SHIM6 WG   E. Nordmark
Internet-Draft   Sun Microsystems
Expires: August 26, 2008   February 23, 2008

Extended Shim6 Design for ID/loc split and Traffic Engineering
draft-nordmark-shim6-esd-01.txt

Status of this Memo

Copyright Notice

Abstract

The Shim6 protocol provides for locator agility while satisfying the 'first, do no harm' security requirements. This document outlines three rather orthogonal sets of extensions to Shim6. The first one is how to procide complete separation between identifiers and locators. The second one is how to allow routers to rewrite the locators in the shim6 packets as a way to provide traffic engineering information to the hosts. The third one is the outline of a simple extension to allow shim6, with a CGA upper-layer ID, to operate using IPv4 addresses as locators.

The purpose of this outline is to stimulate discussions.

Table of Contents

1. Introduction

This document describes extensions to Shim6 [7] to handle complete separation between identifiers and locators and also to provide better feedback to the host for the purposes of traffic engineering.

The Shim6 protocol provides for locator agility while satisfying the 'first, do no harm' security requirements [22]. We believe that the only existing two approaches which satisfy these requirements are shim6 and HIP [30]. Satisfying those requirements seem to imply some additional state on the endnodes, at least when the locator agility mechanism is implemented in layer 3 as opposed to being folded into every transport protocol; if it was folded into the transport protocols one could probably include this information in the transport protocol state and exchanges. As such, these approaches require some additional state management which results in packet exchanges.

This document outlines some extensions to Shim6 that in addition provides complete separation between identifiers and locators, and allows routers to rewrite the locators in the shim6 packets as a way to provide traffic engineering information to the hosts.

1.1. Identifier/Locator split

Shim6 as currently specified states that the upper-layer identifier (ULID) is one of the IPv6 locators of the host. However, the protocol exchanges in shim6 do not require this, and there is already a ULID-pair option that can be conveyed in the shim6 establishment exchange. Thus in the protocol exchanges we can just carry that ULID-pair option in all the shim6 control messages (I1, R1, I2, R2, R1bis, I2bis, Update Request, Update Acknowledgement, Keepalive, and Probe). But that in itself doesn't solve the problem; a large piece of the problem lies in:

- o What is the syntax and semantics of an identifier.
- o How are identifiers allocated to hosts.
- o How do the hosts find a Identifier from a domain name.
- o How do application referrals [27] work.

XXX Add TBD for referals using non-routable ULID - need to know peer application is running on a system capable of looking them up.

This document makes some, not completely arbitrary, assumptions for the above issues, and presents an outline of how things could work

given those assumptions. The document shows that there is some additional setup cost associated with identifier/locator separation compared to shim6 just switching between a set of locators. We argue that some additional overhead is inevitable; the benefits of identifier/locator separation is to provide a layer of indirection between the two, and this indirection necessitates some form of lookup as part of initiating communication to an identifier.

1.2. Traffic Engineering

With the current style of IPv4 multihoming, which uses a single IP address prefix and BGP, there is a range of BGP techniques that can be used to affect the routing for that single prefix, and as a result effect which path incoming traffic use to reach the site. Also, the site itself can use some internal routing control to select which egress path to use should a route be available via multiple egresses.

In any scheme that uses multiple address/locator prefixes per site the mechanisms to control which packets flow over which ingress and egress are likely to be completely different, since BGP would not be able to correlate the different prefixes that are assigned to a single site. This might imply that the traffic engineering capabilities might be quite different.

As specified in shim6, the hosts select the ULID-pair, which by convention is the initial locator pair, using RFC 3484bis [16]. With the extensions specified in these documents there are two cases to consider, depending on whether the ULID is a locator or the ULID is a non-routable identifier:

o For ULIDs that are locators, RFC 3484bis still applies. In addition, DNS SRV records [10] can be used as a way to give the destination site some influence over which ULID = initial locator is used, but that requires that the applications use SRV records.

o With ULIDs that are non-routable identifiers, there will most likely be only one identifier for the destination as well as the source. Thus the role of RFC 3484 is largely removed. But there is an additional step of looking up the identifier to find the locator, and at that point in time it makes sense to consider traffic engineering for selecting the initial locator pair.

In both cases, shim6 already provides the mechanism of a Locator Preference option which can be used by the peers to change the preferences. But this mechanism assumes that there is a mechanism by which a host in a site could be informed of its site's preferences. We outline a DHCPv6 option in this document that could be used to convey this information.

