INTERNET-DRAFT Erik Nordmark
July 9, 2005 Sun Microsystems
Marcelo Bagnulo
UC3M

Multihoming L3 Shim Approach

<draft-ietf-shim6-l3shim-00.txt>

Status of this Memo

This Internet Draft expires January 9, 2006.

Abstract

This document specifies a particular approach to IPv6 multihoming. The approach is based on using a multihoming shim placed between the IP endpoint sublayer and the IP routing sublayer, and, at least initially, using routable IP locators as the identifiers visible above the shim layer. The approach does not introduce a "stack name" type of identifier, instead it ensures that all upper layer protocols can operate unmodified in a multihomed setting while still seeing a stable IPv6 address.

This document does not specify the mechanism for authenticating and authorizing the "rehoming" - this is specified in the HBA document.

Nor does it specify the messages used to establish multihoming state.

The document does not even specify the packet format used for the data packets. Instead it discusses the issue of receive side demultiplexing and the different tradeoffs. The resolution of this issue will affect the packet format for the data packets.

Contents

1. Introduction

The goal of the IPv6 multihoming work is to allow a site to take advantage of multiple attachments to the global Internet without having a specific entry for the site visible in the global routing table. Specifically, a solution should allow users to use multiple attachments in parallel, or to switch between these attachment points dynamically in the case of failures, without an impact on the upper layer protocols.

The goals for this approach is to:

- Preserve established communications through failures, for example, TCP connections and application communications using UDP.

- Have no impact on upper layer protocols in general and on transport protocols in particular.

- Address the security threats in [M6THREATS] through a separate document [HBA]

- No extra roundtrip for setup; deferred setup.

- Take advantage of multiple locators/addresses for load spreading so that different sets of communication to a host (e.g., different connections) might use different locators of the host.

1.1. Non-Goals

The assumption is that the problem we are trying to solve is site multihoming, with the ability to have the set of site locator prefixes change over time due to site renumbering. Further, we assume that such changes to the set of locator prefixes can be relatively slow and managed; slow enough to allow updates to the DNS to propagate. But it is not a goal to try to make communication survive a renumbering event (which causes all the locators of a host to change to a new set of locators). This proposal does not attempt to solve, perhaps related, problems such as host multihoming or host

mobility.

This proposal also does not try to provide an IP identifier. Even though such a concept would be useful to ULPs and applications, especially if the management burden for such a name space was zero and there was an efficient yet secure mechanism to map from identifiers to locators, such a name space isn't necessary (and furthermore doesn't seem to help) to solve the multihoming problem.

1.2. Assumptions

This approach assumes that packets with arbitrary combinations of source and destination locators will make it from end to end unless there is some form of failure. Due to the interaction between ingress filtering [RFC2827] and source address selection, this assumption might not be true in IPv6 today. As a result there is a need to work out a solution that doesn't make the ingress filtering in ISPs drop more packets than needed.

Furthermore, a solution following the outline in this document and companion shim6 WG documents will most likely end up assuming that a host can affect the exit path from the site. In particular, should the host send packets with all combinations of its addresses in the IP source address field and all of the peers addresses in the destination field, that with high probability that set of packets ends up using multiple exits from the site (going via different ISPs), when the site is connected to multiple ISPs. (Of course, the network admin might have disabled this somehow, but in order for the multihoming solution to find a working path it needs to have the ability to explore different exits somehow.)

Some solutions to this have been proposed in [INGRESS].

2. Terminology

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 [NSRG]). 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.

2.1. 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.

3. Overview

This document specifies certain aspects of the approach, yet leaves other aspects open.

The main points are about using locators as the ULIDs, the exact placement of the multihoming shim in the protocol stack, and an outline of a protocol for a protocol for establishing and and managing the necessary state in the multihoming shim.

The draft also discusses issues about receive side demultiplexing, which affects the packet format for data packets.

The approach assumes that there are mechanisms (specified in other drafts) which:

- can prevent redirection attacks [HBA]

- can prevent 3rd party DoS attacks [HBA, M6FUNC]

- can detect whether or not a peer supports the shim6 protocol [M6FUNC, M6DET]

- can explore all the locator pairs to find a working pair when the initial pair does not work [M6DET]

4. Locators as Upper-layer Identifiers

Central to this approach is to not introduce a new identifier name space but instead use one of the locators as the upper-layer ID, while allowing the locators used in the address fields to change over time in response to failures of using the original locator.

This implies that the ULID selection is performed as today's default address selection as specified in [RFC 3484]. Underneath, and transparently, the multihoming shim selects working locator pairs with the initial locator pair being the ULID pair. When communication fails the shim can test and select alternate locators. A subsequent section discusses the issues when the selected ULID is not initially working hence there is a need to switch locators up front.

Using one of the locators as the ULID has certain benefits for applications which have long-lived session state, or performs callbacks or referrals, because both the FQDN and the 128-bit ULID work as handles for the applications. However, using a single 128-bit ULID doesn't provide seamless communication when that locator is unreachable. See [M6REFER] for further discussion of the application implications.

