Network Working Group
Internet-Draft
Updates: rfc1191, rfc4443, rfc7526,
rfc8201 (if approved)
Intended status: Standards Track
Expires: September 25, 2021

F. Templin, Ed.
The Boeing Company
A. Whyman
MWA Ltd c/o Inmarsat Global Ltd

March 24, 2021

Transmission of IP Packets over Overlay Multilink Network (OMNI) Interfaces
draft-templin-6man-omni-interface-99

Abstract

Mobile nodes (e.g., aircraft of various configurations, terrestrial vehicles, seagoing vessels, enterprise wireless devices, etc.) communicate with networked correspondents over multiple access network data links and configure mobile routers to connect end user networks. A multilink interface specification is presented that allows mobile nodes to coordinate with a network-based mobility service and/or with other mobile node peers. This document specifies the transmission of IP packets over Overlay Multilink Network (OMNI) Interfaces.

Status of This Memo

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

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

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

This Internet-Draft will expire on September 25, 2021.

Table of Contents

1. Introduction

Mobile Nodes (MNs) (e.g., aircraft of various configurations, terrestrial vehicles, seagoing vessels, enterprise wireless devices, pedestrians with cellphones, etc.) often have multiple interface connections to wireless and/or wired-line data links used for communicating with networked correspondents. These data links may have diverse performance, cost and availability properties that can

change dynamically according to mobility patterns, flight phases, proximity to infrastructure, etc. MNs coordinate their data links in a discipline known as "multilink", in which a single virtual interface is configured over the node's underlying interface connections to the data links.

The MN configures a virtual interface (termed the "Overlay Multilink Network Interface (OMNI)") as a thin layer over the underlying interfaces. The OMNI interface is therefore the only interface abstraction exposed to the IP layer and behaves according to the Non-Broadcast, Multiple Access (NBMA) interface principle, while underlying interfaces appear as link layer communication channels in the architecture. The OMNI interface internally employs the "OMNI Adaptation Layer (OAL)" to ensure that original IP packets are delivered without loss due to size restrictions. The OMNI interface connects to a virtual overlay service known as the "OMNI link". The OMNI link spans one or more Internetworks that may include private-use infrastructures and/or the global public Internet itself.

Each MN receives a Mobile Network Prefix (MNP) for numbering downstream-attached End User Networks (EUNs) independently of the access network data links selected for data transport. The MN performs router discovery over the OMNI interface (i.e., similar to IPv6 customer edge routers [RFC7084]) and acts as a mobile router on behalf of its EUNs. The router discovery process is iterated over each of the OMNI interface's underlying interfaces in order to register per-link parameters (see Section 15).

The OMNI interface provides a multilink nexus for exchanging inbound and outbound traffic via the correct underlying interface(s). The IP layer sees the OMNI interface as a point of connection to the OMNI link. Each OMNI link has one or more associated Mobility Service Prefixes (MSPs), which are typically IP Global Unicast Address (GUA) prefixes from which MNPs are derived. If there are multiple OMNI links, the IPv6 layer will see multiple OMNI interfaces.

MNs may connect to multiple distinct OMNI links within the same OMNI domain by configuring multiple OMNI interfaces, e.g., omni0, omni1, omni2, etc. Each OMNI interface is configured over a set of underlying interfaces and provides a nexus for Safety-Based Multilink (SBM) operation. Each OMNI interface within the same OMNI domain configures a common ULA prefix [ULA]::/48, and configures a unique 16-bit Subnet ID '*' to construct the sub-prefix [ULA*]::/64 (see: Section 9). The IP layer applies SBM routing to select an OMNI interface, which then applies Performance-Based Multilink (PBM) to select the correct underlying interface. Applications can apply Segment Routing [RFC8402] to select independent SBM topologies for fault tolerance.

The OMNI interface interacts with a network-based Mobility Service (MS) through IPv6 Neighbor Discovery (ND) control message exchanges [RFC4861]. The MS provides Mobility Service Endpoints (MSEs) that track MN movements and represent their MNPs in a global routing or mapping system.

Many OMNI use cases have been proposed. In particular, the International Civil Aviation Organization (ICAO) Working Group-I Mobility Subgroup is developing a future Aeronautical Telecommunications Network with Internet Protocol Services (ATN/IPS) and has issued a liaison statement requesting IETF adoption [ATN] in support of ICAO Document 9896 [ATN-IPS]. The IETF IP Wireless Access in Vehicular Environments (ipwave) working group has further included problem statement and use case analysis for OMNI in a document now in AD evaluation for RFC publication [I-D.ietf-ipwave-vehicular-networking]. Still other communities of interest include AEEC, RTCA Special Committee 228 (SC-228) and NASA programs that examine commercial aviation, Urban Air Mobility (UAM) and Unmanned Air Systems (UAS). Pedestrians with handheld devices represent another large class of potential OMNI users.