1.3. Shim6 using IPv4 addresses as locators

When CGA is used to prevent redirection attacks in shim6, there is no constraint on the locators that are used apart from host B must know its own locators so it can pass it them to host A. In particular, we can use IPv4 addresses as locators; this doesn't require anything more than defining how an IPv4 address is carried in the 128-bit fields in the Locator List option.

As is the norm when suggesting to run an protocol over IPv4, after 5 milliseconds the question is raised how the protocol can operate over IPv4 NAT boxes. For all protocols that adds one or two orders of magnitude or more complexity, and shim6 is no exception. The complexities lie in:

- Handling initial contact from the public side of the NAT box to a host on the private side of the NAT box. All protocols end up with some form of 3rd party device on the public side in order to handle this.

- Maintaining the NAT state in the NAT box, and detecting when this state has been lost and must be recreated. For shim6 the failure detection mechanism [9] can detect this.

- Being able to recreate that state in the NAT after it has been lost.

Thus while supporting IPv4 addresses as locators is easy, running across NATs is complex. It is far from clear that this is worth-while pursuing.

2. Identifier/Locator Split

This section outlines how non-routable identifiers can work in the whole system, including their lookup and how they are used in shim6. There are likely to be alternative designs, so this is by no means believed to be the optimal approach, but working out some number of details is still a useful exercise and helps have more concrete discussions.

2.1. Syntax, Semantics, and Allocation of non-routable identifiers

In order for applications, which have been ported to use the IPv6 APIs that carry 128-bit IPv6 addresses, to not have to be ported again, it makes sense to try to make do with an syntax that fits in 128 bits. Note that one could wish that we could apply this to the IPv4 32-bit APIs, but with a future Internet with more than 4 billion hosts, trying to support the IPv4 APIs to identify each host would be impossible.

Since we are likely to have applications communicate to both hosts which have an IP identifier and those which only have the IP locators, it is highly desirable to be able to have a syntactic means to tell an IP identifier apart from an IP locator. A reasonable approach is to allocate a small subset of the IPv6 address space [23] to be non-routable IP identifiers, in a similar means to the KHI approach [24].

The desired semantics of an IP identifier is to be a name that refers to an instance of the IP protocol. Hence it should not be bound to a particular network interface. Also, it is desirable for the identifier to be long term stable, for instance ensuring that the identifier survives renumbering.

As we will discuss below, for application referrals to work using 128-bit IP "addresses" as handles for another host, there has to be an efficient and scalable way to look up an identifier and find the locators. An obvious way to get scalability is to do hierarchical allocation of the identifiers, since this allows for a scalable, hierarchical organization of a lookup system and also ensures that the identifiers are unique.

While it certainly isn't the only possibility, in order to work out a complete picture, this document suggests such a hierarchical allocation, in such a way that at least 64 bits of the identifier is left to each site to allocate. (With 64 bits we can use CGA to "prove" identifier ownership as a way to prevent redirection attacks from off-path attackers.)

2.2. Domain name lookup to find the Identifier

Many applications start their communication by knowing the fully-qualified domain name of the peer, and they look this up in the DNS to find AAAA and A records for that peer. As we introduce a separate IP Identifier we want the same capability, but we also need to avoid disturbing existing IPv6 hosts which would not be able to deal with an IP identifier, since they assume all AAAA records contain routable locators.

A conceptually clean way to handle this is to introduce a separate ID resource record type in the DNS, which has the same right-hand side as the AAAA records. Then hosts which implement support for the IP identifier would first look for an ID record, and if none found, look for AAAA and A records. This would work, but adds inefficiencies until (or unless) most hosts have ID records.

An alternative would be to overload AAAA records to contain both identifiers and locators and rely on RFC 3484 to avoid accidentally picking an identifier on a host which doesn't understand what an IP identifier is. Then the hosts which do understand IP identifiers can ignore the routable locators in the AAAA RRset. The choice between those two, or other alternatives, doesn't materially affect how the rest of identifier/locator separation would work.

2.3. Identifier Lookup to find the Locators using the DNS

One can do different forms of lookup systems for the identifier to locator lookup. The least "novel" one would be to reuse the DNS. Since the identifier is drawn from the IPv6 address space, a possible way to do this in the DNS is to use the ip6.arpa tree for this, with its ability to delegate on nibble boundaries.

