Network Working Group
Internet-Draft
Obsoletes: rfc6706 (if approved)
Intended status: Standards Track
Expires: December 29, 2014

F. Templin, Ed.
Boeing Research & Technology

June 27, 2014

Transmission of IP Packets over AERO Links
draft-templin-aerolink-28.txt

Abstract

This document specifies the operation of IP 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 that provide forwarding services to reach off-link destinations and redirection services for route optimization. AERO provides an IPv6 link-local address format known as the AERO address that supports operation of the IPv6 Neighbor Discovery (ND) protocol and links IPv6 ND to IP routing. Admission control and provisioning are supported by the Dynamic Host Configuration Protocol for IPv6 (DHCPv6), and node mobility is naturally supported through dynamic neighbor cache updates. Although IPv6 ND messaging is used in the control plane, both IPv4 and IPv6 are supported in the data plane.

Table of Contents

1. Introduction

This document specifies the operation of IP over tunnel virtual Non-Broadcast, Multiple Access (NBMA) links using Asymmetric Extended Route Optimization (AERO). 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. Nodes attached to AERO links can exchange packets via trusted intermediate routers that provide forwarding services to reach off-link destinations and redirection services for route optimization that addresses the requirements outlined in [RFC5522].

AERO provides an IPv6 link-local address format known as the AERO address that supports operation of the IPv6 Neighbor Discovery (ND) [RFC4861] protocol and links IPv6 ND to IP routing. Admission control and provisioning are supported by the Dynamic Host Configuration Protocol for IPv6 (DHCPv6) [RFC3315], and node mobility is naturally supported through dynamic neighbor cache updates. Although IPv6 ND message signalling is used in the control plane, both IPv4 and IPv6 are supported in the data plane. 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 the overlay IP layer.

AERO interface
: a node's attachment to an AERO link.

AERO address
: an IPv6 link-local address constructed as specified in Section 3.2 and assigned to a Client's AERO interface.

AERO node

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

AERO Client ("Client")
a node that assigns an AERO address on an AERO interface and receives an IP prefix delegation.

AERO Server ("Server")
a node that assigns the IPv6 link-local subnet router anycast address (fe80::) and an administratively provisioned IPv6 link-local unicast address on an AERO interface over which it can provide default forwarding and redirection services for AERO Clients.

AERO Relay ("Relay")
a node that relays IP packets between Servers on the same AERO link, and/or that forwards IP packets between the AERO link and the native Internetwork. 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 IP packets.

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

link-layer address
an IP 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 link-layer address. Link-layer addresses are used as the encapsulation header source and destination addresses.

network layer address
the source or destination address of the encapsulated IP packet.

end user network (EUN)

an internal virtual or external edge IP network that an AERO Client connects to the AERO interface.

end user network prefix
an IP prefix delegated to an end user network.

aggregated prefix
an IP prefix assigned to the AERO link and from which end user network prefixes are derived. (For example, and end user network prefix 2001:db8:1:2::/64 is derived from the aggregated prefix 2001:db8::/32.)

Throughout the document, the simple terms "Client", "Server" and "Relay" refer to "AERO Client", "AERO Server" and "AERO Relay", respectively. Capitalization is used to distinguish these terms from DHCPv6 client/server/relay. This is an important distinction, since an AERO Server may be a DHCPv6 relay, and an AERO Relay may be a DHCPv6 server.

The terminology of [RFC4861] (including the names of node variables and protocol constants) applies to this document. Also throughout the document, the term "IP" is used to generically refer to either Internet Protocol version (i.e., IPv4 or IPv6).

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 IP 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 Internetwork. The relaying process entails re-encapsulation of IP packets that were received from a first AERO node and are to be forwarded without modification to a second AERO node.

AERO Servers 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 IP prefixes through a DHCPv6 PD exchange via AERO Servers over the AERO link. (Each

client MAY associate with multiple Servers, but associating with many Servers may result in excessive control message overhead.) Each IPv6 AERO Client receives at least a /64 IPv6 prefix delegation, and may receive even shorter prefixes. Similarly, each IPv4 AERO Client receives at least a /32 IPv4 prefix delegation (i.e., a singleton IPv4 address), and may receive even shorter prefixes.

AERO Clients that act as routers sub-delegate portions of their received prefix delegations to links on EUNs. End system applications on AERO Clients that act as routers bind to EUN interfaces (i.e., and not the AERO interface).

AERO Clients that act as ordinary hosts assign one or more IP 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 Client assigns the delegated prefix to a "black hole" route so that unused portions of the prefix are nullified. End system applications on AERO Clients that act as hosts bind directly to the AERO interface.

3.2. AERO Addresses

An AERO address is an IPv6 link-local address with an embedded IP prefix and assigned to a Client's AERO interface. The AERO address is formatted as follows:

    fe80::[IP prefix]

For IPv6, the AERO address begins with the prefix fe80::/64 and includes in its interface identifier the base 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 the AERO Client receives the IPv6 prefix delegation:

    2001:db8:1000:2000::/56

it constructs its AERO address as:

    fe80::2001:db8:1000:2000

For IPv4, the AERO address begins with the prefix fe80::/96 and includes in its interface identifier the base prefix taken from the Client's delegated IPv4 prefix. For example, if the AERO Client receives the IPv4 prefix delegation:

    192.0.2.32/28

it constructs its AERO address as:

fe80::192.0.2.32

The AERO address remains stable as the Client moves between topological locations, i.e., even if its link-layer addresses change.

NOTE: In some cases, prospective neighbors may not have a priori knowledge of the Client's delegated prefix length and may therefore send initial IPv6 ND messages with an AERO destination address that matches the delegated prefix but does not correspond to the base prefix. In that case, the Client MUST accept the address as equivalent to the base address, but then use the base address as the source address of any IPv6 ND message replies. For example, if the Client receives the IPv6 prefix delegation 2001:db8:1000:2000::/56 then subsequently receives an IPv6 ND message with destination address fe80::2001:db8:1000:2001, it accepts the message but uses fe80::2001:db8:1000:2000 as the source address of any IPv6 ND replies.

