Network Working Group
Internet-Draft
Intended status: Informational
Expires: December 3, 2021

F. Templin, Ed.
Boeing Research & Technology

June 1, 2021

Asymmetric Extended Route Optimization (AERO)
draft-templin-6man-aero-09

Abstract

This document specifies an Asymmetric Extended Route Optimization (AERO) service for IP internetworking over Overlay Multilink Network (OMNI) interfaces. AERO/OMNI use an IPv6 link-local address format that supports operation of the IPv6 Neighbor Discovery (ND) protocol and links ND to IP forwarding. Prefix delegation/registration services are employed for network admission and to manage the routing system. Secure multilink operation, mobility management, multicast, traffic selector signaling and route optimization are naturally supported through dynamic neighbor cache updates. AERO is a widely-applicable mobile internetworking service especially well-suited to aviation services, intelligent transportation systems, mobile Virtual Private Networks (VPNs) and many other applications.

Table of Contents

1. Introduction

Asymmetric Extended Route Optimization (AERO) fulfills the requirements of Distributed Mobility Management (DMM) [RFC7333] and route optimization [RFC5522] for aeronautical networking and other

network mobility use cases including intelligent transportation systems and enterprise mobile device users. AERO is a secure internetworking and mobility management service that employs the Overlay Multilink Network Interface (OMNI) [I-D.templin-6man-omni] Non-Broadcast, Multiple Access (NBMA) virtual link model. The OMNI link is a virtual overlay configured over one or more underlying Internetworks, and nodes on the link can exchange original IP packets as single-hop neighbors. The OMNI Adaptation Layer (OAL) supports end system multilink operation for increased reliability, bandwidth optimization and traffic path selection while performing fragmentation and reassembly to support Internetwork segment routing and Maximum Transmission Unit (MTU) diversity.

The AERO service comprises Clients, Proxy/Servers and Relays that are seen as OMNI link neighbors as well as Bridges that interconnect diverse Internetworks as OMNI link segments through OAL forwarding at a layer below IP. Each node's OMNI interface uses an IPv6 link-local address format that supports operation of the IPv6 Neighbor Discovery (ND) protocol [RFC4861] and links ND to IP forwarding. A node's OMNI interface can be configured over multiple underlying interfaces, and therefore appears as a single interface with multiple link-layer addresses. Each link-layer address is subject to change due to mobility and/or multilink fluctuations, and link-layer address changes are signaled by ND messaging the same as for any IPv6 link.

AERO provides a secure cloud-based service where mobile node Clients may use any Proxy/Server acting as a Mobility Anchor Point (MAP) and fixed nodes may use any Relay on the link for efficient communications. Fixed nodes forward original IP packets destined to other AERO nodes via the nearest Relay, which forwards them through the cloud. A mobile node's initial packets are forwarded through the Proxy/Server, and direct routing is supported through route optimization while packets are flowing. Both unicast and multicast communications are supported, and mobile nodes may efficiently move between locations while maintaining continuous communications with correspondents and without changing their IP Address.

AERO Bridges are interconnected in a secured private BGP overlay routing instance to provide an OAL routing/bridging service that joins the underlying Internetworks of multiple disjoint administrative domains into a single unified OMNI link at a layer below IP. Each OMNI link instance is characterized by the set of Mobility Service Prefixes (MSPs) common to all mobile nodes. Relays provide an optimal route from correspondent nodes on the underlying Internetwork to nodes on the OMNI link. To the underlying Internetwork, the Relay is the source of a route to the MSP, and hence uplink traffic to the mobile node is naturally routed to the nearest Relay.

AERO can be used with OMNI links that span private-use Internetworks and/or public Internetworks such as the global Internet. In the latter case, some end systems may be located behind global Internet Network Address Translators (NATs). A means for robust traversal of NATs while avoiding "triangle routing" is therefore provided.

AERO assumes the use of PIM Sparse Mode in support of multicast communication. In support of Source Specific Multicast (SSM) when a Mobile Node is the source, AERO route optimization ensures that a shortest-path multicast tree is established with provisions for mobility and multilink operation. In all other multicast scenarios there are no AERO dependencies.

AERO was designed as a secure aeronautical internetworking service for both manned and unmanned aircraft, where the aircraft is treated as a mobile node that can connect an Internet of Things (IoT). AERO is also applicable to a wide variety of other use cases. For example, it can be used to coordinate the links of mobile nodes (e.g., cellphones, tablets, laptop computers, etc.) that connect into a home enterprise network via public access networks using tunneling software such as OpenVPN [OVPN] with VPN or non-VPN services enabled according to the appropriate security model. AERO can also be used to facilitate terrestrial vehicular and urban air mobility (as well as pedestrian communication services) for future intelligent transportation systems [I-D.ietf-ipwave-vehicular-networking][I-D.templin-ipwave-uam-its]. Other applicable use cases are also in scope.

Along with OMNI, AERO provides secured optimal routing support for the "6M's" of modern Internetworking, including:

1. Multilink - a mobile node's ability to coordinate multiple diverse underlying data links as a single logical unit (i.e., the OMNI interface) to achieve the required communications performance and reliability objectives.

2. Multinet - the ability to span the OMNI link across multiple diverse network administrative segments while maintaining seamless end-to-end communications between mobile nodes and correspondents such as air traffic controllers, fleet administrators, etc.

3. Mobility - a mobile node's ability to change network points of attachment (e.g., moving between wireless base stations) which may result in an underlying interface address change, but without disruptions to ongoing communication sessions with peers over the OMNI link.

4. Multicast - the ability to send a single network transmission that reaches multiple nodes belonging to the same interest group, but without disturbing other nodes not subscribed to the interest group.

5. Multihop - a mobile node vehicle-to-vehicle relaying capability useful when multiple forwarding hops between vehicles may be necessary to "reach back" to an infrastructure access point connection to the OMNI link.

6. MTU assurance - the ability to deliver packets of various robust sizes between peers without loss due to a link size restriction, and to dynamically adjust packets sizes to achieve the optimal performance for each independent traffic flow.

The following numbered sections present the AERO specification. The appendices at the end of the document are non-normative.

2. Terminology

The terminology in the normative references applies; especially, the terminology in the OMNI specification [I-D.templin-6man-omni] is used extensively throughout. The following terms are defined within the scope of this document:

IPv6 Neighbor Discovery (ND)
: a control message service for coordinating neighbor relationships between nodes connected to a common link. AERO uses the IPv6 ND messaging service specified in [RFC4861].

IPv6 Prefix Delegation
: a networking service for delegating IPv6 prefixes to nodes on the link. The nominal service is DHCPv6 [RFC8415], however alternate services (e.g., based on ND messaging) are also in scope. Most notably, a minimal form of prefix delegation known as "prefix registration" can be used if the Client knows its prefix in advance and can represent it in the IPv6 source address of an ND message.

Access Network (ANET)
: a node's first-hop data link service network (e.g., a radio access network, cellular service provider network, corporate enterprise network, etc.) that often provides link-layer security services such as IEEE 802.1X and physical-layer security (e.g., "protected spectrum") to prevent unauthorized access internally and with border network-layer security services such as firewalls and proxys that prevent unauthorized outside access.

ANET interface
a node's attachment to a link in an ANET.

Internetwork (INET)
a connected IP network topology with a coherent routing and addressing plan and that provides a transit backbone service for ANET end systems. INETs also provide an underlay service over which the AERO virtual link is configured. Example INETs include corporate enterprise networks, aviation networks, and the public Internet itself. When there is no administrative boundary between an ANET and the INET, the ANET and INET are one and the same.

INET interface
a node's attachment to a link in an INET.

*NET
a "wildcard" term referring to either ANET or INET when it is not necessary to draw a distinction between the two.

*NET interface
a node's attachment to a link in a *NET.

*NET Partition
frequently, *NETs such as large corporate enterprise networks are sub-divided internally into separate isolated partitions (a technique also known as "network segmentation"). Each partition is fully connected internally but disconnected from other partitions, and there is no requirement that separate partitions maintain consistent Internet Protocol and/or addressing plans. (Each *NET partition is seen as a separate OMNI link segment as discussed below.)

*NET address
an IP address assigned to a node's interface connection to a *NET.

*NET encapsulation
the encapsulation of a packet in an outer header or headers that can be routed within the scope of the local *NET partition.

OMNI link
the same as defined in [I-D.templin-6man-omni], and manifested by IPv6 encapsulation [RFC2473]. The OMNI link spans underlying *NET segments joined by virtual bridges in a spanning tree the same as a bridged campus LAN. AERO nodes on the OMNI link appear as single-hop neighbors even though they may be separated by multiple underlying *NET hops, and can use Segment Routing [RFC8402] to cause packets to visit selected waypoints on the link.

OMNI Interface
: a node's attachment to an OMNI link. Since the addresses assigned to an OMNI interface are managed for uniqueness, OMNI interfaces do not require Duplicate Address Detection (DAD) and therefore set the administrative variable 'DupAddrDetectTransmits' to zero [RFC4862].

OMNI Adaptation Layer (OAL)
: an OMNI interface process whereby original IP packets admitted into the interface are wrapped in a mid-layer IPv6 header and subject to fragmentation and reassembly. The OAL is also responsible for generating MTU-related control messages as necessary, and for providing addressing context for spanning multiple segments of a bridged OMNI link.

original IP packet
: a whole IP packet or fragment admitted into the OMNI interface by the network layer prior to OAL encapsulation and fragmentation, or an IP packet delivered to the network layer by the OMNI interface following OAL decapsulation and reassembly.

OAL packet
: an original IP packet encapsulated in OAL headers and trailers before OAL fragmentation, or following OAL reassembly.

OAL fragment
: a portion of an OAL packet following fragmentation but prior to *NET encapsulation, or following *NET encapsulation but prior to OAL reassembly.

(OAL) atomic fragment
: an OAL packet that does not require fragmentation is always encapsulated as an "atomic fragment" with a Fragment Header with Fragment Offset and More Fragments both set to 0, but with a valid Identification value.

(OAL) carrier packet
: an encapsulated OAL fragment following *NET encapsulation or prior to *NET decapsulation. OAL sources and destinations exchange carrier packets over underlying interfaces, and may be separated by one or more OAL intermediate nodes. OAL intermediate nodes may perform re-encapsulation on carrier packets by removing the *NET headers of the first hop network and replacing them with new *NET headers for the next hop network.

OAL source

an OMNI interface acts as an OAL source when it encapsulates original IP packets to form OAL packets, then performs OAL fragmentation and *NET encapsulation to create carrier packets.

OAL destination
an OMNI interface acts as an OAL destination when it decapsulates carrier packets, then performs OAL reassembly and decapsulation to derive the original IP packet.

OAL intermediate node
an OMNI interface acts as an OAL intermediate node when it removes the *NET headers of carrier packets received on a first segment, then re-encapsulates the carrier packets in new *NET headers and forwards them into the next segment. OAL intermediate nodes decrement the Hop Limit of the OAL IPv6 header during re-encapsulation, and discard the packet if the Hop Limit reaches 0.

underlying interface
a *NET interface over which an OMNI interface is configured.

Mobility Service Prefix (MSP)
an aggregated IP Global Unicast Address (GUA) prefix (e.g., 2001:db8::/32, 192.0.2.0/24, etc.) assigned to the OMNI link and from which more-specific Mobile Network Prefixes (MNPs) are delegated. OMNI link administrators typically obtain MSPs from an Internet address registry, however private-use prefixes can alternatively be used subject to certain limitations (see: [I-D.templin-6man-omni]). OMNI links that connect to the global Internet advertise their MSPs to their interdomain routing peers.

Mobile Network Prefix (MNP)
a longer IP prefix delegated from an MSP (e.g., 2001:db8:1000:2000::/56, 192.0.2.8/30, etc.) and delegated to an AERO Client or Relay.

Mobile Network Prefix Link Local Address (MNP-LLA)
an IPv6 Link Local Address that embeds the most significant 64 bits of an MNP in the lower 64 bits of fe80::/64, as specified in [I-D.templin-6man-omni].

Mobile Network Prefix Unique Local Address (MNP-ULA)
an IPv6 Unique-Local Address derived from an MNP-LLA.

Administrative Link Local Address (ADM-LLA)
an IPv6 Link Local Address that embeds a 32-bit administratively-assigned identification value in the lower 32 bits of fe80::/96, as specified in [I-D.templin-6man-omni].

Administrative Unique Local Address (ADM-ULA)
an IPv6 Unique-Local Address derived from an ADM-LLA.

AERO node
a node that is connected to an OMNI link and participates in the AERO internetworking and mobility service.

AERO Client ("Client")
an AERO node that connects over one or more underlying interfaces and requests MNP delegation/registration service from AERO Proxy/Servers. The Client assigns an MNP-LLA to the OMNI interface for use in ND exchanges with other AERO nodes and forwards original IP packets to correspondents according to OMNI interface neighbor cache state.

AERO Proxy/Server ("Proxy/Server")
a dual-function node that provides a proxying service between AERO Clients and external peers on its Client-facing ANET interfaces (i.e., in the same fashion as for an enterprise network proxy) as well as default forwarding and Mobility Anchor Point (MAP) services for coordination with correspondents on its INET-facing interfaces. (Proxy/Servers in the open INET instead configure only an INET interface and no ANET interfaces.) The Proxy/Server configures an OMNI interface and assigns an ADM-LLA to support the operation of IPv6 ND services, while advertising all of its associated MNPs via BGP peerings with Bridges. Note that the Proxy and Server functions can be considered logically separable, but since each Proxy/Server must be informed of all of the Client's other multilink Proxy/Server affiliations the AERO service is best supported when the two functions are coresident on the same physical or logical platform.

AERO Relay ("Relay")
a Proxy/Server that provides forwarding services between nodes reached via the OMNI link and correspondents on connected downstream links. AERO Relays configure an OMNI interface and assign an ADM-LLA the same as Proxy/Servers. AERO Relays also run a dynamic routing protocol to discover any non-MNP IP GUA routes in service on its connected downstream network links. In both cases, the Relay advertises the MSP(s) to its downstream networks, and distributes all of its associated non-MNP IP GUA routes via BGP peerings with Bridges (i.e., the same as for Proxy/Servers).

AERO Bridge ("Bridge")
a node that provides hybrid routing/bridging services (as well as a security trust anchor) for nodes on an OMNI link. The Bridge forwards carrier packets between OMNI link segments as OAL intermediate nodes while decrementing the OAL IPv6 header Hop

Limit but without decrementing the network layer IP TTL/Hop Limit. AERO Bridges peer with Proxy/Servers and other Bridges over secured tunnels to discover the full set of MNPs for the link as well as any non-MNP IP GUA routes that are reachable via Relays.

link-layer address
an IP address used as an encapsulation header source or destination address from the perspective of the OMNI interface. When an upper layer protocol (e.g., UDP) is used as part of the encapsulation, the port number is also considered as part of the link-layer address.

network layer address
the source or destination address of an original IP packet presented to the OMNI interface.

end user network (EUN)
an internal virtual or external edge IP network that an AERO Client or Relay connects to the rest of the network via the OMNI interface. The Client/Relay sees each EUN as a "downstream" network, and sees the OMNI interface as the point of attachment to the "upstream" network.

Mobile Node (MN)
an AERO Client and all of its downstream-attached networks that move together as a single unit, i.e., an end system that connects an Internet of Things.

Mobile Router (MR)
a MN's on-board router that forwards original IP packets between any downstream-attached networks and the OMNI link. The MR is the MN entity that hosts the AERO Client.

Route Optimization Source (ROS)
the AERO node nearest the source that initiates route optimization. The ROS may be a Proxy/Server or Relay acting on behalf of the source, or may be the source Client itself.

Route Optimization responder (ROR)
the AERO node nearest the target destination that responds to route optimization requests. The ROR may be a Proxy/Server acting on behalf of a target MNP Client or a Relay for a non-MNP destination.

MAP List
a geographically and/or topologically referenced list of addresses of all Proxy/Servers within the same OMNI link. Each OMNI link has its own MAP list.

Distributed Mobility Management (DMM)
: a BGP-based overlay routing service coordinated by Proxy/Servers and Bridges that tracks all Proxy/Server-to-Client associations.

Mobility Service (MS)
: the collective set of all Proxy/Servers, Bridges and Relays that provide the AERO Service to Clients.

Mobility Service Endpoint MSE)
: an individual Proxy/Server, Bridge or Relay in the Mobility Service.

Throughout the document, the simple terms "Client", "Proxy/Server", "Bridge" and "Relay" refer to "AERO Client", "AERO Proxy/Server", "AERO Bridge" and "AERO Relay", respectively. Capitalization is used to distinguish these terms from other common Internetworking uses in which they appear without capitalization.

The terminology of DHCPv6 [RFC8415] and IPv6 ND [RFC4861] (including the names of node variables, messages and protocol constants) is used throughout this document. The terms "All-Routers multicast", "All-Nodes multicast", "Solicited-Node multicast" and "Subnet-Router anycast" are defined in [RFC4291]. Also, the term "IP" is used to generically refer to either Internet Protocol version, i.e., IPv4 [RFC0791] or IPv6 [RFC8200].

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119][RFC8174] when, and only when, they appear in all capitals, as shown here.

3. Asymmetric Extended Route Optimization (AERO)

The following sections specify the operation of IP over OMNI links using the AERO service:

3.1. AERO Node Types

AERO Clients are Mobile Nodes (MNs) that configure OMNI interfaces over underlying interfaces with addresses that may change when the Client moves to a new network connection point. AERO Clients register their Mobile Network Prefixes (MNPs) with the AERO service, and distribute the MNPs to nodes on EUNs. AERO Bridges, Proxy/Servers and Relays are critical infrastructure elements in fixed (i.e., non-mobile) INET deployments and hence have permanent and unchanging INET addresses. Together, they constitute the AERO

service which provides an OMNI link virtual overlay for connecting AERO Clients.

AERO Bridges provide hybrid routing/bridging services (as well as a security trust anchor) for nodes on an OMNI link. Bridges use standard IPv6 routing to forward carrier packets both within the same *NET partition and between disjoint *NET partitions based on an IPv6 encapsulation mid-layer known as the OMNI Adaptation Layer (OAL) [I-D.templin-6man-omni]. During forwarding, the inner IP layer experiences a virtual bridging service since the inner IP TTL/Hop Limit is not decremented. Each Bridge also peers with Proxy/Servers and other Bridges in a dynamic routing protocol instance to provide a Distributed Mobility Management (DMM) service for the list of active MNPs (see Section 3.2.3). Bridges present the OMNI link as a set of one or more Mobility Service Prefixes (MSPs) and configure secured tunnels with Proxy/Servers, Relays and other Bridges; they further maintain IP forwarding table entries for each MNP and any other reachable non-MNP prefixes.

AERO Proxy/Servers in distributed *NET locations provide default forwarding and mobility/multilink services for AERO Client Mobile Nodes (MNs). Each Proxy/Server also peers with Bridges in a dynamic routing protocol instance to advertise its list of associated MNPs (see Section 3.2.3). Proxy/Servers facilitate prefix delegation/ registration exchanges with Clients, where each delegated prefix becomes an MNP taken from an MSP. Proxy/Servers forward carrier packets between OMNI interface neighbors and track each Client's mobility profiles. Proxy/Servers at ANET/INET boundaries provide a conduit for ANET Clients to associate with peers reached through external INETs. Proxy/Servers in the open INET support INET Clients through authenticated IPv6 ND message exchanges.

AERO Relays are Proxy/Servers that provide forwarding services to exchange original IP packets between the OMNI interface and INET/EUN interfaces. Relays are provisioned with MNPs the same as for an AERO Client, and also run a dynamic routing protocol to discover any non-MNP IP routes. The Relay advertises the MSP(s) to its connected networks, and distributes all of its associated MNP and non-MNP routes via BGP peerings with Bridges

3.2. The AERO Service over OMNI Links

3.2.1. AERO/OMNI Reference Model

Figure 1 presents the basic OMNI link reference model:

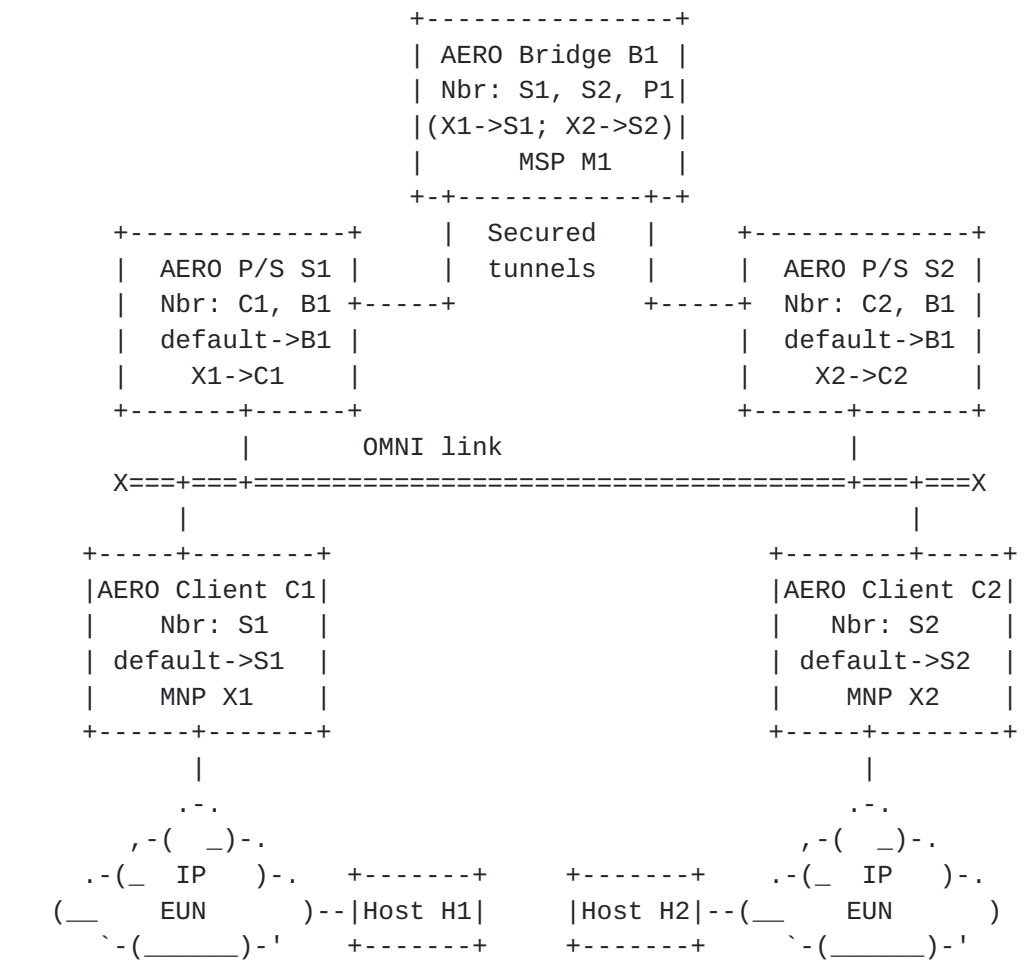

Figure 1: AERO/OMNI Reference Model

In this model:

- the OMNI link is an overlay network service configured over one or more underlying *NET partitions which may be managed by different administrative authorities and have incompatible protocols and/or addressing plans.

- AERO Bridge B1 aggregates Mobility Service Prefix (MSP) M1, discovers Mobile Network Prefixes (MNPs) X* and advertises the MSP via BGP peerings over secured tunnels to Proxy/Servers (S1, S2). Bridges connect the disjoint segments of a partitioned OMNI link.

- AERO Proxy/Servers S1 and S2 configure secured tunnels with Bridge B1 and also provide mobility, multilink, multicast and default router services for the MNPs of their associated Clients C1 and C2. (AERO Proxy/Servers that act as Relays can also advertise non-MNP routes for non-mobile correspondent nodes the same as for MNP Clients.)

- AERO Clients C1 and C2 associate with Proxy/Servers S1 and S2, respectively. They receive MNP delegations X1 and X2, and also act as default routers for their associated physical or internal virtual EUNs. Simple hosts H1 and H2 attach to the EUNs served by Clients C1 and C2, respectively.

An OMNI link configured over a single *NET appears as a single unified link with a consistent underlying network addressing plan. In that case, all nodes on the link can exchange carrier packets via simple *NET encapsulation, since the underlying *NET is connected. In common practice, however, an OMNI link may be partitioned into multiple "segments", where each segment is a distinct *NET potentially managed under a different administrative authority (e.g., as for worldwide aviation service providers such as ARINC, SITA, Inmarsat, etc.). Individual *NETs may also themselves be partitioned internally, in which case each internal partition is seen as a separate segment.

