Network Working Group
Internet-Draft
Obsoletes: rfc6706 (if approved)
Intended status: Standards Track
Expires: September 18, 2014

F. Templin, Ed.
Boeing Research & Technology

March 17, 2014

Transmission of IPv6 Packets over AERO Links
draft-templin-aerolink-07.txt

Abstract

This document specifies the operation of IPv6 over tunnel virtual Non-Broadcast, Multiple Access (NBMA) links using Asymmetric Extended Route Optimization (AERO). Nodes attached to AERO links can exchange packets via trusted intermediate routers on the link that provide forwarding services to reach off-link destinations and/or redirection services to inform the node of an on-link neighbor that is closer to the final destination. Operation of the IPv6 Neighbor Discovery (ND) protocol over AERO links is based on an IPv6 link local address format known as the AERO address.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

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

This Internet-Draft will expire on September 18, 2014.

Table of Contents

1. Introduction 3
2. Terminology 3
3. Asymmetric Extended Route Optimization (AERO) 5
3.1. AERO Node Types 5
3.2. AERO Interface Characteristics 6
3.3. AERO Interface MTU Considerations 8
3.4. AERO Interface Encapsulation, Re-encapsulation and Decapsulation 9
3.5. AERO Addresses 10
3.6. AERO Reference Operational Scenario 11
3.7. AERO Router Discovery and Prefix Delegation 13
3.7.1. AERO Client Behavior 13
3.7.2. AERO Server Behavior 13
3.8. AERO Neighbor Solicitation and Advertisement 14
3.9. AERO Redirection 15
3.9.1. Classical Redirection Approaches 15
3.9.2. AERO Redirection Concept of Operations 16
3.9.3. AERO Redirection Message Format 17
3.9.4. Sending Predirects 18
3.9.5. Processing Predirects and Sending Redirects 19
3.9.6. Re-encapsulating and Relaying Redirects 20
3.9.7. Processing Redirects 20
3.10. Neighbor Reachability Considerations 21
3.11. Mobility and Link-Layer Address Change Considerations . . 21
3.12. Underlying Protocol Version Considerations 22
3.13. Multicast Considerations 22
3.14. Operation on Server-less AERO Links 22
3.15. Other Considerations 23
4. Implementation Status 23
5. IANA Considerations 23
6. Security Considerations 23
7. Acknowledgements 24
8. References 24
8.1. Normative References 24
8.2. Informative References 25
Appendix A. AERO Server and Relay Interworking 26
Author's Address 28

1. Introduction

This document specifies the operation of IPv6 over tunnel virtual Non-Broadcast, Multiple Access (NBMA) links using Asymmetric Extended Route Optimization (AERO). Nodes attached to AERO links can exchange packets via trusted intermediate routers on the link that provide forwarding services to reach off-link destinations and/or redirection services to inform the node of an on-link neighbor that is closer to the final destination.

Nodes on AERO links use an IPv6 link-local address format known as the AERO Address. This address type has properties that statelessly link IPv6 Neighbor Discovery (ND) to IPv6 routing. The AERO link can be used for tunneling to neighboring nodes on either IPv6 or IPv4 networks, i.e., AERO views the IPv6 and IPv4 networks as equivalent links for tunneling. The remainder of this document presents the AERO specification.

2. Terminology

The terminology in the normative references applies; the following terms are defined within the scope of this document:

AERO link
: a Non-Broadcast, Multiple Access (NBMA) tunnel virtual overlay configured over a node's attached IPv6 and/or IPv4 networks. All nodes on the AERO link appear as single-hop neighbors from the perspective of IPv6. Note that the AERO link Maximum Transmission Unit (MTU) is 64KB minus the encapsulation overhead for IPv4 and 4GB minus the encapsulation overhead for IPv6.

AERO interface
: a node's attachment to an AERO link. The AERO interface MTU is less than or equal to the AERO link MTU.

AERO address
: an IPv6 link-local address assigned to an AERO interface and constructed as specified in Section 3.5.

AERO node
: a node that is connected to an AERO link and that participates in IPv6 Neighbor Discovery over the link.

AERO Client ("client")
: a node that configures either a host interface or a router interface on an AERO link.

AERO Server ("server")
: a node that configures a router interface on an AERO link over which it can provide default forwarding and redirection services for other AERO nodes.

AERO Relay ("relay")
: a node that relays IPv6 packets between Servers on the same AERO link, and/or that forwards IPv6 packets between the AERO link and the IPv6 Internet. An AERO Relay may or may not also be configured as an AERO Server.

ingress tunnel endpoint (ITE)
: an AERO interface endpoint that injects tunneled packets into an AERO link.

egress tunnel endpoint (ETE)
: an AERO interface endpoint that receives tunneled packets from an AERO link.

underlying network
: a connected IPv6 or IPv4 network routing region over which AERO nodes tunnel IPv6 packets.

underlying interface
: an AERO node's interface point of attachment to an underlying network.

underlying address
: an IPv6 or IPv4 address assigned to an AERO node's underlying interface. When UDP encapsulation is used, the UDP port number is also considered as part of the underlying address. Underlying addresses are used as the source and destination addresses of the AERO encapsulation header.

link-layer address
: the same as defined for "underlying address" above.

network layer address
: an IPv6 address used as the source or destination address of the inner IPv6 packet header.

end user network (EUN)
: an IPv6 network attached to a downstream interface of an AERO Client (where the AERO interface is seen as the upstream interface).

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this

document are to be interpreted as described in [RFC2119].

3. Asymmetric Extended Route Optimization (AERO)

The following sections specify the operation of IPv6 over Asymmetric Extended Route Optimization (AERO) links:

3.1. AERO Node Types

AERO Relays relay packets between nodes connected to the same AERO link and also forward packets between the AERO link and the native IPv6 network. The relaying process entails re-encapsulation of IPv6 packets that were received from a first AERO node and are to be forwarded without modification to a second AERO node.

AERO Servers configure their AERO interfaces as router interfaces, and provide default routing services to AERO Clients. AERO Servers configure a DHCPv6 Relay or Server function and facilitate DHCPv6 Prefix Delegation (PD) exchanges. An AERO Server may also act as an AERO Relay.

AERO Clients act as requesting routers to receive IPv6 prefixes through a DHCPv6 PD exchange via an AERO Server over the AERO link. Each AERO Client receives at least a /64 prefix delegation, and may receive even shorter prefixes.

AERO Clients that act as routers configure their AERO interfaces as router interfaces, i.e., even if the AERO Client otherwise displays the outward characteristics of an ordinary host (for example, the Client may internally configure both an AERO interface and (internal virtual) End User Network (EUN) interfaces). AERO Clients that act as routers sub-delegate portions of their received prefix delegations to links on EUNs.

AERO Clients that act as ordinary hosts configure their AERO interfaces as host interfaces and assign one or more IPv6 addresses taken from their received prefix delegations to the AERO interface but DO NOT assign the delegated prefix itself to the AERO interface. Instead, the host assigns the delegated prefix to a "black hole" route so that unused portions of the prefix are nullified.

End system applications on AERO hosts bind directly to the AERO interface, while applications on AERO routers (or IPv6 hosts served by an AERO router) bind to EUN interfaces.

3.2. AERO Interface Characteristics

AERO interfaces use IPv6-in-IPv6 encapsulation [RFC2473] to exchange tunneled packets with AERO neighbors attached to an underlying IPv6 network, and use IPv6-in-IPv4 encapsulation [RFC4213] to exchange tunneled packets with AERO neighbors attached to an underlying IPv4 network. AERO interfaces can also use IPsec encapsulation [RFC4301] (either IPv6-in-IPsec-in-IPv6 or IPv6-in-IPsec-in-IPv4) in environments where strong authentication and confidentiality are required. When NAT traversal and/or filtering middlebox traversal is necessary, a UDP header is further inserted between the outer IP encapsulation header and the inner packet.