A simple way to do this would be to just have normal PTR records in the reverse tree that "point" to a fully-qualified domain name, and then do lookups for AAAA records matching that domain name. But this doesn't allow the destination domain to influence the locator selection as part of the lookup. Thus it might make sense to instead deploy DNS SRV records in the reverse zone since this would allow specifying a priority and a weight for primary/backup and load spreading, respectively.

2.4. Handling Application Referrals

A referral is the case when three or more instances of an application interacts and the first instance communicates with the second instance, and then communicates with the third instances and wishes to tell the third instance how to contact the second instance. There

are application patters, called callbacks and long-lived application associations in [27] that have similar issues. The application can handle these by using and passing the fully-qualified domain name, or it can do it by storing and passing the IP address. In the latter case, this application ends up depending on the IP address being universally useful as a way to contact another host.

Assuming we want to keep such applications functioning with the introduction of the identifier/locator split, it is necessary that the 128 bit identifier can be useful on a host which didn't do the DNS lookup and does not know the fully-qualified domain name of that peer.

The approach outlined in this section handles this case, since the identifier will be looked up in ip6.arpa to find the locators.

2.5. Walkthrough of an example

The following example shows how the above use of DNS interacts with shim6. This assumes that the destination host, www.example.com, has been allocated an IP identifier and that IP identifier has been entered as an ID RR in the DNS.

1. The application calls getaddrinfo() for www.example.com, and that performs a DNS lookup for an ID record for www.example.com. Since a record was found, it returns this as a 128-bit IPv6 address. (If no records was found, then getaddrinfo() would have looked for AAAA and A records.)

2. The application, without making any distinction between an IP identifier and a locator, passes this address to the connect() or sendto() calls, depending on which transport protocol it uses.

3. The transport protocol proceeds as normal, which includes picking a source address which "matches" the destination address. The RFC 3484 rules might very well be sufficient for this; alternatively the transport protocols would be modified to explicitly know that when the destination is an identifier (it has a well-known, KHI-like prefix) it must pick a source address which is also an identifier, and vice versa for a destination which is a locator.

4. A packet (for instance, a TCP SYN) from the transport protocol arrives at the shim6 shim layer on the host. The shim looks for a context which matches the ULID pair. Since we assume this is the first attempt to communicate with this ULID, there is no existing context.

5. The shim checks if the destination ULID is a locator or an identifier. If it is a locator it proceeds as specified in Shim6 [7] and none of the items below apply. If it is an identifier, then the shim needs to find the set of locators for the identifier.

6. The shim performs the identifier to locator lookup very similarly to normal IPv6 reverse lookups (form a query name based on the nibbles in reverse order and append ip6.arpa), but it queries for SRV records.

7. Assuming one or more SRV records are returned, the shim takes the priority and and weight into account as specified in [10] to order the locators.

8. The shim then forms a I1 packet as specified in [7] and includes a ULID pair option (to carry the ULIDs which are different than the locators).

9. The rest of the context establishment follows as in [7] with the ULID option. The presence of the ULID option implies that the peer will verify the locators using the CGA ULID properties when receiving the I2 message, and the host itself will verify its peer's locators the same way when receiving the R2 message.

The rest of shim6 remain unmodified, except that the Locator Update, Keepalive, and Probe messages should probably carry the ULID option as well. Alternatively, one could rely on the 47-bit context tag to identify the context.

Note that the above assumes that the site controls the DNS reverse tree for their identifier prefix. But there is no dependency on the reverse tree for the locator prefixes that the site is using.

In order to provide an IP identifier that is not a routable IP address, we would presumably need to provide host autoconfiguration capabilities for the identifier. This could be done as some minor extensions to DHCPv6 and Stateless Address Autoconfiguration, as long as the identifiers would be automatically registered in the DNS as part of such assignment.

2.6. Design Alternatives

If the identifiers are placed in the DNS using AAAA records, then the lookup for the AAAA record set (to find the identifier) might also return a list of locators. Such a list can potentially be useful to avoid the ip6.arpa lookup to find the locators. But relying on this means that the reverse lookup from the identifier will only be used

in uncommon cases such as:

- The shim6 context state having been garbage collected too early, and the upper-layer protocol sends down a packet with a destination ULID which is a non-routable identifier.

- Application callbacks, referrals, and long-lived application handles [27] that are IP addresses.

For this reason it makes sense to be more consistent and always rely on the reverse lookup when the context is established.