This document specifies the transmission of IP packets and MN/MS control messages over OMNI interfaces. The OMNI interface supports either IP protocol version (i.e., IPv4 [RFC0791] or IPv6 [RFC8200]) as the network layer in the data plane, while using IPv6 ND messaging as the control plane independently of the data plane IP protocol(s). The OAL operates as a sublayer between L3 and L2 based on IPv6 encapsulation [RFC2473] as discussed in the following sections. OMNI interfaces enable Multilink, Mobility, Multihop, Multicast and MTU services (i.e., the "five M's"), with provisions for both Vehicle-to-Infrastructure (V2I) communications and Vehicle-to-Vehicle (V2V) communications outside the context of infrastructure.

2. Terminology

The terminology in the normative references applies; especially, the terms "link" and "interface" are the same as defined in the IPv6 [RFC8200] and IPv6 Neighbor Discovery (ND) [RFC4861] specifications. Additionally, this document assumes the following IPv6 ND message types: Router Solicitation (RS), Router Advertisement (RA), Neighbor Solicitation (NS), Neighbor Advertisement (NA) and Redirect.

The Protocol Constants defined in Section 10 of [RFC4861] are used in their same format and meaning in this document. The terms "All-Routers multicast", "All-Nodes multicast" and "Subnet-Router anycast" are the same as defined in [RFC4291] (with Link-Local scope assumed).

The term "IP" is used to refer collectively to either Internet Protocol version (i.e., IPv4 [RFC0791] or IPv6 [RFC8200]) when a specification at the layer in question applies equally to either version.

The following terms are defined within the scope of this document:

Mobile Node (MN)
: an end system with a mobile router having multiple distinct upstream data link connections that are grouped together in one or more logical units. The MN's data link connection parameters can change over time due to, e.g., node mobility, link quality, etc. The MN further connects a downstream-attached End User Network (EUN). The term MN used here is distinct from uses in other documents, and does not imply a particular mobility protocol.

End User Network (EUN)
: a simple or complex downstream-attached mobile network that travels with the MN as a single logical unit. The IP addresses assigned to EUN devices remain stable even if the MN's upstream data link connections change.

Mobility Service (MS)
: a mobile routing service that tracks MN movements and ensures that MNs remain continuously reachable even across mobility events. Specific MS details are out of scope for this document.

Mobility Service Endpoint (MSE)
: an entity in the MS (either singular or aggregate) that coordinates the mobility events of one or more MN.

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: Section 10). 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 assigned to a MN. MNs sub-delegate the MNP to devices located in EUNs. Note that OMNI link Relay nodes may also service non-MNP routes (i.e., GUA prefixes not covered by an MSP) but that these correspond to fixed correspondent nodes and not MNs. Other than this distinction, MNP

and non-MNP routes are treated exactly the same by the OMNI routing system.

Access Network (ANET)
: a data link service network (e.g., an aviation radio access network, satellite service provider network, cellular operator network, WiFi network, etc.) that connects MNs. Physical and/or data link level security is assumed, and sometimes referred to as "protected spectrum". Private enterprise networks and ground domain aviation service networks may provide multiple secured IP hops between the MN's point of connection and the nearest Access Router.

Access Router (AR)
: a router in the ANET for connecting MNs to correspondents in outside Internetworks. The AR may be located on the same physical link as the MN, or may be located multiple IP hops away. In the latter case, the MN uses encapsulation to communicate with the AR as though it were on the same physical link.

ANET interface
: a MN's attachment to a link in an ANET.

Internetwork (INET)
: a connected network region with a coherent IP addressing plan that provides transit forwarding services between ANETs and nodes that connect directly to the open INET via unprotected media. No physical and/or data link level security is assumed, therefore security must be applied by upper layers. The global public Internet itself is an example.

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

*NET
: a "wildcard" term used when a given specification applies equally to both ANET and INET cases.

OMNI link
: a Non-Broadcast, Multiple Access (NBMA) virtual overlay configured over one or more INETs and their connected ANETs. An OMNI link can comprise multiple INET segments joined by bridges the same as for any link; the addressing plans in each segment may be mutually exclusive and managed by different administrative entities.

OMNI interface
: a node's attachment to an OMNI link, and configured over one or more underlying *NET interfaces. If there are multiple OMNI links

in an OMNI domain, a separate OMNI interface is configured for each link.

OMNI Adaptation Layer (OAL)
: an OMNI interface sublayer service whereby original IP packets admitted into the interface are wrapped in an 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.

OMNI Option
an IPv6 Neighbor Discovery option providing multilink parameters for the OMNI interface as specified in Section 12.

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 Section 8.

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 Section 8.