There has been some discussion of using non-routable locators, such as unique-local addresses [ULA], as ULIDs in a multihoming solution. While this document doesn't specify all aspects of this, it is believed that the approach can be extended to handle such a case. For example, the protocol already needs to handle ULIDs that are not initially reachable. Thus the same mechanism can handle ULIDs that are permanently unreachable from outside their site. The issue becomes how to make the protocol perform well when the ULID is not reachable, for instance, avoiding any timeout and retries in this case. In addition one would need to understand how the ULAs would be entered in the DNS to avoid a performance impact on existing, non-shim6 aware, IPv6 hosts potentially trying to communicate to the (unreachable) ULA.

4.1. IP Multicast

IP Multicast requires that the IP source address field contain a topologically correct locator for interface that is used to send the packet, since IP multicast routing uses both the source address and the destination group to determine where to forward the packet. (This isn't much different than the situation with widely implemented ingress filtering [RFC2827] for unicast.)

While in theory it would be possible to apply the shim re-mapping of the IP address fields between ULIDs and locators, the fact that all

the multicast receivers would need to know the mapping to perform, makes such an approach difficult in practice. Thus it makes sense to have multicast ULPs operate directly on locators and not use the shim. This is quite a natural fit for protocols which use RTP [RFC3550], since RTP already has an explicit identifier in the form of the SSRC field in the RTP headers. Thus the actual IP address fields are not important to the application.

4.2. Renumbering Implications

As stated above, this approach does not try to make communication survive renumbering. However, the fact that a ULID might be used with a different locator over time open up the possibility that communication between two ULIDs might continue to work after one or both of those ULIDs are no longer reachable as locators, for example due to a renumbering event. This opens up the possibility that the ULID (or at least the prefix on which it is based) is reassigned to another site while it is still being used (with another locator) for existing communication.

Worst case we could end up with two separate hosts using the same ULID while both of them are communicating with the same host.

This potential source for confusion can be avoided if we require that any communication using a ULID must be terminated when the ULID becomes invalid (due to the underlying prefix becoming invalid).

However, this might be an overkill. Even when an IPv6 prefix is retired and reassigned to some other site, there is still a very small probability that another host in that site picks the same 128 bit address (whether using DHCPv6, stateless address autoconfiguration, or picking a random interface ID [RFC 3041]). Should the identical address be used by another host, then there still wouldn't be a problem until that host attempts to communicate with the same host with which the initial user of the IPv6 address was communicating.

5. Placement of the multihoming shim

```
                            -----------------------
                            | Transport Protocols |
                            -----------------------

             ------ ------- -------------- -------------     IP endpoint
             | AH | | ESP | | Frag/reass | | Dest opts |     sub-layer
             ------ ------- -------------- -------------

                         ---------------------
                         | shim6 shim layer |
                         ---------------------

                                ------                      IP routing
                                | IP |                      sub-layer
                                ------
```

Figure 1: Protocol stack

The proposal uses an multihoming shim layer within the IP layer, i.e., below the ULPs, as shown in figure 1, in order to provide ULP independence. Conceptually the multihoming shim layer behaves as if it is associated with an extension header, which would be ordered immediately after any hop-by-hop options in the packet. However, the amount of data that needs to be carried in an actual shim6 extension header is close to zero, thus it might not be necessary to add bytes to each packet. See section 9.

We refer to packets that at least conceptually have this extension header, i.e., packets that should be processed by the multihoming shim on the receiver, as "shim6 packets" (analogous to "ESP packets" or "TCP packets").

Layering AH and ESP above the multihoming shim means that IPsec can be made to be unaware of locator changes the same way that transport protocols can be unaware. Thus the IPsec security associations remain stable even though the locators are changing. The MOBIKE WG is looking at ways to have IPsec security associations survive even though the IP addresses changes, which is a different approach.

Layering the fragmentation header above the multihoming shim makes reassembly robust in the case that there is broken multi-path routing which results in using different paths, hence potentially different source locators, for different fragments. Thus, effectively the multihoming shim layer is placed between the IP endpoint sublayer, which handles fragmentation, reassembly, and IPsec, and the IP routing sublayer, which on a host selects which default router to use etc.

Applications and upper layer protocols use ULIDs which the shim6 layer will map to/from different locators. The shim6 layer maintains state, called host-pair context, per ULID pairs (that is, applies to all ULP connections between the ULID pair) in order to perform this mapping. The mapping is performed consistently at the sender and the receiver, thus from the perspective of the upper layer protocols, packets appear to be sent using ULIDs from end to end, even though the packets travel through the network containing locators in the IP address fields, and even though those locators might be changed by the transmitting shim6 layer.