3. Traffic Engineering Support

The traffic engineering pieces that might be desirable and that are easy to implement in this model are outlined in this section. If the ULID is a routable locator, then it makes sense to recommend that applications use DNS SRV records for the initial (non-shimmed) contact, and also provide at least a DHCPv6 option by which a site administrator can control what each host in the site uses in the shim6 Locator Preference option.

In the case when the ULID is a non-routable identifier, a different set of mechanisms are desirable. Instead of using DNS SRV records for the lookup of the domain name of the peer, we want similar control when looking up an identifier to find the set of locators.

For both cases it has been argued that allowing the routers, in particular routers located close to the egress from a site, to rewrite the source locators, as a mechanism to indicate to the hosts a dynamic change in the preferred locators. We outline the mechanism for this below and we also discuss the policy issues which are introduced with this additional information from the routers.

Note that we do not currently have a set of agreed upon requirements for traffic engineering. The closest we have to requirements are Jason Schiller's slides at
http://www.iab.org/documents/open-mtgs/2005-10-23-schiller.pdf

3.1. Recommending use of DNS SRV

In shim6 as specified, the host rely on existing DNS mechanisms, such as AAAA records or any other mechanism, to find a list of locators to try. When AAAA records are used, there is no mechanism for the destination site to express any ranking for primary/fallback, or any mechanism to spread load across the paths that are represented by the locators, since the AAAA resource record set is treated as a set with no implied order.

If we could use DNS SRV records instead, then we would have the combination of the SRV priority (for primary/backup) and SRV weight (for specifying a weighted, random load spreading) as tools for the site administrator to use.

Unfortunately it is the application software developer which needs to explicitly decide to use SRV records, thus we can not require that shim6 implementations also have the applications use SRV, since the application developer is in most cases different than the implementor of the TCP/IP protocol stack. But it would be helpful to the IETF to make a strong recommendation that applications should use SRV

records.

Note that in some cases it quite complex to specify how an application uses SRV records. SIP is an example of this [13], which needs to deal with SIP running over multiple transport protocols, SRV vs. AAAA and A record ordering, and various other issues. But in common cases like http that run over a single transport, the needed change is just to precede the lookups for AAAA and A records at www.example.com with a lookup for SRV at _http._tcp.www.example.com.

3.2. DHCPv6 option for Locator Preferences

Shim6 [7] specifies a Locator Preference option which can carry some number of octets, with the protocol currently specifying the semantics of the first three octets of each preference element. One way to allow the site administrator to control this is to introduce a new DHCPv6 [14] option which carries the list of IPv6 locator prefixes that the site uses and the octets of shim6 preference element; this can be done without the host having to understand the semantics of the octets it is sending - it can just copy them from the DHCPv6 option, even though the host will interpret the locator preference octet string it receives from the peer.

To be concrete, we show an example of what such a DHCPv6 option can look like in Section 4.1.

3.3. Identifier to Locator lookup Preferences

As specified in Section 2.3 it makes sense to use SRV records for the reverse lookup used to go from an identifier to a set of locators, since this allows the site manager to provide priority and weight to the peer before the initial contact takes place, thus most likely the initial locator chosen by the host will be a reasonable one from the perspective of traffic engineering for the peer site.

3.4. Locator Rewriting by Routers

Shim6 as specified [7] allows routers to rewrite the locators on shim6 packets that are payload extension headers, but it does not allow such rewrites on the shim6 control messages. In this section we outline what it would take to allow such rewriting on all the shim6 messages, and also provide the host with an indication of how their locators are rewritten so that they can take this into account.

With these extensions the routers are free to rewrite the locators with alternate values in any IPv6 packet that has a shim6 header (i.e., where some next header value is IPPROTO_SHIM6).

The support for this rewriting consists of:

- o Introducing a new Sent Locator Pair option, which the receiver or a packet can use to see how the routers rewrote the locators in a packet sent towards it. The receiver also "echos" this information to its peer using the next option.

- o Introducing a new Received Locator Pair option, which is a echo of the content of a received Sent Locator Pair option.

- o The hosts SHOULD include a shim6 extension header for ULP payloads where the sending shim does not rewrite the locators, in order to allow the routers to rewrite the locators.

- o To maximize the utility of the locator rewriting, we should avoid the use of deferred context establishment. If the first packet sent is an I1 packet, and the above SHOULD is honored, then the routers are free to rewrite the locators in all the packets that are exchanged between a pair of shim6 hosts. (If we are also doing complete identifier/locator split, then there is no deferred context establishment, thus this doesn't add any additional costs.)

