other options.

o Which messages need sequence numbers to prevent parts of the protocol to operate on stale information should the shim6 information get out of date? Just the Locator List Update message?

o The CGA PDS might not need to be included in every LLU message. If it is associated with the ULID, it is sufficient to exchange it once. Then a HBA-protected LLU would not need anything (it can just change the preferences for the locators in any case), and a CGA-protected LLU would just need the signature option.

o In the LLU do we need to indicate which locators need to be validated using HBA vs. CGA? Or it could tell which locators are in the HBA extension in the PDS, and assume any others need CGA validation.

o What happens when a host runs out of 20 bit context tags? When is it safe for a host to reuse a context tag? With the unilateral taredown one end might discard the context state long before the other end.

18. Design Alternatives

This document has picked a certain set of design choices in order to try to work out a bunch of the details, and stimulate discussion. But as has been discussed on the mailing list, there are other choices that make sense. This section tries to enumerate some alternatives.

18.1 State Cleanup

This document uses a timer based cleanup mechanism, as specified in Section 11.

An alternative would be to use an explicit CLOSE mechanism, akin to the one specified in HIP [17]. If an explicit CLOSE handshake and associated timer is used, then there would no longer be a need for the No Context Error message due to a peer having garbage collected its end of the context. However, there is still potentially a need to have a No Context Error message in the case of a complete state loss of the peer (also known as a crash followed by a reboot). Only if we assume that the reboot takes at least the CLOSE timer, or that it is ok to not provide complete service until CLOSE timer minutes after the crash, can we completely do away with the No Context Error message.

If there is no need for the No Context Error message, this also means that it might be possible to remove the need to explicitly identifying the shim6 payload packets after a locator switch, neither using the foo-inside-shim6 protocol number nor using the shim6 Payload message. In essence, the receiver could identify the context based on the locator pair and the Flow Label in the received packets.

There might be some debugging and operational issues with removing the explicit identification of the shim6 packets after a locator switch. Should the receiver have lost the context state, then there will be no indication that something is going wrong. The shim on the receiver would happily pass up the packets unmodified to the ULP, and the ULP would most likely see a checksum error. The checksum error is caused by the ULP packet having different IP addresses than the packet that the sending ULP passed down to its shim.

18.2 Not Overloading the Flow Label

This document overloads the Flow Label field as a context tag for packets that are sent after the locators have been switched, that is, the packets where the sending shim has replaced the ULIDs with some other locator pair.

An alternative would be to not do this, and instead always use the shim6 Payload message to encapsulate the payloads when the locators are different than the ULIDs. While this doesn't remove the need to have any QoS signaling protocol be aware of the shim6 architectural implications Section 19, it does offer some other simplifications to the protocol, namely that there would no longer be a need to use designated protocol number values for the "foo-inside-shim6"; the cases when those protocol numbers are used would instead use the Payload message. The downside of always using the Payload message after a failure is that the path MTU usable by the ULP would be 8 octets less.

18.3 Detecting Context Loss

This document specifies that context loss is detected by receiving a No Such Context error message from the peer. Such messages are generated in response to a shim6 message that contain a the peer's context tag, including the shim6 Payload messages, when the receiver doesn't have matching context. They are also generated in response to data packets after a locator switch (because such payload packets are identified as such using the overloaded protocol field specified in Section 4.2).

This approach has the disadvantage of the overloaded protocol type, and it also doesn't detect the loss of context state when the original ULIDs are used as locators, because there might be no shim6 messages exchanged if the reachability detection manages to suppress any extra packets.

Discussion: it isn't clear we could remove the protocol type overloading with this approach, because without protocol type overloading it is undefined in what order the receiver would do things. Normally the receiver follows the next header chain and processes things in order. This also works with an overloaded protocol type; that next header value is basically indicating that there is a zero length shim payload header. Thus the sender can control whether this happens before the processing of some other extension header or after. Without any such indication in the packet, the receiver would find a shim context based on the <Source Locator, Destination Locator, Flow Label>. But would it process this before or after some other extension header, such as a MIPv6 Home Address Option, or IP-in-IP encapsulation header?

An alternative would be to remove the protocol field overloading and mandate that there be some low-frequency periodic Reachability Probe/ Reply messages, even when there is bidirectional communication and the ULPs report that they are doing fine. Such an approach would be able to detect state loss even before there is a locator switch.

Presumably such probes can be suppressed when there are no ULP packets being sent to the peer.

19. Implications Elsewhere

The general shim6 approach, as well as the specifics of this proposed solution, has implications elsewhere. The key implications are:

- Applications that perform referals, or callbacks using IP addresses as the 'identifiers' can still function in limited ways, as described in [12]. But in order for such applications to be able to take advantage of the multiple locators for redundancy, the applications need to be modified to either use fully qualified domain names as the 'identifiers', or they need to pass all the locators as the 'identifiers' i.e., the 'identifier' from the applications perspective becomes a set of IP addresses instead of a single IP address.

- Firewalls that today pass limited traffic, e.g., outbound TCP connections, would presumably block the shim6 protocol. This means that even when shim6 capable hosts are communicating, the I1 packets would be dropped, hence the hosts would not discover that their peer is shim6 capable. This is in fact a feature, since if the hosts managed to establish a host-pair context, then the firewall would probably drop the "different" packets that are sent after a failure (either using a "TCP-inside-shim6" protocol number, or using the shim6 payload packet with a TCP packet inside it). Thus stateful firewalls that are modified to allow shim6 packets through should also be modified to allow the payload packets through after a failure. This presumably implies that the firewall needs to track the set of locators in use by looking at the shim6 exchanges.