AERO interfaces maintain a neighbor cache and use a variation of standard unicast IPv6 ND messaging. AERO interfaces use Neighbor Solicitation (NS), Neighbor Advertisement (NA) and Redirect messages the same as for any IPv6 link. They do not use Router Solicitation (RS) and Router Advertisement (RA) messages for several reasons. First, default router discovery is supported through other means more appropriate for AERO links as described below. Second, discovery of more-specific routes is through the receipt of NS, NA and Redirect messages. Finally, AERO nodes are pre-provisioned with IPv6 prefixes that they register using DHCPv6 PD; hence, there is no need for RA-based prefix discovery.

AERO Neighbor Solicitation (NS) and Neighbor Advertisement (NA) messages do not include Source/Target Link Layer Address Options (S/TLLAO). Instead, AERO nodes determine the link-layer addresses of neighbors by examining the encapsulation source address of any NS/NA messages they receive and ignore any S/TLLAOs included in these messages. This is vital to the operation of AERO links for which neighbors are separated by Network Address Translators (NATs) - either IPv4 or IPv6.

AERO Redirect messages include a TLLAO the same as for any IPv6 link. The TLLAO includes the link-layer address of the target node, including both the IP address and the UDP source port number used by the target when it sends UDP-encapsulated packets over the AERO interface (the TLLAO instead encodes the value 0 when the target does not use UDP encapsulation). TLLAOs for target nodes that use an IPv6 underlying address include the full 16 bytes of the IPv6 address as shown in Figure 1, while TLLAOs for target nodes that use an IPv4 underlying address include only the 4 bytes of the IPv4 address as shown in Figure 2.

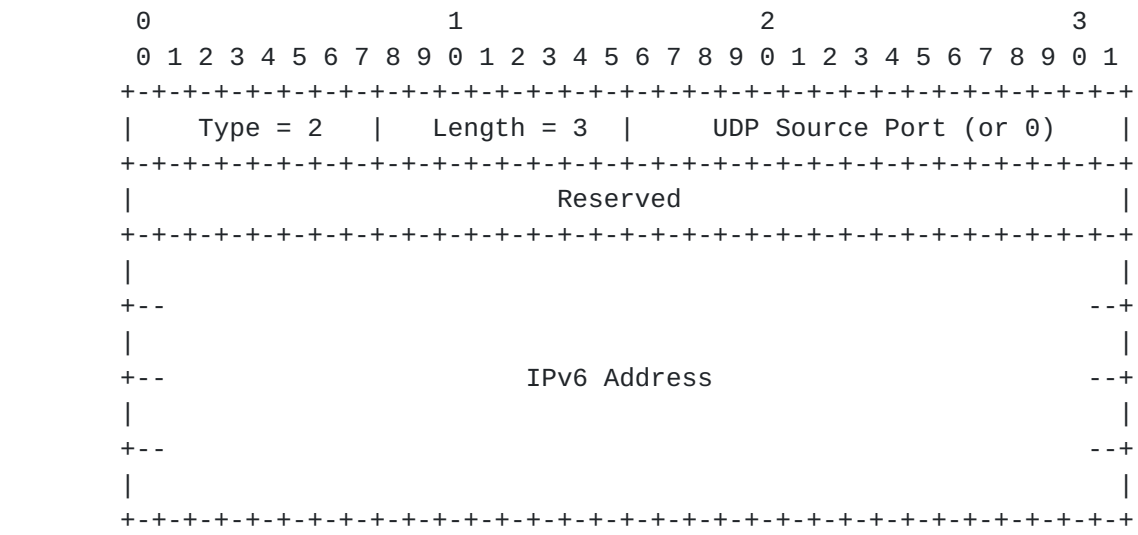

Figure 1: AERO TLLAO Format for IPv6

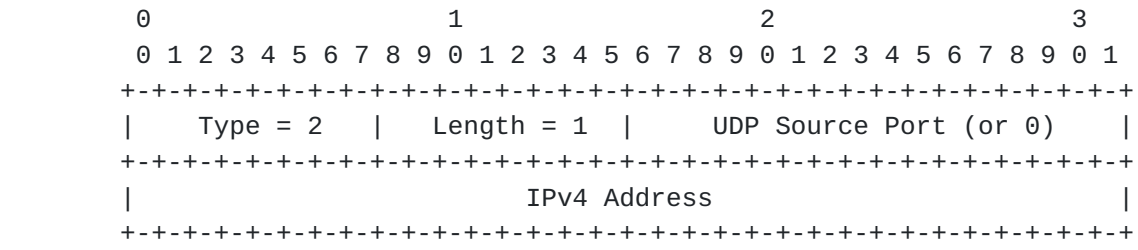

Figure 2: AERO TLLAO Format for IPv4

Finally, nodes on AERO interfaces use a simple data origin authentication for encapsulated packets they receive from other nodes. In particular, AERO Clients accept encapsulated packets with a link-layer source address belonging to their current AERO Server. AERO nodes also accept encapsulated packets with a link-layer source address that is correct for the network-layer source address. The AERO node considers the link-layer source address correct for the network-layer source address if there is an IPv6 route that matches the network-layer source address as well as a neighbor cache entry corresponding to the next hop that includes the link-layer address. (An exception is that NS, NA and Redirect messages may include a different link-layer address than the one currently in the neighbor cache, and the new link-layer address updates the neighbor cache entry.)

3.3. AERO Interface MTU Considerations

The base tunneling specifications for IPv4 and IPv6 typically set a static MTU on the tunnel interface to 1500 bytes minus the encapsulation overhead or smaller still if the tunnel is likely to incur additional encapsulations such as IPsec on the path. This can result in path MTU related black holes when packets that are too large to be accommodated over the AERO link are dropped, but the resulting ICMP Packet Too Big (PTB) messages are lost on the return path. As a result, AERO nodes use the following MTU mitigations to accommodate larger packets.

AERO nodes set their AERO interface MTU to the larger of 1500 bytes and the underlying interface MTU minus the encapsulation overhead. AERO nodes optionally cache other per-neighbor MTU values in the underlying IP path MTU discovery cache initialized to the underlying interface MTU.

AERO nodes admit packets that are no larger than 1280 bytes minus the encapsulation overhead (*) as well as packets that are larger than 1500 bytes into the tunnel without fragmentation, i.e., as long as they are no larger than the AERO interface MTU before encapsulation and also no larger than the cached per-neighbor MTU following encapsulation. For IPv4, the node sets the "Don't Fragment" (DF) bit to 0 for packets no larger than 1280 bytes minus the encapsulation overhead (*) and sets the DF bit to 1 for packets larger than 1500 bytes. If a large packet is lost in the path, the node may optionally cache the MTU reported in the resulting PTB message or may ignore the message, e.g., if there is a possibility that the message is spurious.

For packets destined to an AERO node that are larger than 1280 bytes minus the encapsulation overhead (*) but no larger than 1500 bytes, the node uses outer IP fragmentation to fragment the packet into two pieces (where the first fragment contains 1024 bytes of the fragmented inner packet) then admits the fragments into the tunnel. If the outer protocol is IPv4, the node admits the packet into the tunnel with DF set to 0 and subject to rate limiting to avoid reassembly errors [RFC4963][RFC6864]. For both IPv4 and IPv6, the node also sends a 1500 byte probe message (**) to the neighbor, subject to rate limiting. To construct a probe, the node prepares an ICMPv6 Neighbor Solicitation (NS) message with trailing padding octets added to a length of 1500 bytes but does not include the length of the padding in the IPv6 Payload Length field. The node then encapsulates the NS in the outer encapsulation headers (while including the length of the padding in the outer length fields), sets DF to 1 (for IPv4) and sends the padded NS message to the neighbor. If the neighbor returns an NA message, the node may then send whole

packets within this size range and (for IPv4) relax the rate limiting requirement.