The context state in this approach is maintained per remote ULID i.e. approximately per peer host, and not at any finer granularity. In particular, it is independent of the ULPs and any ULP connections. When two hosts are communicating with each other using multiple different ULID pairs, there is an option to "merge" the context state for the two (or more) ULID pairs. Doing so would mean that the protocol to test and select working locators after a failure can be shared across the multiple ULID pairs between the two hosts. However, if it will be uncommon that two hosts communicate using multiple ULID pairs, the added complexity of merging the state might not be worth while. Thus this is for further study.

```
   ----------------------------          ----------------------------
   | Sender A                 |          | Receiver B               |
   |                          |          |                          |
   |     ULP                  |          |     ULP                  |
   |      | src ULID(A)=L1(A) |          |      ^                   |
   |      | dst ULID(B)=L1(B) |          |      | src ULID(A)=L1(A) |
   |      v                   |          |      | dst ULID(B)=L1(B) |
   |   multihoming shim       |          |   multihoming shim       |
   |      | src L2(A)         |          |      ^                   |
   |      | dst L3(B)         |          |      | src L2(A)         |
   |      v                   |          |      | dst L3(B)         |
   |      IP                  |          |      IP                  |
   ----------------------------          ----------------------------
          |                                     ^
          ------- cloud with some routers -------
```

Figure 2: Mapping with changed locators.

The result of this consistent mapping is that there is no impact on the ULPs. In particular, there is no impact on pseudo-header checksums and connection identification.

Conceptually one could view this approach as if both ULIDs and locators are being present in every packet, but with a header compression mechanism applied that removes the need for the ULIDs

once the state has been established. In order for the receiver to recreate a packet with the correct ULIDs there might be a need to include some "compression tag" in the data packets. This would serve to indicate the correct context to use for decompression when the locator pair in the packet is insufficient to uniquely identify the context.

5.1. Shim Implications on Flow Label Usage

The use of a shim has implications on a class of protocols which have host as well as router functions. This section discusses the implications for the flow label field, which might be used for QoS handling where the hosts set the flow label and initiate some flow signaling protocols, and the routers participate in the flow signaling protocol and as a result perform different action on packets based on the flow (which is identified by the IP address fields and the flow label field).

The shim will leave the flow label unmodified. This means that the {Flow Label, Source ULID, Dest ULID} that the upper-layer protocol sends will appear on the wire as the set of {Flow Label, Source Locator, Destination Locator} for the different locators.

This does have implications for protocols which do explicit signaling to create flow state; such protocols would somehow need to be made shim6 aware so that they can perform the signaling for all the tuples of {Flow Label, Source Locator, Destination Locator} that are used on the wire. Note that this need to modify such signaling protocols would apply even if the flows were identified without the use of the flow label field, due to the different locators which might be used.

5.2. Shim Implications on ICMP errors

Another protocol which requires consistency between the upper layer protocols on the hosts and the routers are the ICMP errors. The routers (and in some cases the peer host) will generate ICMP errors based on the locators contained in the IP address fields, but when the host implementation delivers a notification to the ULP that an ICMP error was received, the ULP instance needs to be discovered based on the ULIDs. This means that for ICMP error reception the host needs to be able to take the initial part of a shim6 packet (the part returned in an ICMP error) and do the inverse translation of the IP address fields that it did when sending the packet, and then deliver the notification to the ULP.

Being able to demultiplex based on the information returned in an

ICMP error presumably has some implications on the packet format and mechanism for receive-side demultiplexing.

5.3. Other Shim Protocol Implications

As stated above, there might be other protocols in addition to flow management and ICMP errors that assume that the ULPs at the hosts and the IP address fields seen by the routers are the same.

Any such protocol is potentially impacted by the introduction of the multihoming shim.

5.4. MTU Implications

Depending on how demultiplexing is handled at the receiver (see subsequent sections) there may or may not be a need for the shim to add an extension header to the packets. In some cases such an extension header would only need to be added to some and not all data packets.

This has implications on the MTU that is available to the upper layer protocols hence will require some extra handling in the host implementation. At some level this is similar to the MTU implementations of IPsec, in that the IP layer would add bytes to some ULP packets and not others, but in the case of IPsec one would expect all or no packets in a particular ULP connection to be affected, whereas in shim6 some packets, such as those sent after a locator failure, would be subject to a reduction in the available MTU.

The MTU can also change because after a rehoming event a different path (with a different MTU) is being used to exchange packets between the peers. So, after a rehoming event, the need to include an additional extension header, and the usage of a different path affect the MTU. However, it should be noted that the shim6 is aware of the path change and that the MTU discovery techniques already have a mechanism to cope with changes in the path MTU when the path changes.

While the effects of these are local to the host implementation they are likely to be a bit complicated. There needs to be a mechanism so that the ULP can be notified when the available MTU changes due to an extension header either being added, or no longer being added. Also, ICMPv6 packet too big errors need to result in a notification to the ULP which takes into account whether or not an extension header is being added. Finally, certain ULPs such as UDP might need to rely on IP fragmentation down to the available MTU, while other ULPs such as

TCP will adapt their segment size to the available MTU.

6. Deferred Context Establishment

The protocol will use some context establishment exchange in order to setup shim6 state at the two endpoints. Similar to MAST [MAST] this initial exchange can be performed asynchronously with data packets flowing between the two hosts; until context state has been established at both ends the packets would flow just as for unmodified IPv6 hosts i.e., without the ability for the hosts to switch locators. This approach allows the hosts to have some local policy on when to attempt to establish shim6 state with a peer; perhaps based on the transport protocols and port numbers, or perhaps based on the number of packets that have flowed to/from the peer.