The addressing plan of each segment is consistent internally but will often bear no relation to the addressing plans of other segments. Each segment is also likely to be separated from others by network security devices (e.g., firewalls, proxys, packet filtering gateways, etc.), and in many cases disjoint segments may not even have any common physical link connections. Therefore, nodes can only be assured of exchanging carrier packets directly with correspondents in the same segment, and not with those in other segments. The only means for joining the segments therefore is through inter-domain peerings between AERO Bridges.

The same as for traditional campus LANs, multiple OMNI link segments can be joined into a single unified link via a virtual bridging service using the OMNI Adaptation Layer (OAL) [I-D.templin-6man-omni] which inserts a mid-layer IPv6 encapsulation header that supports inter-segment forwarding (i.e., bridging) without decrementing the network-layer TTL/Hop Limit. This bridging of OMNI link segments is shown in Figure 2:

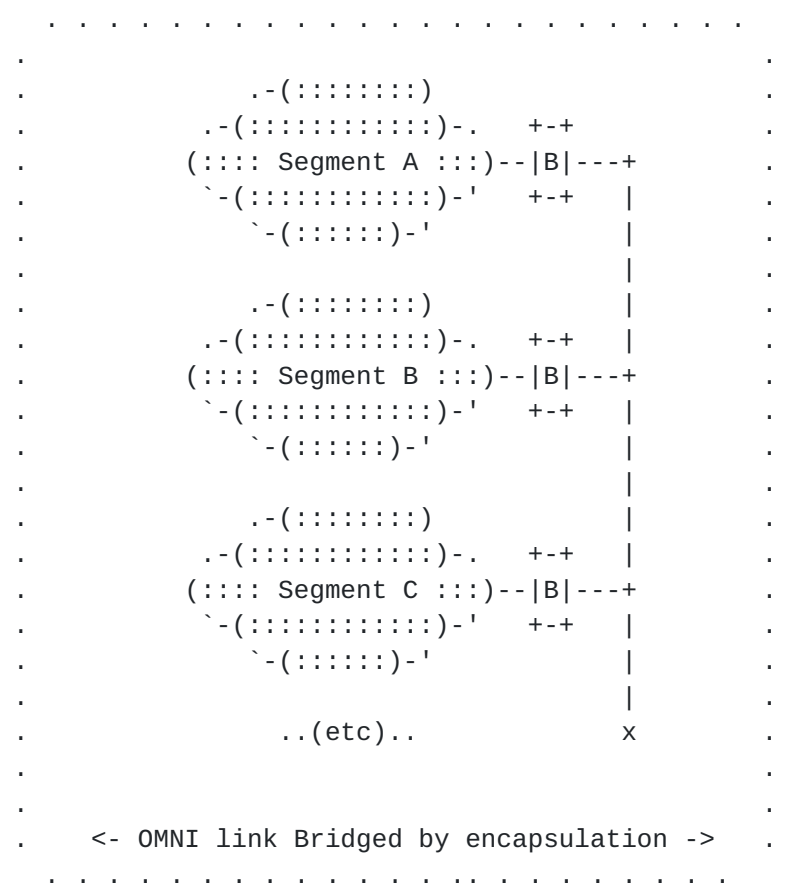

Figure 2: Bridging OMNI Link Segments

Bridges, Proxy/Servers and Relay OMNI interfaces are configured over both secured tunnels and open INET underlying interfaces over their respective segments in a spanning tree topology rooted at the Bridges. The "secured" spanning tree supports strong authentication for control plane messages and may also be used to convey the initial carrier packets in a flow. The "unsecured" spanning tree conveys ordinary carrier packets without security codes and that must be treated by destinations according to data origin authentication procedures. Route optimization can be employed to cause carrier packets to take more direct paths between OMNI link neighbors without having to follow strict spanning tree paths.

3.2.2. Addressing and Node Identification

AERO nodes on OMNI links use the Link-Local Address (LLA) prefix fe80::/64 [RFC4291] to assign LLAs used for network-layer addresses in link-scoped IPv6 ND and data messages. AERO Clients use LLAs constructed from MNPs (i.e., "MNP-LLAs") while other AERO nodes use LLAs constructed from administrative identification values ("ADM-LLAs") as specified in [I-D.templin-6man-omni]. Non-MNP routes are

also represented the same as for MNP-LLAs, but may include a prefix that is not properly covered by the MSP.

AERO nodes also use the Unique Local Address (ULA) prefix fd00::/8 followed by a pseudo-random 40-bit OMNI domain identifier to form the prefix [ULA]::/48, then include a 16-bit OMNI link identifier '*' to form the prefix [ULA*]::/64 [RFC4291]. The AERO node then uses the prefix [ULA*]::/64 to form "MNP-ULAs" or "ADM-ULA"s as specified in [I-D.templin-6man-omni] to support OAL addressing. (The prefix [ULA*]::/64 appearing alone and with no suffix represents "default".) AERO Clients also use Temporary ULAs constructed per [I-D.templin-6man-omni], where the addresses are typically used only in initial control message exchanges until a stable MNP-LLA/ULA is assigned.

AERO MSPs, MNPs and non-MNP routes are typically based on Global Unicast Addresses (GUAs), but in some cases may be based on private-use addresses. See [I-D.templin-6man-omni] for a full specification of LLAs, ULAs and GUAs used by AERO nodes on OMNI links.

Finally, AERO Clients and Proxy/Servers configure node identification values as specified in [I-D.templin-6man-omni].

3.2.3. AERO Routing System

The AERO routing system comprises a private instance of the Border Gateway Protocol (BGP) [RFC4271] that is coordinated between Bridges and Proxy/Servers and does not interact with either the public Internet BGP routing system or any underlying INET routing systems.

In a reference deployment, each Proxy/Server is configured as an Autonomous System Border Router (ASBR) for a stub Autonomous System (AS) using a 32-bit AS Number (ASN) [RFC4271] that is unique within the BGP instance, and each Proxy/Server further uses eBGP to peer with one or more Bridges but does not peer with other Proxy/Servers. Each *NET of a multi-segment OMNI link must include one or more Bridges, which peer with the Proxy/Servers within that *NET. All Bridges within the same *NET are members of the same hub AS, and use iBGP to maintain a consistent view of all active routes currently in service. The Bridges of different *NETs peer with one another using eBGP.

Bridges maintain forwarding table entries only for the MNP-ULAs corresponding to MNP and non-MNP routes that are currently active, and carrier packets destined to all other MNP-ULAs will correctly incur Destination Unreachable messages due to the black-hole route. In this way, Proxy/Servers and Relays have only partial topology knowledge (i.e., they know only about the routes their directly

associated Clients and non-AERO links) and they forward all other carrier packets to Bridges which have full topology knowledge.

Each OMNI link segment assigns a unique ADM-ULA sub-prefix of [ULA*]::/96. For example, a first segment could assign [ULA*]::1000/116, a second could assign [ULA*]::2000/116, a third could assign [ULA*]::3000/116, etc. Within each segment, each Proxy/Server configures an ADM-ULA within the segment's prefix, e.g., the Proxy/Servers within [ULA*]::2000/116 could assign the ADM-ULAs [ULA*]::2011/116, [ULA*]::2026/116, [ULA*]::2003/116, etc.

The administrative authorities for each segment must therefore coordinate to assure mutually-exclusive ADM-ULA prefix assignments, but internal provisioning of ADM-ULAs an independent local consideration for each administrative authority. For each ADM-ULA prefix, the Bridge(s) that connect that segment assign the all-zero's address of the prefix as a Subnet Router Anycast address. For example, the Subnet Router Anycast address for [ULA*]::1023/116 is simply [ULA*]::1000.

ADM-ULA prefixes are statically represented in Bridge forwarding tables. Bridges join multiple segments into a unified OMNI link over multiple diverse administrative domains. They support a bridging function by first establishing forwarding table entries for their ADM-ULA prefixes either via standard BGP routing or static routes. For example, if three Bridges ('A', 'B' and 'C') from different segments serviced [ULA*]::1000/116, [ULA*]::2000/116 and [ULA*]::3000/116 respectively, then the forwarding tables in each Bridge are as follows:

A: [ULA*]::1000/116->local, [ULA*]::2000/116->B, [ULA*]::3000/116->C

B: [ULA*]::1000/116->A, [ULA*]::2000/116->local, [ULA*]::3000/116->C

C: [ULA*]::1000/116->A, [ULA*]::2000/116->B, [ULA*]::3000/116->local

These forwarding table entries are permanent and never change, since they correspond to fixed infrastructure elements in their respective segments.

MNP ULAs are instead dynamically advertised in the AERO routing system by Proxy/Servers and Relays that provide service for their corresponding MNPs. For example, if three Proxy/Servers ('D', 'E' and 'F') service the MNPs 2001:db8:1000:2000::/56, 2001:db8:3000:4000::/56 and 2001:db8:5000:6000::/56 then the routing system would include:

D: [ULA*]:2001:db8:1000:2000/120

E: [ULA*]:2001:db8:3000:4000/120

F: [ULA*]:2001:db8:5000:6000/120

A full discussion of the BGP-based routing system used by AERO is found in [I-D.ietf-rtgwg-atn-bgp].

3.2.4. OMNI Link Segment Routing

With the Client and partition prefixes in place in Bridge forwarding tables, the OMNI interface sends control and data carrier packets toward AERO destination nodes located in different OMNI link segments over the spanning tree. The OMNI interface uses the OMNI Adaptation Layer (OAL) encapsulation service [I-D.templin-6man-omni], and includes an OMNI Routing Header (ORH) as an extension to the OAL header if final segment forwarding information is available, e.g., in the neighbor cache. (For nodes located in the same OMNI link segment the ORH may instead be omitted.) In its full form, the ORH is formatted as shown in Figure 3:

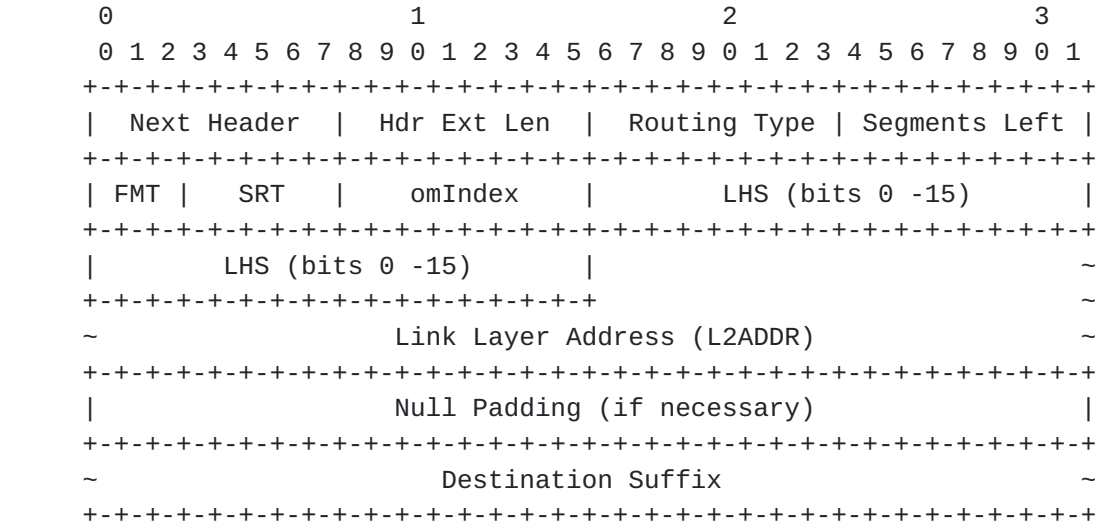

Figure 3: OMNI Routing Header (ORH) Format

In this format:

- Next Header identifies the type of header immediately following the ORH.

- Hdr Ext Len is the length of the Routing header in 8-octet units (not including the first 8 octets). The field must encode a value between 0 and 4 (all other values are treated as a parameter problem).

- Routing Type is set to TBD1 (see IANA Considerations).

- Segments Left encodes the value 0 or 1 (all other values are treated as a parameter problem).

- FMT - a 3-bit "Forward/Mode/Trailer" code corresponding to the included Link Layer Address as follows:

  * When the most significant bit (i.e., "FMT-Forward") is clear, the Last Hop Segment (LHS) Proxy/Server must reassemble. When FMT-Forward is set, the LHS Proxy/Server must forward the fragments to the Client (while changing the OAL destination address to the MNP-ULA of the Client if necessary) without reassembling.

  * When the next most significant bit (i.e., "FMT-Mode") is clear, L2ADDR is the INET address of the LHS Proxy/Server and the Client must be reached through the LHS Proxy/Server. When FMT-Mode is set, the Client is eligible for route optimization over the open INET where it may be located behind one or more NATs, and L2ADDR is either the INET address of the LHS Proxy/Server (when FMT-Forward is set) or the native INET address of the Client itself (when FMT-Forward is clear).

  * When the least significant bit (i.e., "FMT-Trailer") is set and Hdr Ext Len is 1, a trailing 8-octet Destination Suffix is included; otherwise, an LHS with L2ADDR for IPv4 is included. FMT-Trailer is otherwise ignored (see below).

- SRT - a 5-bit Segment Routing Topology prefix length consulted only when Segments Left is 1, and encodes a value that (when added to 96) determines the prefix length to apply to the ADM-ULA formed from concatenating [ULA*]::/96 with the 32 bit LHS value (for example, the value 16 corresponds to the prefix length 112).

- omIndex - a 1-octet field consulted only when Segments Left is 0; encodes the index for the target Client underlying interface and informs the LHS Proxy-Server of the specific interface for forwarding when there are multiple alternatives. When FMT-Forward is clear, omIndex determines the interface for forwarding the ORH packet following reassembly; when FMT-Forward is set, omIndex determines the interface for forwarding the raw carrier packets without first reassembling.

- LHS - a 4-octet field present only when indicated by the ORH length (see below) and consulted only when Segments Left is 1. The field encodes the 32-bit ADM-ULA suffix of a Last Hop Segment (LHS) Proxy/Server for the target. When SRT and LHS are both set

to 0, the LHS must be reached directly via INET encapsulation instead of over the spanning tree. When SRT is set to 0 and LHS is non-zero, the prefix length is set to 128. SRT and LHS determine the ADM-ULA of the LHS Proxy/Server over the spanning tree.

- Link Layer Address (L2ADDR) - an IP encapsulation address present only when indicated by the ORH length (see below) and consulted only when Segments Left is 1. The L2ADDR IP version is determined by consulting the ORH length (see below), since the field will always contain exactly 6 octets for UDP/IPv4 or 18 octets for UDP/IPv6. When present, provides the link-layer address (i.e., the encapsulation address) of the Proxy/Server or the target Client itself. The UDP Port Number appears in the first two octets and the IP address appears in the remaining octets. The Port Number and IP address are recorded in network byte order, and in ones-compliment "obfuscated" form per [RFC4380]. The OMNI interface forwarding algorithm uses L2ADDR as the INET encapsulation address for forwarding when SRT/LHS is located in the same OMNI link segment. If direct INET encapsulation is not permitted, L2ADDR is instead set to all-zeros and the packet must be forwarded to the LHS Proxy-Server via the spanning tree.

- Null Padding - zero-valued octets added as necessary to pad the portion of the ORH included up to this point to an even 8-octet boundary.

- Destination Suffix - a trailing 8-octet field present only when indicated by the ORH length (see below). When ORH length is 1, FMT-Trailer determines whether the option includes a Destination Suffix or an LHS/L2ADDR since there is only enough space available for one. When present, encodes the 64-bit MNP-ULA suffix for the target Client.

The ORH Hdr Ext Len field value also serves as an implicit ORH "Type" such that 5 distinct Types (i.e., ORH-0 through ORH-4) are supported. All Types include the same 6-octet preamble beginning with the Next Header field up to and including the omIndex field. The Types are defined as follows:

- ORH-0 - The preamble Hdr Ext Len and Segments Left fields must both be 0. Two null padding octets follow the preamble, and all other fields are omitted.

- ORH-1 - The preamble Hdr Ext Len is set to 1. When FMT-Trailer is clear, the LHS and L2ADDR for IPv4 fields are included and the Destination Suffix is omitted. When FMT-Trailer is set, the LHS

and L2ADDR fields are omitted, the Destination Suffix field is included and Segments Left must be 0.

- ORH-2 - The preamble Hdr Ext Len is set to 2. The LHS, L2ADDR for IPv4 and Destination Suffix fields are all included.

- ORH-3 - The preamble Hdr Ext Len is set to 3. The LHS and L2ADDR for IPv6 fields are included and the Destination Suffix field is omitted.

- ORH-4 - The preamble Hdr Ext Len is set to 4. The LHS, L2ADDR for IPv6 and Destination Suffix fields are all included.

AERO neighbors use OAL encapsulation and fragmentation to exchange OAL packets as specified in [I-D.templin-6man-omni]. When an AERO node's OMNI interface (acting as an OAL source) uses OAL encapsulation for an original IP packet with source address 2001:db8:1:2::1 and destination address 2001:db8:1234:5678::1, it sets the OAL header source address to its own ULA (e.g., [ULA*]::2001:db8:1:2), sets the destination address to the MNP-ULA corresponding to the IP destination address (e.g., [ULA*]::2001:db8:1234:5678), sets the Traffic Class, Flow Label, Hop Limit and Payload Length as discussed in [I-D.templin-6man-omni], then finally selects an Identification and appends an OAL checksum.

If the neighbor cache information indicates that the target is in a different segment, the OAL source next inserts an ORH immediately following the OAL header while including Destination Suffix for non-first-fragments only when necessary (in this case, the Destination Suffix is 2001:db8:1234:5678). Next, to direct the packet to a first-hop Proxy/Server or a Bridge, the source prepares an ORH with Segments Left set to 1 and with LHS/L2ADDR included, then overwrites the OAL header destination address with the LHS Subnet Router Anycast address (for example, for LHS 3000:4567 with SRT 16, the Subnet Router Anycast address is [ULA*]::3000:0000). To send the packet to the LHS Proxy/Server either directly or via the spanning tree, the OAL source instead includes an ORH (Type 0 or 1) with Segments Left set to 0 and LHS/L2ADDR omitted, and overwrites the OAL header destination address with either the LHS Proxy/Server ADM-ULA or the MNP-ULA of the Client itself.)

The OAL source then fragments the OAL packet, with each resulting OAL fragment including the OAL/ORH headers while only the first fragment includes the original IPv6 header. If FMT-Forward is set, the Identification used for fragmentation must be within the window for the Client and a Destination Suffix must be included with each non-first-fragment when necessary; otherwise the Identification must be within the window for the Client's Proxy/Server and no Destination

Suffix is needed. (Note that if no actual fragmentation is required the OAL source still prepares the packet as an "atomic" fragment that includes a Fragment Header with Offset and More Fragments both set to 0.) The OAL source finally encapsulates each resulting OAL fragment in an *NET header to form an OAL carrier packet, with source address set to its own *NET address (e.g., 192.0.2.100) and destination set to the *NET address of the last hop itself or the next hop in the spanning tree (e.g., 192.0.2.1).

The carrier packet encapsulation format in the above example is shown in Figure 4:

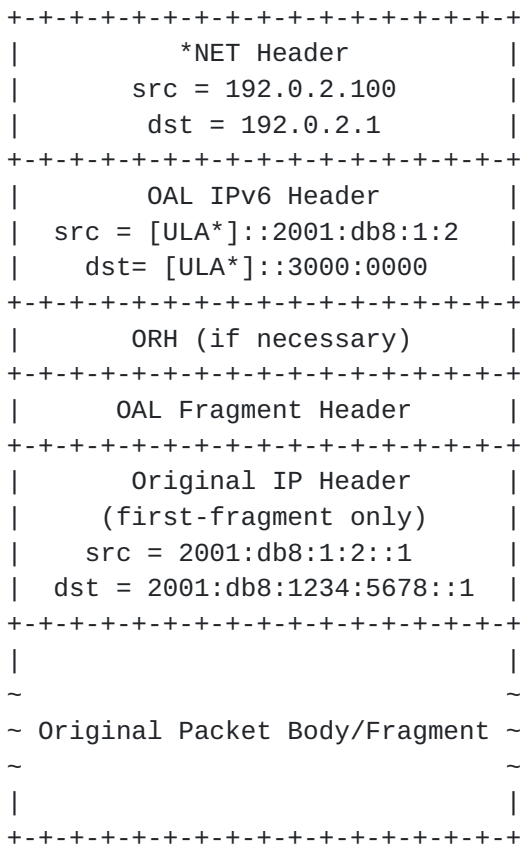

Figure 4: Carrier Packet Format

In this format, the original IP header and packet body/fragment are from the original IP packet, the OAL header is an IPv6 header prepared according to [RFC2473], the ORH is a Routing Header extension of the OAL header, the Fragment Header identifies each fragment, and the INET header is prepared as discussed in Section 3.6. When the OAL source transmits the resulting carrier packets, they are forwarded over possibly multiple OAL intermediate nodes in the OMNI link spanning tree until they arrive at the OAL destination.

This gives rise to a routing system that contains both Client MNP-ULA routes that may change dynamically due to regional node mobility and per-partition ADM-ULA routes that rarely if ever change. The spanning tree can therefore provide link-layer bridging by sending carrier packets over the spanning tree instead of network-layer routing according to MNP routes. As a result, opportunities for loss due to node mobility between different segments are mitigated.

Note: The document suggests in several places that AERO nodes transform ORHs with Segments Left set to 1 into ORH-0 during forwarding. While this may yield a substantial savings in encapsulation overhead in some cases, the AERO node may instead simply set Segments Left to 0 and leave the original ORH in place. The LHS Proxy/Server or target Client that processes the ORH will receive the same information in both cases.

Note: When the OAL source and destination are on the same INET segment, OAL header compression can be used to significantly reduce encapsulation overhead [I-D.templin-6man-omni].

Note: When the OAL source has multiple original IP packets to send to the same OAL destination, it can perform "packing" to generate a "super-packet" [I-D.templin-6man-omni]. In that case, the OAL/ORH super-packet may include up to N original IP packets as long as the total length of the super-packet does not exceed the OMNI interface MTU.

Note: Use of an IPv6 "minimal encapsulation" format (i.e., an IPv6 variant of [RFC2004]) based on extensions to the ORH was considered and abandoned. In the approach, the ORH would be inserted as an extension header to the original IPv6 packet header. The IPv6 destination address would then be written into the ORH, and the ULA corresponding to the destination would be overwritten in the IPv6 destination address. This would seemingly convey enough forwarding information so that OAL encapsulation could be avoided. However, this "minimal encapsulation" IPv6 packet would then have a non-ULA source address and ULA destination address, an incorrect value in upper layer protocol checksums, and a Hop Limit that is decremented within the spanning tree when it should not be. The insertion and removal of the ORH would also entail rewriting the Payload Length and Next Header fields - again, invalidating upper layer checksums. These irregularities would result in implementation challenges and the potential for operational issues, e.g., since actionable ICMPv6 error reports could not be delivered to the original source. In order to address the issues, still more information such as the original IPv6 source address could be written into the ORH. However, with the additional information the benefit of the "minimal

encapsulation" savings quickly diminishes, and becomes overshadowed by the implementation and operational irregularities.

3.2.5. Segment Routing Topologies (SRTs)

The 64-bit sub-prefixes of [ULA]::/48 identify up to 2^16 distinct Segment Routing Topologies (SRTs). Each SRT is a mutually-exclusive OMNI link overlay instance using a distinct set of ULAs, and emulates a Virtual LAN (VLAN) service for the OMNI link. In some cases (e.g., when redundant topologies are needed for fault tolerance and reliability) it may be beneficial to deploy multiple SRTs that act as independent overlay instances. A communication failure in one instance therefore will not affect communications in other instances.

Each SRT is identified by a distinct value in bits 48-63 of [ULA]::/48, i.e., as [ULA0]::/64, [ULA1]::/64, [ULA2]::/64, etc. Each OMNI interface is identified by a unique interface name (e.g., omni0, omni1, omni2, etc.) and assigns an anycast ADM-ULA corresponding to its SRT prefix length. The anycast ADM-ULA is used for OMNI interface determination in Safety-Based Multilink (SBM) as discussed in [I-D.templin-6man-omni]. Each OMNI interface further applies Performance-Based Multilink (PBM) internally.

3.2.6. Segment Routing For OMNI Link Selection

An original IPv6 source can direct an IPv6 packet to an AERO node by including a standard IPv6 Segment Routing Header (SRH) [RFC8754] with the anycast ADM-ULA for the selected SRT as either the IPv6 destination or as an intermediate hop within the SRH. This allows the original source to determine the specific OMNI link topology an original IPv6 packet will traverse when there may be multiple alternatives.

When the AERO node processes the SRH and forwards the original IPv6 packet to the correct OMNI interface, the OMNI interface writes the next IPv6 address from the SRH into the IPv6 destination address and decrements Segments Left. If decrementing would cause Segments Left to become 0, the OMNI interface deletes the SRH before forwarding. This form of Segment Routing supports Safety-Based Multilink (SBM).