3.3. AERO Interface Characteristics

AERO interfaces use IP-in-IPv6 encapsulation [RFC2473] to exchange tunneled packets with AERO neighbors attached to an underlying IPv6 network, and use IP-in-IPv4 encapsulation [RFC2003][RFC4213] to exchange tunneled packets with AERO neighbors attached to an underlying IPv4 network. AERO interfaces can also operate over secured tunnel types such as IPsec [RFC4301] or TLS [RFC5246]. When Network Address Translator (NAT) traversal and/or filtering middlebox traversal may be necessary, a UDP header is further inserted immediately above the IP encapsulation header.

Servers assign the address fe80:: to their AERO interfaces as a link-local Subnet Router Anycast address. Servers and Relays also assign a link-local address fe80::ID to support the operation of the IPv6 ND protocol and the inter-Server/Relay routing system (see: Appendix A). Each fe80::ID address MUST be unique among all Servers and Relays on the AERO link, and MUST NOT collide with any potential AERO addresses (e.g., the addresses for Servers and Relays on the link could be assigned as fe80::1, fe80::2, fe80::3, etc.). Servers accept IPV6 ND messages with either fe80::ID or fe80:: as the IPv6 destination address, but MUST use the fe80::ID address as the IPv6 source address of any IPv6 ND messages they generate.

When a Client does not know the fe80::ID address of a Server, it can use fe80:: as a temporary destination address in IPv6 ND messages. The Client may also use fe80::, e.g., as the link-local address in a neighbor cache entry for a Server when the Server's fe80::ID address is not known in advance.

When a Client enables an AERO interface, it invokes DHCPv6 PD using the temporary IPv6 link-local source address fe80::ffff:ffff:ffff:ffff. After the Client receives a prefix delegation, it assigns the corresponding AERO address to the AERO interface and deprecates the temporary address, i.e., the Client invokes DHCPv6 to bootstrap the provisioning of a unique link-local address before invoking IPv6 ND.

AERO interfaces maintain a neighbor cache and use an adaptation of standard unicast IPv6 ND messaging. AERO interfaces use unicast Neighbor Solicitation (NS), Neighbor Advertisement (NA), Router Solicitation (RS) and Router Advertisement (RA) messages the same as for any IPv6 link. AERO interfaces use two redirection message types -- the first known as a Predirect message and the second being the standard Redirect message (see Section 3.9). AERO links further use link-local-only addressing; hence, Clients ignore any Prefix Information Options (PIOs) they may receive in RA messages.

AERO interface Redirect/Predirect messages include Target Link-Layer Address Options (TLLAOs) formatted as shown in Figure 1:

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |    Type = 2   |  Length = 3   |           Reserved            |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |    Link ID    |  Preference   |     UDP Port Number (or 0)    |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                                                               |
    +--                                                           --+
    |                                                               |
    +--                          IP Address                       --+
    |                                                               |
    +--                                                           --+
    |                                                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 1: AERO Target Link-Layer Address Option (TLLAO) Format

In this format, Link ID is an integer value between 0 and 255 corresponding to an underlying interface of the target node, and Preference is an integer value between 0 and 255 indicating the node's preference for this underlying interface, with 0 being highest preference and 255 being lowest. UDP Port Number and IP Address are set to the addresses used by the target node when it sends encapsulated packets over the underlying interface. When no UDP encapsulation is used, UDP Port Number is set to 0. When the

encapsulation IP address family is IPv4, IP Address is formed as an IPv4-compatible IPv6 address [RFC4291], i.e., 96 bits of leading 0's followed by a 32-bit IPv4 address

AERO interface Redirect/Predirect messages can both update and create neighbor cache entries, including link-layer address information. Redirect/Predirect messages SHOULD include a Timestamp option (see Section 5.3 of [RFC3971]) that other AERO nodes can use to verify the message time of origin.

AERO interface NS/NA/RS/RA messages used for neighbor reachability verification update timers in existing neighbor cache entires but do not update link-layer addresses nor create new neighbor cache entries. AERO interface unsolicited NA messages are used to update a neighbor's cached link-layer address for the sender, e.g., following a link-layer address change due to node mobility. AERO interface NS/RS messages SHOULD include a Nonce option (see Section 5.3 of [RFC3971]) that the recipient echoes back in the corresponding NA/RA response.

3.3.1. Coordination of Multiple Underlying Interfaces

AERO interfaces may be configured over multiple underlying interfaces. For example, common handheld devices have both wireless local area network ("WLAN") and cellular wireless links. These links are typically used "one at a time" with low-cost WLAN preferred and highly-available cellular wireless as a standby. In a more complex example, aircraft frequently have many wireless data link types (e.g. satellite-based, terrestrial, air-to-air directional, etc.) with diverse performance and cost properties.

If a Client's multiple underlying interfaces are used "one at a time" (i.e., all other interfaces are in standby mode while one interface is active), then Predirect/Redirect messages include only a single TLLAO with Link ID set to 0.

If the Client has multiple active underlying interfaces, then from the perspective of IPv6 ND it would appear to have a single link-local address with multiple link-layer addresses. In that case, Predirect/Redirect messages MAY include multiple TLLAOs -- each with a different Link ID that corresponds to an underlying interface of the Client. Further details on coordination of multiple active underlying interfaces are outside the scope of this specification.

3.4. AERO Interface Neighbor Cache Maintenace

Each AERO interface maintains a conceptual neighbor cache that includes an entry for each neighbor it communicates with on the AERO link, the same as for any IPv6 interface [RFC4861]. Neighbor cache entries are created and maintained as follows:

When an AERO Server relays a DHCPv6 Reply message to an AERO Client, it creates or updates a neighbor cache entry for the Client based on the AERO address corresponding to the Client's prefix delegation as the network-layer address and with the Client's encapsulation IP address and UDP port number as the link-layer address.

When an AERO Client receives a DHCPv6 Reply message from an AERO Server, it creates or updates a neighbor cache entry for the Server based on the Reply message link-local source address as the network-layer address, and the encapsulation IP source address and UDP source port number as the link-layer address.

When an AERO Client receives a valid Predirect message it creates or updates a neighbor cache entry for the Predirect target network-layer and link-layer addresses, and also creates an IP forwarding table entry for the predirected (source) prefix. The node then sets an "AcceptTime" variable for the neighbor and uses this value to determine whether messages received from the predirected neighbor can be accepted.