AERO nodes MUST be capable of reassembling packets up to 1500 bytes plus the encapsulation overhead length. It is therefore RECOMMENDED that AERO nodes be capable of reassembling at least 2KB.

(*) Note that if it is known that the minimum Path MTU to an AERO node is MINMTU bytes (where 1280 < MINMTU < 1500) then MINMTU can be used instead of 1280 in the fragmentation threshold considerations listed above.

(**) It is RECOMMENDED that no probes smaller than 1500 bytes be used for MTU probing purposes, since smaller probes may be fragmented if there is a nested tunnel somewhere on the path to the neighbor.

3.4. AERO Interface Encapsulation, Re-encapsulation and Decapsulation

AERO interfaces encapsulate IPv6 packets according to whether they are entering the AERO interface for the first time or if they are being forwarded out the same AERO interface that they arrived on. This latter form of encapsulation is known as "re-encapsulation".

AERO interfaces encapsulate packets per the specifications in , [RFC2473], [RFC4213] except that the interface copies the "TTL/Hop Limit", "Type of Service/Traffic Class" and "Congestion Experienced" values in the inner network layer header into the corresponding fields in the outer IP header. For packets undergoing re-encapsulation, the AERO interface instead copies the "TTL/Hop Limit", "Type of Service/Traffic Class" and "Congestion Experienced" values in the original outer IP header into the corresponding fields in the new outer IP header (i.e., the values are transferred between outer headers and *not* copied from the inner network layer header).

When UDP encapsulation is used, the AERO interface inserts a UDP header between the inner packet and outer IP header. If the outer header is IPv6 and is followed by an IPv6 Fragment header, the AERO interface inserts the UDP header between the outer IPv6 header and IPv6 Fragment header. The AERO interface sets the UDP source port to a constant value that it will use in each successive packet it sends, sets the UDP destination port to 8060 (i.e., the IANA-registered port number for AERO), sets the UDP checksum field to zero (see: [RFC6935][RFC6936]) and sets the UDP length field to the length of the inner packet plus 8 bytes for the UDP header itself.

The AERO interface next sets the outer IP protocol number to the appropriate value for the first protocol layer within the encapsulation (e.g., IPv6, IPv6 Fragment Header, UDP, etc.). When

IPv6 is used as the outer IP protocol, the ITE then sets the flow label value in the outer IPv6 header the same as described in [RFC6438]. When IPv4 is used as the outer IP protocol, the AERO interface sets the DF bit as discussed in Section 3.2.

AERO interfaces decapsulate packets destined either to the localhost or to a destination reached via an interface other than the receiving AERO interface per the specifications in , [RFC2473], [RFC4213]. When the encapsulated packet includes a UDP header, the AERO interfaces examines the first octet of data following the UDP header to determine the inner header type. If the most significant four bits of the first octet encode the value '0110', the inner header is an IPv6 header. Otherwise, the interface considers the first octet as an IP protocol type that encodes the value '44' for IPv6 fragment header, the value '50' for Encapsulating Security Payload, the value '51' for Authentication Header etc. (If the first octet encodes the value '0', the interface instead discards the packet, since this is the value reserved for experimentation under , [RFC6706]). During the decapsulation, the AERO interface records the UDP source port in the neighbor cache entry for this neighbor then discards the UDP header.

3.5. AERO Addresses

An AERO address is an IPv6 link-local address assigned to an AERO interface and with an IPv6 prefix embedded within the interface identifier. The AERO address is formatted as:

```
fe80::[IPv6 prefix]
```

Each AERO Client configures an AERO address based on the delegated prefix it has received from the AERO link prefix delegation authority. The address begins with the prefix fe80::/64 and includes in its interface identifier the base /64 prefix taken from the Client's delegated IPv6 prefix. The base prefix is determined by masking the delegated prefix with the prefix length. For example, if an AERO Client has received the prefix delegation:

```
2001:db8:1000:2000::/56
```

it would construct its AERO address as:

```
fe80::2001:db8:1000:2000
```

An AERO Client may have multiple non-contiguous IPv6 prefix delegations, in which case it would configure multiple AERO addresses - one for each prefix. Note that, in order for the DHCPv6 PD function to operate correctly, the AERO Client must already hold a

delegated IPv6 prefix so that it can construct an AERO address to use as the source address in the DHCPv6 exchange. This means that the DHCPv6 PD function is really just a registration of a pre-provisioned prefix.

Each AERO Server configures the special AERO address fe80::1 to support the operation of IPv6 Neighbor Discovery over the AERO link; the address therefore has the properties of an IPv6 Anycast address. While all Servers configure the same AERO address and therefore cannot be distinguished from one another at the network layer, Clients can still distinguish Servers at the link layer by examining the Servers' link-layer addresses.

Nodes that are configured as pure AERO Relays (i.e., and that do not also act as Servers) do not configure an IPv6 address of any kind on their AERO interfaces. The Relay's AERO interface is therefore used purely for transit and does not participate in IPv6 ND message exchanges.

3.6. AERO Reference Operational Scenario

Figure 3 depicts the AERO reference operational scenario. The figure shows an AERO Server('A'), two AERO Clients ('B', 'D') and three ordinary IPv6 hosts ('C', 'E', 'F'):