Once the initial exchange has completed there is host-pair context state at both hosts, and both ends know a set of locators for the peer that are acceptable as the source in received packets. This will trigger some verification of the set of locators, which is the subject of the security scheme.

7. Assumptions about the DNS

This approach assumes that hosts in multihomed sites have multiple AAAA records under a single name, in order to allow initial communication to try all the locators. For shim6 capable hosts, the content of those records are the locators (which also serve as ULIDs).

However, the approach does not assume that all the AAAA records for a given name refer to the same host. For instance, the AAAA records could point at different hosts providing the same service. This is handled by the context establishment mechanism allowing each host to pass its locators to the peer. This set contains a non-null subset of the locators presented in the AAAA record set and it may contain additional locators, not contained in the AAAA record set

The approach makes no assumption about the reverse tree since the approach does not use it. However, applications might rely on the reverse tree whether shim6 is used or not.

When the reverse DNS tree is populated so that any locator for the host can be used to find a FQDN and that FQDN can be used to find all the locators of the host, then this property can be used by the

applications when doing referals by the application only passing one locator to the peer and having the peer application do the DNS lookups to find the other locators. However, such a mechanism does not work when the reverse DNS is not setup to satisfy this property. Suggestions for how to handle the general case of application referals is captured in [APP-REFER]

Note that in some cases, even when the reverse DNS is populated, it can be hard to satisfy the above property. For instance, a home site using two "consumer" ISPs each providing DNS service including reverse DNS. In this case there might be no way for the site to control what goes in the DNS for the reverse tree.

7.1. DNS and Unique-local Addresses

Earlier we've mentioned that the protocol might provide the basic mechanism to use Unique-local addresses [ULA] as ULIDs.

In the cases where hosts have been assigned centrally assigned ULAs [ULA-CENTRAL], one can potentially take advantage of this to provide better support for applications. With centrally assigned ULAs it is possible to register them in the reverse DNS tree. As a result, one could use the DNS not only for applications which care about reverse and forward tree being consistent, but also to find the full set of locators from the ULID, the same way as outlined in the previous section. But again, this doesn't handle the general case of application referals.

For ULAs that are not centrally allocated it is not likely to be possible to register them in the global DNS, thus this possibility does not exist.

8. Protocol Walkthrough

8.1. Initial Context Establishment

Here is the sequence of events when A starts talking to B:

1. A looks up FQDN(B) in the DNS which returns a locator set which includes some locators for B. (The set could include locators for other hosts since e.g., www.example.com might include AAAA records for multiple hosts.) The default address selection [RFC3484] would make that set into an ordered list. The application would typically try to connect using the first locator in the list i.e., ULID(B) = L1(B). The application is prepared to try the other locators should the first one fail.

2. The ULP creates "connection" state between ULID(A)=L1(A) and ULID(B) and sends the first packet down to the IP/shim6 shim layer on A. L1(A) was picked using regular source address selection mechanisms [RFC3484]. Note that should communication fail using the initial locator/ULID pair, there has to be a mechanism to retry with both different destination locators and different source locators.

3. The packet passes through the shim6 layer, which has no state for ULID(B). A local policy will be used to determine when, if at all, to attempt to setup shim6 state with the peer. Until this state triggers packets pass back and forth between A and B as they do in unmodified IPv6 today.

   When the policy is triggered, which could be on either A or B, an initial context establishment takes place. This exchange will fail if the peer does not support the shim6 protocol. If it succeeds it results in both ends receiving the locator sets from their respective peer, and the security mechanism provides some way to verify these sets.

   At this point in time it is possible for the hosts to change to a different locator in the set. But until they have exchanged the locator sets, and probably until they rehome the context to use different locators, they continue sending and receiving IPv6 packets as before.

   As long as both hosts have been informed of the state at the peer i.e., know the locators of the peer and know that the peer has received its locators, each host can make an independent decision when it sees a need to change either the source or destination locator in the packets it is sending. Thus the approach does not require coordinating the actual locator changes between the peers.

8.2. Locator Change

When a host detects that communication is no longer working it can try to switch to a different locator pair. A host might suspect that communication isn't working due to

- lack of positive advice from the ULP (akin to the NUD advice in [RFC2461]

- negative advice from the ULP

- failure of some explicit shim6 "heartbeat" messages

- local indications such as the local locator becoming invalid [RFC2462] or the interface being disabled

Given that each host knows the locator set for its peer, the host can just switch to using a different locator pair. It might make sense for the host to test the locator pair before using it for ULP traffic, both to verify that the locator pair is working and to verify that it is indeed the peer that is present at the other end; the latter to prevent 3rd party DoS attacks. Such testing needs to complete before using the locator as a destination in order to prevent 3rd party DoS attacks [M6THREATS].

8.3. Concurrent Context Establishment

Should both A and B attempt to contact each other at about the same time using the same ULIDs for each other, the context establishment should create a single host-pair context. The NOID draft [NOID] contains a proof-of-concept that a 4-way context establishment exchange can ensure that a single context is created in this case.