When an AERO Client receives a valid Redirect message it creates or updates a neighbor cache entry for the Redirect target network-layer and link-layer addresses, and also creates an IP forwarding table entry for the redirected (destination) prefix. The node then sets a "ForwardTime" variable for the neighbor and uses this value to determine whether packets can be sent directly to the redirected neighbor. The node also maintains a constant value MAX_RETRY to limit the number of keepalives sent when a neighbor may have gone unreachable.

When an AERO Client receives a valid NS message it (re)sets AcceptTime for the neighbor to ACCEPT_TIME.

When an AERO Client receives a valid solicited NA message, it (re)sets ForwardTime for the neighbor to FORWARD_TIME. (When an AERO Client receives a valid unsolicited NA message, it updates the neighbor's link-layer address but DOES NOT reset ForwardTime.)

It is RECOMMENDED that FORWARD_TIME be set to the default constant value 30 seconds to match the default REACHABLE_TIME value specified for IPv6 ND [RFC4861].

It is RECOMMENDED that ACCEPT_TIME be set to the default constant value 40 seconds to allow a 10 second window so that the AERO redirection procedure can converge before AcceptTime decrements below FORWARD_TIME.

It is RECOMMENDED that MAX_RETRY be set to 3 the same as described for IPv6 ND address resolution in Section 7.3.3 of [RFC4861].

Different values for FORWARD_TIME, ACCEPT_TIME, and MAX_RETRY MAY be administratively set, if necessary, to better match the AERO link's performance characteristics; however, if different values are chosen, all nodes on the link MUST consistently configure the same values. In particular, ACCEPT_TIME SHOULD be set to a value that is sufficiently longer than FORWARD_TIME to allow the AERO redirection procedure to converge.

3.5. AERO Interface Data Origin Authentication

AERO nodes use a simple data origin authentication for encapsulated packets they receive from other nodes. In particular, AERO nodes accept encapsulated packets with a link-layer source address belonging to one of their current AERO Servers and 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 IP forwarding table entry 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 and AcceptTime is non-zero.

Note that this simple data origin authentication only applies to environments in which link-layer addresses cannot be spoofed. Additional security mitigations may be necessary in other environments.

3.6. AERO Interface MTU Considerations

The AERO link Maximum Transmission Unit (MTU) is 64KB minus the encapsulation overhead for IPv4 as the link-layer [RFC0791] and 4GB minus the encapsulation overhead for IPv6 as the link layer [RFC2675]. This is the most that IPv4 and IPv6 (respectively) can convey within the constraints of protocol constants, but actual sizes available for tunneling will frequently be much smaller.

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 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 the underlying interface MTU minus the encapsulation overhead, and 1500 bytes. (If there are multiple underlying interfaces, the node sets the AERO interface MTU according to the largest underlying interface MTU, or 64KB /4G minus the encapsulation overhead if the largest MTU cannot be determined.) 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 IP fragmentation to fragment the encapsulated packet into two pieces (where the first fragment contains 1024 bytes of the original IP packet) then admits the fragments into the tunnel. If the link-layer protocol is IPv4, the node admits each fragment 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 NS message with a Nonce option plus trailing padding octets added to a length of 1500 bytes without including the length of the padding in the IPv6 Payload Length field. The node then encapsulates the NS in the encapsulation headers (while including the length of the padding in the encapsulation header length fields), sets DF to 1 (for IPv4) and sends the padded NS message to the neighbor. If the neighbor returns an NA message with a correct Nonce value, the node may then send

whole packets within this size range and (for IPv4) relax the rate limiting requirement. (Note that the trailing padding SHOULD NOT be included within the Nonce option itself but rather as padding beyond the last option in the NS message; otherwise, the (large) Nonce option would be echoed back in the solicited NA message and may be lost at a link with a small MTU along the reverse path.)

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 without probing 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. Probe sizes larger than 1500 bytes MAY be used, but may be unnecessary since original sources are expected to implement [RFC4821] when sending large packets.

3.7. AERO Interface Encapsulation, Re-encapsulation and Decapsulation

AERO interfaces encapsulate IP 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 [RFC2003][RFC2473][RFC4213][RFC4301][RFC5246] except that the interface copies the "TTL/Hop Limit", "Type of Service/Traffic Class" and "Congestion Experienced" values in the packet's IP header into the corresponding fields in the encapsulation 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 encapsulation header into the corresponding fields in the new encapsulation header (i.e., the values are transferred between encapsulation headers and *not* copied from the encapsulated packet's network-layer header).

When AERO UDP encapsulation is used, the AERO interface encapsulates the packet per the specifications in [RFC2003][RFC2473][RFC4213] except that it inserts a UDP header between the encapsulation header and the packet's IP 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 checksum field to zero (see:

[RFC6935][RFC6936]) and sets the UDP length field to the length of the IP packet plus 8 bytes for the UDP header itself. For packets sent via a Server, the AERO interface sets the UDP destination port to 8060 (i.e., the IANA-registered port number for AERO) when AERO-only encapsulation is used. For packets sent to a neighboring Client, the AERO interface sets the UDP destination port to the port value stored in the neighbor cache entry for this neighbor.

The AERO interface next sets the IP protocol number in the encapsulation header to the appropriate value for the first protocol layer within the encapsulation (e.g., IPv4, IPv6, UDP, IPsec, etc.). When IPv6 is used as the encapsulation protocol, the interface then sets the flow label value in the encapsulation header the same as described in [RFC6438]. When IPv4 is used as the encapsulation protocol, the AERO interface sets the DF bit as discussed in Section 3.6.

AERO interfaces decapsulate packets destined either to the node itself or to a destination reached via an interface other than the receiving AERO interface. When AERO UDP encapsulation is used (i.e., when a UDP header with destination port 8060 is present) the interface examines the first octet of the encapsulated packet. If the most significant four bits of the first octet encode the value '0110' (i.e., the version number value for IPv6) or the value '0100' (i.e., the version number value for IPv4), the packet is accepted and the encapsulating UDP header is discarded; otherwise, the packet is discarded.