3.2.7. Segment Routing Within the OMNI Link

OAL sources can insert an ORH for Segment Routing within the OMNI link to influence the paths of OAL packets sent to OAL destinations in remote segments without requiring all carrier packets to traverse strict spanning tree paths.

When an AERO node's OMNI interface has an original IP packet to send to a target discovered through route optimization located in the same OMNI link segment, it acts as an OAL source to perform OAL encapsulation and fragmentation. The node then uses L2ADDR for INET encapsulation while including an ORH-0 when sending the resulting carrier packets to the LHS Proxy/Server, or omitting the ORH-0 when sending directly to the target Client itself. When the node sends carrier packets with an ORH-0 to the LHS Proxy/Server, it sets the OAL destination to the ADM-ULA of the Proxy/Server if the Proxy/Server is responsible for reassembly; otherwise, it sets the OAL destination to the MNP-ULA of the target Client to cause the Proxy/Server to forward without reassembling. The node also sets omIndex to identify a specific target Client underlying interface.

When an AERO node's OMNI interface has an original IP packet to send to a route optimization target located in a remote OMNI link segment, it acts as an OAL source the same as above but also includes an appropriate ORH type with Segments Left set to 1 and with SRT/LHS/L2ADDR information while setting the OAL destination to the Subnet Router Anycast address for the LHS OMNI link segment. The OAL source can alternatively include an ORH with Segments Left set to 0 while setting the OAL destination to the ADM-ULA of the LHS Proxy/Server, but this would cause the carrier packets to follow strict spanning tree paths all the way to the LHS Proxy/Server. The OMNI interface then forwards the resulting carrier packets into the spanning tree.

When a Bridge receives a carrier packet destined to its Subnet Router Anycast address with any ORH type with Segments Left set to 1 and SRT/LHS/L2ADDR values corresponding to the local segment, it examines FMT-Mode to determine whether the target Client can accept packets directly (i.e., following any NAT traversal procedures necessary) while bypassing the LHS Proxy/Server. If the Client can be reached directly and NAT traversal has converged, the Bridge then writes the MNP-ULA (found in the inner IPv6 header for first fragments or the ORH Destination Suffix for non-first fragments) into the OAL destination address, decrements the OAL IPv6 header Hop Limit (and discards the packet if Hop Limit reaches 0), removes the ORH, re-encapsulates the carrier packet according to L2ADDR then forwards the carrier packet directly to the target Client. If the Client cannot be reached directly (or if NAT traversal has not yet converged), the Bridge instead transforms the ORH into an ORH-0, re-encapsulates the packet according to L2ADDR, changes the OAL destination to the ADM-ULA of the LHS Proxy/Server if FMT-Forward is clear or the MNP-ULA of the Client if FMT-Forward is set and forwards the carrier packet to the LHS Proxy/Server.

When a Bridge receives a carrier packet destined to its Subnet Router Anycast address with any ORH type with Segments Left set to 1 and

L2ADDR set to 0, the Bridge instead forwards the packet toward the LHS Proxy/Server via the spanning tree. The Bridge changes the OAL destination to the ADM-ULA of the LHS Proxy/Server, transforms the ORH into an ORH-0 (or an ORH-1 with FMT-Trailer set and Segments Left 0), then forwards the packet to the next hop in the spanning tree. The Bridge may also elect to forward via the spanning tree as above even when it receives a carrier packet with any ORH even if it includes a valid L2ADDR Port Number and IP address, however this may result in a longer path than necessary. If the carrier packet arrived via the secured spanning tree, the Bridge forwards to the next hop also via the secured spanning tree. If the carrier packet arrived via the unsecured spanning tree, the Bridge forwards to the next hop also via the unsecured spanning tree.

When an LHS Proxy/Server receives carrier packets with any ORH type with Segments Left set to 0 and with OAL destination set to its own ADM-ULA, it proceeds according to FMT-Forward and omIndex. If FMT-Forward is set, the LHS Proxy/Server changes the OAL destination to the MNP-ULA of the target Client found in the IPv6 header for first fragments or in the ORH Destination Suffix for non-first-fragments, removes the ORH and forwards to the target Client interface identified by omIndex. If FMT-Forward is clear, the LHS Proxy/Server instead reassembles then re-encapsulates while refragmenting if necessary, removes the ORH and forwards to the target Client according to omIndex.

When an LHS Proxy/Server receives carrier packets with any ORH type with Segments Left set to 0 and with OAL destination set to the MNP-ULA of the target Client, it removes the ORH and forwards to the target Client according to omIndex.

When a target Client receives carrier packets with OAL destination set to is MNP-ULA, it reassembles to obtain the OAL packet, then decapsulates and delivers the original IP packet to upper layers.

Note: Special handling procedures are employed for the exchange of IPv6 ND messages used to establish neighbor cache state as discussed in Section 3.14. The procedures call for hop-by-hop authentication and neighbor cache state establishment based on the encapsulation ULA, with next-hop determination based on the IPv6 ND message LLA.

3.3. OMNI Interface Characteristics

OMNI interfaces are virtual interfaces configured over one or more underlying interfaces classified as follows:

- INET interfaces connect to an INET either natively or through one or more NATs. Native INET interfaces have global IP addresses

that are reachable from any INET correspondent. The INET-facing interfaces of Proxy/Servers are native interfaces, as are Relay and Bridge interfaces. NATed INET interfaces connect to a private network behind one or more NATs that provide INET access. Clients that are behind a NAT are required to send periodic keepalive messages to keep NAT state alive when there are no carrier packets flowing.

- ANET interfaces connect to an ANET that is separated from the open INET by a Proxy/Server. Proxy/Servers can actively issue control messages over the INET on behalf of the Client to reduce ANET congestion.

- VPNed interfaces use security encapsulation over the INET to a Virtual Private Network (VPN) server that also acts as a Proxy/Server. Other than the link-layer encapsulation format, VPNed interfaces behave the same as Direct interfaces.

- Direct (i.e., single-hop point-to-point) interfaces connect a Client directly to a Proxy/Server without crossing any ANET/INET paths. An example is a line-of-sight link between a remote pilot and an unmanned aircraft. The same Client considerations apply as for VPNed interfaces.

OMNI interfaces use OAL encapsulation and fragmentation as discussed in Section 3.2.4. OMNI interfaces use *NET encapsulation (see: Section 3.6) to exchange carrier packets with OMNI link neighbors over INET or VPNed interfaces as well as over ANET interfaces for which the Client and Proxy/Server may be multiple IP hops away. OMNI interfaces do not use link-layer encapsulation over Direct underlying interfaces or ANET interfaces when the Client and Proxy/Server are known to be on the same underlying link.

OMNI interfaces maintain a neighbor cache for tracking per-neighbor state the same as for any interface. OMNI interfaces use ND messages including Router Solicitation (RS), Router Advertisement (RA), Neighbor Solicitation (NS) and Neighbor Advertisement (NA) for neighbor cache management. In environments where spoofing may be a threat, OMNI neighbors should employ OAL Identification window synchronization in their ND message exchanges.

OMNI interfaces send ND messages with an OMNI option formatted as specified in [I-D.templin-6man-omni]. The OMNI option includes prefix registration information and Interface Attributes containing link information parameters for the OMNI interface's underlying interfaces. Each OMNI option may include multiple Interface Attributes sub-options, each identified by an omIndex value.

A Client's OMNI interface may be configured over multiple underlying interface connections. For example, common mobile handheld devices have both wireless local area network ("WLAN") and cellular wireless links. These links are often used "one at a time" with low-cost WLAN preferred and highly-available cellular wireless as a standby, but a simultaneous-use capability could provide benefits. In a more complex example, aircraft frequently have many wireless data link types (e.g. satellite-based, cellular, 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 ND message OMNI options include Interface Attributes sub-options with the same underlying interface index. In that case, the Client would appear to have a single interface but with a dynamically changing link-layer address.

If the Client has multiple active underlying interfaces, then from the perspective of ND it would appear to have multiple link-layer addresses. In that case, ND message OMNI options MAY include Interface Attributes sub-options with different underlying interface indexes. Every ND message need not include Interface Attributes for all underlying interfaces; for any attributes not included, the neighbor considers the status as unchanged.

Bridge and Proxy/Server OMNI interfaces are configured over secured tunnel underlying interfaces for carrying IPv6 ND and BGP protocol control plane messages, plus open INET underlying interfaces for carrying unsecured messages. The OMNI interface configures both an ADM-LLA and its corresponding ADM-ULA, and acts as an OAL source to encapsulate and fragment original IP packets while presenting the resulting carrier packets to a secured or unsecured underlying interface. Note that Bridge and Proxy/Server BGP protocol TCP sessions are run directly over the OMNI interface using ADM-ULA source and destination addresses. The OMNI interface will encapsulate these as carrier packets even though the OAL header may use the same ADM-ULAs as the original IP header; these carrier packets must then be sent over a secured underlying interface.

3.4. OMNI Interface Initialization

AERO Proxy/Servers and Clients configure OMNI interfaces as their point of attachment to the OMNI link. AERO nodes assign the MSPs for the link to their OMNI interfaces (i.e., as a "route-to-interface") to ensure that original IP packets with destination addresses covered by an MNP not explicitly assigned to a non-OMNI interface are directed to the OMNI interface.

OMNI interface initialization procedures for Proxy/Servers, Clients and Bridges are discussed in the following sections.

3.4.1. AERO Proxy/Server and Relay Behavior

When a Proxy/Server enables an OMNI interface, it assigns an ADM-{LLA,ULA} appropriate for the given OMNI link segment. The Proxy/Server also configures secured tunnels with one or more neighboring Bridges and engages in a BGP routing protocol session with each Bridge.

The OMNI interface provides a single interface abstraction to the IP layer, but internally includes an NBMA nexus for sending carrier packets to OMNI interface neighbors over underlying INET interfaces and secured tunnels. The Proxy/Server further configures a service to facilitate ND exchanges with AERO Clients and manages per-Client neighbor cache entries and IP forwarding table entries based on control message exchanges.

Relays are simply Proxy/Servers that run a dynamic routing protocol to redistribute routes between the OMNI interface and INET/EUN interfaces (see: Section 3.2.3). The Relay provisions MNPs to networks on the INET/EUN interfaces (i.e., the same as a Client would do) and advertises the MSP(s) for the OMNI link over the INET/EUN interfaces. The Relay further provides an attachment point of the OMNI link to a non-MNP-based global topology.

3.4.2. AERO Client Behavior

When a Client enables an OMNI interface, it assigns either an MNP-{LLA, ULA} or a Temporary ULA and sends RS messages with ND parameters over its underlying interfaces to a Proxy/Server, which returns an RA message with corresponding parameters. The RS/RA messages may pass through one or more NATs in the case of a Client's INET interface. (Note: if the Client used a Temporary ULA in its initial RS message, it will discover an MNP-{LLA, ULA} in the corresponding RA that it receives from the Proxy/Server and begin using these new addresses. If the Client is operating outside the context of AERO infrastructure such as in a Mobile Ad-hoc Network (MANET), however, it may continue using Temporary ULAs for Client-to-Client communications until it encounters an infrastructure element that can provide an MNP.)

3.4.3. AERO Bridge Behavior

AERO Bridges configure an OMNI interface and assign the ADM-ULA Subnet Router Anycast address for each OMNI link segment they connect to. Bridges configure secured tunnels with Proxy/Servers and other

Bridges, and engage in a BGP routing protocol session with neighbors on the spanning tree (see: Section 3.2.3).

3.5. OMNI Interface Neighbor Cache Maintenance

Each OMNI interface maintains a conceptual neighbor cache that includes a Neighbor Cache Entry (NCE) for each neighbor it communicates with on the OMNI link per [RFC4861]. Each route optimization source NCE is indexed by the LLA of the neighbor, which must match the ULA used during OAL encapsulation. In addition to ordinary neighbor cache entries, proxy neighbor cache entries are created and maintained by AERO Proxy/Servers when they proxy Client ND message exchanges [RFC4389]. AERO Proxy/Servers maintain proxy neighbor cache entries for each of their associated Clients.

To the list of NCE states in Section 7.3.2 of [RFC4861], Proxy/Server OMNI interfaces add an additional state DEPARTED that applies to Clients that have recently departed. The interface sets a "DepartTime" variable for the NCE to "DEPART_TIME" seconds. DepartTime is decremented unless a new ND message causes the state to return to REACHABLE. While a NCE is in the DEPARTED state, the Proxy/Server forwards carrier packets destined to the target Client to the Client's new location instead. When DepartTime decrements to 0, the NCE is deleted. It is RECOMMENDED that DEPART_TIME be set to the default constant value REACHABLE_TIME plus 10 seconds (40 seconds by default) to allow a window for carrier packets in flight to be delivered while stale route optimization state may be present.

Proxy/Servers can act as RORs on behalf of dependent Clients according to the Proxy Neighbor Advertisement specification in Section 7.2.8 of [RFC4861], while well-connected Clients can act as an ROR on their own behalf. When a Proxy/Server ROR receives an authentic NS message used for route optimization, it first searches for a proxy NCE for the target Client and accepts the message only if there is an entry. The Proxy/Server then returns a solicited NA message while creating a NCE for the ROS. Proxy/Server RORs also create or update a "Report List" entry for the ROS in the target Client's NCE with a "ReportTime" variable set to REPORT_TIME seconds. The ROR resets ReportTime when it receives a new authentic NS message, and otherwise decrements ReportTime while no authentic NS messages have been received. It is RECOMMENDED that REPORT_TIME be set to the default constant value REACHABLE_TIME plus 10 seconds (40 seconds by default) to allow a window for route optimization to converge before ReportTime decrements below REACHABLE_TIME.

When the ROS receives a solicited NA message response to its NS message used for route optimization, it creates or updates a NCE for the target network-layer and link-layer addresses. The ROS then

(re)sets ReachableTime for the NCE to REACHABLE_TIME seconds and uses this value to determine whether carrier packets can be forwarded directly to the target, i.e., instead of via a default route. The ROS otherwise decrements ReachableTime while no further solicited NA messages arrive. It is RECOMMENDED that REACHABLE_TIME be set to the default constant value 30 seconds as specified in [RFC4861].

AERO nodes also use the value MAX_UNICAST_SOLICIT to limit the number of NS messages sent when a correspondent may have gone unreachable, the value MAX_RTR_SOLICITATIONS to limit the number of RS messages sent without receiving an RA and the value MAX_NEIGHBOR_ADVERTISEMENT to limit the number of unsolicited NAs that can be sent based on a single event. It is RECOMMENDED that MAX_UNICAST_SOLICIT, MAX_RTR_SOLICITATIONS and MAX_NEIGHBOR_ADVERTISEMENT be set to 3 the same as specified in [RFC4861].

Different values for DEPART_TIME, REPORT_TIME, REACHABLE_TIME, MAX_UNICAST_SOLICIT, MAX_RTR_SOLCITATIONS and MAX_NEIGHBOR_ADVERTISEMENT MAY be administratively set; however, if different values are chosen, all nodes on the link MUST consistently configure the same values. Most importantly, DEPART_TIME and REPORT_TIME SHOULD be set to a value that is sufficiently longer than REACHABLE_TIME to avoid packet loss due to stale route optimization state.

3.5.1. OMNI ND Messages

OMNI interface IPv6 ND messages include OMNI options [I-D.templin-6man-omni] with per-neighbor information including Interface Attributes that provide Link-Layer Address and traffic selector information for the neighbor's underlying interfaces. This information is stored in the neighbor cache and provides the basis for the forwarding algorithm specified in Section 3.10. The information is cumulative and reflects the union of the OMNI information from the most recent ND messages received from the neighbor; it is therefore not required that each ND message contain all neighbor information.

The OMNI option Interface Attributes for each underlying interface includes a two-part "Link-Layer Address" consisting of a simple IP encapsulation address determined by the FMT and L2ADDR fields and an ADM-ULA determined by the SRT and LHS fields. Underlying interfaces are further selected based on their associated traffic selectors.

The OMNI option is distinct from any Source/Target Link-Layer Address Options (S/TLLAOs) that may appear in an ND message according to the appropriate IPv6 over specific link layer specification (e.g., [RFC2464]). If both an OMNI option and S/TLLAO appear, the former

pertains to encapsulation addresses while the latter pertains to the native L2 address format of the underlying media

OMNI interface IPv6 ND messages may also include other IPv6 ND options. In particular, solicitation messages may include Nonce and/or Timestamp options if required for verification of advertisement replies. If an OMNI ND solicitation message includes a Nonce option, the advertisement reply must echo the same Nonce. If an OMNI ND solicitation message includes a Timestamp option, the advertisement reply should also include a Timestamp option.

AERO Clients send RS messages to the All-Routers multicast address while using unicast link-layer addresses. AERO Proxy/Servers respond by returning unicast RA messages. During the RS/RA exchange, AERO Clients and Servers include state synchronization parameters to establish Identification windows and other state.

AERO nodes use NS/NA messages for the following purposes:

- NS/NA(AR) messages are used for address resolution only. The ROS sends an NS(AR) to the solicited-node multicast address of the target, and an ROR in the network with addressing information for the target returns a unicast NA(AR). The NA(AR) contains authentic and current target address resolution information, but only an implicit third-party assertion of target reachability. NS/NA(AR) messages must be secured.

- NS/NA(WIN) messages are used for establishing and maintaining window synchronization (and any other) state. The source sends an NS(WIN) to the unicast address of the target, and the target returns a unicast NA(WIN). The NS/NA(WIN) exchange synchronizes the sequence numbers the neighbors will include in subsequent packets, and asserts reachability for the target without necessarily testing a specific underlying interface pair. NS/NA(WIN) messages must be secured.

- NS/NA(NUD) messages are used for determining target reachability. The source sends an NS(NUD) to the unicast address of the target over a specific underlying interface pair, and the target returns a unicast NA(NUD). NS/NA(NUD) messages that use an in-window sequence number and do not update any other state need not be secured. NS/NA(NUD) messages may also be used in combination with window synchronization (i.e., NUD+WIN), in which case the messages must be secured.

- Unsolicited NA (uNA) messages are used to signal addressing and/or other neighbor state changes (e.g., due to mobility, signal

degradation, traffic selector updates, etc.). uNA messages that include state update information must be secured.

- NS/NA(DAD) messages are not used in AERO, since Duplicate Address Detection is not required.

Additionally, nodes may send NA/RA messages with the OMNI option PNG flag set to receive a solicited NA response from the neighbor. The solicited NA response MUST set the ACK flag (without also setting the SYN or PNG flags) and include the Identification used in the PNG message in the Acknowledgement.

3.5.2. OMNI Neighbor Advertisement Message Flags

As discussed in Section 4.4 of [RFC4861] NA messages include three flag bits R, S and O. OMNI interface NA messages treat the flags as follows:

- R: The R ("Router") flag is set to 1 in the NA messages sent by all AERO/OMNI node types. Simple hosts that would set R to 0 do not occur on the OMNI link itself, but may occur on the downstream links of Clients and Relays.

- S: The S ("Solicited") flag is set exactly as specified in Section 4.4. of [RFC4861], i.e., it is set to 1 for Solicited NAs and set to 0 for uNAs (both unicast and multicast).

- O: The O ("Override") flag is set to 0 for solicited NAs returned by a Proxy/Server ROR and set to 1 for all other solicited and unsolicited NAs. For further study is whether solicited NAs for anycast targets apply for OMNI links. Since MNP-LLAs must be uniquely assigned to Clients to support correct ND protocol operation, however, no role is currently seen for assigning the same MNP-LLA to multiple Clients.

3.5.3. OMNI Neighbor Window Synchronization

In secured environments (e.g., such as between nodes on the same secured ANET), OMNI interface neighbors can exchange OAL packets using randomly-initialized and monotonically-increasing Identification values (modulo 2*32) without window synchronization. In environments where spoofing is considered a threat, OMNI interface neighbors instead invoke window synchronization in ND message exchanges to maintain send/receive window state in their respective neighbor cache entries as specified in [I-D.templin-6man-omni].

In the asymmetric window synchronization case, the initial ND message exchange establishes only the initiator's send window and the

responder's receive window such that a corresponding exchange would be needed to establish the reverse direction. In the symmetric case, the initiator and responder engage in a three-way handshake to symmetrically establish the send/receive windows of both parties.

3.6. OMNI Interface Encapsulation and Re-encapsulation

The OMNI interface admits original IP packets then (acting as an OAL source) performs OAL encapsulation and fragmentation as specified in [I-D.templin-6man-omni] while including an ORH if necessary as specified in Section 3.2.4. OAL encapsulation produces OAL packets, while OAL fragmentation turns them into OAL fragments which are then encapsulated in *NET headers as carrier packets.

For carrier packets undergoing re-encapsulation at an OAL intermediate node, the OMNI interface decrements the OAL IPv6 header Hop Limit and discards the carrier packet if the Hop Limit reaches 0. The intermediate node next removes the *NET encapsulation headers from the first segment and re-encapsulates the packet in new *NET encapsulation headers for the next segment.

When a Proxy/Server or Relay re-encapsulates a carrier packet received from a Client that includes an OAL but no ORH, it inserts an ORH immediately following the OAL header and adjusts the OAL payload length and destination address field. The inserted ORH will be removed by the LHS Bridge or Proxy/Server, but its insertion and removal will not interfere with reassembly at the final destination. For this reason, Clients must reserve 40 bytes for a maximum-length ORH when they perform OAL encapsulation (see: Section 3.9).

3.7. OMNI Interface Decapsulation

OMNI interfaces (acting as OAL destinations) decapsulate and reassemble OAL packets into original IP packets destined either to the AERO node itself or to a destination reached via an interface other than the OMNI interface the original IP packet was received on. When carrier packets containing OAL fragments arrive, the OMNI interface reassembles as discussed in Section 3.9.

3.8. OMNI Interface Data Origin Authentication

AERO nodes employ simple data origin authentication procedures. In particular:

- AERO Bridges and Proxy/Servers accept carrier packets received from secured underlying interfaces.

- AERO Proxy/Servers and Clients accept carrier packets and original IP packets that originate from within the same secured ANET.

- AERO Clients and Relays accept original IP packets from downstream network correspondents based on ingress filtering.

- AERO Clients, Relays and Proxy/Servers verify carrier packet UDP/IP encapsulation addresses according to [I-D.templin-6man-omni].

- AERO Clients (as well as Proxy/Servers and Relays when acting as OAL destinations) accept only carrier packets with Identification values within the current window for the OAL source neighbor when window synchronization is employed.

AERO nodes silently drop any packets that do not satisfy the above data origin authentication procedures. Further security considerations are discussed in Section 6.

3.9. OMNI Interface MTU

The OMNI interface observes the link nature of tunnels, including the Maximum Transmission Unit (MTU), Maximum Reassembly Unit (MRU) and the role of fragmentation and reassembly [I-D.ietf-intarea-tunnels]. The OMNI interface employs an OMNI Adaptation Layer (OAL) that accommodates multiple underlying links with diverse MTUs while observing both a minimum and per-path Maximum Payload Size (MPS). The functions of the OAL and the OMNI interface MTU/MRU/MPS are specified in [I-D.templin-6man-omni] with MTU/MRU both set to the constant value 9180 bytes, with minimum MPS set to 400 bytes, and with per-path MPS set to potentially larger values depending on the underlying path.

When the network layer presents an original IP packet to the OMNI interface, the OAL source encapsulates and fragments the original IP packet if necessary. When the network layer presents the OMNI interface with multiple original IP packets bound to the same OAL destination, the OAL source can concatenate them together into a single OAL super-packet as discussed in [I-D.templin-6man-omni]. The OAL source then fragments the OAL packet if necessary according to the minimum/path MPS such that the OAL headers appear in each fragment while the original IP packet header appears only in the first fragment. The OAL source then encapsulates each OAL fragment in *NET headers for transmission as carrier packets over an underlying interface connected to either a physical link such as Ethernet, WiFi and the like or a virtual link such as an Internet or higher-layer tunnel (see the definition of link in [RFC8200]).

Note: A Client that does not (yet) have neighbor cache state for a target may omit the ORH in carrier packets with the understanding that a Proxy/Server may insert an ORH on its behalf. For this reason, Clients reserve 40 bytes for the largest possible ORH in their OAL fragment size calculations.

Note: Although the ORH may be removed or replaced by a Bridge on the path (see: Section 3.10.3), this does not interfere with the destination's ability to reassemble. This is due to the fact that the ORH is not included in the fragmentable part; therefore, its removal does not invalidate the offset values in any fragment headers.

3.10. OMNI Interface Forwarding Algorithm

Original IP packets enter a node's OMNI interface either from the network layer (i.e., from a local application or the IP forwarding system) while carrier packets enter from the link layer (i.e., from an OMNI interface neighbor). All original IP packets and carrier packets entering a node's OMNI interface first undergo data origin authentication as discussed in Section 3.8. Those that satisfy data origin authentication are processed further, while all others are dropped silently.