However, if different ULIDs are used this would result in two completely independent contexts between the two hosts following the basic content establishment above; the context is per ULID pair. As noted above, in this case it might be desirable to "merge" i.e., share certain information, such as the reachability of different locator pairs, across the different ULID pairs that are between the same pair of hosts.

8.4. Handling Initial Locator Failures

Should not all locators be working when the communication is initiated some extra complexity arises, because the ULP has already been told which ULIDs to use. If the locators that were selected to be ULIDs are not working and the shim6 layer does not know of alternate locators, it has no other choice than to have the application try a different ULID. Note that the mechanism need to be able to try both different source ULIDs as well as different destination ULIDs, to make sure all combinations are explored.

Thus the simplest approach is to always punt initial locator failures up the stack to the application. However, this might imply significant delays while transport protocol times out. Also, the current default address selection mechanism [RFC3484] doesn't have a mechanism to try different source addresses for a single destination address; it can only cycle through different destination addresses with each destination address being used with a single source

address.

It is possible to optimize initial locator failures when the shim6 layer already has alternate locators for the peer. This might be the case when the two hosts have already communicated, or it might be possible to have the DNS resolver library provide alternate locators to the shim in the speculation that they might be useful. Such an optimization must not assume that the AAAA records refer to the same host, since it isn't uncommon that a FQDN have multiple AAAA records for the same *service* but for different hosts. For instance, the protocol would need to verify with the peer that the ULID in question is in fact assigned to the peer in this case. Potentially the trust level for the different locators retrieved from the DNS in this case, as opposed to retrieving the ULIDs from the DNS and the locators from the peer itself using the multihoming protocol, might be different. Note however, that this is an optimization and is not required for the protocol to work.

Should the shim6 layer know alternate locators for the peer, it needs to perform the shim6 protocol before upper layer protocol packets are exchanged. This means that the context establishment can not be deferred, and that there is a rehoming event, with the necessary security checks, before the first ULP packets can be successfully exchanged.

9. Demultiplexing of data packets in shim6 communications

The mechanisms for preserving established communications through outages that reside in the multihoming shim layer manage the multiple locators available in the multihomed node so that a reachable locator is used in the communication. Since reachability may vary during the communication lifetime, different locators may have to be used in order to keep packets flowing. However, the locators presented by the shim layer to the upper layer protocols must remain constant through the locator changes, so that received packets are recognized by the upper layer protocols as belonging to the established communication. In other words, in order to preserve established communications through outages, the shim layer will use different locators for exchanging packets while presenting the same ULIDs to the upper layer protocols. This means that upon the reception of an incoming packet with a pair of locators, the shim layer will need to translate the received locators to the ULIDs that are being used by the upper layer protocols in the particular communication. This operation is called demultiplexing.

For example, if a host has address A1 and A2 and starts communicating

with a peer with addresses B1 and B2, then some communication (connections) might use the pair <A1, B1> as ULID and others might use e.g., <A2, B2>. Initially there are no failures so these address pairs are used as locators i.e. in the IP address fields in the packets on the wire. But when there is a failure the shim6 layer on A might decide to send packets that used <A1, B1> as ULIDs using <A2, B2> as the locators. In this case B needs to be able to rewrite the IP address field for some packets and not others, but the packets all have the same locator pair.

Either we must prevent this from happening, or provide some additional information to B so that it can tell which packets need to have the IP address fields rewritten.

In this section, we will analyze different approaches to perform the demultiplexing operation. The possible approaches can be classified into two categories: First, the approaches that prevent the existence of ambiguities on the demultiplexing operation i.e. each received locator corresponds to one and only one ULID. Second, the approaches that use a context tag to provide additional information to the receiver that indicates the ULIDs that correspond to the locators contained in the packets.

Note that the sender also needs to be able to demultiplex ICMP errors as noted in Section 5.2, however the analysis below does not take that added constraint into account.

9.1. Approaches preventing the existence of ambiguities

One could think this problem can be avoided if the host never used the same locators for different ULIDs when communicating with the same peer host. However, the host can't tell a priori whether two peers share an IP stack. For instance, if A connects to www.foo.com with AAAA=B1 and www.bar.com with AAAA=B2 it can't tell whether B1 or B2 are assigned to the same IP stack or not until it communicates with B1 and B2 and retrieves there complete locator sets. (And even this might not suffice, since the peer might want to preserve the illusion of being two different hosts by returning B3 as an alternative locator for B1 and B4 as an alternate for B2.)

9.1.1. Pre-agreed identifiers

The simplest approach of this type is to designate one of the available addresses as the ULID to be used for all the communications while the remaining addresses will only be used as locators. This means that the upper layer protocols will only be aware of a single address, the one used as ULID, and all the remaining addresses that

are used as locators will remain invisible to them. Consequently, only the ULID can be returned by the resolver to the applications. The addresses used as locators cannot be returned to the applications by the resolver. So, if no additional information about the role of the addresses is placed in the DNS, only the ULID can be published in the DNS. This configuration has reduced fault tolerance capabilities during the initial contact, since the initiator will have only one address available to reach the receiver. If the ULID placed in the DNS is not reachable, the communication will fail. It would be possible to overcome this limitation by defining a new DNS record for storing information about addresses that can be only used as locators. If such record is defined, the initiator can use an alternative locator, even for initial contact, while still presenting the ULID to the upper layer protocols. However, this approach requires support from the initiator node, implying that only upgraded nodes will obtain improved fault tolerance while legacy nodes that don't support the new DNS record will still obtain reduced fault tolerance capabilities.