- o As in the current shim6 specification, any source and destination locator can be used on a shim6 payload message as long as the context tag matches. But in addition, when the receiver of a payload message observes a change in the locator pair on which it receives the payload extension headers, it SHOULD notify the peer of this by sending an Update Request message with a Received Locator Pair option. Such update messages MUST be rate limited since the locators can flap when routing is flapping.

3.5. Walkthrough using locator rewriting

The following example shows how the Sent Locator Pair and Received Locator Pair options are used as part of the Shim6 context establishment exchange, and indicates some of the policy choices that exist in terms of what locator is chosen. The same exchange applies whether the ULID is a reachable locator or not; when the ULID is an unreachable identifier then there would be an ULID option in most of the shim6 control messages, but we omit that aspect here as we focus on the locator exchanges.

We assume that host A, which locators A1, A2, and A3, is communicating with host B, with locators B1 and B2.

The walkthrough starts when A is sending an I1 message to B:

1. The shim then forms a I1 packet as specified in [7] and includes a Sent Locator Pair option. The shim has chosen A1 to be the source locator and B1 to be the destination locator, thus it places <A1, B1> in both IPv6 header (as source and destination), and in the Sent Locator Pair option.

2. The routers, for instance, the egress router from A's site, might rewrite the IPv6 source address field with a different prefix so that when the packet is receive by B it has <A2, B1> in the IPv6 header (and the Sent Locator Pair option is unmodified with <A1, B1>).

3. B processes the I1 message as specified in [7] to generate a R1 message. In addition, it copies the content of the Sent Locator Pair option into a Received Locator Pair option. Host B must decide whether it should send the R1 message to the IP source address of the R1 message, or send it to the potentially different Sender Locator in the Sent Locator Pair option in the I1 message. The safe thing would be to send it to the source address of the R1, because the alternative can be used for reflection attacks. Once B has made this decision, it puts the addresses, in this example <B1, A2> in the IPv6 header as well as into a Sent Locator Pair option.

4. Host A receives the R1 message and finds the context state. At this point in time A can see how the router rewrote the I1 message by comparing the Lp(local), Lp(peer) with the Received Locator Pair option. It can also see how the locators were rewritten for the return path by comparing the IPv6 header in the R1 message with the Sent Locator Pair option. It might be that the Received Locator Pair option has a Sender Locator which is not one of the locators that host A knows as its own. In this case it it might reasonable for A to treat this locator as its own, at least in terms of accepting packets destined to it. Host A forms a normal I2 message, but also includes a Received Locator Pair option in it. The host chooses the locator pair to use when sending the I2 message, and if the source address of the R1 message is one of B's locators, then this might be the best choice. Just as for the I1 message, the host adds a Sent Locator Pair option to the I2 message which contains a copy of the source and destination fields in the IPv6 header.

5. Host B receives the I2 message and finds the context state. At this point in time B can see how the routers on the return path rewrote the I2 message by comparing the IPv6 header in the I2 message with the Sent Locator Pair option. It might be that the Received Locator Pair option has a Sender Locator which is not one of the locators that host B knows as its own. In this case

it it might reasonable for B to treat this locator as its own, at least in terms of accepting packets destined to it. Host B forms a normal R2 message, but also includes a Received Locator Pair option in it. The host chooses the locator pair to use when sending the R2 message, and if the source address of the I2 message is one of A's locators (that it in the Locator List option), then this might be the best choice. Just as for the R1 message, the host adds a Sent Locator Pair option to the R2 message which contains a copy of the source and destination fields in the IPv6 header.

The host maintains Lp(local) and Lp(peer) to use as the preferred source and destination address when sending packets to the peer. But in addition the hosts main a Lr(local) ["r" for "received"] and Lr(peer) that is the locator pair in the most recently received packet for the context.

When a packet with a Payload Extension Header is received for the context and the source or destination address does not match Lr(peer)/Lr(local), then, subject to rate limiting, the host forms an Update Request message with a Received Locator Pair option. This Update Request message is transmitted and retransmitted as specified in [7]. This allows the peer to be notified should the routers change the way the rewrite the locators, which the host can use to make sure the packets have the correct locators as they are originated.