Original IP packets that enter the OMNI interface from the network layer are forwarded to an OMNI interface neighbor using OAL encapsulation and fragmentation to produce carrier packets for transmission over underlying interfaces. (If routing indicates that the original IP packet should instead be forwarded back to the network layer, the packet is dropped to avoid looping). Carrier packets that enter the OMNI interface from the link layer are either re-encapsulated and re-admitted into the OMNI link, or reassembled and forwarded to the network layer where they are subject to either local delivery or IP forwarding. In all cases, the OAL MUST NOT decrement the network layer TTL/Hop-count since its forwarding actions occur below the network layer.

OMNI interfaces may have multiple underlying interfaces and/or neighbor cache entries for neighbors with multiple underlying interfaces (see Section 3.3). The OAL uses Interface Attribute traffic selectors (e.g., port number, flow specification, etc.) to select an outbound underlying interface for each OAL packet based on the node's own interface attributes, and also to select a destination link-layer address based on the neighbor's underlying interface attributes. AERO implementations SHOULD allow for traffic selector values to be modified at runtime through network management.

If multiple outgoing interfaces and/or neighbor interfaces match the traffic selectors, the AERO node replicates the OAL packet and sends one copy via each of the (outgoing / neighbor) interface pairs; otherwise, the node sends a single copy of the OAL packet via an interface with a matching traffic selector. (While not strictly required, successful delivery may be more likely when all fragments of the same OAL packet are sent over the same underlying interface.) AERO nodes keep track of which underlying interfaces are currently "reachable" or "unreachable", and only use "reachable" interfaces for forwarding purposes.

The following sections discuss the OMNI interface forwarding algorithms for Clients, Proxy/Servers and Bridges. In the following discussion, an original IP packet's destination address is said to "match" if it is the same as a cached address, or if it is covered by a cached prefix (which may be encoded in an MNP-LLA).

3.10.1. Client Forwarding Algorithm

When an original IP packet enters a Client's OMNI interface from the network layer the Client searches for a NCE that matches the destination. If there is a match, the Client selects one or more "reachable" neighbor interfaces in the entry for forwarding purposes. If there is no NCE, the Client instead either enqueues the original IP packet and invokes route optimization or forwards the original IP packet toward a Proxy/Server. The Client (acting as an OAL source) performs OAL encapsulation and sets the OAL destination address to the MNP-ULA of the target if there is a matching NCE; otherwise, it sets the OAL destination to the ADM-ULA of the Proxy/Server. If the Client has multiple original IP packets to send to the same neighbor, it can concatenate them in a single super-packet [I-D.templin-6man-omni]. The OAL source then performs fragmentation to create OAL fragments (see: Section 3.9), appends any *NET encapsulation, and sends the resulting carrier packets over underlying interfaces to the neighbor acting as an OAL destination.

If the neighbor interface selected for forwarding is located on the same OMNI link segment and not behind a NAT, the Client forwards the carrier packets directly according to the L2ADDR information for the neighbor. If the neighbor interface is behind a NAT on the same OMNI link segment, the Client instead forwards the initial carrier packets to its Proxy/Server (while inserting an ORH-0) and initiates NAT traversal procedures. If the Client's intended source underlying interface is also behind a NAT and located on the same OMNI link segment, it sends a "direct bubble" over the interface per [RFC6081][RFC4380] to the L2ADDR found in the neighbor cache in order to establish state in its own NAT by generating traffic toward the neighbor (note that no response to the bubble is expected).

The Client next sends an NS(NUD) message toward the MNP-ULA of the neighbor via its Proxy/Server as discussed in Section 3.15. If the Client receives an NA(NUD) from the neighbor over the underlying interface, it marks the neighbor interface as "trusted" and sends future carrier packets directly to the L2ADDR information for the neighbor instead of indirectly via the Proxy/Server. The Client must honor the neighbor cache maintenance procedure by sending additional direct bubbles and/or NS/NA(NUD) messages as discussed in [RFC6081][RFC4380] in order to keep NAT state alive as long as carrier packets are still flowing.

When a carrier packet enters a Client's OMNI interface from the link-layer, if the OAL destination matches one of the Client's ULAs the Client (acting as an OAL destination) verifies that the Identification is in-window for this OAL source, then reassembles and decapsulates as necessary and delivers the original IP packet to the network layer. Otherwise, the Client drops the original IP packet and MAY return a network-layer ICMP Destination Unreachable message subject to rate limiting (see: Section 3.11).

Note: Clients and their Proxy/Server (and other Client) peers can exchange original IP packets over ANET underlying interfaces without invoking the OAL, since the ANET is secured at the link and physical layers. By forwarding original IP packets without invoking the OAL, however, the ANET peers can engage only in classical path MTU discovery since the packets are subject to loss and/or corruption due to the various per-link MTU limitations that may occur within the ANET. Moreover, the original IP packets do not include either the OAL integrity check or per-packet Identification values that can be used for data origin authentication and link-layer retransmissions. The tradeoff therefore involves an assessment of the per-packet encapsulation overhead saved by bypassing the OAL vs. inheritance of classical network "brittleness". (Note however that ANET peers can send small original IP packets without invoking the OAL, while invoking the OAL for larger packets. This presents the beneficial aspects of both small packet efficiency and large packet robustness.)

3.10.2. Proxy/Server and Relay Forwarding Algorithm

When the Proxy/Server receives an original IP packet from the network layer, it drops the packet if routing indicates that it should be forwarded back to the network layer to avoid looping. Otherwise, the Proxy/Server regards the original IP packet the same as if it had arrived as carrier packets with OAL destination set to its own ADM-ULA. When the Proxy/Server receives carrier packets on underlying interfaces with OAL destination set to its own ADM-ULA, it performs OAL reassembly if necessary to obtain the original IP packet.

The Proxy/Server next searches for a NCE that matches the original IP destination and proceeds as follows:

- if the original IP packet destination matches a NCE, the Proxy/Sever uses one or more "reachable" neighbor interfaces in the entry for packet forwarding using OAL encapsulation and fragmentation according to the cached link-layer address information. If the neighbor interface is in a different OMNI link segment, the Proxy/Server performs OAL encapsulation and fragmentation, inserts an appropriate ORH and forwards the resulting carrier packets via the spanning tree to a Bridge; otherwise, it forwards the carrier packets directly to the neighbor. If the neighbor is behind a NAT, the Proxy/Server instead forwards initial carrier packets via a Bridge while sending an NS(NUD) to the neighbor. When the Proxy/Server receives the NA(NUD), it can begin forwarding carrier packets directly to the neighbor the same as discussed in Section 3.10.1 while sending additional NS(NUD) messages as necessary to maintain NAT state. Note that no direct bubbles are necessary since the Proxy/Server is by definition not located behind a NAT.

- else, if the original IP destination matches a non-MNP route in the IP forwarding table or an ADM-LLA assigned to the Proxy/Server's OMNI interface, the Proxy/Server acting as a Relay presents the original IP packet to the network layer for local delivery or IP forwarding.

- else, the Proxy/Server initiates address resolution as discussed in Section 3.14, while retaining initial original IP packets in a small queue awaiting address resolution completion.

When the Proxy/Server receives a carrier packet with OAL destination set to an MNP-ULA that does not match the MSP, it accepts the carrier packet only if data origin authentication succeeds and if there is a network layer routing table entry for a GUA route that matches the MNP-ULA. If there is no route, the Proxy/Server drops the carrier packet; otherwise, it reassembles and decapsulates to obtain the original IP packet and acts as a Relay to present it to the network layer where it will be delivered according to standard IP forwarding.

When the Proxy/Server receives a carrier packet from one of its Client neighbors with OAL destination set to another node, it forwards the packets via a matching NCE or via the spanning tree if there is no matching entry. When the Proxy/Server receives a carrier packet with OAL destination set to the MNP-ULA of one of its Client neighbors established through RS/RA exchanges, it accepts the carrier packet only if data origin authentication succeeds. If the NCE state is DEPARTED, the Proxy/Server inserts an ORH that encodes the MNP-ULA

destination suffix and changes the OAL destination address to the ADM-ULA of the new Proxy/Server, then re-encapsulates the carrier packet and forwards it to a Bridge which will eventually deliver it to the new Proxy/Server.

If the neighbor cache state for the MNP-ULA is REACHABLE, the Proxy/Server forwards the carrier packets to the Client which then must reassemble. (Note that the Proxy/Server does not reassemble carrier packets not explicitly addressed to its own ADM-ULA, since routing could direct some of the carrier packet of the same original IP packet through a different Proxy/Server.) In that case, the Client may receive fragments that are smaller than its link MTU but can still be reassembled.

Note: Proxy/Servers may receive carrier packets with ORHs that include additional forwarding information. Proxy/Servers use the forwarding information to determine the correct interface for forwarding to the target destination Client, then remove the ORH and forward the carrier packet. If the ORH information instead indicates that the Proxy/Server is responsible for reassembly, the Proxy/Server reassembles first before re-encapsulating (and possibly also re-fragmenting) then forwards to the target Client. For a full discussion of cases when the Proxy/Server may receive carrier packets with ORHs, see: Section 3.14.6.

Note: Clients and their Proxy/Server peers can exchange original IP packets over ANET underlying interfaces without invoking the OAL, since the ANET is secured at the link and physical layers. By forwarding original IP packets without invoking the OAL, however, the Client and Proxy/Server can engage only in classical path MTU discovery since the packets are subject to loss and/or corruption due to the various per-link MTU limitations that may occur within the ANET. Moreover, the original IP packets do not include either the OAL integrity check or per-packet Identification values that can be used for data origin authentication and link-layer retransmissions. The tradeoff therefore involves an assessment of the per-packet encapsulation overhead saved by bypassing the OAL vs. inheritance of classical network "brittleness". (Note however that ANET peers can send small original IP packets without invoking the OAL, while invoking the OAL for larger packets. This presents the beneficial aspects of both small packet efficiency and large packet robustness.)

Note: When a Proxy/Server receives a (non-OAL) original IP packet from an ANET Client, or a carrier packet with OAL destination set to its own ADM-ULA from any Client, the Proxy/Server reassembles if necessary then performs ROS functions on behalf of the Client. The Client may at some later time begin sending carrier packets to the OAL address of the actual target instead of the Proxy/Server, at

which point it may begin functioning as an ROS on its own behalf and thereby "override" the Proxy/Server's ROS role.

Note: Proxy/Servers forward secure control plane carrier packets via the secured spanning tree and forwards other carrier packets via the unsecured spanning tree. When a Proxy/Server receives a carrier packet from the secured spanning tree, it considers the message as authentic without having to verify upper layer authentication signatures. When a Proxy/Server receives a carrier packet from the unsecured spanning tree, it verifies any upper layer authentication signatures and/or forwards the unsecured message toward the destination which must apply data origin authentication.

Note: If the Proxy/Server has multiple original IP packets to send to the same neighbor, it can concatenate them in a single OAL super-packet [I-D.templin-6man-omni].

3.10.3. Bridge Forwarding Algorithm

Bridges forward carrier packets the same as any IPv6 router. Bridges convey carrier packets that encapsulate IPv6 ND control messages or routing protocol control messages using security encapsulations, and may convey carrier packets that encapsulate ordinary data without including security encapsulations. When the Bridge receives a carrier packet, it removes the outer *NET header and searches for a forwarding table entry that matches the OAL destination address. The Bridge then processes the packet as follows:

- if the carrier packet destination matches its ADM-ULA or the corresponding ADM-ULA Subnet Router Anycast address and the OAL header is followed by an ORH, the Bridge reassembles if necessary then sets aside the ORH and processes the carrier packet locally before forwarding. If the OAL packet contains an NA(NUD) message, the Bridge replaces the OMNI option Interface Attributes sub-option with information for its own interface while retaining the omIndex value supplied by the NA(NUD) message source. The Bridge next examines the ORH FMT code. If FMT-Mode indicates the destination is a Client on the open *NET (or, a Client behind a NAT for which NAT traversal procedures have already converged) the Bridge writes the MNP-ULA formed from the ORH Destination Suffix into the OAL destination. The Bridge then removes the ORH and forwards the packet using encapsulation based on L2ADDR. If the LHS Proxy/Server will forward to the Client without reassembly, the Bridge also writes the MNP-ULA into the OAL destination, then replaces the ORH with an ORH-0 and forwards the carrier packet to the LHS Proxy/Server, while also invoking NAT traversal procedures if necessary noting that no direct bubbles are necessary since only the target Client and not the Bridge is behind a NAT. If the

LHS Proxy/Server must perform reassembly before forwarding to the Client, the Bridge instead writes the ADM-ULA formed from the ORH SRT/LHS into the OAL destination address, replaces the ORH with an ORH-0 and forwards the carrier packet to the LHS Proxy/Server.

- else, if the carrier packet destination matches its ADM-ULA or the corresponding ADM-ULA Subnet Router Anycast address and the OAL header is not followed by an ORH with Segments Left set to 1, the Bridge submits the packet for reassembly. When reassembly is complete, the Bridge submits the original packet to the IP layer to support local applications such as BGP routing protocol sessions.

- else, if the carrier packet destination matches a forwarding table entry the Bridge forwards the carrier packet to the next hop. (If the destination matches an MSP without matching an MNP, however, the Bridge instead drops the packet and returns an ICMP Destination Unreachable message subject to rate limiting - see: Section 3.11).

- else, the Bridge drops the packet and returns an ICMP Destination Unreachable as above.

As for any IP router, the Bridge decrements the OAL IPv6 header Hop Limit when it forwards the carrier packet and drops the packet if the Hop Limit reaches 0. Therefore, only the Hop Limit in the OAL header is decremented and not the TTL/Hop Limit in the original IP packet header. Bridges do not insert OAL/ORH headers themselves; instead, they act as IPv6 routers and forward carrier packets based on their destination addresses while also possibly transforming larger ORHs into an ORH-0.

Bridges forward carrier packets received from a first segment via the unsecured spanning tree to the next segment also via the unsecured spanning tree. Bridges forward carrier packets received from a first segment via the secured spanning tree to the next segment also via the secured spanning tree. Bridges use a single IPv6 routing table that always determines the same next hop for a given OAL destination, where the secured/unsecured spanning tree is determined through the selection of the underlying interface to be used for transmission (i.e., a secured tunnel or an open INET interface).

3.11. OMNI Interface Error Handling

When an AERO node admits an original IP packet into the OMNI interface, it may receive link-layer or network-layer error indications.

A link-layer error indication is an ICMP error message generated by a router in the INET on the path to the neighbor or by the neighbor itself. The message includes an IP header with the address of the node that generated the error as the source address and with the link-layer address of the AERO node as the destination address.

The IP header is followed by an ICMP header that includes an error Type, Code and Checksum. Valid type values include "Destination Unreachable", "Time Exceeded" and "Parameter Problem" [RFC0792][RFC4443]. (OMNI interfaces ignore link-layer IPv4 "Fragmentation Needed" and IPv6 "Packet Too Big" messages for carrier packets that are no larger than the minimum/path MPS as discussed in Section 3.9, however these messages may provide useful hints of probe failures during path MPS probing.)

The ICMP header is followed by the leading portion of the carrier packet that generated the error, also known as the "packet-in-error". For ICMPv6, [RFC4443] specifies that the packet-in-error includes: "As much of invoking packet as possible without the ICMPv6 packet exceeding the minimum IPv6 MTU" (i.e., no more than 1280 bytes). For ICMPv4, [RFC0792] specifies that the packet-in-error includes: "Internet Header + 64 bits of Original Data Datagram", however [RFC1812] Section 4.3.2.3 updates this specification by stating: "the ICMP datagram SHOULD contain as much of the original datagram as possible without the length of the ICMP datagram exceeding 576 bytes".

The link-layer error message format is shown in Figure 5 (where, "L2" and "L3" refer to link-layer and network-layer, respectively):

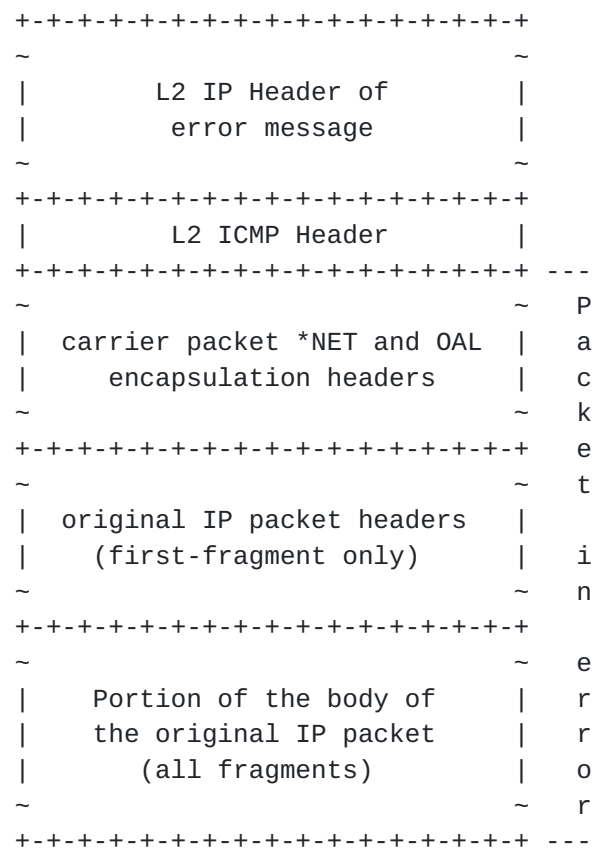

Figure 5: OMNI Interface Link-Layer Error Message Format

The AERO node rules for processing these link-layer error messages are as follows:

- When an AERO node receives a link-layer Parameter Problem message, it processes the message the same as described as for ordinary ICMP errors in the normative references [RFC0792][RFC4443].

- When an AERO node receives persistent link-layer Time Exceeded messages, the IP ID field may be wrapping before earlier fragments awaiting reassembly have been processed. In that case, the node should begin including integrity checks and/or institute rate limits for subsequent packets.

- When an AERO node receives persistent link-layer Destination Unreachable messages in response to carrier packets that it sends to one of its neighbor correspondents, the node should process the message as an indication that a path may be failing, and optionally initiate NUD over that path. If it receives Destination Unreachable messages over multiple paths, the node should allow future carrier packets destined to the correspondent

to flow through a default route and re-initiate route optimization.

- When an AERO Client receives persistent link-layer Destination Unreachable messages in response to carrier packets that it sends to one of its neighbor Proxy/Servers, the Client should mark the path as unusable and use another path. If it receives Destination Unreachable messages on many or all paths, the Client should associate with a new Proxy/Server and release its association with the old Proxy/Server as specified in Section 3.16.5.

- When an AERO Proxy/Server receives persistent link-layer Destination Unreachable messages in response to carrier packets that it sends to one of its neighbor Clients, the Proxy/Server should mark the underlying path as unusable and use another underlying path.

- When an AERO Proxy/Server receives link-layer Destination Unreachable messages in response to a carrier packet that it sends to one of its permanent neighbors, it treats the messages as an indication that the path to the neighbor may be failing. However, the dynamic routing protocol should soon reconverge and correct the temporary outage.

When an AERO Bridge receives a carrier packet for which the network-layer destination address is covered by an MSP, the Bridge drops the packet if there is no more-specific routing information for the destination and returns a network-layer Destination Unreachable message subject to rate limiting. The Bridge writes the network-layer source address of the original IP packet as the destination address and uses one of its non link-local addresses as the source address of the message.

When an AERO node receives a carrier packet for which reassembly is currently congested, it returns a network-layer Packet Too Big (PTB) message as discussed in [I-D.templin-6man-omni] (note that the PTB messages could indicate either "hard" or "soft" errors).

AERO nodes that act as OAL destinations include ICMPv6 error messages intended for the OAL source as sub-options in the OMNI option of secured uNA messages. When the OAL source receives the uNA message, it can extract the ICMPv6 error message enclosed in the OMNI option.

3.12. AERO Router Discovery, Prefix Delegation and Autoconfiguration

AERO Router Discovery, Prefix Delegation and Autoconfiguration are coordinated as discussed in the following Sections.

3.12.1. AERO Service Model

Each AERO Proxy/Server on the OMNI link is configured to facilitate Client prefix delegation/registration requests. Each Proxy/Server is provisioned with a database of MNP-to-Client ID mappings for all Clients enrolled in the AERO service, as well as any information necessary to authenticate each Client. The Client database is maintained by a central administrative authority for the OMNI link and securely distributed to all Proxy/Servers, e.g., via the Lightweight Directory Access Protocol (LDAP) [RFC4511], via static configuration, etc. Clients receive the same service regardless of the Proxy/Servers they select.

AERO Clients and Proxy/Servers use ND messages to maintain neighbor cache entries. AERO Proxy/Servers configure their OMNI interfaces as advertising NBMA interfaces, and therefore send unicast RA messages with a short Router Lifetime value (e.g., ReachableTime seconds) in response to a Client's RS message. Thereafter, Clients send additional RS messages to keep Proxy/Server state alive.

AERO Clients and Proxy/Servers include prefix delegation and/or registration parameters in RS/RA messages (see [I-D.templin-6man-omni]). The ND messages are exchanged between Client and Proxy/Server according to the prefix management schedule required by the service. If the Client knows its MNP in advance, it can employ prefix registration by including its MNP-LLA as the source address of an RS message and with an OMNI option with valid prefix registration information for the MNP. If the Proxy/Server accepts the Client's MNP assertion, it injects the MNP into the routing system and establishes the necessary neighbor cache state. If the Client does not have a pre-assigned MNP, it can instead employ prefix delegation by including the unspecified address (::) as the source address of an RS message and with an OMNI option with prefix delegation parameters to request an MNP.

The following sections specify the Client and Proxy/Server behavior.

3.12.2. AERO Client Behavior

AERO Clients discover the addresses of Proxy/Servers in a similar manner as described in [RFC5214]. Discovery methods include static configuration (e.g., from a flat-file map of Proxy/Server addresses and locations), or through an automated means such as Domain Name System (DNS) name resolution [RFC1035]. Alternatively, the Client can discover Proxy/Server addresses through a layer 2 data link login exchange, or through a unicast RA response to a multicast/anycast RS as described below. In the absence of other information, the Client can resolve the DNS Fully-Qualified Domain Name (FQDN)

"linkupnetworks.[domainname]" where "linkupnetworks" is a constant text string and "[domainname]" is a DNS suffix for the OMNI link (e.g., "example.com").

To associate with a Proxy/Server, the Client acts as a requesting router to request MNPs by preparing an RS message with prefix management parameters. If the Client already knows the Proxy/Server's ADM-LLA, it includes the LLA as the network-layer destination address; otherwise, the Client includes the (link-local) All-Routers multicast as the network-layer destination. If the Client already knows its own MNP-LLA, it can use the MNP-LLA as the network-layer source address and include an OMNI option with prefix registration information. Otherwise, the Client uses the unspecified address (::) as the network-layer source address and includes prefix delegation parameters in the OMNI option (see: [I-D.templin-6man-omni]).

The Client next includes Interface Attributes corresponding to the underlying interface over which it will send the RS message, and MAY include additional Interface Attributes specific to other underlying interfaces. Next, the Client submits the RS for OAL encapsulation and fragmentation if necessary with its own MNP-ULA and the Proxy/Server's ADM-ULA as the OAL addresses while selecting an Identification value and invoking window synchronization as specified in [I-D.templin-6man-omni].

The Client then sends the RS (either directly via Direct interfaces, via a VPN for VPNed interfaces, via an access router for ANET interfaces or via INET encapsulation for INET interfaces) then waits up to RetransTimer milliseconds for an RA message reply (see Section 3.12.3) (retrying up to MAX_RTR_SOLICITATIONS). If the Client receives no RAs, or if it receives an RA with Router Lifetime set to 0, the Client SHOULD abandon attempts through the first Proxy/Server and try another Proxy/Server. Otherwise, the Client processes the prefix information found in the RA message.

When the Client processes an RA, it first performs OAL reassembly and decapsulation if necessary then creates a NCE with the Proxy/Server's ADM-LLA as the network-layer address and the Proxy/Server's encapsulation and/or link-layer addresses as the link-layer address. The Client next records the RA Router Lifetime field value in the NCE as the time for which the Proxy/Server has committed to maintaining the MNP in the routing system via this underlying interface, and caches the other RA configuration information including Cur Hop Limit, M and O flags, Reachable Time and Retrans Timer. The Client then autoconfigures MNP-LLAs for any delegated MNPs and assigns them to the OMNI interface. The Client also caches any MSPs included in Route Information Options (RIOs) [RFC4191] as MSPs to associate with

the OMNI link, and assigns the MTU value in the MTU option to the underlying interface.