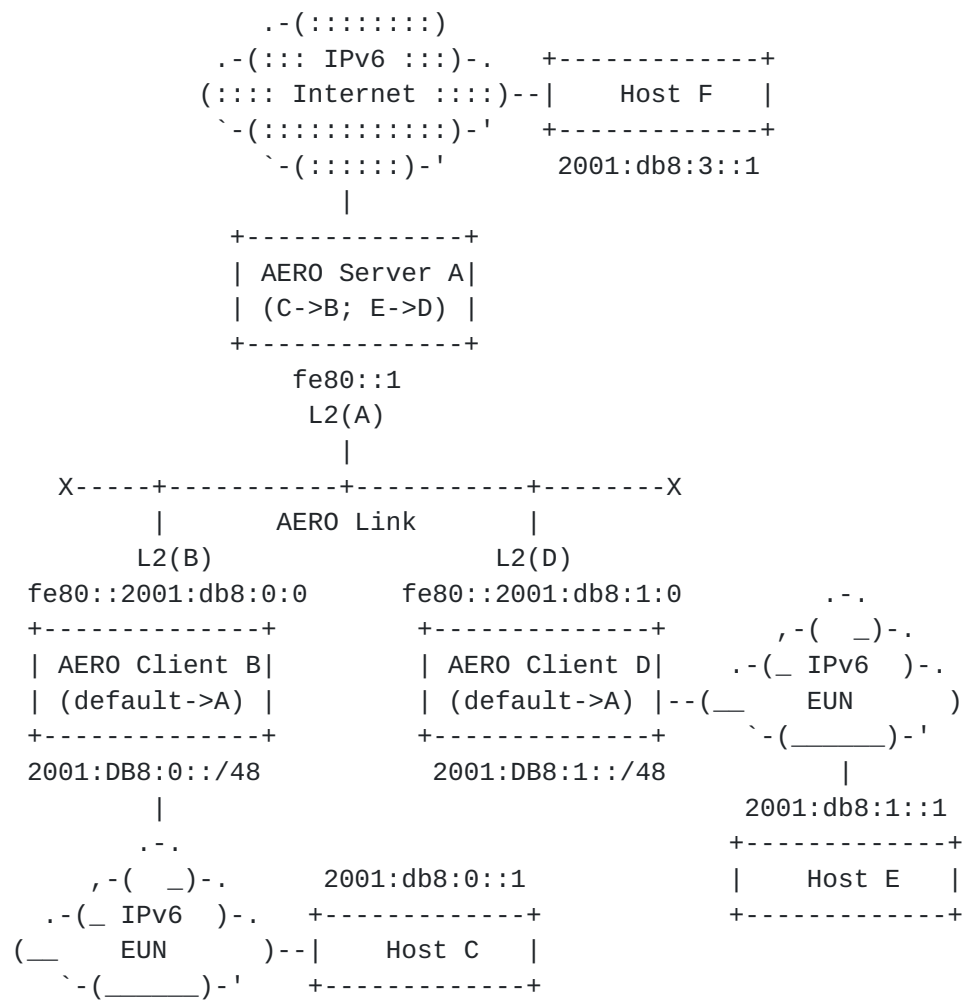

Figure 3: AERO Reference Operational Scenario

In Figure 3, AERO Server ('A') connects to the AERO link and connects to the IPv6 Internet, either directly or via an AERO Relay (not shown). Server ('A') assigns the address fe80::1 to its AERO interface with link-layer address L2(A). Server ('A') next arranges to add L2(A) to a published list of valid Servers for the AERO link.

AERO Client ('B') assigns the address fe80::2001:db8:0:0 to its AERO interface with link-layer address L2(B) and registers the IPv6 prefix 2001:db8:0::/48 in a DHCPv6 PD exchange via Server ('A'). Client ('B') configures a default route via the AERO interface with next-hop address fe80::1 and link-layer address L2(A), then sub-delegates the prefix 2001:db8:0::/48 to its attached EUNs. IPv6 host ('C') connects to the EUN, and configures the address 2001:db8:0::1.

AERO Client ('D') assigns the address fe80::2001:db8:1:0 to its AERO interface with link-layer address L2(D) and registers the IPv6 prefix 2001:db8:1::/48 in a DHCPv6 PD exchange via Server ('A'). Client

('D') configures a default route via the AERO interface with next-hop address fe80::1 and link-layer address L2(A), then sub-delegates the prefix 2001:db8:1::/48 to its attached EUNs. IPv6 host ('E') connects to the EUN, and configures the address 2001:db8:1::1.

Finally, IPv6 host ('F') connects to an IPv6 network outside of the AERO link domain. Host ('F') configures its IPv6 interface in a manner specific to its attached IPv6 link, and assigns the address 2001:db8:3::1 to its IPv6 link interface.

3.7. AERO Router Discovery and Prefix Delegation

3.7.1. AERO Client Behavior

AERO Clients observe the IPv6 router requirements defined in [RFC6434]. AERO Clients first discover the link-layer address of an AERO Server via static configuration, or through an automated means such as DNS name resolution. In the absence of other information, the Client resolves the Fully-Qualified Domain Name (FQDN) "linkupnetworks.domainname", where "domainname" is the DNS domain appropriate for the Client's attached underlying network. The Client then creates a neighbor cache entry with the IPv6 link-local address fe80::1 and the discovered address as the link-layer address. The Client further creates a default route with the link-local address fe80::1 as the next hop.

Next, the Client assigns the link-local AERO address(es) taken from its delegated prefix(es) to the AERO interface (see: Section 3.5). It then acts as a requesting router to register its IPv6 prefixes through DHCPv6 PD [RFC3633] via the Server. After the Client registers its prefixes, it sub-delegates them to nodes and links within its attached EUNs.

After configuring a default route and registering its prefixes, the Client sends periodic NS messages to the server to obtain new NAs in order to keep neighbor cache entries alive. The Client can also forward IPv6 packets destined to networks beyond its local EUNs via the Server as an IPv6 default router. The Server may in turn return a Redirect message informing the Client of a neighbor on the AERO link that is topologically closer to the final destination as specified in Section 3.9.

3.7.2. AERO Server Behavior

AERO Servers observe the IPv6 router requirements defined in [RFC6434]. They further configure a DHCPv6 relay/server function on their AERO links. When the Server delegates prefixes, it also establishes forwarding table and neighbor cache entries that list the

AERO address of the Client as the next hop toward the delegated IPv6 prefixes (where the AERO address is constructed as specified in Section 3.5).

Servers respond to NS messages from Clients on their AERO interfaces by returning an NA message. When the Server receives an NS message, it updates the neighbor cache entry using the network layer source address as the neighbor's network layer address and using the link-layer source address of the NS message as the neighbor's link-layer address.

When the Server forwards a packet via the same AERO interface on which it arrived, it initiates an AERO route optimization procedure as specified in Section 3.9.

3.8. AERO Neighbor Solicitation and Advertisement

After an AERO node has received a prefix delegation, it creates an AERO address as specified in Section 3.5. It can then send NS messages to elicit NA messages from other AERO nodes. When the AERO node sends NS/NA messages, however, it must also include the length of the prefix corresponding to the AERO address. AERO NS/NA messages therefore include an 8-bit "Prefix Length" field take from the low-order 8 bits of the Reserved field as shown in Figure 4 and Figure 5.