Further decapsulation then proceeds according to the appropriate tunnel type [RFC2003][RFC2473][RFC4213][RFC4301][RFC5246].

3.8. AERO Router Discovery, Prefix Delegation and Address Configuration

3.8.1. AERO Client Behavior

AERO Clients discover the link-layer addresses of AERO Servers 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. After discovering the link-layer addresses, the Client associates with one or more of the corresponding Servers.

To associate with a Server, the Client acts as a requesting router to request an IP prefix through DHCPv6 PD [RFC3315][RFC3633][RFC6355] using fe80::ffff:ffff:ffff:ffff as the IPv6 source address (see Section 3.3), 'All_DHCP_Relay_Agents_and_Servers' as the IPv6

destination address and the link-layer address of the Server as the link-layer destination address. The Client includes a DHCPv6 Unique Identifier (DUID) in the Client Identifier option of its DHCPv6 messages (as well as a DHCPv6 authentication option if necessary) to identify itself to the DHCPv6 server. If the Client is pre-provisioned with an IP prefix associated with the AERO service, it MAY also include the prefix in its DHCPv6 PD Request to indicate its preferred prefix to the DHCPv6 server. The Client then sends the encapsulated DHCPv6 request via an underlying interface.

When the Client receives its prefix delegation via a Reply from the DHCPv6 server, it creates a neighbor cache entry with the Server's link-local address (i.e., fe80::ID) as the network-layer address and the Server's encapsulation address as the link-layer addresses. Next, the Client assigns the AERO address derived from the delegated prefix to the AERO interface and sub-delegates the prefix to nodes and links within its attached EUNs (the AERO address thereafter remains stable as the Client moves). The Client also sets both AcceptTime and ForwardTime for each Server to the constant value REACHABLE_TIME. The Client further renews its prefix delegation by performing DHCPv6 Renew/Reply exchanges with its AERO address as the IPv6 source address, 'All_DHCP_Relay_Agents_and_Servers' as the IPv6 destination address, the link-layer address of a Server as the link-layer destination address and the same DUID and authentication information. If the Client wishes to associate with multiple Servers, it can perform DHCPv6 Renew/Reply exchanges via each of the Servers.

The Client then sends an RS message to each of its associated Servers to receive an RA message with a default router lifetime and any other link-specific parameters. When the Client receives an RA message, it configures or updates a default route according to the default router lifetime but ignores any Prefix Information Options (PIOs) included in the RA message since the AERO link is link-local-only. The Client further ignores any RS messages it might receive, since only Servers may process RS messages.

The Client then sends periodic RS messages to each Server before AcceptTime and ForwardTime expire to obtain new RA messages for Neighbor Unreachability Detection (NUD), to refresh any network state, and to update the default router lifetime and any other link-specific parameters. When the Client receives a new RA message, it resets AcceptTime and ForwardTime to REACHABLE_TIME. The Client can also forward IP packets destined to networks beyond its local EUNs via a Server as a default router. The Server may in turn return a redirection message informing the Client of a neighbor on the AERO link that is topologically closer to the final destination (see Section 3.9).

Note that, since the Client's AERO address is configured from the unique DHCPv6 prefix delegation it receives, there is no need for Duplicate Address Detection (DAD) on AERO links. Other nodes maliciously attempting to hijack an authorized Client's AERO address will be denied access to the network by the DHCPv6 server due to an unacceptable link-layer address and/or security parameters (see: Security Considerations).

3.8.2. AERO Server Behavior

AERO Servers configure a DHCPv6 relay function on their AERO links. AERO Servers arrange to add their encapsulation layer IP addresses (i.e., their link-layer addresses) to the DNS resource records for the FQDN "linkupnetworks.domainname" before entering service.

When an AERO Server relays a prospective Client's DHCPv6 PD messages to the DHCPv6 server, it wraps each message in a "Relay-forward" message per [RFC3315] and includes a DHCPv6 Interface Identifier option that encodes a value that identifies the AERO link to the DHCPv6 server. Without creating internal state, the Server then includes the Client's link-layer address in a DHCPv6 Client Link Layer Address Option (CLLAO) [RFC6939] with the link-layer address format shown in Figure 1 (i.e., Link ID followed by Preference followed by UDP Port Number followed by IP Address). The Server sets the CLLAO 'option-length' field to 22 (2 plus the length of the link-layer address) and sets the 'link-layer type' field to TBD (see: IANA Considerations). The Server finally includes a DHCPv6 Echo Request Option (ERO) [RFC4994] that encodes the option code for the CLLAO in a 'requested-option-code-n' field, then relays the message to the DHCPv6 server. The CLLAO information will therefore subsequently be echoed back in the DHCPv6 server's "Relay-reply" message.

When the DHCPv6 server issues the prefix delegation in a "Relay-reply" message via the AERO Server (acting as a DHCPv6 relay), the Server obtains the Client's link-layer address from the echoed CLLAO option and also obtains the Client's delegated prefix from the message. The Server then creates a neighbor cache entry for the Client's AERO address with the Client's link-layer address as the link-layer address for the neighbor cache entry. The neighbor cache entry is created with both AcceptTime and ForwardTime set to REACHABLE_TIME, since the Client will continue to send RS messages within REACHABLE_TIME seconds as long as it wishes to remain associated with this Server.

The Server also configures an IP forwarding table entry that lists the Client's AERO address as the next hop toward the delegated IP prefix with a lifetime derived from the DHCPv6 lease lifetime. The Server finally injects the Client's prefix as an IP route into the

inter-Server/Relay routing system (see: Appendix A) then relays the DHCPv6 message to the Client while using fe80::ID as the IPv6 source address, the link-local address found in the "peer address" field of the Relay-reply message as the IPv6 destination address, and the Client's link-layer address as the destination link-layer address.

Servers respond to NS/RS messages from Clients on their AERO interfaces by returning an NA/RA message. The Server SHOULD NOT include PIOs in the RA messages it sends to Clients, since the Client will ignore any such options. When the Server receives an NS/RS message from the Client, it resets AcceptTime and ForwardTime to REACHABLE_TIME.