The Client then registers additional underlying interfaces with the Proxy/Server by sending RS messages via each additional interface as described above. The RS messages include the same parameters as for the initial RS/RA exchange, but with destination address set to the Proxy/Server's ADM-LLA. The Client finally sub-delegates the MNPs to its attached EUNs and/or the Client's own internal virtual interfaces as described in [I-D.templin-v6ops-pdhost] to support the Client's downstream attached "Internet of Things (IoT)". The Client then sends additional RS messages over each underlying interface before the Router Lifetime received for that interface expires.

After the Client registers its underlying interfaces, it may wish to change one or more registrations, e.g., if an interface changes address or becomes unavailable, if traffic selectors change, etc. To do so, the Client prepares an RS message to send over any available underlying interface as above. The RS includes an OMNI option with prefix registration/delegation information, with Interface Attributes specific to the selected underlying interface, and with any additional Interface Attributes specific to other underlying interfaces. When the Client receives the Proxy/Server's RA response, it has assurance that the Proxy/Server has been updated with the new information.

If the Client wishes to discontinue use of a Proxy/Server it issues an RS message over any underlying interface with an OMNI option with a prefix release indication. When the Proxy/Server processes the message, it releases the MNP, sets the NCE state for the Client to DEPARTED and returns an RA reply with Router Lifetime set to 0. After a short delay (e.g., 2 seconds), the Proxy/Server withdraws the MNP from the routing system.

3.12.3. AERO Proxy/Server Behavior

AERO Proxy/Servers act as IP routers and support a prefix delegation/registration service for Clients. Proxy/Servers arrange to add their ADM-LLAs to a static map of Proxy/Server addresses for the link and/or the DNS resource records for the FQDN "linkupnetworks.[domainname]" before entering service. Proxy/Server addresses should be geographically and/or topologically referenced, and made available for discovery by Clients on the OMNI link.

When a Proxy/Server receives a prospective Client's RS message on its OMNI interface, it SHOULD return an immediate RA reply with Router Lifetime set to 0 if it is currently too busy or otherwise unable to service the Client. Otherwise, the Proxy/Server performs OAL

reassembly and decapsulation if necessary, then authenticates the RS message and processes the prefix delegation/registration parameters. The Proxy/Server first determines the correct MNPs to provide to the Client by processing the MNP-LLA prefix parameters and/or the DHCPv6 OMNI sub-option. When the Proxy/Server returns the MNPs, it also creates a forwarding table entry for the MNP-ULA corresponding to each MNP so that the MNPs are propagated into the routing system (see: Section 3.2.3). For IPv6, the Proxy/Server creates an IPv6 forwarding table entry for each MNP. For IPv4, the Proxy/Server creates an IPv6 forwarding table entry with the IPv4-compatibility MNP-ULA prefix corresponding to the IPv4 address.

The Proxy/Server next creates a NCE for the Client using the base MNP-LLA as the network-layer address and with lifetime set to no more than the smallest prefix lifetime. Next, the Proxy/Server updates the NCE by recording the information in each Interface Attributes sub-option in the RS OMNI option. The Proxy/Server also records the actual OAL/*NET addresses and RS message window synchronization parameters (if any) in the NCE.

Next, the Proxy/Server prepares an RA message using its ADM-LLA as the network-layer source address and the network-layer source address of the RS message as the network-layer destination address. The Proxy/Server sets the Router Lifetime to the time for which it will maintain both this underlying interface individually and the NCE as a whole. The Proxy/Server also sets Cur Hop Limit, M and O flags, Reachable Time and Retrans Timer to values appropriate for the OMNI link. The Proxy/Server includes the MNPs, any other prefix management parameters and an OMNI option with no Interface Attributes but with an Origin Indication sub-option per [I-D.templin-6man-omni] with the mapped and obfuscated Port Number and IP address corresponding to the Client's own INET address in the case of INET Clients or to the Proxy/Server's INET-facing address for all other Clients. The Proxy/Server then includes one or more RIOs that encode the MSPs for the OMNI link, plus an MTU option (see Section 3.9). The Proxy/Server finally forwards the message to the Client using OAL encapsulation/fragmentation if necessary while including an acknowledgement if the RS invoked window synchronization.

After the initial RS/RA exchange, the Proxy/Server maintains a ReachableTime timer for each of the Client's underlying interfaces individually (and for the Client's NCE collectively) set to expire after ReachableTime seconds. If the Client (or Proxy) issues additional RS messages, the Proxy/Server sends an RA response and resets ReachableTime. If the Proxy/Server receives an ND message with a prefix release indication it sets the Client's NCE to the DEPARTED state and withdraws the MNP from the routing system after a short delay (e.g., 2 seconds). If ReachableTime expires before a new

RS is received on an individual underlying interface, the Proxy/Server marks the interface as DOWN. If ReachableTime expires before any new RS is received on any individual underlying interface, the Proxy/Server sets the NCE state to STALE and sets a 10 second timer. If the Proxy/Server has not received a new RS or ND message with a prefix release indication before the 10 second timer expires, it deletes the NCE and withdraws the MNP from the routing system.

The Proxy/Server processes any ND messages pertaining to the Client and returns an NA/RA reply in response to solicitations. The Proxy/Server may also issue unsolicited RA messages, e.g., with reconfigure parameters to cause the Client to renegotiate its prefix delegation/registrations, with Router Lifetime set to 0 if it can no longer service this Client, etc. Finally, If the NCE is in the DEPARTED state, the Proxy/Server deletes the entry after DepartTime expires.

Note: Clients SHOULD notify former Proxy/Servers of their departures, but Proxy/Servers are responsible for expiring neighbor cache entries and withdrawing routes even if no departure notification is received (e.g., if the Client leaves the network unexpectedly). Proxy/Servers SHOULD therefore set Router Lifetime to ReachableTime seconds in solicited RA messages to minimize persistent stale cache information in the absence of Client departure notifications. A short Router Lifetime also ensures that proactive RS/RA messaging between Clients and Proxy/Servers will keep any NAT state alive (see above).

Note: All Proxy/Servers on an OMNI link MUST advertise consistent values in the RA Cur Hop Limit, M and O flags, Reachable Time and Retrans Timer fields the same as for any link, since unpredictable behavior could result if different Proxy/Servers on the same link advertised different values.

3.12.3.1. DHCPv6-Based Prefix Registration

When a Client is not pre-provisioned with an MNP-LLA, it will need for the Proxy/Server to select one or more MNPs on its behalf and set up the correct state in the AERO routing service. (A Client with a pre-provisioned MNP may also request the Proxy/Server to select additional MNPs.) The DHCPv6 service [RFC8415] is used to support this requirement.

When a Client needs to have the Proxy/Server select MNPs, it sends an RS message with source address set to the unspecified address (::) and with an OMNI option that includes a DHCPv6 message sub-option with DHCPv6 Prefix Delegation (DHCPv6-PD) parameters. When the Proxy/Server receives the RS message, it extracts the DHCPv6-PD message from the OMNI option.

The Proxy/Server then acts as a "Proxy DHCPv6 Client" in a message exchange with the locally-resident DHCPv6 server, which delegates MNPs and returns a DHCPv6-PD Reply message. (If the Proxy/Server wishes to defer creation of MN state until the DHCPv6-PD Reply is received, it can instead act as a Lightweight DHCPv6 Relay Agent per [RFC6221] by encapsulating the DHCPv6-PD message in a Relay-forward/ reply exchange with Relay Message and Interface ID options.)

When the Proxy/Server receives the DHCPv6-PD Reply, it adds a route to the routing system and creates an MNP-LLA based on the delegated MNP. The Proxy/Server then sends an RA back to the Client with the (newly-created) MNP-LLA as the destination address and with the DHCPv6-PD Reply message coded in the OMNI option. When the Client receives the RA, it creates a default route, assigns the Subnet Router Anycast address and sets its MNP-LLA based on the delegated MNP.

Note: See [I-D.templin-6man-omni] for an MNP delegation alternative that avoids including a DHCPv6 message sub-option in the RS. Namely, when the Client requests a single MNP it can set the RS source to the unspecified address (::) and include a Node Identification sub-option and Preflen in the OMNI option (but with no DHCPv6 message sub-option). When the Proxy/Server receives the RS message, it forwards a self-generated DHCPv6 Solicit message to the DHCPv6 server on behalf of the Client. When the Proxy/Server receives the DHCPv6 Reply, it prepares an RA message with an OMNI option with Preflen information (but with no DHCPv6 message sub-option), then places the (newly-created) MNP-LLA in the RA destination address and returns the message to the Client.

3.13. The AERO Proxy Function

Clients connect to the OMNI link via Proxy/Servers, with one Proxy/ Server for each underlying interface. Each of the Client's Proxy/ Servers must be informed of all of the Client's additional underlying interfaces. For Clients on Direct and VPNed underlying interfaces the Proxy/Server "A" for that interface is directly connected, for Clients on ANET underlying interfaces Proxy/Server "A" is located on the ANET/INET boundary, and for Clients on INET underlying interfaces Proxy/Server "A" is located somewhere in the connected Internetwork. When the Client registers with Proxy/Server "A", it must also report the registration to any other Proxy/Servers for other underlying interfaces "B", "C", "D", etc. for which an underlying interface relationship has already been established. The Proxy/Server satisfies these requirements as follows:

- when Proxy/Server "A" receives an RS message from a new Client, it first verifies that the OAL Identification is within the window

for the NCE that matches the MNP-ULA for this Client neighbor and authenticates the message. (If no NCE was found, Proxy/Server "A instead creates one in the STALE state and returns an RA message with an authentication signature and any window synchronization parameters.) Proxy/Server "A" then examines the network-layer destination address. If the destination address is the ADM-LLA of a different Proxy/Server "B" (or, if the OMNI option included an MS-Register sub-option with the ADM-LLAs of one or more different Proxy/Servers "B", "C", "D", etc.), Proxy/Server "A" prepares a separate proxyed version of the RS message with an OAL header with source set to its own ADM-ULA and destination set to Proxy/Server "B"'s ADM-ULA for each such "B". Proxy/Server "A" also includes an OMNI header with an Interface Attributes option that includes its own INET address, a unique UDP Port Number for this Client, and SRT/LHS information. Proxy/Server "A" then sets the S/T-omIndex to the appropriate value for this Client underlying interface, then forwards the message into the OMNI link secured spanning tree. (Note: including a unique Port Number allows Proxy/Server "B" to distinguish different Clients located behind the same Proxy/Server "A" at the link-layer, whereas the link-layer addresses would otherwise be indistinguishable.)

- when Proxy/Server "B" receives the RS, it authenticates the message then creates or updates a NCE for the Client with Proxy/Server "A"'s Interface Attributes as the link-layer address information for this S/T-omIndex and caches any window synchronization parameters supplied by the Client. Proxy/Server "B" then prepares an RA message with source set to its own LLA and destination set to the Client's MNP-LLA, and with any window synchronization acknowledgements. Proxy/Server "B" then encapsulates the RA in an OAL header with source set to its own ADM-ULA and destination set to the ADM-ULA of Proxy/Server "A, performs fragmentation if necessary, then sends the resulting carrier packets into the secured spanning tree.

- when Proxy/Server "A" reassembles the RA, it locates the Client NCE based on the RA destination LLA. Proxy/Server "A" then re-encapsulates the RA message with OAL source set to its own ADM-ULA and OAL destination set to the MNP-ULA of the Client, includes an authentication signature if necessary, fragments if necessary and returns the fragments to the Client.

- The Client repeats this process with each Proxy/Server "B", "C", "D" for each of its additional underlying interfaces. When the Client includes multiple Proxy/Server IDs in the MS-Register option, it may receive multiple RAs - each with identical window acknowledgements. The Client can then create an independent NCE for each responding Proxy/Server and complete the window

synchronization even though all Proxy/Servers received the same ISS.

After the initial RS/RA exchanges each Proxy/Server forwards any of the Client's carrier packets with OAL destinations for which there is no matching NCE to a Bridge using OAL encapsulation with its own ADM-ULA as the source and the destination determined by the ORH supplied by the Client. The Proxy/Server instead forwards any carrier packets destined to a neighbor cache target directly to the target according to the OAL/link-layer information - the process of establishing neighbor cache entries is specified in Section 3.14.

While the Client is still associated with each Proxy/Server "A", "A" can send NS, RS and/or unsolicited NA messages to update the neighbor cache entries of other AERO nodes on behalf of the Client and/or to convey Interface Attribute updates. This allows for higher-frequency Proxy-initiated RS/RA messaging over well-connected INET infrastructure supplemented by lower-frequency Client-initiated RS/RA messaging over constrained ANET data links.

If any Proxy/Server "B", "C", "D" ceases to send solicited advertisements, Proxy/Server "A" sends unsolicited RAs to the Client with destination set to (link-local) All-Nodes multicast and with Router Lifetime set to zero to inform Clients that a Proxy/Server has failed. Although Proxy/Server "A" can engage in ND exchanges on behalf of the Client, the Client can also send ND messages on its own behalf, e.g., if it is in a better position than "A" to convey Interface Attribute changes, etc. The ND messages sent by the Client include the Client's MNP-LLA as the source in order to differentiate them from the ND messages sent by Proxy/Server "A".

If the Client becomes unreachable over an underlying interface, Proxy/Server "A" sets the NCE state to DEPARTED and retains the entry for DepartTime seconds. While the state is DEPARTED, Proxy/Server "A" forwards any carrier packets destined to the Client to a Bridge via OAL/ORH encapsulation. When DepartTime expires, Proxy/Server "A" deletes the NCE and discards any further carrier packets destined to the former Client.

In some ANETs that employ a Proxy/Server, the Client's MNP can be injected into the ANET routing system. In that case, the Client can send original IP packets without invoking the OAL so that the ANET routing system transports the original IP packets to the Proxy. This can be very beneficial, e.g., if the Client connects to the ANET via low-end data links such as some aviation wireless links.

If the ANET first-hop access router is on the same underlying link as the Client and recognizes the AERO/OMNI protocol, the Client can

avoid OAL encapsulation for both its control and data messages. When the Client connects to the link, it can send an unencapsulated RS message with source address set to its own MNP-LLA (or to a Temporary LLA), and with destination address set to the ADM-LLA of the Client's selected Proxy/Server or to (link-local) All-Routers multicast. The Client includes an OMNI option formatted as specified in [I-D.templin-6man-omni]. The Client then sends the unencapsulated RS message, which will be intercepted by the AERO-Aware access router.

The ANET access router then performs OAL encapsulation on the RS message and forwards it to a Proxy/Server at the ANET/INET boundary. When the access router and Proxy/Server are one and the same node, the Proxy/Server would share and underlying link with the Client but its message exchanges with outside correspondents would need to pass through a security gateway at the ANET/INET border. The method for deploying access routers and Proxys (i.e. as a single node or multiple nodes) is an ANET-local administrative consideration.

Note: The Proxy/Server can apply packing as discussed in [I-D.templin-6man-omni] if an opportunity arises to concatenate multiple original IP packets that will be destined to the same neighbor.

3.13.1. Detecting and Responding to Proxy/Server Failures

In environments where fast recovery from Proxy/Server failure is required, Proxy/Server "A" SHOULD use proactive Neighbor Unreachability Detection (NUD) to track peer Proxy/Server "B" reachability in a similar fashion as for Bidirectional Forwarding Detection (BFD) [RFC5880]. Proxy/Server "A" can then quickly detect and react to failures so that cached information is re-established through alternate paths. The NUD control messaging is carried only over well-connected ground domain networks (i.e., and not low-end aeronautical radio links) and can therefore be tuned for rapid response.

Proxy/Server "A" performs proactive NUD with peer Proxy/Server "B" for which there are currently active Clients by sending continuous NS messages in rapid succession, e.g., one message per second. Proxy/Server "A" sends the NS message via the spanning tree with its own ADM-LLA as the source and the ADM-LLA of the peer Proxy/Server "B" as the destination. When Proxy/Server "A" is also sending RS messages to the peer Proxy/Server "B" on behalf of ANET Clients, the resulting RA responses can be considered as equivalent hints of forward progress. This means that Proxy/Server "B" need not also send a periodic NS if it has already sent an RS within the same period. If the peer Proxy/Server "B" fails (i.e., if "A" ceases to receive

advertisements), Proxy/Server "A" can quickly inform Clients by sending multicast RA messages on the ANET interface.

Proxy/Server "A" sends RA messages on the ANET interface with source address set to Proxy/Server "B"'s address, destination address set to (link-local) All-Nodes multicast, and Router Lifetime set to 0. Proxy/Server "A" SHOULD send MAX_FINAL_RTR_ADVERTISEMENTS RA messages separated by small delays [RFC4861]. Any Clients on the ANET that had been using the failed Proxy/Server "B" will receive the RA messages and associate with a new Proxy/Server.

3.13.2. Point-to-Multipoint Proxy/Server Coordination

In environments where Client messaging over ANETs is bandwidth-limited and/or expensive, Clients can enlist the services of Proxy/Server "A" to coordinate with multiple Proxy/Servers "B", "C", "D" etc. in a single RS/RA message exchange. The Client can send a single RS message to (link-local) All-Routers multicast that includes the ID's of multiple Proxy/Servers in MS-Register sub-options of the OMNI option.

When Proxy/Server "A" receives the RS and processes the OMNI option, it sends a separate RS to each MS-Register Proxy/Server ID. When Proxy/Server "A" receives an RA, it can optionally return an immediate "singleton" RA to the Client or record the Proxy/Server's ID for inclusion in a pending "aggregate" RA message. Proxy/Server "A" can then return aggregate RA messages to the Client including multiple Proxy/Server IDs in order to conserve bandwidth. Each RA includes a proper subset of the Proxy/Server IDs from the original RS message, and Proxy/Server "A" must ensure that the message contents of each RA are consistent with the information received from the (aggregated) additional Proxy/Servers.

Clients can thereafter employ efficient point-to-multipoint Proxy/Server coordination under the assistance of Proxy/Server "A" to reduce the number of messages sent over the ANET while enlisting the support of multiple Proxy/Servers for fault tolerance. Clients can further include MS-Release sub-options in IPv6 ND messages to request Proxy/Server "A" to release from former Proxy/Servers via the procedures discussed in Section 3.16.5.

The OMNI interface specification [I-D.templin-6man-omni] provides further discussion of the RS/RA messaging involved in point-to-multipoint coordination.

3.14. AERO Route Optimization

AERO nodes invoke route optimization when they need to forward packets to new target destinations. Route optimization is based on IPv6 ND Address Resolution messaging between a Route Optimization Source (ROS) and Route Optimization Responder (ROR). Route optimization is initiated by the first eligible ROS closest to the source as follows:

- For Clients on VPNed and Direct interfaces, the Proxy/Server is the ROS.
- For Clients on ANET interfaces, either the Client or the Proxy/Server may be the ROS.
- For Clients on INET interfaces, the Client itself is the ROS.
- For correspondent nodes on INET/EUN interfaces serviced by a Relay, the Relay is the ROS.

The route optimization procedure is conducted between the ROS and the nearest Proxy/Server/Relay for the target selected by routing as the ROR. In this arrangement, the ROS is always the Client or Proxy/Server/Relay nearest the source over the selected source underlying interface, while the ROR is always a Proxy/Server/Relay that services the target regardless of the target underlying interface.

The AERO routing system directs a route optimization solicitation sent by the ROS to the nearest available ROR, which returns a route optimization reply. The exact ROR selected is unimportant; all that matters is that the route optimization information returned must be current and authentic. The ROS is responsible for periodically refreshing the route optimization, and the ROR is responsible for quickly informing the ROS of any changes.

The procedures are specified in the following sections.

3.14.1. Route Optimization Initiation

When an original IP packet from a source node destined to a target node arrives, the ROS checks for a NCE with an MNP-LLA that matches the target destination. If there is a NCE in the REACHABLE state, the ROS invokes the OAL and forwards the resulting carrier packets according to the cached state then returns from processing. Otherwise, if there is no NCE the ROS creates one in the INCOMPLETE state.

The ROS next places the original IP packet on a short queue then sends an NS message for Address Resolution (NS(AR)) to receive a solicited NA(AR) message from an ROR. The NS(AR) message must be sent securely, and includes:

- the LLA of the ROS as the source address.
- the MNP-LLA corresponding to the original IP packet's destination as the Target Address, e.g., for 2001:db8:1:2::10:2000 the Target Address is fe80::2001:db8:1:2.
- the Solicited-Node multicast address [RFC4291] formed from the lower 24 bits of the original IP packet's destination as the destination address, e.g., for 2001:db8:1:2::10:2000 the NS(AR) destination address is ff02:0:0:0:0:1:ff10:2000.

The NS(AR) message also includes an OMNI option with an Interface Attributes entry for the sending interface, with S/T-omIndex set to 0 and with Preflen set to the prefix length associated with the NS(AR) source. The ROS then selects an Identification value submits the NS(AR) message for OAL encapsulation with OAL source set to its own ULA and OAL destination set to the ULA corresponding to the target. (The ROS does not include any window synchronization parameters, since it will never forward OAL packets directly to the ROR).

The ROS then sends the resulting carrier packet(s) into the secured spanning tree without decrementing the network-layer TTL/Hop Limit field. (When the ROS is an INET Client, it instead sends the resulting carrier packets to the ADM-ULA of one of its current Proxy/Servers. The Proxy/Server then reassembles if necessary, verifies the NS(AR) signature, then re-encapsulates with the OAL source set to its own ADM-ULA and OAL destination set to the ULA corresponding to the target. The Proxy/Server then fragments if necessary and sends the resulting carrier packets into the secured spanning tree on behalf of the Client.)

3.14.2. Relaying the NS(AR) *NET Packet(s)

When the Bridge receives the carrier packet(s) containing the RS from the ROS, it discards the *NET headers and determines the next hop by consulting its standard IPv6 forwarding table for the OAL header destination address. The Bridge then decrements the OAL header Hop-Limit, then re-encapsulates and forwards the carrier packet(s) via the secured spanning tree the same as for any IPv6 router, where it may traverse multiple OMNI link segments. The final-hop Bridge will deliver the carrier packet(s) via the secured spanning tree to a Proxy/Server or Relay that services the target.

3.14.3. Processing the NS(AR) and Sending the NA(AR)

When the target Proxy/Server (or Relay) receives the secured carrier packet(s), it reassembles if necessary then examines the NS(AR) target to determine whether it has a matching NCE and/or non-MNP route. If there is no match, the Proxy/Server drops the message. Otherwise, the Proxy/Server/Relay continues processing as follows:

- if the NS(AR) target matches a Client NCE in the DEPARTED state, the Proxy/Server re-encapsulates while setting the OAL source to the ULA of the ROS and OAL destination address to the ADM-ULA of the Client's new Proxy/Server. The (old) Proxy/Server then fragments if necessary and forwards the resulting carrier packet(s) over the secured spanning tree then returns from processing.

- If the NS(AR) target matches the MNP-LLA of a Client NCE in the REACHABLE state, the Proxy/Server makes note of whether the NS (AR) arrived from the secured or unsecured spanning tree then acts as an ROR to provide route optimization information on behalf of the Client. (Note that if the message arrived via the secured spanning tree the ROR need not perform further authentication, but if it arrived over an open INET underlying interface it must check for and verify the message authentication signature before accepting.)

- If the NS(AR) target matches one of its non-MNP routes, the Relay acts as both an ROR and a route optimization target, since the Relay forwards IP packets toward the (fixed network) target at the network layer.

The ROR next checks the target NCE for a Report List entry that matches the NS(AR) source LLA/ULA of the ROS. If there is a Report List entry, the ROR refreshes ReportTime for this ROR; otherwise, the ROR creates a new entry for the ROS and records both the LLA and ULA.

The ROR then prepares a (solicited) NA(AR) message to return to the ROS with the source address set to the target's MNP-LLA, the destination address set to the NS(AR) LLA source address and the Target Address set to the same value that appeared in the NS(AR). The ROR then includes an OMNI option with Preflen set to the prefix length associated with the NA(AR) source address. The ROR next includes Interface Attributes in the OMNI option for all of the target's underlying interfaces with current information for each interface.

For each Interface Attributes sub-option, the ROR sets the L2ADDR according to the Proxy/Server's INET address for VPNed or Direct

interfaces, to the INET address of the Proxy/Server for proxyed interfaces or to the Client's INET address for INET interfaces. The ROR then includes the lower 32 bits of the Proxy/Server's ADM-ULA as the LHS, encodes the ADM-ULA prefix length code in the SRT field and sets FMT as specified in Section 3.3.

The ROR then sets the NA(AR) message R flag to 1 (as a router) and S flag to 1 (as a response to a solicitation) and sets the O flag to 0 (as a proxy) and sets the OMNI header S/T-omIndex to 0. The ROR finally submits the NA(AR) for OAL encapsulation with source set to its own ULA and destination set to the same ULA that appeared in the NS(AR) OAL source, then performs OAL encapsulation and fragmentation using the same Identification value that appeared in the NS(AR) and finally forwards the resulting (*NET-encapsulated) carrier packets via the secured spanning tree without decrementing the network-layer TTL/Hop Limit field.