- Signaling protocols for QoS or other things that involve having devices in the network path look at IP addresses and port numbers, or IP addresses and Flow Labels, need to be invoked on the hosts when the locator pair changes due to a failure. At that point in time those protocols need to inform the devices that a new pair of IP addresses will be used for the flow, as well as a new Flow Label being used.

- MTU implications. The path MTU mechanisms we use are robust against different packets taking different paths through the Internet, by computing a minimum over the recently observed path MTUs. When shim6 fails over from using one locator pair to another pair, this means that packets might travel over a different path through the Internt, hence the path MTU might be quite different. Perhaps such a path change would be a good hint to the path MTU mechanism to try a larger MTU?

  The fact that the shim, at least for uncommon payload types, will

add an 8 octet extension header (the payload message) after a locator switch, can also affect the usable path MTU for the ULPs. In this case the MTU change is local to the sending host, thus conveying the change to the ULPs is an implementation matter.

20. Security Considerations

Some of the residual threats in this proposal are:

- An attacker which arrives late on the path (after the context has been established) can use the No Such Context error to cause one peer to recreate the context, and at that point in time the attacker can observe all of the exchange. But this doesn't seem to open any new doors for the attacker since such an attacker can observe the Context tags that are being used, and once known it can use those to send bogus messages.

- An attacker which is present on the path so that it can find out the context tags, can generate a No Such Context error after it has moved off the path. For this packet to be effective it needs to have a source locator which belongs to the context, thus there can not be "too much" ingress filtering between the attackers new location and the communicating peers. But this doesn't seem to be that severe, because once the error causes the context to be torn down and re-established, a new pair of context tags will be used, which will not be known to the attacker. If this is still a concern, we could require a 2-way handshake "did you really loose the state?" in response to the error message.

- It might be possible for an attacker to try random 24-bit context tags and see if they can cause disruption for communication between two hosts. We can make this harder by using a larger context tag (64-bits?) in the shim6 control messages, and use the low-order 24 bits as the flow label.

21. Acknowledgements

Over the years many people active in the multi6 and shim6 WGs have contributed ideas a suggestions that are reflected in this draft.

Thanks to Marcelo Bagnulo for providing comments on earlier versions of this draft.

22. References

22.1 Normative References

[1] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, December 1998.

[2] Narten, T., Nordmark, E., and W. Simpson, "Neighbor Discovery for IP Version 6 (IPv6)", RFC 2461, December 1998.

[3] Thomson, S. and T. Narten, "IPv6 Stateless Address Autoconfiguration", RFC 2462, December 1998.

[4] Bagnulo, M., "Hash Based Addresses (HBA)", draft-ietf-shim6-hba-00 (work in progress), July 2005.

[5] Beijnum, I., "Shim6 Reachability Detection", draft-ietf-shim6-reach-detect-00 (work in progress), July 2005.

[6] Arkko, J., "Failure Detection and Locator Selection Design Considerations", draft-ietf-shim6-failure-detection-00 (work in progress), July 2005.

22.2 Informative References

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

[8] Gulbrandsen, A., Vixie, P., and L. Esibov, "A DNS RR for specifying the location of services (DNS SRV)", RFC 2782, February 2000.

[9] Draves, R., "Default Address Selection for Internet Protocol version 6 (IPv6)", RFC 3484, February 2003.

[10] Abley, J., Black, B., and V. Gill, "Goals for IPv6 Site-Multihoming Architectures", RFC 3582, August 2003.

[11] Rajahalme, J., Conta, A., Carpenter, B., and S. Deering, "IPv6 Flow Label Specification", RFC 3697, March 2004.

[12] Nordmark, E., "Shim6 Application Referral Issues", draft-ietf-shim6-app-refer-00 (work in progress), July 2005.

[13] Abley, J., "Shim6 Applicability Statement", draft-ietf-shim6-applicability-00 (work in progress), July 2005.

[14] Huston, G., "Architectural Commentary on Site Multi-homing using a Level 3 Shim", draft-ietf-shim6-arch-00 (work in progress), July 2005.

[15] Bagnulo, M. and J. Arkko, "Functional decomposition of the multihoming protocol", draft-ietf-shim6-functional-dec-00 (work in progress), July 2005.

[16] Nordmark, E. and M. Bagnulo, "Multihoming L3 Shim Approach", draft-ietf-shim6-l3shim-00 (work in progress), July 2005.

[17] Moskowitz, R., "Host Identity Protocol", draft-ietf-hip-base-03 (work in progress), June 2005.

[18] Lear, E. and R. Droms, "What's In A Name:Thoughts from the NSRG", draft-irtf-nsrg-report-10 (work in progress), September 2003.

Author's Address

Erik Nordmark
Sun Microsystems
17 Network Circle
Menlo Park, CA 94043
USA

Phone: +1 650 786 2921
Email: erik.nordmark@sun.com

Intellectual Property Statement

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at http://www.ietf.org/ipr.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

Disclaimer of Validity

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS 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

Copyright (C) The Internet Society (2005). This document is subject to the rights, licenses and restrictions contained in BCP 78, and except as set forth therein, the authors retain all their rights.

Acknowledgment

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