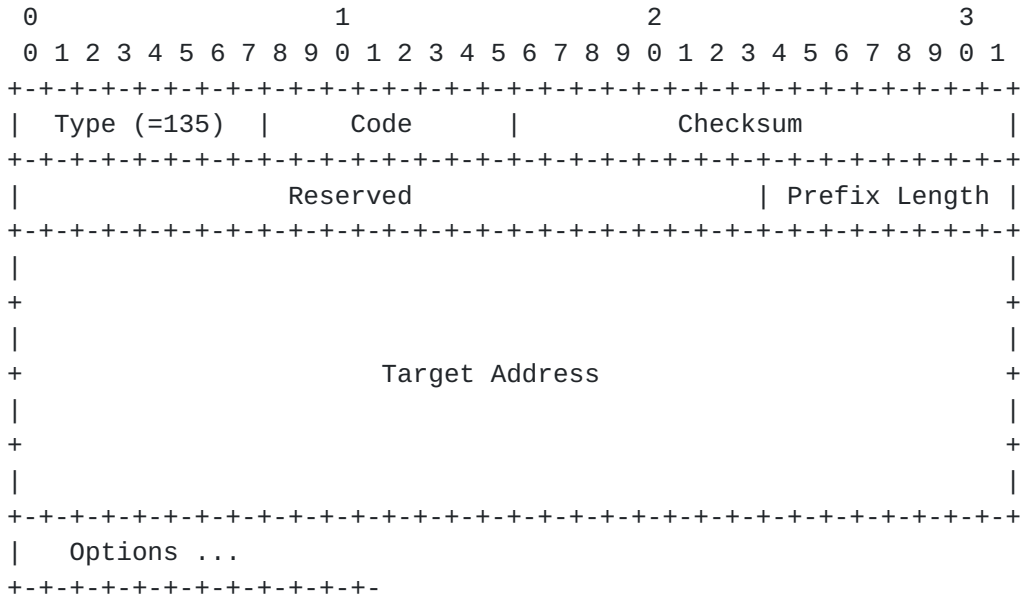

Figure 4: AERO Neighbor Solicitation (NS) Message Format

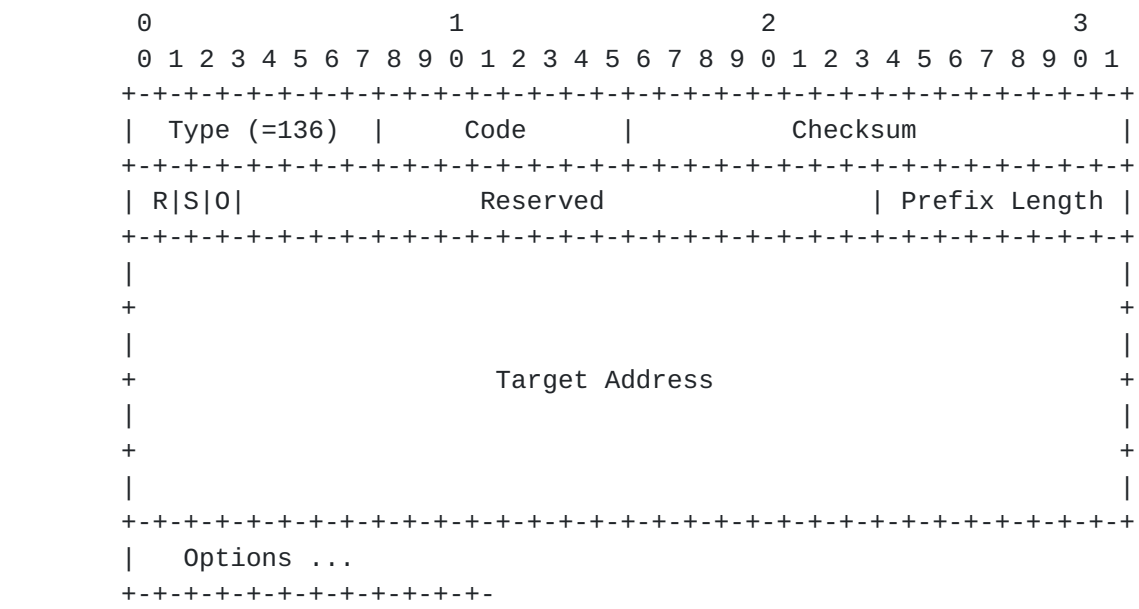

Figure 5: AERO Neighbor Advertisement (NA) Message Format

When an AERO node sends an NS/NA message, it MUST use its AERO address as the IPv6 source address and MUST include its AERO address prefix length in the Prefix Length field.

When an AERO node receives an NS/NA message, it accepts the message if the Prefix Length applied to the source address is correct for the neighbor; otherwise, it ignores the message.

3.9. AERO Redirection

Section 3.6 describes the AERO reference operational scenario. We now discuss the operation and protocol details of AERO Redirection with respect to this reference scenario.

3.9.1. Classical Redirection Approaches

With reference to Figure 3, when the IPv6 source host ('C') sends a packet to an IPv6 destination host ('E'), the packet is first forwarded via the EUN to AERO Client ('B'). Client ('B') then forwards the packet over its AERO interface to AERO Server ('A'), which then re-encapsulates and forwards the packet to AERO Client ('D'), where the packet is finally forwarded to the IPv6 destination host ('E'). When Server ('A') re-encapsulates and forwards the packet back out on its advertising AERO interface, it must arrange to redirect Client ('B') toward Client ('D') as a better next-hop node on the AERO link that is closer to the final destination. However, this redirection process applied to AERO interfaces must be more carefully orchestrated than on ordinary links since the parties may

be separated by potentially many underlying network routing hops.

Consider a first alternative in which Server ('A') informs Client ('B') only and does not inform Client ('D') (i.e., "classical redirection"). In that case, Client ('D') has no way of knowing that Client ('B') is authorized to forward packets from their claimed network-layer source addresses, and it may simply elect to drop the packets. Also, Client ('B') has no way of knowing whether Client ('D') is performing some form of source address filtering that would reject packets arriving from a node other than a trusted default router, nor whether Client ('D') is even reachable via a direct path that does not involve Server ('A').

Consider a second alternative in which Server ('A') informs both Client ('B') and Client ('D') separately, via independent redirection control messages (i.e., "augmented redirection"). In that case, if Client ('B') receives the redirection control message but Client ('D') does not, subsequent packets sent by Client ('B') could be dropped due to filtering since Client ('D') would not have a route to verify their source network-layer addresses. Also, if Client ('D') receives the redirection control message but Client ('B') does not, subsequent packets sent in the reverse direction by Client ('D') would be lost.

Since both of these alternatives have shortcomings, a new redirection technique (i.e., "AERO redirection") is needed.

3.9.2. AERO Redirection Concept of Operations

Again, with reference to Figure 3, when source host ('C') sends a packet to destination host ('E'), the packet is first forwarded over the source host's attached EUN to Client ('B'), which then forwards the packet via its AERO interface to Server ('A').

Server ('A') then re-encapsulates forwards the packet out the same AERO interface toward Client ('D') and also sends an AERO "Predirect" message forward to Client ('D') as specified in Section 3.9.4. The Predirect message includes Client ('B')'s network- and link-layer addresses as well as information that Client ('D') can use to determine the IPv6 prefix used by Client ('B') . After Client ('D') receives the Predirect message, it process the message and returns an AERO Redirect message destined for Client ("B") via Server ('A') as specified in Section 3.9.5. During the process, Client ('D') also creates or updates a neighbor cache entry for Client ('B'), and creates an IPv6 route for Client ('B')'s IPv6 prefix.

When Server ('A') receives the Redirect message, it re-encapsulates the message and forwards it on to Client ('B') as specified in

Section 3.9.6. The message includes Client ('D')'s network- and link-layer addresses as well as information that Client ('B') can use to determine the IPv6 prefix used by Client ('D'). After Client ('B') receives the Redirect message, it processes the message as specified in Section 3.9.7. During the process, Client ('B') also creates or updates a neighbor cache entry for Client ('D'), and creates an IPv6 route for Client ('D')'s IPv6 prefix.

Following the above Predirect/Redirect message exchange, forwarding of packets from Client ('B') to Client ('D') without involving Server ('A') as an intermediary is enabled. The mechanisms that support this exchange are specified in the following sections.

3.9.3. AERO Redirection Message Format

AERO Redirect/Predirect messages use the same format as for ICMPv6 Redirect messages depicted in Section 4.5 of [RFC4861], but also include a new "Prefix Length" field taken from the low-order 8 bits of the Redirect message Reserved field. The Redirect/Predirect messages are formatted as shown in Figure 6:

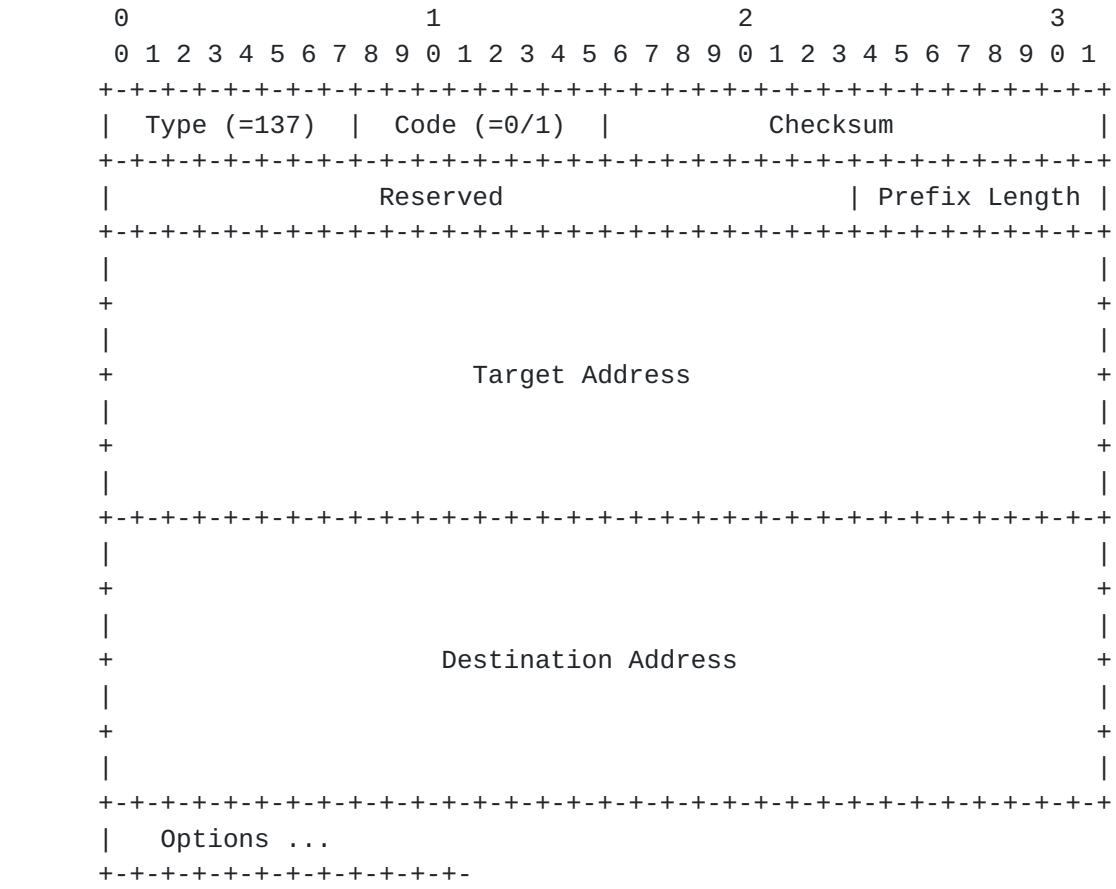

Figure 6: AERO Redirect/Predirect Message Format

3.9.4. Sending Predirects

When an AERO Server forwards a packet out the same AERO interface that it arrived on, the Server sends a Predirect message forward toward the AERO Client nearest the destination instead of sending a Redirect message back to AERO Client nearest the source.

In the reference operational scenario, when Server ('A') forwards a packet sent by Client ('B') toward Client ('D'), it also sends a Predirect message forward toward Client ('D'), subject to rate limiting (see Section 8.2 of [RFC4861]). Server ('A') prepares the Predirect message as follows:

- the link-layer source address is set to 'L2(A)' (i.e., the underlying address of Server ('A')).
- the link-layer destination address is set to 'L2(D)' (i.e., the underlying address of Client ('D')).
- the network-layer source address is set to fe80::1 (i.e., the AERO address of Server ('A')).
- the network-layer destination address is set to fe80::2001:db8:1:0 (i.e., the AERO address of Client ('D')).
- the Type is set to 137.
- the Code is set to 1 to indicate "Predirect".
- the Prefix Length is set to the length of the prefix to be applied to Target address.
- the Target Address is set to fe80::2001:db8:0::0 (i.e., the AERO address of Client ('B')).
- the Destination Address is set to the IPv6 source address of the packet that triggered the Predirection event.
- the message includes a TLLAO set to 'L2(B)' (i.e., the underlying address of Client ('B')).
- the message includes a Redirected Header Option (RHO) that contains the originating packet truncated to ensure that at least the network-layer header is included but the size of the message does not exceed 1280 bytes.

Server ('A') then sends the message forward to Client ('D').

3.9.5. Processing Predirects and Sending Redirects

When Client ('D') receives a Predirect message, it accepts the message only if it has a link-layer source address of the Server, i.e. 'L2(A)'. Client ('D') further accepts the message only if it is willing to serve as a redirection target. Next, Client ('D') validates the message according to the ICMPv6 Redirect message validation rules in Section 8.1 of [RFC4861].

In the reference operational scenario, when the Client ('D') receives a valid Predirect message, it either creates or updates a neighbor cache entry that stores the Target Address of the message as the network-layer address of Client ('B') and stores the link-layer address found in the TLLAO as the link-layer address of Client ('B'). Client ('D') then applies the Prefix Length to the Interface Identifier portion of the Target Address and records the resulting IPv6 prefix in its IPv6 forwarding table.

After processing the message, Client ('D') prepares a Redirect message response as follows:

- the link-layer source address is set to 'L2(D)' (i.e., the link-layer address of Client ('D')).
- the link-layer destination address is set to 'L2(A)' (i.e., the link-layer address of Server ('A')).
- the network-layer source address is set to 'L3(D)' (i.e., the AERO address of Client ('D')).
- the network-layer destination address is set to 'L3(B)' (i.e., the AERO address of Client ('B')).
- the Type is set to 137.
- the Code is set to 0 to indicate "Redirect".
- the Prefix Length is set to the length of the prefix to be applied to the Target and Destination address.
- the Target Address is set to fe80::2001:db8:1::1 (i.e., the AERO address of Client ('D')).
- the Destination Address is set to the IPv6 destination address of the packet that triggered the Redirection event.
- the message includes a TLLAO set to 'L2(D)' (i.e., the underlying address of Client ('D')).

- the message includes as much of the RHO copied from the corresponding AERO Predirect message as possible such that at least the network-layer header is included but the size of the message does not exceed 1280 bytes.

After Client ('D') prepares the Redirect message, it sends the message to Server ('A').

3.9.6. Re-encapsulating and Relaying Redirects

When Server ('A') receives a Redirect message, it accepts the message only if it has a neighbor cache entry that associates the message's link-layer source address with the network-layer source address. Next, Server ('A') validates the message according to the ICMPv6 Redirect message validation rules in Section 8.1 of [RFC4861]. Following validation, Server ('A') re-encapsulates the Redirect then relays the re-encapsulated Redirect on to Client ('B') as follows.

In the reference operational scenario, Server ('A') receives the Redirect message from Client ('D') and prepares to re-encapsulate and forward the message to Client ('B'). Server ('A') first verifies that Client ('D') is authorized to use the Prefix Length in the Redirect message when applied to the AERO address in the network-layer source of the Redirect message, and discards the message if verification fails. Otherwise, Server ('A') re-encapsulates the message by changing the link-layer source address of the message to 'L2(A)', changing the network-layer source address of the message to fe80::1, and changing the link-layer destination address to 'L2(B)' . Server ('A') finally relays the re-encapsulated message to the ingress node ('B') without decrementing the network-layer IPv6 header Hop Limit field.

While not shown in Figure 3, AERO Relays relay Redirect and Predirect messages in exactly this same fashion described above. See Figure 7 in Appendix A for an extension of the reference operational scenario that includes Relays.

3.9.7. Processing Redirects

When Client ('B') receives the Redirect message, it accepts the message only if it has a link-layer source address of the Server, i.e. 'L2(A)'. Next, Client ('B') validates the message according to the ICMPv6 Redirect message validation rules in Section 8.1 of [RFC4861]. Following validation, Client ('B') then processes the message as follows.

In the reference operational scenario, when Client ('B') receives the Redirect message, it either creates or updates a neighbor cache entry

that stores the Target Address of the message as the network-layer address of Client ('D') and stores the link-layer address found in the TLLAO as the link-layer address of Client ('D'). Client ('B') then applies the Prefix Length to the Interface Identifier portion of the Target Address and records the resulting IPv6 prefix in its IPv6 forwarding table.