3.14.4. Relaying the NA(AR)

When the Bridge receives NA(AR) carrier packets from the ROR, it discards the *NET header and determines the next hop by consulting its standard IPv6 forwarding table for the OAL header destination address. The Bridge then decrements the OAL header Hop-Limit, re-encapsulates the carrier packet and forwards it via the secured spanning tree the same as for any IPv6 router, where it may traverse multiple OMNI link segments. The final-hop Bridge will deliver the carrier packet via the secured spanning tree to a Proxy/Server for the ROS.

3.14.5. Processing the NA(AR)

When the ROS receives the NA(AR) message from the ROR, it first searches for a NCE that matches the NA(AR) LLA source address. The ROS then processes the message the same as for standard IPv6 Address Resolution [RFC4861]. In the process, it caches all OMNI option information in the target NCE (including all Interface Attributes).

When the ROS is a Client, the solicited NA(AR) message will first be delivered via the secured spanning tree to the Proxy/Server that forwarded the NS(AR), which reassembles if necessary. The Proxy/Server then forwards the message to the Client while re-encapsulating and re-fragmenting if necessary. If the Client is on an ANET, ANET physical security and protected spectrum ensures security for the unmodified NA(AR); if the Client is on the open INET the Proxy/Server must instead insert an authentication signature. The Proxy/Server uses its own ADM-ULA as the OAL source and the MNP-ULA of the Client as the OAL destination.

3.14.6. Forwarding Packets to Route Optimized Targets

After the ROS receives the route optimization NA(AR) and updates the target NCE, it can begin forwarding packets along the best paths based on the target's Interface Attributes. The ROS selects target underlying interfaces according to traffic selectors and/or any other traffic discriminators, however each underlying interface selected must first establish window synchronization state if necessary.

To establish window synchronization state, the ROS performs a secured unicast NS/NA(WIN) exchange with window synchronization parameters according to Interface Attribute FMTs. The ROS prepares an NS(WIN) with its own LLA as the source and the MNP-LLA of the target Client as the destination if FMT-Forward is set; otherwise, it sets the ADM-LLA of the LHS Proxy/Server as the destination. The ROS then encapsulates the NS(WIN) in an OAL header with its own ULA as the source. If the ROS is the Client, it sets the OAL destination to the ADM-ULA of its Proxy/Server, includes an authentication signature if necessary, and includes an ORH-1 with Trailer Clear. The Client sets the ORH Segments Left to 1 and includes valid SRT/LHS information for the LHS Proxy/Server with L2ADDR set to 0, then forwards the NS(WIN) to its own Proxy/Server which must reassemble and verify the authentication signature if necessary. The Proxy/Server then re-encapsulates, re-fragments and forwards the NS(WIN) carrier packets into the secured spanning tree with its own ADM-ULA as the OAL source and the ADM-ULA of the LHS Proxy/Server as the OAL destination while replacing the ORH-1 with an ORH-0. (If the ROS was the Proxy/Server itself, it instead includes an ORH-0, and forwards the carrier packets into the secured spanning tree.)

When an LHS Proxy/Server receives the NS(WIN) it first reassembles if necessary. If the NS(WIN) destination is its own ADM-LLA, the LHS Proxy/Server creates an NCE based on the NS(WIN) source LLA, caches the window synchronization information, and prepares an NA(WIN) while using its own ADM-LLA as the source and the ROS LLA as the destination. The LHS Proxy/Server then encapsulates the NA(WIN) in an OAL header with source set to its own ADM-ULA and destination set to the NS(WIN) OAL source. The LHS Proxy/Server then fragments if necessary includes an ORH-0 with omIndex set to the S/T-omIndex value found in the NS(WIN) OMNI option, then forwards the resulting carrier packets into the secured spanning tree which will deliver them to the ROS Proxy/Server.

If the NS(WIN) destination is the MNP-LLA of the target Client, the LHS Proxy/Server instead includes an authentication signature if necessary then re-encapsulates using the same OAL source and the MNP-ULA of the target as the OAL destination while removing the ORH-0. The LHS Proxy/Server then forwards the NS(WIN) to the target over the

underlying interface identified by the ORH-0 omIndex (or, over any underlying interface if the omIndex is 0). When the target receives the NS(WIN), it verifies the authentication signature if necessary then creates an NCE for the ROS LLA, caches the window synchronization information and prepares an NA(WIN) to return to the ROS with its MNP-LLA as the source and the LLA of the ROS as the destination, and with an authentication signature if necessary. The target Client then encapsulates the NA(WIN) in an OAL header with its own MNP-ULA as the source, the ADM-ULA of the LHS Proxy/Server as the destination, and with an ORH-1 with SRT/LHS information copied from the ADM-ULA of the ROS Proxy/Server found in the NS(WIN) OAL source address. The target Client then sets the ORH-1 omIndex to the S/T-omIndex value found in the NS(WIN) OMNI option, then forward the message to the LHS Proxy/Server.

When the LHS Proxy/Server receives the message, it reassembles if necessary, verifies the authentication signature if necessary then re-encapsulates using its own ADM-ULA as the source and the ADM-ULA of the ROS Proxy/Server as the destination The LHS Proxy/Server then re-fragments and forwards the NS(WIN) carrier packets into the spanning tree while replacing the ORH-1 with an ORH-0. When the ROS Proxy/Server receives the NA(WIN), it reassembles if necessary then updates the target NCE based on the message contents if the Proxy/Server itself is the ROS. If the NS(WIN) source was the ADM-LLA of the LHS Proxy/Server, the ROS must create and maintain a NCE for the LHS Proxy/Server which it must regard as a companion to the existing MNP-LLA NCE for the target Client. (The NCE for the LHS Proxy/Server can also be shared by multiple target Client NCEs if the ROS communicates with multiple active targets located behind the same LHS Proxy/Server.) If the ROS is the Client, the Proxy/Server instead inserts an authentication signature if necessary, removes the ORH-0 then re-encapsulates and re-fragments if necessary while changing the OAL destination to the MNP-ULA of the ROS Client. The Proxy/Server then forwards the NA(WIN) to the ROS Client over the underlying interface identified by the ORH-0 omIndex which then updates its own NCE based on the target Client MNP-LLA or LHS Proxy/Server ADM-LLA. The ROS (whether the Proxy/Server or the Client itself) finally arranges to return an acknowledgement if requested by the NA(WIN).

After window synchronization state has been established, the ROS can begin forwarding packets while performing additional NS/NA(WIN) exchanges as above to update window state and/or test reachability. When the ROS forwards carrier packets, it proceeds as specified in Section 3.2.7. These forwarding procedures apply to the case where the selected target interface SRT/LHS codes indicate that the interface is located in a foreign OMNI link segment. In that case, the ROS must include ORHs and send the resulting carrier packets into the spanning tree.

If the SRT/LHS codes indicate that the interface is in the local OMNI link segment, the ROS can instead forward carrier packets directly to the LHS Proxy/Server using the L2ADDR for encapsulation, or even to the target Client itself while invoking NAT traversal if necessary. When the ROS sends packets directly to the LHS Proxy/Server, it must still include an ORH-0 so the Proxy/Server can determine the correct target Client interface for (re)forwarding. If the LHS Proxy/Server is required to reassemble, the ROS sets the OAL destination to the ADM-ULA of the LHS Proxy/Server; otherwise, it sets the OAL destination to the MNP-ULA of the target Client itself. When the ROS sends packets directly to the target Client, it need not include an ORH. In both cases, the ROS can also begin applying OAL header compression as specified in [I-D.templin-6man-omni], since the OAL header is not examined by any forwarding nodes in the path.

While the ROS continues to actively forward packets to the target Client, it is responsible for updating window synchronization state and per-interface reachability before expiration. Window synchronization state is shared by all underlying interfaces in the ROS' NCE that use the same destination LLA so that a single NS/NA(NUD) exchange applies for all interfaces regardless of the (single) interface used to conduct the exchange. However, the window synchronization exchange only confirms target Client reachability over the specific interface used to conduct the exchange. Reachability for other underlying interfaces that share the same window synchronization state must be determined individually using NS/NA(NUD) messages which need not be secured as long as they use in-window Identifications and do not update other state information.

3.15. Neighbor Unreachability Detection (NUD)

AERO nodes perform Neighbor Unreachability Detection (NUD) per [RFC4861] either reactively in response to persistent link-layer errors (see Section 3.11) or proactively to confirm reachability. The NUD algorithm is based on periodic control message exchanges and may further be seeded by ND hints of forward progress, but care must be taken to avoid inferring reachability based on spoofed information. For example, IPv6 ND message exchanges that include authentication codes and/or in-window Identifications may be considered as acceptable hints of forward progress, while spurious random carrier packets should be ignored.

AERO nodes can use standard NS/NA(NUD) exchanges sent over the OMNI link secured spanning tree (i.e. the same as described above for NS/NA(WIN)) to test reachability without risk of DoS attacks from nodes pretending to be a neighbor. These NS/NA(NUD) messages use the unicast LLAs and ULAs of the parties involved in the NUD test the same as for standard IPv6 ND over the secured spanning tree. When

only reachability information is required without updating any other NCE state, unsecured NS/NA(NUD) messages may instead be exchanged directly between neighbors as long as they include in-window Identifications.

When an ROR directs an ROS to a target neighbor with one or more link-layer addresses, the ROS probes each unsecured target underlying interface either proactively or on-demand of carrier packets directed to the path by multilink forwarding to maintain the interface's state as reachable. Probing is performed through NS(NUD) messages over either the secured or unsecured spanning tree, or through NS(NUD) messages sent directly to an underlying interface of the target itself. While testing a target underlying interface, the ROS can optionally continue to forward carrier packets via alternate interfaces and/or maintain a small queue of carrier packets until target reachability is confirmed.

NS(NUD) messages are encapsulated, fragmented and transmitted as carrier packets the same as for ordinary original IP data packets, however the encapsulated destinations are the LLA of the ROS and either the ADM-LLA of the LHS Proxy/Server or the MNP-LLA of the target itself. The ROS encapsulates the NS(NUD) message the same as described in Section 3.2.7, however Destination Suffixes (if present) are set according to the LLA destination (i.e., and not a ULA/GUA destination). The ROS sets the S/T-omIndex in the NS(NUD) OMNI header to identify the underlying interface used for forwarding. The ROS also includes an ORH with SRT/LHS/LLADDR information the same as for ordinary data packets, but no authentication signatures are included. The ROS then fragments the OAL packet and forwards the resulting carrier packets into the unsecured spanning tree or directly to the target (or LHS Proxy/Server) if it is in the local segment.

When the target (or LHS Proxy/Server) receives the NS(NUD) carrier packets, it verifies that it has a NCE for this ROS and that the Identification is in-window, then submits the carrier packets for reassembly. The node then searches for Interface Attributes in its NCE for the ROS that match the NS(NUD) S/T-omIndex and uses the SRT/LHS/L2ADDR and FMT information to prepare an ORH for the NA(NUD) reply. The node then prepare the NA(NUD) with the source and destination LLAs reversed, encapsulates and sets the OAL source and destination, sets the NA(NUD) S/T-omIndex to the index of the underlying interface the NS(NUD) arrived on and sets the Target Address to the same value included in the NS(NUD). The target next sets the R flag to 1, the S flag to 1 and the O flag to 1, then selects an in-window Identification for the ROS and performs fragmentation. The node then forwards the carrier packets into the

unsecured spanning tree, directly to the ROS if it is in the local segment or directly to a Bridge in the local segment.

When the ROS receives the NA(NUD), it marks the target underlying interface tested as "reachable". Note that underlying interface states are maintained independently of the overall NCE REACHABLE state, and that a single NCE may have multiple target underlying interfaces in various states "reachable" and otherwise while the NCE state as a whole remains REACHABLE.

Note also that the exchange of NS/NA(NUD) messages has the useful side-benefit of opening holes in NATs that may be useful for NAT traversal.

3.16. Mobility Management and Quality of Service (QoS)

AERO is a Distributed Mobility Management (DMM) service. Each Proxy/Server is responsible for only a subset of the Clients on the OMNI link, as opposed to a Centralized Mobility Management (CMM) service where there is a single network mobility collective entity for all Clients. Clients coordinate with their associated Proxy/Servers via RS/RA exchanges to maintain the DMM profile, and the AERO routing system tracks all current Client/Proxy/Server peering relationships.

Proxy/Servers provide default routing and mobility/multilink services for their dependent Clients. Clients are responsible for maintaining neighbor relationships with their Proxy/Servers through periodic RS/RA exchanges, which also serves to confirm neighbor reachability. When a Client's underlying Interface Attributes change, the Client is responsible for updating the Proxy/Server with this new information. Note that when there is a Proxy/Server in the path, the Proxy function can also perform some RS/RA exchanges on the Client's behalf.

Mobility management messaging is based on the transmission and reception of unsolicited Neighbor Advertisement (uNA) messages. Each uNA message sets the IPv6 source address to the LLA of the ROR and the destination address to the unicast LLA of the ROS.

Mobility management considerations are specified in the following sections.

3.16.1. Mobility Update Messaging

RORs accommodate Client mobility and/or multilink change events by sending secured uNA messages to each ROS in the target Client's Report List. When an ROR sends a uNA message, it sets the IPv6 source address to the its own LLA, sets the destination address to

the ROS LLA (i.e., an MNP-LLA if the ROS is a Client and an ADM-LLA if the ROS is a Proxy/Server) and sets the Target Address to the Client's MNP-LLA. The ROR also includes an OMNI option with Preflen set to the prefix length associated with the Client's MNP-LLA, with Interface Attributes for the target Client's underlying interfaces and with the OMNI header S/T-omIndex set to 0. The ROR then sets the uNA R flag to 1, S flag to 0 and O flag to 1, then encapsulates the message in an OAL header with source set to its own ADM-ULA and destination set to the ROS ULA (i.e., the ADM-ULA of the ROS Proxy/Server) and sends the message into the secured spanning tree.

As discussed in Section 7.2.6 of [RFC4861], the transmission and reception of uNA messages is unreliable but provides a useful optimization. In well-connected Internetworks with robust data links uNA messages will be delivered with high probability, but in any case the Proxy/Server can optionally send up to MAX_NEIGHBOR_ADVERTISEMENT uNAs to each ROS to increase the likelihood that at least one will be received. Alternatively, the Proxy/Server can set the PNG flag in the uNA OMNI option header to request a solicited NA acknowledgement as specified in [I-D.templin-6man-omni].

When the ROS Proxy/Server receives a uNA message prepared as above, it ignores the message if the destination is not its own ADM-ULA or the MNP-ULA of the Client ROS. In the former case, it uses the included OMNI option information to update its NCE for the target, but does not reset ReachableTime since the receipt of an unsolicited NA message from the ROR does not provide confirmation that any forward paths to the target Client are working. If the destination was the MNP-ULA of the ROS Client, the ROS Proxy/Server instead re-encapsulates with the OAL source set to its own ADM-ULA, OAL destination set to the MNP-ULA of the ROS Client with an authentication signature if necessary, and with an in-window Identification for this Client. Finally, if the uNA message PNG flag was set, the ROS returns a solicited NA acknowledgement as specified in [I-D.templin-6man-omni].

In addition to sending uNA messages to the current set of ROSs for the target Client, the ROR also sends uNAs to the MNP-ULA associated with the link-layer address for any underlying interface for which the link-layer address has changed. These uNA messages update an old Proxy/Server that cannot easily detect (e.g., without active probing) when a formerly-active Client has departed. When the ROR sends the uNA, it sets the IPv6 source address to its LLA, sets the destination address to the old Proxy/Server's ADM-LLA, and sets the Target Address to the Client's MNP-LLA. The ROR also includes an OMNI option with Preflen set to the prefix length associated with the Client's MNP-LLA, with Interface Attributes for the changed underlying interface, and with the OMNI header S/T-omIndex set to 0.

The ROR then sets the uNA R flag to 1, S flag to 0 and O flag to 1, then encapsulates the message in an OAL header with source set to its own ULA and destination set to the ADM-ULA of the old Proxy/Server and sends the message into the secured spanning tree.

3.16.2. Announcing Link-Layer Address and/or QoS Preference Changes

When a Client needs to change its underlying Interface Attributes (e.g., due to a mobility event), the Client requests one of its Proxy/Servers to send RS messages to all of its other Proxy/Servers via the secured spanning tree with an OMNI option that includes Interface attributes with the new link quality and address information.

Up to MAX_RTR_SOLICITATIONS RS messages MAY be sent in parallel with sending carrier packets containing user data in case one or more RAs are lost. If all RAs are lost, the Client SHOULD re-associate with a new Proxy/Server.

When the Proxy/Server receives the Client's changes, it sends uNA messages to all nodes in the Report List the same as described in the previous section.

3.16.3. Bringing New Links Into Service

When a Client needs to bring new underlying interfaces into service (e.g., when it activates a new data link), it sends an RS message to the Proxy/Server via the underlying interface with an OMNI option that includes Interface Attributes with appropriate link quality values and with link-layer address information for the new link.

3.16.4. Deactivating Existing Links

When a Client needs to deactivate an existing underlying interface, it sends an RS or uNA message to its Proxy/Server with an OMNI option with appropriate Interface Attribute values - in particular, the link quality value 0 assures that neighbors will cease to use the link.

If the Client needs to send RS/uNA messages over an underlying interface other than the one being deactivated, it MUST include Interface Attributes with appropriate link quality values for any underlying interfaces being deactivated.

Note that when a Client deactivates an underlying interface, neighbors that have received the RS/uNA messages need not purge all references for the underlying interface from their neighbor cache entries. The Client may reactivate or reuse the underlying interface

and/or its omIndex at a later point in time, when it will send RS/uNA messages with fresh Interface Attributes to update any neighbors.

3.16.5. Moving Between Proxy/Servers

The Client performs the procedures specified in Section 3.12.2 when it first associates with a new Proxy/Server or renews its association with an existing Proxy/Server. The Client also includes MS-Release identifiers in the RS message OMNI option per [I-D.templin-6man-omni] if it wants the new Proxy/Server to notify any old Proxy/Servers from which the Client is departing.

When the new Proxy/Server receives the Client's RS message, it returns an RA as specified in Section 3.12.3 and sends uNA messages to any old Proxy/Servers listed in OMNI option MS-Release identifiers. When the new Proxy/Server sends a uNA message, it sets the IPv6 source address to the Client's MNP-LLA, sets the destination address to the old Proxy/Server's ADM-LLA, and sets the Target Address to 0. The new Proxy/Server also includes an OMNI option with Preflen set to the prefix length associated with the Client's MNP-LLA, with Interface Attributes for its own underlying interface, and with the OMNI header S/T-omIndex set to 0. The new Proxy/Server then sets the uNA R flag to 1, S flag to 0 and O flag to 1, then encapsulates the message in an OAL header with source set to its own ADM-ULA and destination set to the ADM-ULA of the old Proxy/Server and sends the message into the secured spanning tree.

When an old Proxy/Server receives the uNA, it notices that the message appears to have originated from the Client's MNP-LLA but that the Target Address is 0. The old Proxy/Server then changes the Client's NCE state to DEPARTED, sets the link-layer address of the Client to the new Proxy/Server's ADM-ULA, and resets DepartTime. After a short delay (e.g., 2 seconds) the old Proxy/Server withdraws the Client's MNP from the routing system. After DepartTime expires, the old Proxy/Server deletes the Client's NCE.

The old Proxy/Server also iteratively forwards a copy of the uNA message to each ROS in the Client's Report List by changing the OAL destination address to the ULA of the ROS while leaving all other fields of the message unmodified. When the ROS receives the uNA, it examines the source address to determine the target Client NCE and verifies that the destination address matches the old Proxy/Server. The ROS then caches the ULA source address as the new Proxy/Server for the existing NCE and marks the entry as STALE. While in the STALE state, the ROS allows new carrier packets to flow according to any alternate reachable underlying interfaces and sends new NS(WIN) messages using its own ULA as the OAL source and the ADM-ULA of the

new Proxy/Server as the OAL destination address to elicit NA(WIN) messages that reset the NCE state to REACHABLE.

Clients SHOULD NOT move rapidly between Proxy/Servers in order to avoid causing excessive oscillations in the AERO routing system. Examples of when a Client might wish to change to a different Proxy/Server include a Proxy/Server that has gone unreachable, topological movements of significant distance, movement to a new geographic region, movement to a new OMNI link segment, etc.

When a Client moves to a new Proxy/Server, some of the carrier packets of a multiple fragment OAL packet may have already arrived at the old Proxy/Server while others are en route to the new Proxy/Server, however no special attention in the reassembly algorithm is necessary since all carrier packets will eventually arrive at the Client which can then reassemble. However, any carrier packets that are somehow lost can often be recovered through retransmissions.

3.17. Multicast

The AERO Client provides an IGMP (IPv4) [RFC2236] or MLD (IPv6) [RFC3810] proxy service for its EUNs and/or hosted applications [RFC4605]. The Client forwards IGMP/MLD messages over any of its underlying interfaces for which group membership is required. The IGMP/MLD messages may be further forwarded by a first-hop ANET access router acting as an IGMP/MLD-snooping switch [RFC4541], then ultimately delivered to an AERO Proxy/Server acting as a Protocol Independent Multicast - Sparse-Mode (PIM-SM, or simply "PIM") Designated Router (DR) [RFC7761]. AERO Relays also act as PIM routers (i.e., the same as AERO Proxys/Servers) on behalf of nodes on INET/EUN networks. The behaviors identified in the following sections correspond to Source-Specific Multicast (SSM) and Any-Source Multicast (ASM) operational modes.

3.17.1. Source-Specific Multicast (SSM)

When an ROS "X" acting as PIM router receives a Join/Prune message from a node on its downstream interfaces containing one or more ((S)ource, (G)roup) pairs, it updates its Multicast Routing Information Base (MRIB) accordingly. For each S belonging to a prefix reachable via X's non-OMNI interfaces, X then forwards the (S, G) Join/Prune to any PIM routers on those interfaces per [RFC7761].

For each S belonging to a prefix reachable via X's OMNI interface, X includes a PIM Join/Prune for each (S,G) in the OMNI option of an NS(AR) message (see: Section 3.14) using its own LLA as the source address and ALL-PIM-ROUTERS as the destination address. X then encapsulates the NS(AR) in an OAL header with source address set to

the ULA of X and destination address set to S then forwards the message into the secured spanning tree, which delivers it to ROR "Y" that services S. The resulting NA(AR) will return the LLA for the prefix that matches S as the network-layer source address and with an OMNI option with the ULA corresponding to any underlying interfaces that are currently servicing S.

When Y processes the NS(AR) it examines the PIM Join/Prune message. If S is located behind any INET, Direct, or VPNed interfaces Y acts as a PIM router and updates its MRIB to list X as the next hop in the reverse path. If S is located behind any Proxys "Z"*, Y then sends an NS(NUD) message containing the PIM message to each Z* with addressing and encapsulation details the same as specified in Section 3.15. Each Z* then updates its MRIB accordingly and maintains the LLA of X as the next hop in the reverse path. Since the Bridges do not examine network layer control messages, this means that the (reverse) multicast tree path is simply from each Z* (and/or Y) to X with no other multicast-aware routers in the path.

Following the initial combined Join/Prune and NS/NA messaging, X maintains a NCE for each S the same as if X was sending unicast data traffic to S. In particular, X performs additional NS/NA exchanges to keep the NCE alive for up to t_periodic seconds [RFC7761]. If no new Joins are received within t_periodic seconds, X allows the NCE to expire. Finally, if X receives any additional Join/Prune messages for (S,G) it forwards the messages in NS/NA exchanges with each Y and Z* in the NCE over the secured spanning tree.

At some later time, Client C that holds an MNP for source S may depart from a first Proxy/Server Z1 and/or connect via a new Proxy/ Server Z2. In that case, Y sends a uNA message to X the same as specified for unicast mobility in Section 3.16. When X receives the uNA message, it updates its NCE for the LLA for source S and sends new Join messages to any new Proxys Z2. There is no requirement to send any Prune messages to old Proxy/Server Z1 since source S will no longer source any multicast data traffic via Z1. Instead, the multicast state for (S,G) in Proxy/Server Z1 will soon time out since no new Joins will arrive.

After some later time, C may move to a new Proxy/Server Y2 and depart from old Sever Y1. In that case, Y1 sends Join messages for any of C's active (S,G) groups to Y2 while including its own LLA as the source address. This causes Y2 to include Y1 in the multicast forwarding tree during the interim time that Y1's NCE for C is in the DEPARTED state. At the same time, Y1 sends a uNA message to X with an OMNI option with S/T-omIndex in the header set to 0 and a release indication to cause X to release its NCE for S. X then sends a new Join message to S via the secured spanning tree and re-initiates

route optimization the same as if it were receiving a fresh Join message from a node on a downstream link.