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

Multilink
an OMNI interface's manner of managing diverse underlying interface connections to data links as a single logical unit. The OMNI interface provides a single unified interface to upper layers, while underlying interface selections are performed on a per-packet basis considering factors such as DSCP, flow label, application policy, signal quality, cost, etc. Multilinking decisions are coordinated in both the outbound (i.e. MN to correspondent) and inbound (i.e., correspondent to MN) directions.

Multihop
an iterative relaying of IP packets between MNs over an OMNI underlying interface technology (such as omnidirectional wireless) without support of fixed infrastructure. Multihop services entail node-to-node relaying within a Mobile/Vehicular Ad-hoc Network (MANET/VANET) for MN-to-MN communications and/or for "range extension" where MNs within range of communications infrastructure elements provide forwarding services for other MNs.

L2
: The second layer in the OSI network model. Also known as "layer-2", "link-layer", "sub-IP layer", "data link layer", etc.

L3
: The third layer in the OSI network model. Also known as "layer-3", "network-layer", "IP layer", etc.

underlying interface
: a *NET interface over which an OMNI interface is configured. The OMNI interface is seen as a L3 interface by the IP layer, and each underlying interface is seen as a L2 interface by the OMNI interface. The underlying interface either connects directly to the physical communications media or coordinates with another node where the physical media is hosted.

Mobility Service Identification (MSID)
: Each MSE and AR is assigned a unique 32-bit Identification (MSID) (see: Section 8). IDs are assigned according to MS-specific guidelines (e.g., see: [I-D.templin-intarea-6706bis]).

Safety-Based Multilink (SBM)
: A means for ensuring fault tolerance through redundancy by connecting multiple affiliated OMNI interfaces to independent routing topologies (i.e., multiple independent OMNI links).

Performance Based Multilink (PBM)
: A means for selecting underlying interface(s) for packet transmission and reception within a single OMNI interface.

OMNI Domain
: The set of all SBM/PBM OMNI links that collectively provides services for a common set of MSPs. Each OMNI domain consists of a set of affiliated OMNI links that all configure the same ::/48 ULA prefix with a unique 16-bit Subnet ID as discussed in Section 9.

3. Requirements

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.

An implementation is not required to internally use the architectural constructs described here so long as its external behavior is consistent with that described in this document.

4. Overlay Multilink Network (OMNI) Interface Model

An OMNI interface is a virtual interface configured over one or more underlying interfaces, which may be physical (e.g., an aeronautical radio link, etc.) or virtual (e.g., an Internet or higher-layer "tunnel"). The OMNI interface architectural layering model is the same as in [RFC5558][RFC7847], and augmented as shown in Figure 1. The IP layer therefore sees the OMNI interface as a single L3 interface nexus for multiple underlying interfaces that appear as L2 communication channels in the architecture.

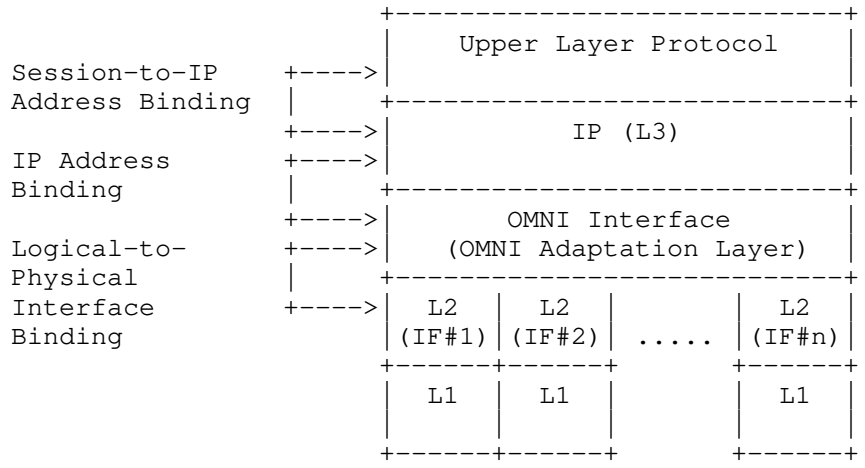


Figure 1: OMNI Interface Architectural Layering Model

Each underlying interface provides an L2/L1 abstraction according to one of the following models:

- INET interfaces connect to an INET either natively or through one or several IPv4 Network Address Translators (NATs). Native INET interfaces have global IP addresses that are reachable from any INET correspondent. NATed INET interfaces typically have private IP addresses and connect to a private network behind one or more NATs that provide INET access.

- ANET interfaces connect to a protected ANET that is separated from the open INET by an AR acting as a proxy. The ANET interface may be either on the same L2 link segment as the AR, or separated from the AR by multiple IP hops.

- VPNed interfaces use security encapsulation over a *NET to a Virtual Private Network (VPN) gateway. Other than the link-layer

encapsulation format, VPNed interfaces behave the same as for Direct interfaces.