3.6. Need for Coordination?

It might be the case that the routers are not aware of the the actual prefixes assigned to each host. For instance, just because a site has 7 different prefixes, there might be some host that due to resource considerations or DHCP sever policy only configure addresses in a subset of these prefixes. Thus the routers might rewrite the source locator to be a locator which is not assigned to the source host. If host B were to respond to an I1 message that had been rewritten this way, then the R1 message would not be accepted by host A. For this reason it is recommended that the I1 and R1 messages include the Locator List option, and that the receiver of these messages reply using the Sender Locator in the the Sent Locator Pair option, in the case when the IPv6 source address is not part of the Locator List option.

Also, it is recommended that hosts "learn" new locators from the Received Locator Pair option, so that they will in the future accept packets destined to the locators that they routers use when rewriting, even if the hosts aren't otherwise configured with those addresses. Note that for shim6 to be able to prove to the peer that

the host in fact "owns" such a locator, the host must have a CGA ULID and sign an Update Request message with the new locator set. Whether it is a good idea to always "adopt" locators from the Received Locator Pair option is TBD.

4. New Option Formats

4.1. New DHCPv6 Option

This option follows the DHCPv6 [14] format for options and allows the DHCP server to specify what the host will put in the Shim6 Locator Preference option. The DHCPv6 option carries IPv6 address prefixes, and the host will apply the logest matching such prefix to each of its IPv6 locators.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          option-code          |           option-len          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Prefix Length1|                                               |
+-+-+-+-+-+-+-+-+         Prefix 1                              |
|                                                               |
|                                               +-+-+-+-+-+-+-+-+
|                                               | Elem len 1    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Element 1 (len octets)       | Prefix Length2|               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+               |
|                      Prefix 2                                 |
|                                                               |
|                                               +-+-+-+-+-+-+-+-+
|                                               | Elem len 2    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Element 2 (len octets)       |                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                               |
|                              ...                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Fields:

option-code: 16-bit unsigned integer. To be allocated by the IANA.

option-len: 16-bit unsigned integer. The length in octets of all the fields in the option which follow the option-len field.

Prefix Length[n]: 8-bit unsigned integer. The number of relevant bits in Prefix[n].

Prefix[n]:   An IPv6 address prefix. The length of this field is the number of octets necessary to carry Prefix Length[2] bits. For instance, a 47-bit Prefix Length would mean that the Prefix field is 6 octets.

Element Len[n]: 8-bit unsigned integer. The number of octets of Element[n] that follows the Element Len field.

Element[n]: One or more octets that can be directly copied to the Shim6 Locator Prefix option's Element field.

4.2. New Shim6 Options

This document defines two new Shim6 options.

Type	Option Name
13	Sent Locator Pair
14	Received Locator Pair

Table 1

4.2.1. Sent Locator Pair Option Format

This option carries the set source and destination locators as sent by the sender, that is, the sender sets them to the same values as the IPv6 source and destination fields. When router rewriting is in use, then the routers might change the IPv6 source and destination fields, this field allows the receiver to observe how the locators were rewritten, and also "echo" that back to the peer. This option SHOULD be included in all the Shim6 control messages.

```
 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 = 13         |0|         Length = 36           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Reserved2                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                           Sender Locator                      +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                          Receiver Locator                     +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Fields:

Reserved2: 32-bit field. Reserved for future use. Zero on transmit. MUST be ignored on receipt. (Needed to make the ULIDs start on a multiple of 8 octet boundary.)

Sender Locator: A 128-bit IPv6 address.

Receiver Locator: A 128-bit IPv6 address.

4.2.2. Received Locator Pair Option Format

When a host receives a shim6 control message, such as an I1 or Update Request message, from its peer and will send a response message, then if the "request" contained a Sent Locator Pair option, the host will "echo" that option in unchanged but with the option type changed to be a Received Locator Pair option. This allows the sending host to see how the router rewrote its locators, which is useful information for its locator selection.

The control messages that carry a Received Locator Pair option should also carry a Sent Locator Pair option, to allow the discovery of the rewriting that happens in the reverse direction.

```
 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 = 14         |0|        Length = 36            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Reserved2                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                          Sender Locator                       +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                         Receiver Locator                      +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Fields:

Reserved2: 32-bit field. Reserved for future use. Zero on transmit. MUST be ignored on receipt. (Needed to make the ULIDs start on a multiple of 8 octet boundary.)

Sender Locator: A 128-bit IPv6 address.