9.1.2. N-square addresses

In order to overcome the limitations presented by the previous scheme, it is possible to create additional addresses that have a pre-determined role. In this approach, each multihomed node that has n prefixes available, will create n^2 addresses, or in other words, the node will have n sets of n addresses each. Each set will contain one address per prefix. So, in each set, one address will be designated as ULID while the remaining addresses will be designated as locators. The ULIDs will have different prefixes in the different sets. The result is that there will be n ULIDs, one per available prefix, and each ULID will have an associated set of n-1 addresses that can only be used as locators. The ULIDs will be published in the DNS while the addresses usable only as locators must not be AAAA records in the DNS to prevent them from ever being used as ULIDs. The applications and default address selection [RFC3484] will only have knowledge of the ULIDs, and only the shim layer will deal with locators. The resulting configuration has full fault tolerance capabilities since n addresses (one per prefix) will be published in the DNS, allowing the usage of different addresses to make the initial contact.

9.2. Providing additional information to the receiver

When two nodes establish a shim6 enabled communication, a context is created at the shim layers of each node. The context stores information about the ULIDs and also about the locator set available for each node. In this approach, data packets carry a context tag

that allows the receiver determine which is the context that has to be used to perform the demultiplexing operation. There are several ways to carry the context tag within the data packets. In this section we will explore the following options: the Flow Label, and an Extension Header.

9.2.1. Flow-label

A possible approach is to carry the context tag in the Flow Label field of the IPv6 header. This means that when a shim6 context is established, a Flow Label value is associated with this context (and perhaps a separate flow label for each direction).

The simplest approach that does this is to have the triple <Flow Label, Source Locator, Destination Locator> identify the context at the receiver.

The sender and receiver needs to agree to allocate the flow labels so that each context between a pair of IP stacks receives a different flow label. While this might seem simple at first sight, the possibilities that different ULIDs refer to the same IP stack, and even that different FQDNs refer to the same IP stack, severely constrains how flow labels can be allocated. For instance, when communication is initiated from a host H to both foo.example (with ULIDs A and B) and bar.example (with ULIDs C and D), then host H might think it is communicating to two different IP stacks, when in fact they might be the same IP stack. Later when the locator sets are updated, for instances, after some failure, it might be told that ULID A has locators A, B, C, and D. Hence the two communications would need to have separate flow labels for the packets sent from H.

A protocol can handle this either by having H (in this example) allocate the flow label for the packets it is sending, or having the intended receiver allocate them. In the former case, H would need to allocate different flow labels for the different ULID pairs, since it doesn't know which peers are the same IP stack or not. In the latter case, the intended receiver would pick a flow label which is unique i.e., it can disambiguate the two contexts in this case. This implies that the flow label will not be assigned until the multihoming protocol has established the context state.

A modification to this suggested on the mailing list, which does not need establish the context state before the flow label is assigned, is to start the communication with unmodified flow label usage (could be zero, or as suggested in [RFC 3697]). The packets sent after a failure when a different locator pair is used would use a completely different flow label, and this flow label can be allocated as part of the shim context establishment. Since it is allocated during the

context establishment, the receiver of the "failed over" packets can pick a flow label of its choosing, without any performance impact, and respecting that for each locator pair, the flow label value used for a given locator pair doesn't change due to the operation of the multihoming shim.

A, perhaps minor, limitation imposed by overloading the flow label all the potential source and destination locators have to be known beforehand by the receiver in order to be recognized. This means that before sending packets with a new locator, the sender has to inform the receiver about the new locator, while for other approaches it is probably possible to start sending packets using a new locator and the same context tag in parallel with carrying information about the new locator to the peer, if the context tag would be sufficient by itself to identify the context i.e., the source locator isn't used to identify the context.

Note that we do not yet understand how beneficial it would be to be able to accept packets from unknown source locators (the rules for packet reception can probably be more relaxed than for where packets are sent, for instance if the context tag matches). Requiring a match on <source locator, destination locator, flow label> would make this impossible. Instead the locator change signaling would need to be acknowledged before the peer can start sending using a new source locator.

An attempt to remove the above limitation would be to try to have the receiver only identify the context based on the flow label field, i.e., without taking the locators into account in the lookup. This requires constraining flow label allocation for the hosts that implement shim6 so that for shim6 packets the receiver wouldn't have to compare the locators but only use the flow label. Due to the deferred shim6 capability discovery this would have to apply to all flow label assignments on a host which implements shim6.