Servers ignore any RA messages they may receive from a Client, but they MAY examine RA messages received from other Servers for consistency verification purposes. Servers do not send NS messages for the purpose of updating Client neighbor cache timers, since Clients are responsible for refreshing neighbor cache state.

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.

3.9. AERO Redirection

3.9.1. Reference Operational Scenario

Figure 2 depicts the AERO redirection reference operational scenario, using IPv6 addressing as the example (while not shown, a corresponding example for IPv4 addressing can be easily constructed). The figure shows an AERO Server('A'), two AERO Clients ('B', 'C') and three ordinary IPv6 hosts ('D', 'E', 'F'):

```
                    .-(::::::::)
                 .-(:::: IP ::::)-.   +-------------+
                (:: Internetwork ::)--|    Host F   |
                 `-(::::::::::::)-'   +-------------+
                    `-(::::::)-'       2001:db8:2::1
                         |
                  +--------------+
                  | AERO Server A|
                  | (D->B; E->C) |
                  +--------------+
                      fe80::ID
                       L2(A)
                         |
       X-----+-----------+-----------+--------X
             |       AERO Link       |
            L2(B)                   L2(C)
     fe80::2001:db8:0:0      fe80::2001:db8:1:0         .-.
     +--------------+         +--------------+       ,-(  _)-.
     | AERO Client B|         | AERO Client C|    .-(_  IP   )-.
     | (default->A) |         | (default->A) |--(__    EUN      )
     +--------------+         +--------------+     `-(______)-'
     2001:DB8:0::/48           2001:DB8:1::/48           |
             |                                     2001:db8:1::1
            .-.                                   +-------------+
         ,-(  _)-.      2001:db8:0::1             |    Host E   |
      .-(_  IP   )-.   +-------------+            +-------------+
    (__    EUN      )--|    Host D   |
       `-(______)-'    +-------------+
```

Figure 2: AERO Reference Operational Scenario

In Figure 2, AERO Server ('A') connects to the AERO link and connects to the IP Internetwork, either directly or via an AERO Relay (not shown). Server ('A') assigns the address fe80::ID 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') receives the prefix 2001:db8:0::/48 in a DHCPv6 PD exchange via AERO Server ('A') then assigns the address fe80::2001:db8:0:0 to its AERO interface with link-layer address L2(B). Client ('B') configures a default route and neighbor cache entry via the AERO interface with next-hop address fe80::ID and link-layer address L2(A), then sub-delegates the prefix 2001:db8:0::/48 to its attached EUNs. IPv6 host ('D') connects to the EUN, and configures the address 2001:db8:0::1.

AERO Client ('C') receives the prefix 2001:db8:1::/48 in a DHCPv6 PD exchange via AERO Server ('A') then assigns the address

fe80::2001:db8:1:0 to its AERO interface with link-layer address L2(C). Client ('C') configures a default route and neighbor cache entry via the AERO interface with next-hop address fe80::ID 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 a 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:db8:2::1 to its IPv6 link interface.

3.9.2. Classical Redirection Approaches

With reference to Figure 2, when the source host ('D') sends a packet to 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 ('C'), where the packet is finally forwarded to 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 ('C') 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 ('C') (i.e., "classical redirection"). In that case, Client ('C') has no way of knowing that Client ('B') is authorized to forward packets from the claimed source address, and it may simply elect to drop the packets. Also, Client ('B') has no way of knowing whether Client ('C') 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 ('C') 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 ('C') separately, via independent redirection control messages (i.e., "augmented redirection"). In that case, if Client ('B') receives the redirection control message but Client ('C') does not, subsequent packets sent by Client ('B') could be dropped due to filtering since Client ('C') would not have a route to verify the claimed source address. Also, if Client ('C') receives the redirection control message but Client ('B') does not, subsequent packets sent in the reverse direction by Client ('C') would be lost.

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

3.9.3. Concept of Operations

Again, with reference to Figure 2, when source host ('D') 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 and forwards the packet out the same AERO interface toward Client ('C') and also sends an AERO "Predirect" message forward to Client ('C') as specified in Section 3.9.5. The Predirect message includes Client ('B')'s network- and link-layer addresses as well as information that Client ('C') can use to determine the IP prefix used by Client ('B') . After Client ('C') 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 3.9.6. During the process, Client ('C') also creates or updates a neighbor cache entry for Client ('B') and creates an IP forwarding table entry for Client ('B')'s prefix.

When Server ('A') receives the Redirect message, it re-encapsulates the message and forwards it on to Client ('B') as specified in Section 3.9.7. The message includes Client ('C')'s network- and link-layer addresses as well as information that Client ('B') can use to determine the IP prefix used by Client ('C'). After Client ('B') receives the Redirect message, it processes the message as specified in Section 3.9.8. During the process, Client ('B') also creates or updates a neighbor cache entry for Client ('C') and creates an IP forwarding table entry for Client ('C')'s prefix.

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

3.9.4. 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. (For IPv6, valid values for the Prefix Length field are 0 through 64; for IPv4, valid values are 0 through 32.) The Redirect/Predirect messages are formatted as shown in Figure 3:

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |  Type (=137)  |  Code (=0/1)  |          Checksum             |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                   Reserved                    | Prefix Length |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                                                               |
   +                                                               +
   |                                                               |
   +                       Target Address                          +
   |                                                               |
   +                                                               +
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                                                               |
   +                                                               +
   |                                                               |
   +                     Destination Address                       +
   |                                                               |
   +                                                               +
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |   Options ...
   +-+-+-+-+-+-+-+-+-+-+-+-
```

Figure 3: AERO Redirect/Predirect Message Format

3.9.5. Sending Predirects

When a Server forwards a packet from one of its associated Clients toward another AERO Client connected to the same AERO link, the Server sends a Predirect message forward toward the destination Client instead of sending a Redirect message back to the source Client.

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

- o the link-layer source address is set to 'L2(A)' (i.e., the link-layer address of Server ('A')).
- o the link-layer destination address is set to 'L2(C)' (i.e., the link-layer address of Client ('C')).

o the network-layer source address is set to fe80::2001:db8:0:0 (i.e., the AERO address of Client ('B')).

o the network-layer destination address is set to fe80::2001:db8:1:0 (i.e., the AERO address of Client ('C')).

o the Type is set to 137.

o the Code is set to 1 to indicate "Predirect".

o the Prefix Length is set to the length of the prefix to be applied to the Target Address.

o the Target Address is set to fe80::2001:db8:0:0 (i.e., the AERO address of Client ('B')).

o the Destination Address is set to the source address of the originating packet that triggered the Predirection event. (If the originating packet is an IPv4 packet, the address is constructed in IPv4-compatible IPv6 address format).

o the message includes a TLLAO with Link ID and Preference set to appropriate values for Client ('B')'s underlying interface, and with UDP Port Number and IP Address set to 'L2(B)'.

o the message includes a Timestamp option.

o 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.

Note that the reference operational scenario applies to the case when the source and destination Clients are associated with the same Server. When the source and destination Clients are associated with different Servers, the source Client's Server forwards the packets and Predirect messages to a Relay which in turn forwards them toward the destination Client. In that case, the Server sets the Predirect link-layer destination address to the link-layer address of the Relay.

Servers therefore require knowledge of all aggregated IP prefixes associated with the AERO link so that they can determine when a prospective destination Client is on-link. See Appendix A for a discussion of AERO Server/Relay interworking.

3.9.6. Processing Predirects and Sending Redirects

When Client ('C') receives a Predirect message, it accepts the message only if the message has a link-layer source address of the Server, i.e. 'L2(A)'. Client ('C') further accepts the message only if it is willing to serve as a redirection target. Next, Client ('C') validates the message according to the ICMPv6 Redirect message validation rules in Section 8.1 of [RFC4861], except that it accepts the message even though the network-layer source address is not that of it's current first-hop router.

In the reference operational scenario, when Client ('C') 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(es) of Client ('B'). Client ('C') then sets AcceptTime for the neighbor cache entry to ACCEPT_TIME. Next, Client ('C') applies the Prefix Length to the Destination Address and records the resulting prefix in its IP forwarding table.

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

- the link-layer source address is set to 'L2(C)' (i.e., the link-layer address of Client ('C')).

- 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 fe80::2001:db8:1:0 (i.e., the AERO address of Client ('C')).

- the network-layer destination address is set to fe80::2001:db8:0:0 (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 Address.

- the Target Address is set to fe80::2001:db8:1:0 (i.e., the AERO address of Client ('C')).

- the Destination Address is set to the destination address of the originating packet that triggered the Redirection event. (If the

originating packet is an IPv4 packet, the address is constructed in IPv4-compatible IPv6 address format).

- the message includes a TLLAO with Link ID and Preference set to appropriate values for Client ('C')'s underlying interface, and with UDP Port Number and IP Address set to '0'.
- the message includes a Timestamp option.
- 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 ('C') prepares the Redirect message, it sends the message to Server ('A').

3.9.7. Re-encapsulating and Relaying Redirects

When Server ('A') receives a Redirect message from Client ('C'), it validates the message according to the ICMPv6 Redirect message validation rules in Section 8.1 of [RFC4861] and also verifies that Client ('C') 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 by searching for the AERO address' embedded prefix in the IP routing table. If validation fails, Server ('A') discards the message; otherwise, it copies the correct UDP Port number and IP Address for Client ('C') into the (previously empty) TLLAO.

Server ('A') then examines the network-layer destination address of the message to determine the next hop toward the prefix of Client ('B') by searching for the AERO address' embedded prefix in the IP routing table. If the next hop is reached via the AERO interface, Server ('A') re-encapsulates the Redirect and relays it on to Client ('B') by changing the link-layer source address of the message to 'L2(A)' and changing the link-layer destination address to 'L2(B)'. Server ('A') finally forwards the re-encapsulated message to Client ('B') without decrementing the network-layer TTL/Hop Limit field.

While not shown in Figure 2, AERO Relays relay Redirect and Predirect messages in exactly this same fashion described above (see: Appendix A).

3.9.8. 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], except that it accepts the message even though the network-layer source address is not that of it's current first-hop router. 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 ('C') and stores the link-layer address found in the TLLAO as the link-layer address of Client ('C'). Client ('B') then sets the neighbor cache entry ForwardTime variable with timeout value FORWARD_TIME. Next, Client ('B') applies the Prefix Length to the Destination Address and records the resulting IP prefix in its IP forwarding table.

Now, Client ('B') has an IP forwarding table entry for Client('C')'s prefix and a neighbor cache entry with a valid ForwardTime value, while Client ('C') has an IP forwarding table entry for Client ('B')'s prefix with a valid AcceptTime value. Thereafter, Client ('B') may forward ordinary network-layer data packets directly to Client ("C") without involving Server ('A') and Client ('C') can verify that the packets came from an acceptable source. (In order for Client ('C') to forward packets to Client ('B') a corresponding Predirect/Redirect message exchange is required in the reverse direction.)

3.9.9. Server-Oriented Redirection

In some environments, the Server nearest the destination Client may need to serve as the redirection target, e.g., if direct Client-to-Client communications are not possible. In that case, the Server prepares the Redirect message the same as if it were the destination Client (see: Section 3.9.6), except that it writes its own link-layer address in the TLLAO option.

3.10. Neighbor Reachability Maintenance

AERO nodes send unicast NS messages to elicit solicited NA messages from neighbors the same as described for Neighbor Unreachability Detection (NUD) in [RFC4861]. When an AERO node sends an NS/NA message, it MUST use its link-local address as the IPv6 source address and the link-local address of the neighbor as the IPv6

destination address. When an AERO node receives an NS message or a solicited NA message, it accepts the message if it has a neighbor cache entry for the neighbor; otherwise, it ignores the message.

When a source Client is redirected to a target Client it SHOULD test the direct path by sending an initial NS message to elicit a solicited NA response. While testing the path, the source Client can optionally continue sending packets via the Server, maintain a small queue of packets until target reachability is confirmed, or (optimistically) allow packets to flow directly to the target. The source Client SHOULD thereafter continue to test the direct path to the target Client (see Section 7.3 of [RFC4861]) periodically in order to keep neighbor cache entries alive.

In particular, while the source Client is actively sending packets to the target Client it SHOULD also send NS messages separated by RETRANS_TIMER milliseconds in order to receive solicited NA messages. If the source Client is unable to elicit a solicited NA response from the target Client after MAX_RETRY attempts, it SHOULD set ForwardTime to 0 and resume sending packets via the Server which may or may not result in a new redirection event. Otherwise, the source Client considers the path usable and SHOULD thereafter process any link-layer errors as a hint that the direct path to the target Client has either failed or has become intermittent.