Now, Client ('B') has an IPv6 forwarding table entry for Client('D')'s prefix, and Client ('D') has an IPv6 forwarding table entry for Client ('B')'s prefix. Thereafter, the clients may exchange ordinary network-layer data packets directly without forwarding through Server ('A').

3.10. Neighbor Reachability Considerations

When a source Client discovers a target neighbor (either through redirection or some other means) it MUST test the direct path to the target, e.g., by sending an initial NS message to elicit a solicited NA response. While testing the path, the Client SHOULD continue sending packets via the Server until target reachability has been confirmed. The Client MUST thereafter follow the Neighbor Unreachability Detection (NUD) procedures in Section 7.3 of [RFC4861], and can resume sending packets via the Server at any time the direct path appears to be failing.

If the Client is unable to elicit a NUD response after MAX_RETRY attempts, it SHOULD consider the direct path unusable for forwarding purposes but still viable for ingress filtering purposes.

If reachability is confirmed, the Client SHOULD thereafter process any link-layer errors as a hint that the direct path to the target has either failed or has become intermittent.

On some AERO links, establishment and maintenance of a direct path between neighbors requires coordination such as through the Internet Key Exchange (IKEv2) protocol [RFC5996]. In those cases, link-specific hints of forward progress can be used instead of NS/NA to test neighbor reachability.

3.11. Mobility and Link-Layer Address Change Considerations

When a Client needs to change its link-layer address (e.g., due to a mobility event, due to a change in underlying network interface, etc.), it sends an immediate NS message forward to any of its correspondents (including the Server and any other Clients) which then discover the new link-layer address. The Client may instead send an immediate NA message if there is strong assurance that the correspondent would receive the message with no need for an

acknowledgement.

If two Clients change their link-layer addresses simultaneously, the NS/NA messages may be lost. In that case, the Clients SHOULD delete their respective neighbor cache entries and allow packets to again flow through their Server(s), which MAY result in a new AERO redirection exchange.

When a Client needs to change to a new Server, it performs a DHCPv6 "Release" message exchange with the delegating router via the old Server then sends a DHCPv6 "Request" message to the delegating router via the new Server. Note that this may result in a temporary service outage during Server "handovers".

3.12. Underlying Protocol Version Considerations

A source Client may connect only to an IPvX underlying network, while the target Client connects only to an IPvY underlying network. In that case, the source Client has no means for reaching the target directly (since they connect to underlying networks of different IP protocol versions) and so must ignore any Redirects and continue to send packets via the Server.

3.13. Multicast Considerations

When the underlying network does not support multicast, AERO nodes map IPv6 link-scoped multicast addresses (including "All_DHCP_Relay_Agents_and_Servers") to the underlying IP address of the AERO Server.

When the underlying network supports multicast, AERO nodes use the multicast address mapping specification found in [RFC2529] for IPv4 underlying networks and use a direct multicast mapping for IPv6 underlying networks. (In the latter case, "direct multicast mapping" means that if the IPv6 multicast destination address of the inner packet is "M", then the IPv6 multicast destination address of the encapsulating header is also "M".)

3.14. Operation on Server-less AERO Links

In some AERO link scenarios, there may be no Server on the link and/or no need for Clients to use a Server as an intermediary trust anchor. In that case, Clients can establish neighbor cache entries and IPv6 routes by performing direct Client-to-Client exchanges, and some other form of trust basis must be applied so that each Client can verify that the prospective neighbor is authorized to use its claimed prefix.

When there is no Server on the link, Clients must arrange to receive prefix delegations and publish the delegations via a secure prefix discovery service through some means outside the scope of this document.

3.15. Other Considerations

IPv6 hosts serviced by an AERO Client can reach IPv4-only services via a NAT64 gateway [RFC6146] within the IPv6 network.

AERO nodes can use the Default Address Selection Policy with DHCPv6 option [RFC7078] the same as on any IPv6 link.

All other (non-multicast) functions that operate over ordinary IPv6 links operate in the same fashion over AERO links.

4. Implementation Status

An early implementation is available at: http://linkupnetworks.com/seal/sealv2-1.0.tgz.

5. IANA Considerations

This document uses the UDP Service Port Number 8060 reserved by IANA for AERO in [RFC6706]. Therefore, there are no new IANA actions required for this document.

6. Security Considerations

AERO link security considerations are the same as for standard IPv6 Neighbor Discovery [RFC4861] except that AERO improves on some aspects. In particular, AERO is dependent on a trust basis between AERO Clients and Servers, where the Clients only engage in the AERO mechanism when it is facilitated by a trust anchor.

AERO links must be protected against link-layer address spoofing attacks in which an attacker on the link pretends to be a trusted neighbor. Links that provide link-layer securing mechanisms (e.g., WiFi networks) and links that provide physical security (e.g., enterprise network LANs) provide a first line of defense that is often sufficient. In other instances, securing mechanisms such as Secure Neighbor Discovery (SeND) [RFC3971] or IPsec [RFC4301] may be necessary.

AERO Clients MUST ensure that their connectivity is not used by

unauthorized nodes to gain access to a protected network. (This concern is no different than for ordinary hosts that receive an IP address delegation but then "share" the address with unauthorized nodes via an IPv6/IPv6 NAT function.)

7. Acknowledgements

Discussions both on the v6ops list and in private exchanges helped shape some of the concepts in this work. Individuals who contributed insights include Mikael Abrahamsson, Fred Baker, Stewart Bryant, Brian Carpenter, Brian Haberman, Joel Halpern, Sascha Hlusiak, Lee Howard and Joe Touch. Members of the IESG also provided valuable input during their review process that greatly improved the document. Special thanks go to Stewart Bryant, Joel Halpern and Brian Haberman for their shepherding guidance.

This work has further been encouraged and supported by Boeing colleagues including Keith Bartley, Balaguruna Chidambaram, Jeff Holland, Cam Brodie, Yueli Yang, Wen Fang, Ed King, Mike Slane, Kent Shuey, Gen MacLean, and other members of the BR&T and BIT mobile networking teams.

Earlier works on NBMA tunneling approaches are found in [RFC2529][RFC5214][RFC5569].

8. References

8.1. Normative References

[RFC0768] Postel, J., "User Datagram Protocol", STD 6, RFC 768, August 1980.

[RFC0791] Postel, J., "Internet Protocol", STD 5, RFC 791, September 1981.

[RFC0792] Postel, J., "Internet Control Message Protocol", STD 5, RFC 792, September 1981.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2460] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, December 1998.

[RFC2473] Conta, A. and S. Deering, "Generic Packet Tunneling in IPv6 Specification", RFC 2473, December 1998.

[RFC3633] Troan, O. and R. Droms, "IPv6 Prefix Options for Dynamic Host Configuration Protocol (DHCP) version 6", RFC 3633, December 2003.

[RFC4213] Nordmark, E. and R. Gilligan, "Basic Transition Mechanisms for IPv6 Hosts and Routers", RFC 4213, October 2005.

[RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, September 2007.

[RFC4862] Thomson, S., Narten, T., and T. Jinmei, "IPv6 Stateless Address Autoconfiguration", RFC 4862, September 2007.

[RFC6434] Jankiewicz, E., Loughney, J., and T. Narten, "IPv6 Node Requirements", RFC 6434, December 2011.

8.2. Informative References

[IRON] Templin, F., "The Internet Routing Overlay Network (IRON)", Work in Progress, June 2012.

[RFC0879] Postel, J., "TCP maximum segment size and related topics", RFC 879, November 1983.

[RFC2529] Carpenter, B. and C. Jung, "Transmission of IPv6 over IPv4 Domains without Explicit Tunnels", RFC 2529, March 1999.

[RFC3315] Droms, R., Bound, J., Volz, B., Lemon, T., Perkins, C., and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 3315, July 2003.