It also requires carrying some additional information in the packet to identify whether the Flow Label field is actually being used as a context tag or not. In other words, additional information is needed to identify shim6 packets from regular IPv6 packets. This is because, the same Flow Label value that is being used as context tag in shim6 enabled communication can be used for other purposes by a non-shim6 enabled host, resulting in two communications using the same Flow Label value. The result of this situation would be that packets of the non-shim6 enabled communication would be demultiplexed using the context associated to the Flow Label value carried in the packets. A possible approach to solve this issue it to use an additional bit to identify data packets that belong to shim6 capable communications and that have to be demultiplexed using the Flow Label

value. However, there are no obvious choices for that bit, since all bits of the IPv6 header are currently in use. A possibility would be to use new Next Header values to indicate that the packet belongs to a shim6 enabled communication and that the Flow Label carries context information as proposed in [NOID].

9.2.2. Extension Header

Another approach is to define a new Extension Header to carry the context tag. This context tag is agreed between the involved parties during the shim6 protocol initial negotiation. Following data packets will be demultiplexed using the tag carried in the Extension Header. This seems a clean approach since it does not overload existing fields. However, it introduces additional overhead in the packet due to the additional header. The additional overhead introduced is 8 octets. However, it should be noted that the context tag is only required when a locator other than the one used as ULID is contained in the packet. Packets where both the source and destination address fields contain the ULIDs do not require a context tag, since no rewriting is necessary at the receiver. This approach would reduce the overhead. On the other hand, this approach would cause changes in the available MTU for some packets, since packets that include the Extension Header will have an MTU 8 octets shorter.

9.3. Host-Pair Context

The host-pair context is established on each end of the communication when one of the endpoints performs the shim6 signaling (the 4-way handshake referred to in [M6FUNC]).

This context is accessed differently in the transmit and receive paths. In the transmit path when the ULP passes down a packet the key to the context state is the tuple <ULID(local), ULID(peer)>; this key must identify at most one state record. In the receive path the context must be found based on what is in the packet, be it just the locators, or the locators plus some additional "context tag" as discussed above, or just a "context tag".

10. IPSEC INTERACTIONS

As specified, all of ESP, AH, and key management is layered above the shim6 layer. Thus they benefit from the stable ULIDs provided above the shim6 layer. This means the IPsec security associations are unaffected by switching locators.

The alternative would be to layer shim6 above IPsec, but that doesn't seem to provide any benefits and it would add the need to create different IPsec SAs when the locators change due to rehoming.

A result of layering shim6 above IPsec is that the shim6 protocol can potentially be used to redirect IPsec protected traffic as a selective DoS mechanism. If we somehow could require IPsec for the shim6 protocol packets when the ULP packets between the same hosts use IPsec, then we could prevent such attacks.

However, due to the richness in IPsec policy, this would be a bit tricky. If only some protocols or port numbers/selectors are to be protected by IPsec per a host's IPsec policy, then how would one determine whether shim6 traffic needs to be protected? Should one take the conservative approach that if any packets between the hosts/ULIDs need to be protected, then the shim6 traffic should also be protected?

For this to be useful both communicating hosts would need to make the same policy decisions, so if we are to take this path there would need to some standardization in this area.

11. OPEN ISSUES

Receive side demultiplexing issue as described above.

How is the shim6 state managed, in particular, what mechanism is used for removing the state? There seems to be two choices:

- a harder state approach which relies on some "CLOSE" message exchange (in combination with timers). An example of this is in [HIP-BASE].

- a soft-state mechanism where a node can discard the shim6 state at any time, combined with an error message "I have no state for you" that triggers the peer to reestablish the context.

Related to the state management and the demultiplexing issues is how the protocol detects a loss of context state, which can occur due to

a complete state loss (a host crashing and rebooting) or due to the shim garbage collecting the shim state even though the peer might continue to rely on it.

The actual packet formats for the shim6 protocol.

Is it possible to facilitate transition to shim6 using some "shim6 proxy" at site boundaries until all important hosts in a site have been upgraded to support shim6? Would would be the properties of such a proxy? Would it place any additional requirements on the protocol itself?

12. ACKNOWLEDGMENTS

This document was originally produced of a MULTI6 design team consisting of (in alphabetical order): Jari Arkko, Iljitsch van Beijnum, Marcelo Bagnulo Braun, Geoff Huston, Erik Nordmark, Margaret Wasserman, and Jukka Ylitalo.

The idea to use a set of locators and not inventing a new identifier name space, as well as using the DNS for verification of the locators, was first brought up by Tony Li.

Greg Daley suggested that the flow label approach can be more easily used if different flow label values are used for the different locator pairs.

13. REFERENCES

13.1. Normative References

[M6THREATS] Nordmark, E., and T. Li, "Threats relating to IPv6 multihoming solutions", draft-ietf-multi6-multihoming-threats-00.txt, July 2004.

[ADDR-ARCH] S. Deering, R. Hinden, Editors, "IP Version 6 Addressing Architecture", RFC 3513, April 2003.

[IPv6] S. Deering, R. Hinden, Editors, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2461.