When a target Client receives an NS message from a source Client, it resets AcceptTime to ACCEPT_TIME if a neighbor cache entry exists; otherwise, it discards the NS message.

When a source Client receives a solicited NA message from a target Client, it resets ForwardTime to FORWARD_TIME if a neighbor cache entry exists; otherwise, it discards the NA message.

When ForwardTime for a neighbor cache entry expires, the source Client resumes sending any subsequent packets via the Server and may (eventually) receive a new Redirect message. When AcceptTime for a neighbor cache entry expires, the target Client discards any subsequent packets received directly from the source Client. When both ForwardTime and AcceptTime for a neighbor cache entry expire, the Client deletes both the neighbor cache entry and the corresponding IP forwarding table entry.

3.11. Mobility Management

When a Client needs to change its link-layer address, e.g., due to a mobility event, it performs an immediate DHCPv6 Renew/Reply via each of its Servers using the new link-layer address as the source. The DHCPv6 Server will re-authenticate the Client and (assuming

authentication succeeds) the DHCPv6 Renew/Reply exchange will update each Server's neighbor cache.

Next, the Client sends unsolicited NA messages to each of its active neighbors using the same procedures as specified in Section 7.2.6 of [RFC4861], except that it sends the messages as unicast to each neighbor via a Server instead of multicast. In this process, the Client should send no more than MAX_NEIGHBOR_ADVERTISEMENT messages separated by no less than RETRANS_TIMER seconds to each neighbor.

With reference to Figure 2, Client ('C') sends unicast unsolicited NA messages to Client ('B') via Server ('A') as follows:

- o the link-layer source address is set to 'L2(C)' (i.e., the link-layer address of Client ('C')).
- o the link-layer destination address is set to 'L2(A)' (i.e., the link-layer address of Server ('A')).
- o the network-layer source address is set to fe80::2001:db8:1:0 (i.e., the AERO address of Client ('C')).
- o the network-layer destination address is set to fe80::2001:db8:0:0 (i.e., the AERO address of Client ('B')).
- o the Type is set to 136.
- o the Code is set to 0.
- o the Solicited flag is set to 0.
- o the Override flag is set to 1.
- o the Target Address is set to fe80::2001:db8:1:0 (i.e., the AERO address of Client ('C')).
- o the message includes a TLLAO with Link ID and Preference set to appropriate values for Client ('C')'s underlying interface, and with UDP Port Number and IP Address set to '0'.
- o the message includes a Timestamp option.

When Server ('A') receives the NA message, it relays the message in the same way as described for relaying Redirect messages in Section 3.9.7. In particular, Server ('A') copies the correct UDP port number and IP address into the TLLAO, changes the link-layer source address to its own address, changes the link-layer destination address to the address of Client ('B'), then forwards the NA message

based on an IP route matching the AERO address in the network-layer destination address. When Client ('B') receives the NA message, it accepts the message only if it already has a neighbor cache entry for Client ('C') then updates the link-layer address for Client ('C') based on the address in the TLLAO. However, Client ('B') MUST NOT update ForwardTime since it has no way of knowing whether Client ('C') has updated AcceptTime.

When a Client associates with a new Server, it issues a new DHCPv6 Renew message via the new Server as the DHCPv6 relay. The new Server then relays the message to the DHCPv6 server and processes the resulting exchange. After the Client receives the resulting DHCPv6 Reply message, it sends an RS message to the new Server to receive a new RA message.

When a Client disassociates with an existing Server, it sends a "terminating RS" message to the old Server. The terminating RS message is prepared exactly the same as for an ordinary RS message, except that the Code field contains the value '1'. When the old Server receives the terminating RS message, it withdraws the IP route from the routing system and deletes the neighbor cache entry and IP forwarding table entry for the Client. The old Server then returns an RA message with default router lifetime set to 0 which the Client can use to verify that the termination signal has been processed. The client then deletes both the default route and the neighbor cache entry for the old Server. (Note that the Client and the old Server MAY impose a small delay before deleting the neighbor cache and IP forwarding table entries so that any packets already in the system can still be delivered to the Client.)

3.12. Encapsulation 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 target and source Clients have no means for reaching each other directly (since they connect to underlying networks of different IP protocol versions) and so must ignore any redirection messages 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 link-layer address of a 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 encapsulated packet is "M", then the IPv6 multicast destination address of the encapsulating header is also "M".)

3.14. Operation on AERO Links Without DHCPv6 Services

When the AERO link does not provide DHCPv6 services, operation can still be accommodated through administrative configuration of prefixes on AERO Clients. In that case, administrative configurations of IP routes and AERO interface neighbor cache entries on both the Server and Client are also necessary. However, this may preclude the ability for Clients to dynamically change to new Servers, and can expose the AERO link to misconfigurations unless the administrative configurations are carefully coordinated.

3.15. Operation on Server-less AERO Links

In some AERO link scenarios, there may be no Servers on the link and/or no need for Clients to use a Server as an intermediary trust anchor. In that case, each Client acts as a Server unto itself to establish neighbor cache entries and IP forwarding table entries by performing direct Client-to-Client Predirect/Redirect 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 alternate prefix delegation authority through some means outside the scope of this document.

4. Implementation Status

An application-layer implementation is in progress.

5. IANA Considerations

The IANA is instructed to assign a new 2-octet Hardware Type number for AERO in the "arp-parameters" registry per Section 2 of [RFC5494]. The number is assigned from the 2-octet Unassigned range with Hardware Type "AERO" and with this document as the reference.

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 uses a trust basis between Clients and Servers, where the Clients only engage in the AERO mechanism when it is facilitated by a trust anchor. AERO also uses DHCPv6 authentication for Client authentication and network admission control.

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., IEEE 802.1X WLANs) and links that provide physical security (e.g., enterprise network wired LANs) provide a first line of defense that is often sufficient. In other instances, additional securing mechanisms such as Secure Neighbor Discovery (SeND) [RFC3971], IPsec [RFC4301] or TLS [RFC5246] may be necessary.

AERO Clients MUST ensure that their connectivity is not used by unauthorized nodes on EUNs to gain access to a protected network, i.e., AERO Clients that act as routers MUST NOT provide routing services for unauthorized nodes. (This concern is no different than for ordinary hosts that receive an IP address delegation but then "share" the address with unauthorized nodes via a NAT function.)

On some AERO links, establishment and maintenance of a direct path between neighbors requires secured coordination such as through the Internet Key Exchange (IKEv2) protocol [RFC5996] to establish a security association.

7. Acknowledgements

Discussions both on IETF lists 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, Wojciech Dec, Brian Haberman, Joel Halpern, Sascha Hlusiak, Lee Howard, Joe Touch and Bernie Volz. 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, Dave Bernhardt, Cam Brodie, Balaguruna Chidambaram, Wen Fang, Anthony Gregory, Jeff Holland, Ed King, Gen MacLean, Kent Shuey, Mike Slane, Julie Wulff, Yueli Yang, 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.

[RFC2003] Perkins, C., "IP Encapsulation within IP", RFC 2003, October 1996.

[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.

[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.

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

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

[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.

[RFC1930] Hawkinson, J. and T. Bates, "Guidelines for creation, selection, and registration of an Autonomous System (AS)", BCP 6, RFC 1930, March 1996.

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

[RFC2675] Borman, D., Deering, S., and R. Hinden, "IPv6 Jumbograms", RFC 2675, August 1999.

[RFC4271] Rekhter, Y., Li, T., and S. Hares, "A Border Gateway Protocol 4 (BGP-4)", RFC 4271, January 2006.

[RFC4291] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, February 2006.

[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.

[RFC4994] Zeng, S., Volz, B., Kinnear, K., and J. Brzozowski, "DHCPv6 Relay Agent Echo Request Option", RFC 4994, September 2007.

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

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008.

[RFC5494] Arkko, J. and C. Pignataro, "IANA Allocation Guidelines for the Address Resolution Protocol (ARP)", RFC 5494, April 2009.

[RFC5522] Eddy, W., Ivancic, W., and T. Davis, "Network Mobility Route Optimization Requirements for Operational Use in Aeronautics and Space Exploration Mobile Networks", RFC 5522, October 2009.

[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.

[RFC6355] Narten, T. and J. Johnson, "Definition of the UUID-Based DHCPv6 Unique Identifier (DUID-UUID)", RFC 6355, August 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.

[RFC6939] Halwasia, G., Bhandari, S., and W. Dec, "Client Link-Layer Address Option in DHCPv6", RFC 6939, May 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 2 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 4.