3.17.2. Any-Source Multicast (ASM)

When an ROS X acting as a PIM router receives a Join/Prune from a node on its downstream interfaces containing one or more (*,G) pairs, it updates its Multicast Routing Information Base (MRIB) accordingly. X then forwards a copy of the message within the OMNI option of an NS(AR) message to the Rendezvous Point (RP) R for each G over the secured spanning tree. X uses its own LLA as the source address and ALL-PIM-ROUTERS as the destination address, then encapsulates the NS(AR) message in an OAL header with source address set to the ULA of X and destination address set to R, then sends the message into the secured spanning tree.

For each source S that sends multicast traffic to group G via R, the Proxy/Server Z* for the Client that aggregates S encapsulates the original IP packets in PIM Register messages and forwards them to R via the secured spanning tree, which may then elect to send a PIM Join to Z*. This will result in an (S,G) tree rooted at Z* with R as the next hop so that R will begin to receive two copies of the original IP packet; one native copy from the (S, G) tree and a second copy from the pre-existing (*, G) tree that still uses PIM Register encapsulation. R can then issue a PIM Register-stop message to suppress the Register-encapsulated stream. At some later time, if C moves to a new Proxy/Server Z*, it resumes sending original IP packets via PIM Register encapsulation via the new Z*.

At the same time, as multicast listeners discover individual S's for a given G, they can initiate an (S,G) Join for each S under the same procedures discussed in Section 3.17.1. Once the (S,G) tree is established, the listeners can send (S, G) Prune messages to R so that multicast original IP packets for group G sourced by S will only be delivered via the (S, G) tree and not from the (*, G) tree rooted at R. All mobility considerations discussed for SSM apply.

3.17.3. Bi-Directional PIM (BIDIR-PIM)

Bi-Directional PIM (BIDIR-PIM) [RFC5015] provides an alternate approach to ASM that treats the Rendezvous Point (RP) as a Designated Forwarder (DF). Further considerations for BIDIR-PIM are out of scope.

3.18. Operation over Multiple OMNI Links

An AERO Client can connect to multiple OMNI links the same as for any data link service. In that case, the Client maintains a distinct OMNI interface for each link, e.g., 'omni0' for the first link, 'omni1' for the second, 'omni2' for the third, etc. Each OMNI link would include its own distinct set of Bridges and Proxy/Servers, thereby providing redundancy in case of failures.

Each OMNI link could utilize the same or different ANET connections. The links can be distinguished at the link-layer via the SRT prefix in a similar fashion as for Virtual Local Area Network (VLAN) tagging (e.g., IEEE 802.1Q) and/or through assignment of distinct sets of MSPs on each link. This gives rise to the opportunity for supporting multiple redundant networked paths, with each VLAN distinguished by a different SRT "color" (see: Section 3.2.5).

The Client's IP layer can select the outgoing OMNI interface appropriate for a given traffic profile while (in the reverse direction) correspondent nodes must have some way of steering their original IP packets destined to a target via the correct OMNI link.

In a first alternative, if each OMNI link services different MSPs, then the Client can receive a distinct MNP from each of the links. IP routing will therefore assure that the correct OMNI link is used for both outbound and inbound traffic. This can be accomplished using existing technologies and approaches, and without requiring any special supporting code in correspondent nodes or Bridges.

In a second alternative, if each OMNI link services the same MSP(s) then each link could assign a distinct "OMNI link Anycast" address that is configured by all Bridges on the link. Correspondent nodes can then perform Segment Routing to select the correct SRT, which will then direct the original IP packet over multiple hops to the target.

3.19. DNS Considerations

AERO Client MNs and INET correspondent nodes consult the Domain Name System (DNS) the same as for any Internetworking node. When correspondent nodes and Client MNs use different IP protocol versions (e.g., IPv4 correspondents and IPv6 MNs), the INET DNS must maintain A records for IPv4 address mappings to MNs which must then be populated in Relay NAT64 mapping caches. In that way, an IPv4 correspondent node can send original IPv4 packets to the IPv4 address mapping of the target MN, and the Relay will translate the IPv4 header and destination address into an IPv6 header and IPv6 destination address of the MN.

When an AERO Client registers with an AERO Proxy/Server, the Proxy/Server can return the address(es) of DNS servers in RDNSS options [RFC6106]. The DNS server provides the IP addresses of other MNs and correspondent nodes in AAAA records for IPv6 or A records for IPv4.

3.20. Transition/Coexistence Considerations

OAL encapsulation ensures that dissimilar INET partitions can be joined into a single unified OMNI link, even though the partitions themselves may have differing protocol versions and/or incompatible addressing plans. However, a commonality can be achieved by incrementally distributing globally routable (i.e., native) IP prefixes to eventually reach all nodes (both mobile and fixed) in all OMNI link segments. This can be accomplished by incrementally deploying AERO Bridges on each INET partition, with each Bridge distributing its MNPs and/or discovering non-MNP IP GUA prefixes on its INET links.

This gives rise to the opportunity to eventually distribute native IP addresses to all nodes, and to present a unified OMNI link view even if the INET partitions remain in their current protocol and addressing plans. In that way, the OMNI link can serve the dual purpose of providing a mobility/multilink service and a transition/coexistence service. Or, if an INET partition is transitioned to a native IP protocol version and addressing scheme that is compatible with the OMNI link MNP-based addressing scheme, the partition and OMNI link can be joined by Bridges.

Relays that connect INETs/EUNs with dissimilar IP protocol versions may need to employ a network address and protocol translation function such as NAT64 [RFC6146].

3.21. Detecting and Reacting to Proxy/Server and Bridge Failures

In environments where rapid failure recovery is required, Proxy/Servers and Bridges SHOULD use Bidirectional Forwarding Detection (BFD) [RFC5880]. Nodes that use BFD can quickly detect and react to failures so that cached information is re-established through alternate nodes. BFD control messaging is carried only over well-connected ground domain networks (i.e., and not low-end radio links) and can therefore be tuned for rapid response.

Proxy/Servers and Bridges maintain BFD sessions in parallel with their BGP peerings. If a Proxy/Server or Bridge fails, BGP peers will quickly re-establish routes through alternate paths the same as for common BGP deployments. Similarly, Proxys maintain BFD sessions with their associated Bridges even though they do not establish BGP peerings with them.

3.22. AERO Clients on the Open Internet

AERO Clients that connect to the open Internet via INET interfaces can establish a VPN or direct link to securely connect to a Proxy/Server in a "tethered" arrangement with all of the Client's traffic transiting the Proxy/Server which acts as a router. Alternatively, the Client can associate with an INET Proxy/Server using UDP/IP encapsulation and control message securing services as discussed in the following sections.

When a Client's OMNI interface enables an INET underlying interface, it first determines whether the interface is likely to be behind a NAT. For IPv4, the Client assumes it is on the open Internet if the INET address is not a special-use IPv4 address per [RFC3330]. Similarly for IPv6, the Client assumes it is on the open Internet if the INET address is a Global Unicast Address (GUA) [RFC4291]. Otherwise, the Client assumes it may be behind one or several NATs.

The Client then prepares an RS message with IPv6 source address set to its MNP-LLA, with IPv6 destination set to (link-local) All-Routers multicast and with an OMNI option with underlying interface attributes. If the Client believes that it is on the open Internet, it SHOULD include an L2ADDR in the Interface Attributes sub-option corresponding to the underlying interface; otherwise, it MAY set L2ADDR to 0. If the underlying address is IPv4, the Client includes the Port Number and IPv4 address written in obfuscated form [RFC4380] as discussed in Section 3.3. If the underlying interface address is IPv6, the Client instead includes the Port Number and IPv6 address in obfuscated form. The Client finally includes an authentication signature sub-option in the OMNI option [I-D.templin-6man-omni] to provide message authentication, selects an Identification value and window synchronization parameters, and submits the RS for OAL encapsulation. The Client then encapsulates the OAL fragment in UDP/IP headers to form a carrier packet, sets the UDP/IP source to its INET address and UDP port, sets the UDP/IP destination to the Proxy/Server's INET address and the AERO service port number (8060), then sends the carrier packet to the Proxy/Server.

When the Proxy/Server receives the RS, it discards the OAL encapsulation, authenticates the RS message, creates a NCE and registers the Client's MNP, window synchronization state and INET interface information according to the OMNI option parameters. If the RS message OMNI option includes Interface Attributes with an L2ADDR, the Proxy/Server compares the encapsulation IP address and UDP port number with the (unobfuscated) values. If the values are the same, the Proxy/Server caches the Client's information as "INET" addresses meaning that the Client is likely to accept direct messages without requiring NAT traversal exchanges. If the values are

different (or, if the OMNI option did not include an L2ADDR) the Proxy/Server instead caches the Client's information as "mapped" addresses meaning that NAT traversal exchanges may be necessary.

The Proxy/Server then prepares an RA message with IPv6 source and destination set corresponding to the addresses in the RS, and with an OMNI option with an Origin Indication sub-option per [I-D.templin-6man-omni] with the mapped and obfuscated Port Number and IP address observed in the encapsulation headers. The Proxy/Server also includes an authentication signature sub-option per [I-D.templin-6man-omni] and a symmetric window synchronization/acknowledgement. The Proxy/Server then performs OAL encapsulation and fragmentation if necessary and encapsulates each fragment in UDP/IP headers with addresses set per the L2ADDR information in the NCE for the Client.

When the Client receives the RA, it authenticates the message then process the window synchronization/acknowledgement and compares the mapped Port Number and IP address from the Origin Indication sub-option with its own address. If the addresses are the same, the Client assumes the open Internet / Cone NAT principle; if the addresses are different, the Client instead assumes that further qualification procedures are necessary to detect the type of NAT and proceeds according to standard procedures [RFC6081][RFC4380]. The Client finally arranges to return an explicit/implicit acknowledgement, and sends periodic RS messages to receive fresh RA messages before the Router Lifetime received on each INET interface expires.

When the Client sends messages to target IP addresses, it also invokes route optimization per Section 3.14. For route optimized targets in the same OMNI link segment, if the target's L2ADDR is on the open INET, the Client forwards carrier packets directly to the target INET address. If the target is behind a NAT, the Client first establishes NAT state for the L2ADDR using the "direct bubble" and NUD mechanisms discussed in Section 3.10.1. The Client continues to send carrier packets via its Proxy/Server until NAT state is populated, then begins forwarding carrier packets via the direct path through the NAT to the target. For targets in different OMNI link segments, the Client uses OAL/ORH encapsulation and forwards carrier packets to the Bridge that returned the NA(AR) message.

The Client can send original IP packets to route-optimized neighbors in the same OMNI link segment no larger than the minimum/path MPS in one piece and with OAL encapsulation as atomic fragments. For larger original IP packets, the Client applies OAL encapsulation and fragmentation if necessary according to Section 3.9, with OAL header with source set to its own MNP-ULA and destination set to the MNP-ULA

of the target, and with an in-window Identification value. The Client then encapsulates each resulting carrier packet in UDP/IP *NET headers and sends them to the next hop.

Note: The NAT traversal procedures specified in this document are applicable for Cone, Address-Restricted and Port-Restricted NATs only. While future updates to this document may specify procedures for other NAT variations (e.g., hairpinning and various forms of Symmetric NATs), it should be noted that continuous communications are always possible through forwarding via a Proxy/Server even if NAT traversal is not employed.

3.23. Time-Varying MNPs

In some use cases, it is desirable, beneficial and efficient for the Client to receive a constant MNP that travels with the Client wherever it moves. For example, this would allow air traffic controllers to easily track aircraft, etc. In other cases, however (e.g., intelligent transportation systems), the MN may be willing to sacrifice a modicum of efficiency in order to have time-varying MNPs that can be changed every so often to defeat adversarial tracking.

The DHCPv6 service offers a way for Clients that desire time-varying MNPs to obtain short-lived prefixes (e.g., on the order of a small number of minutes). In that case, the identity of the Client would not be bound to the MNP but rather to a Node Identification value (see: [I-D.templin-6man-omni]) to be used as the Client ID seed for MNP prefix delegation. The Client would then be obligated to renumber its internal networks whenever its MNP (and therefore also its MNP-LLA) changes. This should not present a challenge for Clients with automated network renumbering services, however presents limits for the durations of ongoing sessions that would prefer to use a constant address.

4. Implementation Status

An early AERO implementation based on OpenVPN (https://openvpn.net/) was announced on the v6ops mailing list on January 10, 2018 and an initial public release of the AERO proof-of-concept source code was announced on the intarea mailing list on August 21, 2015.

AERO Release-3.2 was tagged on March 30, 2021, and is undergoing internal testing. Additional internal releases expected within the coming months, with first public release expected end of 1H2021.

5. IANA Considerations

The IANA is instructed to assign a new type value TBD1 in the IPv6 Routing Types registry (IANA registration procedure is IETF Review or IESG Approval).

The IANA has assigned the UDP port number "8060" for an earlier experimental first version of AERO [RFC6706]. This document obsoletes [RFC6706], and together with [I-D.templin-6man-omni] reclaims the UDP port number "8060" for 'aero' as the service port for UDP/IP encapsulation. (Note that, although [RFC6706] was not widely implemented or deployed, any messages coded to that specification can be easily distinguished and ignored since they use the invalid ICMPv6 message type number '0'.) This document makes no request of IANA, since [I-D.templin-6man-omni] already provides instructions.

No further IANA actions are required.

6. Security Considerations

AERO Bridges configure secured tunnels with AERO Proxy/Servers and Relays within their local OMNI link segments. Applicable secured tunnel alternatives include IPsec [RFC4301], TLS/SSL [RFC8446], DTLS [RFC6347], WireGuard [WG], etc. The AERO Bridges of all OMNI link segments in turn configure secured tunnels for their neighboring AERO Bridges in a secured spanning tree topology. Therefore, control messages exchanged between any pair of OMNI link neighbors over the secured spanning tree are already protected.

AERO nodes acting as Route Optimization Responders (RORs) may also receive packets directly from arbitrary nodes in INET partitions instead of via the secured spanning tree. For INET partitions that apply effective ingress filtering to defeat source address spoofing, the simple data origin authentication procedures in Section 3.8 can be applied.

For INET partitions that require strong security in the data plane, two options for securing communications include 1) disable route optimization so that all traffic is conveyed over secured tunnels, or 2) enable on-demand secure tunnel creation between Client neighbors. Option 1) would result in longer routes than necessary and impose traffic concentration on critical infrastructure elements. Option 2) could be coordinated between Clients using NS/NA messages with OMNI Host Identity Protocol (HIP) "Initiator/Responder" message sub-options [RFC7401][I-D.templin-6man-omni] to create a secured tunnel on-demand.

AERO Clients that connect to secured ANETs need not apply security to their ND messages, since the messages will be intercepted by a perimeter Proxy/Server that applies security on its INET-facing interface as part of the spanning tree (see above). AERO Clients connected to the open INET can use network and/or transport layer security services such as VPNs or can by some other means establish a direct link to a Proxy/Server. When a VPN or direct link may be impractical, however, INET Clients and Proxy/Servers SHOULD include and verify authentication signatures for their IPv6 ND messages as specified in [I-D.templin-6man-omni].

Application endpoints SHOULD use transport-layer (or higher-layer) security services such as TLS/SSL, DTLS or SSH [RFC4251] to assure the same level of protection as for critical secured Internet services. AERO Clients that require host-based VPN services SHOULD use network and/or transport layer security services such as IPsec, TLS/SSL, DTLS, etc. AERO Proxys and Proxy/Servers can also provide a network-based VPN service on behalf of the Client, e.g., if the Client is located within a secured enclave and cannot establish a VPN on its own behalf.

AERO Proxy/Servers and Bridges present targets for traffic amplification Denial of Service (DoS) attacks. This concern is no different than for widely-deployed VPN security gateways in the Internet, where attackers could send spoofed packets to the gateways at high data rates. This can be mitigated through the AERO/OMNI data origin authentication procedures, as well as connecting Proxy/Servers and Bridges over dedicated links with no connections to the Internet and/or when connections to the Internet are only permitted through well-managed firewalls. Traffic amplification DoS attacks can also target an AERO Client's low data rate links. This is a concern not only for Clients located on the open Internet but also for Clients in secured enclaves. AERO Proxy/Servers and Proxys can institute rate limits that protect Clients from receiving packet floods that could DoS low data rate links.

AERO Relays must implement ingress filtering to avoid a spoofing attack in which spurious messages with ULA addresses are injected into an OMNI link from an outside attacker. AERO Clients MUST ensure that their connectivity is not used by unauthorized nodes on their 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 other nodes via some form of Internet connection sharing such as tethering.)

The MAP list MUST be well-managed and secured from unauthorized tampering, even though the list contains only public information. The MAP list can be conveyed to the Client in a similar fashion as in [RFC5214] (e.g., through layer 2 data link login messaging, secure upload of a static file, DNS lookups, etc.).

The AERO service for open INET Clients depends on a public key distribution service in which Client public keys and identities are maintained in a shared database accessible to all open INET Proxy/Servers. Similarly, each Client must be able to determine the public key of each Proxy/Server, e.g. by consulting an online database. When AERO nodes register their public keys indexed by a unique Host Identity Tag (HIT) [RFC7401] in a distributed database such as the DNS, and use the HIT as an identity for applying IPv6 ND message authentication signatures, a means for determining public key attestation is available.

Security considerations for IPv6 fragmentation and reassembly are discussed in [I-D.templin-6man-omni]. In environments where spoofing is considered a threat, OMNI nodes SHOULD employ Identification window synchronization and OAL destinations SHOULD configure an (end-system-based) firewall.

SRH authentication facilities are specified in [RFC8754]. Security considerations for accepting link-layer ICMP messages and reflected packets are discussed throughout the document.

7. Acknowledgements

Discussions in the IETF, aviation standards communities and private exchanges helped shape some of the concepts in this work. Individuals who contributed insights include Mikael Abrahamsson, Mark Andrews, Fred Baker, Bob Braden, Stewart Bryant, Brian Carpenter, Wojciech Dec, Pavel Drasil, Ralph Droms, Adrian Farrel, Nick Green, Sri Gundavelli, Brian Haberman, Bernhard Haindl, Joel Halpern, Tom Herbert, Sascha Hlusiak, Lee Howard, Zdenek Jaron, Andre Kostur, Hubert Kuenig, Ted Lemon, Andy Malis, Satoru Matsushima, Tomek Mrugalski, Madhu Niraula, Alexandru Petrescu, Behcet Saikaya, Michal Skorepa, Joe Touch, Bernie Volz, Ryuji Wakikawa, Tony Whyman, Lloyd Wood and James Woodyatt. 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 during the publication of the AERO first edition.

This work has further been encouraged and supported by Boeing colleagues including Kyle Bae, M. Wayne Benson, Dave Bernhardt, Cam Brodie, John Bush, Balaguruna Chidambaram, Irene Chin, Bruce Cornish,

Claudiu Danilov, Don Dillenburg, Joe Dudkowski, Wen Fang, Samad Farooqui, Anthony Gregory, Jeff Holland, Seth Jahne, Brian Jaury, Greg Kimberly, Ed King, Madhuri Madhava Badgandi, Laurel Matthew, Gene MacLean III, Kyle Mikos, Rob Muszkiewicz, Sean O'Sullivan, Vijay Rajagopalan, Greg Saccone, Rod Santiago, Kent Shuey, Brian Skeen, Mike Slane, Carrie Spiker, Katie Tran, Brendan Williams, Amelia Wilson, Julie Wulff, Yueli Yang, Eric Yeh and other members of the Boeing mobility, networking and autonomy teams. Kyle Bae, Wayne Benson, Madhuri Madhava Badgandi, Vijayasarathy Rajagopalan, Katie Tran and Eric Yeh are especially acknowledged for implementing the AERO functions as extensions to the public domain OpenVPN distribution. Chuck Klabunde is honored and remembered for his early leadership, and we mourn his untimely loss.

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

Many of the constructs presented in this second edition of AERO are based on the author's earlier works, including:

- The Internet Routing Overlay Network (IRON) [RFC6179][I-D.templin-ironbis]

- Virtual Enterprise Traversal (VET) [RFC5558][I-D.templin-intarea-vet]

- The Subnetwork Encapsulation and Adaptation Layer (SEAL) [RFC5320][I-D.templin-intarea-seal]

- AERO, First Edition [RFC6706]

Note that these works cite numerous earlier efforts that are not also cited here due to space limitations. The authors of those earlier works are acknowledged for their insights.

This work is aligned with the NASA Safe Autonomous Systems Operation (SASO) program under NASA contract number NNA16BD84C.

This work is aligned with the FAA as per the SE2025 contract number DTFAWA-15-D-00030.

This work is aligned with the Boeing Commercial Airplanes (BCA) Internet of Things (IoT) and autonomy programs.

This work is aligned with the Boeing Information Technology (BIT) MobileNet program.

8. References

8.1. Normative References

[I-D.templin-6man-omni]
Templin, F. L. and T. Whyman, "Transmission of IP Packets over Overlay Multilink Network (OMNI) Interfaces", draft-templin-6man-omni-03 (work in progress), April 2021.

[RFC0791] Postel, J., "Internet Protocol", STD 5, RFC 791, DOI 10.17487/RFC0791, September 1981, <https://www.rfc-editor.org/info/rfc791>.

[RFC0792] Postel, J., "Internet Control Message Protocol", STD 5, RFC 792, DOI 10.17487/RFC0792, September 1981, <https://www.rfc-editor.org/info/rfc792>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC2473] Conta, A. and S. Deering, "Generic Packet Tunneling in IPv6 Specification", RFC 2473, DOI 10.17487/RFC2473, December 1998, <https://www.rfc-editor.org/info/rfc2473>.

[RFC3971] Arkko, J., Ed., Kempf, J., Zill, B., and P. Nikander, "SEcure Neighbor Discovery (SEND)", RFC 3971, DOI 10.17487/RFC3971, March 2005, <https://www.rfc-editor.org/info/rfc3971>.

[RFC3972] Aura, T., "Cryptographically Generated Addresses (CGA)", RFC 3972, DOI 10.17487/RFC3972, March 2005, <https://www.rfc-editor.org/info/rfc3972>.

[RFC4191] Draves, R. and D. Thaler, "Default Router Preferences and More-Specific Routes", RFC 4191, DOI 10.17487/RFC4191, November 2005, <https://www.rfc-editor.org/info/rfc4191>.

[RFC4193] Hinden, R. and B. Haberman, "Unique Local IPv6 Unicast Addresses", RFC 4193, DOI 10.17487/RFC4193, October 2005, <https://www.rfc-editor.org/info/rfc4193>.

[RFC4380] Huitema, C., "Teredo: Tunneling IPv6 over UDP through Network Address Translations (NATs)", RFC 4380, DOI 10.17487/RFC4380, February 2006, <https://www.rfc-editor.org/info/rfc4380>.

[RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, DOI 10.17487/RFC4861, September 2007, <https://www.rfc-editor.org/info/rfc4861>.

[RFC4862] Thomson, S., Narten, T., and T. Jinmei, "IPv6 Stateless Address Autoconfiguration", RFC 4862, DOI 10.17487/RFC4862, September 2007, <https://www.rfc-editor.org/info/rfc4862>.

[RFC6081] Thaler, D., "Teredo Extensions", RFC 6081, DOI 10.17487/RFC6081, January 2011, <https://www.rfc-editor.org/info/rfc6081>.

[RFC7401] Moskowitz, R., Ed., Heer, T., Jokela, P., and T. Henderson, "Host Identity Protocol Version 2 (HIPv2)", RFC 7401, DOI 10.17487/RFC7401, April 2015, <https://www.rfc-editor.org/info/rfc7401>.

[RFC7739] Gont, F., "Security Implications of Predictable Fragment Identification Values", RFC 7739, DOI 10.17487/RFC7739, February 2016, <https://www.rfc-editor.org/info/rfc7739>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8200] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", STD 86, RFC 8200, DOI 10.17487/RFC8200, July 2017, <https://www.rfc-editor.org/info/rfc8200>.

[RFC8415] Mrugalski, T., Siodelski, M., Volz, B., Yourtchenko, A., Richardson, M., Jiang, S., Lemon, T., and T. Winters, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", RFC 8415, DOI 10.17487/RFC8415, November 2018, <https://www.rfc-editor.org/info/rfc8415>.

8.2. Informative References

[BGP] Huston, G., "BGP in 2015, http://potaroo.net", January 2016.

[I-D.bonica-6man-comp-rtg-hdr] Bonica, R., Kamite, Y., Alston, A., Henriques, D., and L. Jalil, "The IPv6 Compact Routing Header (CRH)", draft-bonica-6man-comp-rtg-hdr-24 (work in progress), January 2021.