[RFC3971] Arkko, J., Kempf, J., Zill, B., and P. Nikander, "SEcure Neighbor Discovery (SEND)", RFC 3971, March 2005.

[RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, December 2005.

[RFC4821] Mathis, M. and J. Heffner, "Packetization Layer Path MTU Discovery", RFC 4821, March 2007.

[RFC4963] Heffner, J., Mathis, M., and B. Chandler, "IPv4 Reassembly Errors at High Data Rates", RFC 4963, July 2007.

[RFC5214] Templin, F., Gleeson, T., and D. Thaler, "Intra-Site Automatic Tunnel Addressing Protocol (ISATAP)", RFC 5214, March 2008.

[RFC5569] Despres, R., "IPv6 Rapid Deployment on IPv4 Infrastructures (6rd)", RFC 5569, January 2010.

[RFC5996] Kaufman, C., Hoffman, P., Nir, Y., and P. Eronen, "Internet Key Exchange Protocol Version 2 (IKEv2)", RFC 5996, September 2010.

[RFC6146] Bagnulo, M., Matthews, P., and I. van Beijnum, "Stateful NAT64: Network Address and Protocol Translation from IPv6 Clients to IPv4 Servers", RFC 6146, April 2011.

[RFC6204] Singh, H., Beebee, W., Donley, C., Stark, B., and O. Troan, "Basic Requirements for IPv6 Customer Edge Routers", RFC 6204, April 2011.

[RFC6438] Carpenter, B. and S. Amante, "Using the IPv6 Flow Label for Equal Cost Multipath Routing and Link Aggregation in Tunnels", RFC 6438, November 2011.

[RFC6691] Borman, D., "TCP Options and Maximum Segment Size (MSS)", RFC 6691, July 2012.

[RFC6706] Templin, F., "Asymmetric Extended Route Optimization (AERO)", RFC 6706, August 2012.

[RFC6864] Touch, J., "Updated Specification of the IPv4 ID Field", RFC 6864, February 2013.

[RFC6935] Eubanks, M., Chimento, P., and M. Westerlund, "IPv6 and UDP Checksums for Tunneled Packets", RFC 6935, April 2013.

[RFC6936] Fairhurst, G. and M. Westerlund, "Applicability Statement for the Use of IPv6 UDP Datagrams with Zero Checksums", RFC 6936, April 2013.

[RFC6980] Gont, F., "Security Implications of IPv6 Fragmentation with IPv6 Neighbor Discovery", RFC 6980, August 2013.

[RFC7078] Matsumoto, A., Fujisaki, T., and T. Chown, "Distributing Address Selection Policy Using DHCPv6", RFC 7078, January 2014.

Appendix A. AERO Server and Relay Interworking

Figure 3 depicts a reference AERO operational scenario with a single Server on the AERO link. In order to support scaling to larger numbers of nodes, the AERO link can deploy multiple Servers and

Relays, e.g., as shown in Figure 7.

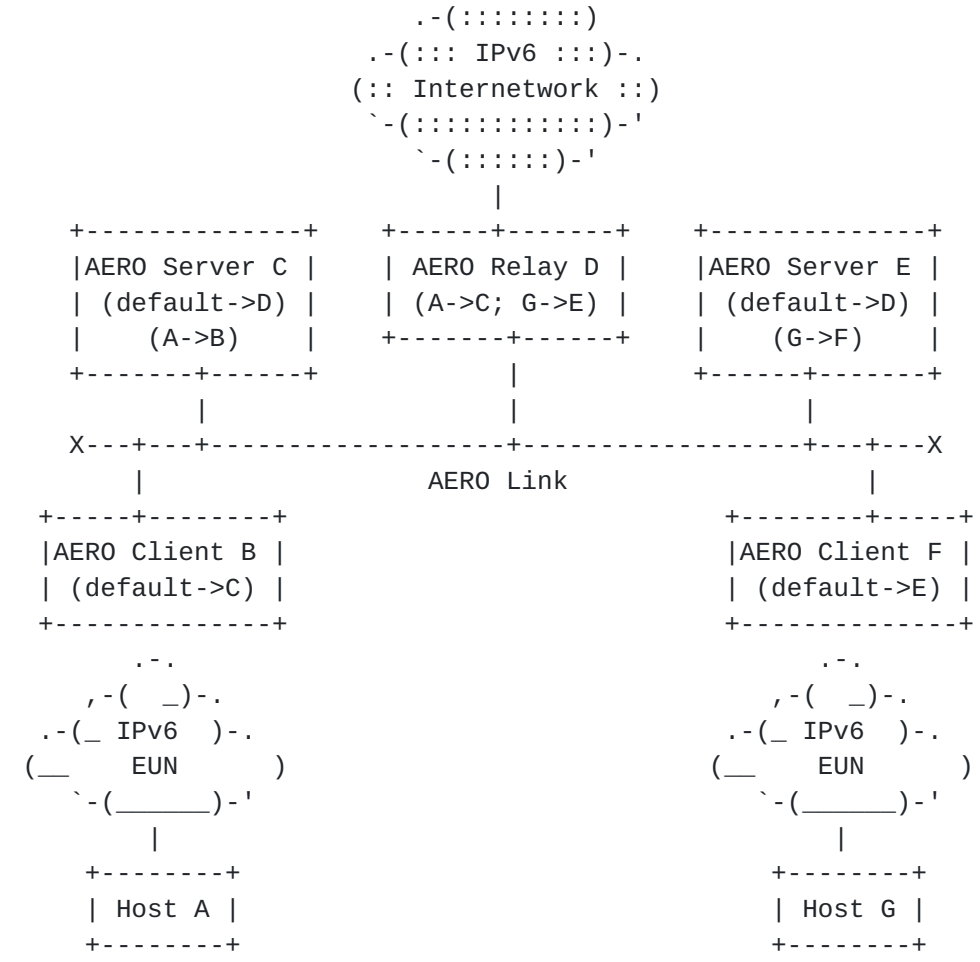

Figure 7: AERO Server/Relay Interworking

In this example, AERO Client ('B') associates with AERO Server ('C'), while AERO Client ('F') associates with AERO Server ('E'). Furthermore, AERO Servers ('C') and ('E') do not associate with each other directly, but rather have an association with AERO Relay ('D') (i.e., a router that has full topology information concerning its associated Servers and their Clients). Relay ('D') connects to the AERO link, and also connects to the native IPv6 Internetwork.

When host ('A') sends a packet toward destination host ('G'), IPv6 forwarding directs the packet through the EUN to Client ('B'), which forwards the packet to Server ('C') in absence of more-specific forwarding information. Server ('C') forwards the packet, and it also generates an AERO Predirect message that is then forwarded through Relay ('D') to Server ('E'). When Server ('E') receives the message, it forwards the message to Client ('F').

After processing the AERO Predirect message, Client ('F') sends an AERO Redirect message to Server ('E'). Server ('E'), in turn, forwards the message through Relay ('D') to Server ('C'). When Server ('C') receives the message, it forwards the message to Client ('B') informing it that host 'G's EUN can be reached via Client ('F'), thus completing the AERO redirection.

The network layer routing information shared between Servers and Relays must be carefully coordinated in a manner outside the scope of this document. In particular, Relays require full topology information, while individual Servers only require partial topology information (i.e., they only need to know the EUN prefixes associated with their current set of Clients). See [IRON] for an architectural discussion of routing coordination between Relays and Servers.

Author's Address

Fred L. Templin (editor)
Boeing Research & Technology
P.O. Box 3707
Seattle, WA 98124
USA

Email: fltemplin@acm.org