```
                             .-(::::::::)
                          .-(:::: IP ::::)-.
                         (:: Internetwork ::)
                          `-(::::::::::::)-'
                             `-(::::::)-'
                                  |
       +--------------+    +------+-------+    +--------------+
       |AERO Server C |    | AERO Relay D |    |AERO Server E |
       | (default->D) |    | (A->C; G->E) |    | (default->D) |
       |    (A->B)    |    +-------+------+    |    (G->F)    |
       +-------+------+            |           +------+-------+
               |                   |                  |
       X---+---+-------------------+------------------+---+---X
           |                  AERO Link                   |
     +-----+--------+                            +--------+-----+
     |AERO Client B |                            |AERO Client F |
     | (default->C) |                            | (default->E) |
     +--------------+                            +--------------+
           .-.                                         .-.
        ,-(  _)-.                                   ,-(  _)-.
     .-(_   IP  )-.                              .-(_   IP  )-.
    (__    EUN      )                           (__    EUN      )
       `-(______)-'                                `-(______)-'
            |                                           |
        +--------+                                  +--------+
        | Host A |                                  | Host G |
        +--------+                                  +--------+
```

Figure 4: AERO Server/Relay Interworking

In this example, Client ('B') associates with Server ('C'), while Client ('F') associates with Server ('E'). Furthermore, Servers ('C') and ('E') do not associate with each other directly, but rather have an association with 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 IP Internetwork.

When source host ('A') sends a packet toward destination host ('G'), IP forwarding directs the packet through the EUN to Client ('B'), which forwards the packet to Server ('C'). Server ('C') forwards both the packet and a Predirect message through Relay ('D'). Relay ('D') then forwards both the original packet and Predirect to Server ('E'). When Server ('E') receives the packet and Predirect message, it forwards them to Client ('F').

After processing the Predirect message, Client ('F') sends a Redirect message to Server ('E'). Server ('E'), in turn, forwards the message through Relay ('D') to Server ('C'). When Server ('C') receives the Redirect 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 particular, Relays require full topology information, while individual Servers only require partial topology information, i.e., they only need to know the set of aggregated prefixes associated with the AERO link and the EUN prefixes associated with their current set of associated Clients. This can be accomplished in a number of ways, but a prominent example is through the use of an internal instance of the Border Gateway Protocol (BGP) [RFC4271] coordinated between Servers and Relays. This internal BGP instance does not interact with the public Internet BGP instance; therefore, the AERO link is presented to the IP Internetwork as a small set of aggregated prefixes as opposed to the full set of individual Client prefixes.

In a reference BGP arrangement, each AERO Server is configured as an Autonomous System Border Router (ASBR) for a stub Autonomous System (AS) (possibly using a private AS Number (ASN) [RFC1930]), and each Server further peers with each Relay but does not peer with other Servers. Each Server maintains a working set of associated Clients, and dynamically announces new Client prefixes and withdraws departed Client prefixes in its BGP updates. The Relays therefore discover the full topology of the AERO link in terms of the working set of Clients associated with each Server. Since Clients are expected to remain associated with their current set of Servers for extended timeframes, the amount of BGP control messaging between Servers and

Relays should be minimal. However, Servers SHOULD dampen any route oscillations caused by impatient Clients that repeatedly associate and disassociate with the Server.

See [IRON] for further architectural discussion.

Author's Address

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

Email: fltemplin@acm.org