[I-D.bonica-6man-crh-helper-opt]
Li, X., Bao, C., Ruan, E., and R. Bonica, "Compressed Routing Header (CRH) Helper Option", draft-bonica-6man-crh-helper-opt-03 (work in progress), April 2021.

[I-D.ietf-intarea-frag-fragile]
Bonica, R., Baker, F., Huston, G., Hinden, R. M., Troan, O., and F. Gont, "IP Fragmentation Considered Fragile", draft-ietf-intarea-frag-fragile-17 (work in progress), September 2019.

[I-D.ietf-intarea-tunnels]
Touch, J. and M. Townsley, "IP Tunnels in the Internet Architecture", draft-ietf-intarea-tunnels-10 (work in progress), September 2019.

[I-D.ietf-ipwave-vehicular-networking]
(editor), J. (. J., "IPv6 Wireless Access in Vehicular Environments (IPWAVE): Problem Statement and Use Cases", draft-ietf-ipwave-vehicular-networking-20 (work in progress), March 2021.

[I-D.ietf-rtgwg-atn-bgp]
Templin, F. L., Saccone, G., Dawra, G., Lindem, A., and V. Moreno, "A Simple BGP-based Mobile Routing System for the Aeronautical Telecommunications Network", draft-ietf-rtgwg-atn-bgp-10 (work in progress), January 2021.

[I-D.templin-6man-dhcpv6-ndopt]
Templin, F. L., "A Unified Stateful/Stateless Configuration Service for IPv6", draft-templin-6man-dhcpv6-ndopt-11 (work in progress), January 2021.

[I-D.templin-intarea-seal]
Templin, F. L., "The Subnetwork Encapsulation and Adaptation Layer (SEAL)", draft-templin-intarea-seal-68 (work in progress), January 2014.

[I-D.templin-intarea-vet]
Templin, F. L., "Virtual Enterprise Traversal (VET)", draft-templin-intarea-vet-40 (work in progress), May 2013.

[I-D.templin-ipwave-uam-its]
Templin, F. L., "Urban Air Mobility Implications for Intelligent Transportation Systems", draft-templin-ipwave-uam-its-04 (work in progress), January 2021.

[I-D.templin-ironbis]
Templin, F. L., "The Interior Routing Overlay Network (IRON)", draft-templin-ironbis-16 (work in progress), March 2014.

[I-D.templin-v6ops-pdhost]
Templin, F. L., "IPv6 Prefix Delegation and Multi-Addressing Models", draft-templin-v6ops-pdhost-27 (work in progress), January 2021.

[OVPN] OpenVPN, O., "http://openvpn.net", October 2016.

[RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, DOI 10.17487/RFC1035, November 1987, <https://www.rfc-editor.org/info/rfc1035>.

[RFC1812] Baker, F., Ed., "Requirements for IP Version 4 Routers", RFC 1812, DOI 10.17487/RFC1812, June 1995, <https://www.rfc-editor.org/info/rfc1812>.

[RFC2003] Perkins, C., "IP Encapsulation within IP", RFC 2003, DOI 10.17487/RFC2003, October 1996, <https://www.rfc-editor.org/info/rfc2003>.

[RFC2004] Perkins, C., "Minimal Encapsulation within IP", RFC 2004, DOI 10.17487/RFC2004, October 1996, <https://www.rfc-editor.org/info/rfc2004>.

[RFC2236] Fenner, W., "Internet Group Management Protocol, Version 2", RFC 2236, DOI 10.17487/RFC2236, November 1997, <https://www.rfc-editor.org/info/rfc2236>.

[RFC2464] Crawford, M., "Transmission of IPv6 Packets over Ethernet Networks", RFC 2464, DOI 10.17487/RFC2464, December 1998, <https://www.rfc-editor.org/info/rfc2464>.

[RFC2529] Carpenter, B. and C. Jung, "Transmission of IPv6 over IPv4 Domains without Explicit Tunnels", RFC 2529, DOI 10.17487/RFC2529, March 1999, <https://www.rfc-editor.org/info/rfc2529>.

[RFC2983] Black, D., "Differentiated Services and Tunnels", RFC 2983, DOI 10.17487/RFC2983, October 2000, <https://www.rfc-editor.org/info/rfc2983>.

[RFC3168] Ramakrishnan, K., Floyd, S., and D. Black, "The Addition of Explicit Congestion Notification (ECN) to IP", RFC 3168, DOI 10.17487/RFC3168, September 2001, <https://www.rfc-editor.org/info/rfc3168>.

[RFC3330] IANA, "Special-Use IPv4 Addresses", RFC 3330, DOI 10.17487/RFC3330, September 2002, <https://www.rfc-editor.org/info/rfc3330>.

[RFC3810] Vida, R., Ed. and L. Costa, Ed., "Multicast Listener Discovery Version 2 (MLDv2) for IPv6", RFC 3810, DOI 10.17487/RFC3810, June 2004, <https://www.rfc-editor.org/info/rfc3810>.

[RFC4122] Leach, P., Mealling, M., and R. Salz, "A Universally Unique IDentifier (UUID) URN Namespace", RFC 4122, DOI 10.17487/RFC4122, July 2005, <https://www.rfc-editor.org/info/rfc4122>.

[RFC4251] Ylonen, T. and C. Lonvick, Ed., "The Secure Shell (SSH) Protocol Architecture", RFC 4251, DOI 10.17487/RFC4251, January 2006, <https://www.rfc-editor.org/info/rfc4251>.

[RFC4271] Rekhter, Y., Ed., Li, T., Ed., and S. Hares, Ed., "A Border Gateway Protocol 4 (BGP-4)", RFC 4271, DOI 10.17487/RFC4271, January 2006, <https://www.rfc-editor.org/info/rfc4271>.

[RFC4291] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, DOI 10.17487/RFC4291, February 2006, <https://www.rfc-editor.org/info/rfc4291>.

[RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, DOI 10.17487/RFC4301, December 2005, <https://www.rfc-editor.org/info/rfc4301>.

[RFC4389] Thaler, D., Talwar, M., and C. Patel, "Neighbor Discovery Proxies (ND Proxy)", RFC 4389, DOI 10.17487/RFC4389, April 2006, <https://www.rfc-editor.org/info/rfc4389>.

[RFC4443] Conta, A., Deering, S., and M. Gupta, Ed., "Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification", STD 89, RFC 4443, DOI 10.17487/RFC4443, March 2006, <https://www.rfc-editor.org/info/rfc4443>.

[RFC4511] Sermersheim, J., Ed., "Lightweight Directory Access Protocol (LDAP): The Protocol", RFC 4511, DOI 10.17487/RFC4511, June 2006, <https://www.rfc-editor.org/info/rfc4511>.

[RFC4541] Christensen, M., Kimball, K., and F. Solensky, "Considerations for Internet Group Management Protocol (IGMP) and Multicast Listener Discovery (MLD) Snooping Switches", RFC 4541, DOI 10.17487/RFC4541, May 2006, <https://www.rfc-editor.org/info/rfc4541>.

[RFC4605] Fenner, B., He, H., Haberman, B., and H. Sandick, "Internet Group Management Protocol (IGMP) / Multicast Listener Discovery (MLD)-Based Multicast Forwarding ("IGMP/MLD Proxying")", RFC 4605, DOI 10.17487/RFC4605, August 2006, <https://www.rfc-editor.org/info/rfc4605>.

[RFC4982] Bagnulo, M. and J. Arkko, "Support for Multiple Hash Algorithms in Cryptographically Generated Addresses (CGAs)", RFC 4982, DOI 10.17487/RFC4982, July 2007, <https://www.rfc-editor.org/info/rfc4982>.

[RFC5015] Handley, M., Kouvelas, I., Speakman, T., and L. Vicisano, "Bidirectional Protocol Independent Multicast (BIDIR-PIM)", RFC 5015, DOI 10.17487/RFC5015, October 2007, <https://www.rfc-editor.org/info/rfc5015>.

[RFC5214] Templin, F., Gleeson, T., and D. Thaler, "Intra-Site Automatic Tunnel Addressing Protocol (ISATAP)", RFC 5214, DOI 10.17487/RFC5214, March 2008, <https://www.rfc-editor.org/info/rfc5214>.

[RFC5320] Templin, F., Ed., "The Subnetwork Encapsulation and Adaptation Layer (SEAL)", RFC 5320, DOI 10.17487/RFC5320, February 2010, <https://www.rfc-editor.org/info/rfc5320>.

[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, DOI 10.17487/RFC5522, October 2009, <https://www.rfc-editor.org/info/rfc5522>.

[RFC5558] Templin, F., Ed., "Virtual Enterprise Traversal (VET)", RFC 5558, DOI 10.17487/RFC5558, February 2010, <https://www.rfc-editor.org/info/rfc5558>.

[RFC5569] Despres, R., "IPv6 Rapid Deployment on IPv4 Infrastructures (6rd)", RFC 5569, DOI 10.17487/RFC5569, January 2010, <https://www.rfc-editor.org/info/rfc5569>.

[RFC5880] Katz, D. and D. Ward, "Bidirectional Forwarding Detection (BFD)", RFC 5880, DOI 10.17487/RFC5880, June 2010, <https://www.rfc-editor.org/info/rfc5880>.

[RFC6106] Jeong, J., Park, S., Beloeil, L., and S. Madanapalli, "IPv6 Router Advertisement Options for DNS Configuration", RFC 6106, DOI 10.17487/RFC6106, November 2010, <https://www.rfc-editor.org/info/rfc6106>.

[RFC6139] Russert, S., Ed., Fleischman, E., Ed., and F. Templin, Ed., "Routing and Addressing in Networks with Global Enterprise Recursion (RANGER) Scenarios", RFC 6139, DOI 10.17487/RFC6139, February 2011, <https://www.rfc-editor.org/info/rfc6139>.

[RFC6146] Bagnulo, M., Matthews, P., and I. van Beijnum, "Stateful NAT64: Network Address and Protocol Translation from IPv6 Clients to IPv4 Servers", RFC 6146, DOI 10.17487/RFC6146, April 2011, <https://www.rfc-editor.org/info/rfc6146>.

[RFC6179] Templin, F., Ed., "The Internet Routing Overlay Network (IRON)", RFC 6179, DOI 10.17487/RFC6179, March 2011, <https://www.rfc-editor.org/info/rfc6179>.

[RFC6221] Miles, D., Ed., Ooghe, S., Dec, W., Krishnan, S., and A. Kavanagh, "Lightweight DHCPv6 Relay Agent", RFC 6221, DOI 10.17487/RFC6221, May 2011, <https://www.rfc-editor.org/info/rfc6221>.

[RFC6273] Kukec, A., Krishnan, S., and S. Jiang, "The Secure Neighbor Discovery (SEND) Hash Threat Analysis", RFC 6273, DOI 10.17487/RFC6273, June 2011, <https://www.rfc-editor.org/info/rfc6273>.

[RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, DOI 10.17487/RFC6347, January 2012, <https://www.rfc-editor.org/info/rfc6347>.

[RFC6355] Narten, T. and J. Johnson, "Definition of the UUID-Based DHCPv6 Unique Identifier (DUID-UUID)", RFC 6355, DOI 10.17487/RFC6355, August 2011, <https://www.rfc-editor.org/info/rfc6355>.

[RFC6438] Carpenter, B. and S. Amante, "Using the IPv6 Flow Label for Equal Cost Multipath Routing and Link Aggregation in Tunnels", RFC 6438, DOI 10.17487/RFC6438, November 2011, <https://www.rfc-editor.org/info/rfc6438>.

[RFC6706] Templin, F., Ed., "Asymmetric Extended Route Optimization (AERO)", RFC 6706, DOI 10.17487/RFC6706, August 2012, <https://www.rfc-editor.org/info/rfc6706>.

[RFC6935] Eubanks, M., Chimento, P., and M. Westerlund, "IPv6 and UDP Checksums for Tunneled Packets", RFC 6935, DOI 10.17487/RFC6935, April 2013, <https://www.rfc-editor.org/info/rfc6935>.

[RFC6936] Fairhurst, G. and M. Westerlund, "Applicability Statement for the Use of IPv6 UDP Datagrams with Zero Checksums", RFC 6936, DOI 10.17487/RFC6936, April 2013, <https://www.rfc-editor.org/info/rfc6936>.

[RFC7333] Chan, H., Ed., Liu, D., Seite, P., Yokota, H., and J. Korhonen, "Requirements for Distributed Mobility Management", RFC 7333, DOI 10.17487/RFC7333, August 2014, <https://www.rfc-editor.org/info/rfc7333>.

[RFC7761] Fenner, B., Handley, M., Holbrook, H., Kouvelas, I., Parekh, R., Zhang, Z., and L. Zheng, "Protocol Independent Multicast - Sparse Mode (PIM-SM): Protocol Specification (Revised)", STD 83, RFC 7761, DOI 10.17487/RFC7761, March 2016, <https://www.rfc-editor.org/info/rfc7761>.

[RFC8402] Filsfils, C., Ed., Previdi, S., Ed., Ginsberg, L., Decraene, B., Litkowski, S., and R. Shakir, "Segment Routing Architecture", RFC 8402, DOI 10.17487/RFC8402, July 2018, <https://www.rfc-editor.org/info/rfc8402>.

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/info/rfc8446>.

[RFC8754] Filsfils, C., Ed., Dukes, D., Ed., Previdi, S., Leddy, J., Matsushima, S., and D. Voyer, "IPv6 Segment Routing Header (SRH)", RFC 8754, DOI 10.17487/RFC8754, March 2020, <https://www.rfc-editor.org/info/rfc8754>.

[WG] Wireguard, "WireGuard, https://www.wireguard.com", August 2020.

Appendix A. Non-Normative Considerations

AERO can be applied to a multitude of Internetworking scenarios, with each having its own adaptations. The following considerations are provided as non-normative guidance:

A.1. Implementation Strategies for Route Optimization

Route optimization as discussed in Section 3.14 results in the route optimization source (ROS) creating a NCE for the target neighbor. The NCE state is set to REACHABLE for at most ReachableTime seconds. In order to refresh the NCE lifetime before the ReachableTime timer expires, the specification requires implementations to issue a new NS/NA exchange to reset ReachableTime while data packets are still flowing. However, the decision of when to initiate a new NS/NA exchange and to perpetuate the process is left as an implementation detail.

One possible strategy may be to monitor the NCE watching for data packets for (ReachableTime - 5) seconds. If any data packets have been sent to the neighbor within this timeframe, then send an NS to receive a new NA. If no data packets have been sent, wait for 5 additional seconds and send an immediate NS if any data packets are sent within this "expiration pending" 5 second window. If no additional data packets are sent within the 5 second window, reset the NCE state to STALE.

The monitoring of the neighbor data packet traffic therefore becomes an ongoing process during the NCE lifetime. If the NCE expires, future data packets will trigger a new NS/NA exchange while the packets themselves are delivered over a longer path until route optimization state is re-established.

A.2. Implicit Mobility Management

OMNI interface neighbors MAY provide a configuration option that allows them to perform implicit mobility management in which no ND messaging is used. In that case, the Client only transmits packets over a single interface at a time, and the neighbor always observes packets arriving from the Client from the same link-layer source address.

If the Client's underlying interface address changes (either due to a readdressing of the original interface or switching to a new interface) the neighbor immediately updates the NCE for the Client and begins accepting and sending packets according to the Client's new address. This implicit mobility method applies to use cases such as cellphones with both WiFi and Cellular interfaces where only one

of the interfaces is active at a given time, and the Client automatically switches over to the backup interface if the primary interface fails.

A.3. Direct Underlying Interfaces

When a Client's OMNI interface is configured over a Direct interface, the neighbor at the other end of the Direct link can receive packets without any encapsulation. In that case, the Client sends packets over the Direct link according to traffic selectors. If the Direct interface is selected, then the Client's IP packets are transmitted directly to the peer without going through an ANET/INET. If other interfaces are selected, then the Client's IP packets are transmitted via a different interface, which may result in the inclusion of Proxy/Servers and Bridges in the communications path. Direct interfaces must be tested periodically for reachability, e.g., via NUD.

A.4. AERO Critical Infrastructure Considerations

AERO Bridges can be either Commercial off-the Shelf (COTS) standard IP routers or virtual machines in the cloud. Bridges must be provisioned, supported and managed by the INET administrative authority, and connected to the Bridges of other INETs via inter-domain peerings. Cost for purchasing, configuring and managing Bridges is nominal even for very large OMNI links.

AERO cloud Proxy/Servers can be standard dedicated server platforms, but most often will be deployed as virtual machines in the cloud. The only requirements for cloud Proxy/Servers are that they can run the AERO user-level code and have at least one network interface connection to the INET. Cloud Proxy/Servers must be provisioned, supported and managed by the INET administrative authority. Cost for purchasing, configuring and managing cloud Proxy/Servers is nominal especially for virtual machines.

AERO ANET Proxy/Servers are most often standard dedicated server platforms with one underlying interface connected to the ANET and a second interface connected to an INET. As with cloud Proxy/Servers, the only requirements are that they can run the AERO user-level code and have at least one interface connection to the INET. ANET Proxy/Servers must be provisioned, supported and managed by the ANET administrative authority. Cost for purchasing, configuring and managing Proxys is nominal, and borne by the ANET administrative authority.

AERO Relays are simply Proxy/Servers connected to INETs and/or EUNs that provide forwarding services for non-MNP destinations. The Relay

connects to the OMNI link and engages in eBGP peering with one or more Bridges as a stub AS. The Relay then injects its MNPs and/or non-MNP prefixes into the BGP routing system, and provisions the prefixes to its downstream-attached networks. The Relay can perform ROS/ROR services the same as for any Proxy/Server, and can route between the MNP and non-MNP address spaces.

A.5. AERO Server Failure Implications

AERO Proxy/Servers may appear as a single point of failure in the architecture, but such is not the case since all Proxy/Servers on the link provide identical services and loss of a Proxy/Server does not imply immediate and/or comprehensive communication failures. Proxy/Server failure is quickly detected and conveyed by Bidirectional Forward Detection (BFD) and/or proactive NUD allowing Clients to migrate to new Proxy/Servers.

If a Proxy/Server fails, ongoing packet forwarding to Clients will continue by virtue of the neighbor cache entries that have already been established in route optimization sources (ROSs). If a Client also experiences mobility events at roughly the same time the Proxy/Server fails, unsolicited NA messages may be lost but neighbor cache entries in the DEPARTED state will ensure that packet forwarding to the Client's new locations will continue for up to DepartTime seconds.

If a Client is left without a Proxy/Server for a considerable length of time (e.g., greater than ReachableTime seconds) then existing neighbor cache entries will eventually expire and both ongoing and new communications will fail. The original source will continue to retransmit until the Client has established a new Proxy/Server relationship, after which time continuous communications will resume.

Therefore, providing many Proxy/Servers on the link with high availability profiles provides resilience against loss of individual Proxy/Servers and assurance that Clients can establish new Proxy/Server relationships quickly in event of a Proxy/Server failure.

A.6. AERO Client / Server Architecture

The AERO architectural model is client / server in the control plane, with route optimization in the data plane. The same as for common Internet services, the AERO Client discovers the addresses of AERO Proxy/Servers and connects to one or more of them. The AERO service is analogous to common Internet services such as google.com, yahoo.com, cnn.com, etc. However, there is only one AERO service for the link and all Proxy/Servers provide identical services.

Common Internet services provide differing strategies for advertising server addresses to clients. The strategy is conveyed through the DNS resource records returned in response to name resolution queries. As of January 2020 Internet-based 'nslookup' services were used to determine the following:

- When a client resolves the domainname "google.com", the DNS always returns one A record (i.e., an IPv4 address) and one AAAA record (i.e., an IPv6 address). The client receives the same addresses each time it resolves the domainname via the same DNS resolver, but may receive different addresses when it resolves the domainname via different DNS resolvers. But, in each case, exactly one A and one AAAA record are returned.

- When a client resolves the domainname "ietf.org", the DNS always returns one A record and one AAAA record with the same addresses regardless of which DNS resolver is used.

- When a client resolves the domainname "yahoo.com", the DNS always returns a list of 4 A records and 4 AAAA records. Each time the client resolves the domainname via the same DNS resolver, the same list of addresses are returned but in randomized order (i.e., consistent with a DNS round-robin strategy). But, interestingly, the same addresses are returned (albeit in randomized order) when the domainname is resolved via different DNS resolvers.

- When a client resolves the domainname "amazon.com", the DNS always returns a list of 3 A records and no AAAA records. As with "yahoo.com", the same three A records are returned from any worldwide Internet connection point in randomized order.

The above example strategies show differing approaches to Internet resilience and service distribution offered by major Internet services. The Google approach exposes only a single IPv4 and a single IPv6 address to clients. Clients can then select whichever IP protocol version offers the best response, but will always use the same IP address according to the current Internet connection point. This means that the IP address offered by the network must lead to a highly-available server and/or service distribution point. In other words, resilience is predicated on high availability within the network and with no client-initiated failovers expected (i.e., it is all-or-nothing from the client's perspective). However, Google does provide for worldwide distributed service distribution by virtue of the fact that each Internet connection point responds with a different IPv6 and IPv4 address. The IETF approach is like google (all-or-nothing from the client's perspective), but provides only a single IPv4 or IPv6 address on a worldwide basis. This means that the addresses must be made highly-available at the network level with

no client failover possibility, and if there is any worldwide service distribution it would need to be conducted by a network element that is reached via the IP address acting as a service distribution point.

In contrast to the Google and IETF philosophies, Yahoo and Amazon both provide clients with a (short) list of IP addresses with Yahoo providing both IP protocol versions and Amazon as IPv4-only. The order of the list is randomized with each name service query response, with the effect of round-robin load balancing for service distribution. With a short list of addresses, there is still expectation that the network will implement high availability for each address but in case any single address fails the client can switch over to using a different address. The balance then becomes one of function in the network vs function in the end system.

The same implications observed for common highly-available services in the Internet apply also to the AERO client/server architecture. When an AERO Client connects to one or more ANETs, it discovers one or more AERO Proxy/Server addresses through the mechanisms discussed in earlier sections. Each Proxy/Server address presumably leads to a fault-tolerant clustering arrangement such as supported by Linux-HA, Extended Virtual Synchrony or Paxos. Such an arrangement has precedence in common Internet service deployments in lightweight virtual machines without requiring expensive hardware deployment. Similarly, common Internet service deployments set service IP addresses on service distribution points that may relay requests to many different servers.

For AERO, the expectation is that a combination of the Google/IETF and Yahoo/Amazon philosophies would be employed. The AERO Client connects to different ANET access points and can receive 1-2 Proxy/Server ADM-LLAs at each point. It then selects one AERO Proxy/Server address, and engages in RS/RA exchanges with the same Proxy/Server from all ANET connections. The Client remains with this Proxy/Server unless or until the Proxy/Server fails, in which case it can switch over to an alternate Proxy/Server. The Client can likewise switch over to a different Proxy/Server at any time if there is some reason for it to do so. So, the AERO expectation is for a balance of function in the network and end system, with fault tolerance and resilience at both levels.

Appendix B. Change Log

<< RFC Editor - remove prior to publication >>

Changes from draft-templin-6man-aero-08 to draft-templin-6man-aero-09:

- Final editorial review pass resulting in multiple changes. Document now submit for final approval (with reference to rfcdiff from previous version).

Changes from draft-templin-6man-aero-07 to draft-templin-6man-aero-08:

- Final editorial review pass resulting in multiple changes. Document now submit for final approval (with reference to rfcdiff from previous version).

Changes from draft-templin-6man-aero-06 to draft-templin-6man-aero-07:

- Final editorial review pass resulting in multiple changes. Document now submit for final approval (with reference to rfcdiff from previous version).

Changes from draft-templin-6man-aero-05 to draft-templin-6man-aero-06:

- Final editorial review pass resulting in multiple changes. Document now submit for final approval.

Changes from draft-templin-6man-aero-04 to draft-templin-6man-aero-05:

- Changed to use traffic selectors instead of the former multilink selection strategy.

Changes from draft-templin-6man-aero-03 to draft-templin-6man-aero-04:

- Removed documents from "Obsoletes" list.

- Introduced the concept of "secured" and "unsecured" spanning tree.

- Additional security considerations.

- Additional route optimization considerations.

Changes from draft-templin-6man-aero-02 to draft-templin-6man-aero-03:

- Support for extended route optimization from ROR to target over target's underlying interfaces.

Changes from draft-templin-6man-aero-01 to draft-templin-6man-aero-02:

- Changed reference citations to "draft-templin-6man-omni".
- Several important updates to IPv6 ND cache states and route optimization message addressing.
- Included introductory description of the "6M's".
- Updated Multicast specification.

Changes from draft-templin-6man-aero-00 to draft-templin-6man-aero-01:

- Changed category to "Informational".
- Updated implementation status.

Changes from earlier versions to draft-templin-6man-aero-00:

- Established working baseline reference.

Author's Address

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

Email: fltemplin@acm.org