[M6FUNC] Functional decomposition of the M6 protocol, draft-ietf-shim6-functional-dec-00.txt

[HBA] Hash Based Addresses (HBA), draft-ietf-shim6-hba-00.txt

[M6DET] Jari Arkko, Failure Detection and Locator Selection in Multi6, draft-ietf-shim6-failure-detection-00.txt

13.2. Informative References

[NSRG] Lear, E., and R. Droms, "What's In A Name: Thoughts from the NSRG", draft-irtf-nsrg-report-09.txt (work in progress), March 2003.

[ULA] R. Hinden, and B. Haberman, Unique Local IPv6 Unicast Addresses, draft-ietf-ipv6-unique-local-addr-08.txt

[ULA-CENTRAL] R. Hinden, and B. Haberman, Centrally Assigned Unique Local IPv6 Unicast Addresses, draft-ietf-ipv6-ula-central-00.txt

[APP-REFER] Shim6 Application Referral Issues, July 2005, <draft-ietf-shim6-app-refer-00.txt>

[NOID] Erik Nordmark, "Multihoming without IP Identifiers", Oct 27, 2003, <draft-nordmark-multi6-noid-01.txt>

[MAST] D. Crocker, "MULTIPLE ADDRESS SERVICE FOR TRANSPORT (MAST): AN EXTENDED PROPOSAL", draft-crocker-mast-protocol-01.txt, October, 2003.

[RFC3041] T. Narten, R. Draves, "Privacy Extensions for Stateless Address Autoconfiguration in IPv6", RFC 3041, January 2001.

[RFC2827] Ferguson P., and D. Senie, "Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing", RFC 2827, May 2000.

[RFC3550] H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", July 2003, RFC 3550.

[INGRESS] C. Huitema, R. Draves, and M. Bagnulo, "Ingress filtering compatibility for IPv6 multihomed sites", Oct 2004, <draft-huitema-multi6-ingress-filtering-00>

[RFC3484] R. Draves, "Default Address Selection for Internet Protocol version 6 (IPv6)", February 2003, RFC 3484.

[RFC3697] J. Rajahalme, A. Conta, B. Carpenter, S. Deering, "IPv6 Flow Label Specification", March 2004, RFC 3697.

[HIP-BASE] Robert Moskowitz, "Host Identity Protocol", draft-ietf-hip-base-03.txt

14. CHANGE LOG

Changes since draft-ietf-multi6-l3shim-00.txt:

- o Renamed to draft-ietf-shim6-l3shim
- o Renamed protocol and layer from "multi6" to "shim6".
- o Using "address" vs. "locator" and "ULID" more consistently and carefully.
- o Made it more clear that the ULID is just an IPv6 address. (Requested on mailing list.)
- o In "Renumbering Implications" added text to point out the small probability of there being a problem. (Requested on mailing list.)
- o Extended the assumption about ingress filtering and exit selection. (Requested on mailing list.)
- o Added clarification to MTU implications. (Requested on mailing list.)
- o Clarified what Centrally assigned ULAs can do which regular IPv6 addresses can't do with respect to the DNS. (Requested on mailing list.)
- o Added suggestion from mailing list that one can use different flow label for the communication when ULIDs=locators, and when they are different.
- o Listed a few more open issues.

Changes since draft-nordmark-multi6dt-shim-00.txt:

- o Added assumption that something else handles the interaction between ingress filtering and source address selection.
- o Clarified things with respect to using ULAs in general, and added

separate text about centrally assigned ULAs.

- o Added more text about MTU dropping implications and ICMP too big re-mapping

- o Added text specifying how the shim handles the flow label field, and the impact on flow setup protocols.

- o Added text about the need for the sender to handle ICMP errors

- o Added text that there might be other protocols than flow setup protocols and ICMP errors that might be impacted by the shim.

- o Added text about IP multicast in a new section.

- o Added clarification in section 8.4 about AAAA records being for a service and not a host needing some care.

- o Added a clarification in section 8.4 that learning the different locators during initial communication from the DNS potentially has different trust issues than learning them from the peer.

- o Clarified the two models of flow label usage for demultiplexing

- o In section 5 clarified that state maintenance is not per ULP connection.

- o In section 5 clarified merging option.

- o Clarified in section 9.1 why it isn't sufficient to avoid using the same locators for different ULIDs for the same peer host.

- o Clarified in section 9.1.1/9.1.2 that there is multi6 state at the receiver to tell how/whether to rewrite the source address field.

- o Clarified the aspect of section 9.2.1 which talks about not being able to use a new locator until the peer has been told of the new locator.

- o Added text about the implications of renumbering and reassignment.

- o Clarified section on flow labels to first talk about the simple case of using <source locator, destination locator, flow label> and its complexities, and then about the potential to just use the flow label by itself to identify the context.

AUTHORS' ADDRESSES

Erik Nordmark
Sun Microsystems, Inc.
17 Network Circle
Mountain View, CA
USA

phone: +1 650 786 2921
fax: +1 650 786 5896
email: erik.nordmark@sun.com

Marcelo Bagnulo
Universidad Carlos III de Madrid
Av. Universidad 30
Leganes, Madrid 28911
SPAIN

Phone: 34 91 6249500
EMail: marcelo@it.uc3m.es
URI: http://www.it.uc3m.es

Intellectual Property Statement

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

Acknowledgment

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