Receiver Locator: A 128-bit IPv6 address.

5. Security Considerations

This document isn't known to introduce any new security considerations other than those listed for Shim6 [7]. Since Shim6 satisfies the concerns specified in [22] that is likely to be sufficient.

The suggest looking of identifiers to find the set of locators in this document uses DNS. If DNSsec is not applied to the part of the ip6.arpa tree where the IP identifiers are placed, then an attacker could spoof the set of locators. This would result in a form of DoS attack and not a redirection attack, since a host would verify the CGA property when it receives the R2 message, and only the host which "owns" the CGA-based ULID would be able to correctly sign the R2 message.

Of course, the DNS lookups of www.example.com to find the identifier is still subject to DNS spoofing attacks, including redirection attacks to a different ULID.

6. Acknowledgements

Over the years many people active in the multi6 and shim6 WGs have contributed ideas a suggestions that are reflected in this specification. The original ideas that has stimulated this work was Mike O'Dell's GSE proposal.

7. References

7.1. Normative References

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

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

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

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

[5] Conta, A. and S. Deering, "Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification", RFC 2463, December 1998.

[6] Aura, T., "Cryptographically Generated Addresses (CGA)", RFC 3972, March 2005.

[7] Nordmark, E. and M. Bagnulo, "Shim6: Level 3 Multihoming Shim Protocol for IPv6", draft-ietf-shim6-proto-10 (work in progress), February 2008.

[8] Bagnulo, M., "Hash Based Addresses (HBA)", draft-ietf-shim6-hba-05 (work in progress), December 2007.

[9] Arkko, J. and I. Beijnum, "Failure Detection and Locator Pair Exploration Protocol for IPv6 Multihoming", draft-ietf-shim6-failure-detection-11 (work in progress), February 2008.

7.2. Informative References

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

[11] Ferguson, P. and D. Senie, "Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing", BCP 38, RFC 2827, May 2000.

[12] Narten, T. and R. Draves, "Privacy Extensions for Stateless Address Autoconfiguration in IPv6", RFC 3041, January 2001.

[13] Rosenberg, J. and H. Schulzrinne, "Session Initiation Protocol (SIP): Locating SIP Servers", RFC 3263, June 2002.

[14] Droms, R., Bound, J., Volz, B., Lemon, T., Perkins, C., and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3315, July 2003.

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

[16] Bagnulo, M., "Updating RFC 3484 for multihoming support", draft-bagnulo-ipv6-rfc3484-update-00 (work in progress), December 2005.

[17] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, July 2003.

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

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

[20] Eastlake, D., Schiller, J., and S. Crocker, "Randomness Requirements for Security", BCP 106, RFC 4086, June 2005.

[21] Hinden, R. and B. Haberman, "Unique Local IPv6 Unicast Addresses", RFC 4193, October 2005.

[22] Nordmark, E. and T. Li, "Threats Relating to IPv6 Multihoming Solutions", RFC 4218, October 2005.

[23] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, February 2006.

[24] Nikander, P., "An IPv6 Prefix for Overlay Routable Cryptographic Hash Identifiers (ORCHID)", draft-laganier-ipv6-khi-07 (work in progress), February 2007.

[25] Huitema, C., "Ingress filtering compatibility for IPv6 multihomed sites", draft-huitema-shim6-ingress-filtering-00 (work in progress), September 2005.

[26] Bagnulo, M. and E. Nordmark, "SHIM - MIPv6 Interaction", draft-bagnulo-shim6-mip-00 (work in progress), July 2005.

[27] Nordmark, E., "Shim6 Application Referral Issues",

draft-ietf-shim6-app-refer-00 (work in progress), July 2005.

[28] Bagnulo, M. and J. Abley, "Applicability Statement for the Level 3 Multihoming Shim Protocol (Shim6)", draft-ietf-shim6-applicability-03 (work in progress), July 2007.

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

[30] Moskowitz, R., Nikander, P., Jokela, P., and T. Henderson, "Host Identity Protocol", draft-ietf-hip-base-10 (work in progress), October 2007.

[31] Eronen, P., "IKEv2 Mobility and Multihoming Protocol (MOBIKE)", draft-ietf-mobike-protocol-08 (work in progress), February 2006.

Author's Address

Erik Nordmark
Sun Microsystems
17 Network Circle
Menlo Park, CA 94025
USA

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

Full Copyright Statement

Intellectual Property

Acknowledgment

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).