- Direct (aka "point-to-point") interfaces connect directly to a peer without crossing any *NET paths. An example is a line-of-sight link between a remote pilot and an unmanned aircraft.

The OMNI interface forwards original IP packets from the network layer (L3) using the OMNI Adaptation Layer (OAL) (see: Section 5) as an encapsulation and fragmentation sublayer service. This "OAL source" then further encapsulates the resulting OAL packets/fragments in *NET headers to create OAL carrier packets for transmission over underlying interfaces (L2/L1). The target OMNI interface receives the carrier packets from underlying interfaces (L1/L2) and discards the *NET headers. If the resulting OAL packets/fragments are addressed to itself, the OMNI interface acts as an "OAL destination" and performs reassembly if necessary, discards the OAL encapsulation, and delivers the original IP packet to the network layer (L3). If the OAL fragments are addressed to another node, the OMNI interface instead acts as an "OAL intermediate node" by re-encapsulating in new *NET headers and forwarding the new carrier packets over an underlying interface without reassembling or discarding the OAL encapsulation. The OAL source and OAL destination are seen as "neighbors" on the OMNI link, while OAL intermediate nodes are seen as "bridges".

The OMNI interface can send/receive original IP packets to/from underlying interfaces while including/omitting various encapsulations including OAL, UDP, IP and L2. The network layer can also access the underlying interfaces directly while bypassing the OMNI interface entirely when necessary. This architectural flexibility may be beneficial for underlying interfaces (e.g., some aviation data links) for which encapsulation overhead may be a primary consideration. OMNI interfaces that send original IP packets directly over underlying interfaces without invoking the OAL can only reach peers located on the same OMNI link segment. However, an ANET proxy that receives the original IP packet can forward it further by performing OAL encapsulation with source set to its own address and destination set to the OAL destination corresponding to the final destination (i.e., even if the OAL destination is on a different OMNI link segment).

Original IP packets sent directly over underlying interfaces are subject to the same path MTU related issues as for any Internetworking path, and do not include per-packet identifications that can be used for data origin verification and/or link-layer retransmissions. Original IP packets presented directly to an underlying interface that exceed the underlying network path MTU are

dropped with an ordinary ICMPv6 Packet Too Big (PTB) message returned. These PTB messages are subject to loss [RFC2923] the same as for any non-OMNI IP interface.

The OMNI interface encapsulation/decapsulation layering possibilities are shown in Figure 2 below. In the figure, imaginary vertical lines drawn between the Network Layer and Underlying interfaces denote the encapsulation/decapsulation layering combinations possible. Common combinations include NULL (i.e., direct access to underlying interfaces with or without using the OMNI interface), OMNI/IP, OMNI/UDP/IP, OMNI/UDP/IP/L2, OMNI/OAL/UDP/IP, OMNI/OAL/UDP/L2, etc.

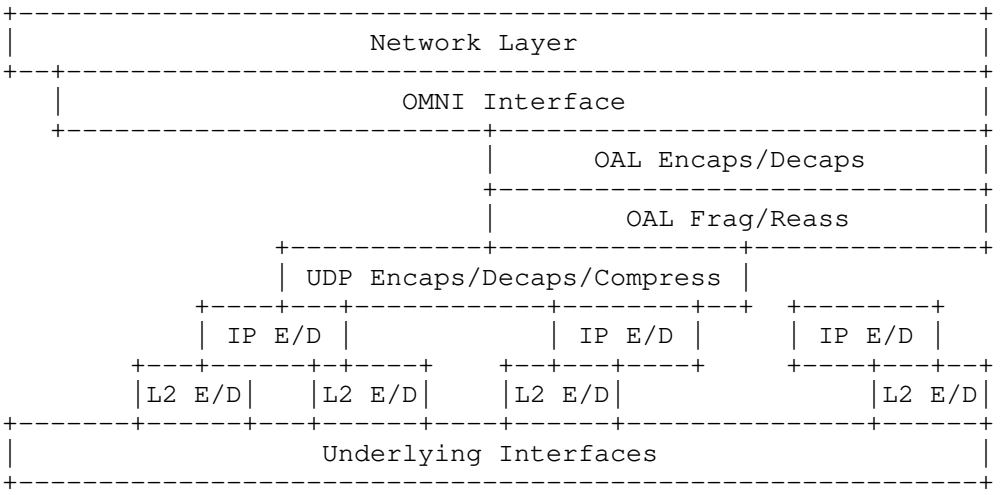


Figure 2: OMNI Interface Layering

The OMNI/OAL model gives rise to a number of opportunities:

- MNs receive a MNP from the MS, and coordinate with the MS through IPv6 ND message exchanges. The MN uses the MNP to construct a unique Link-Local Address (MNP-LLA) through the algorithmic derivation specified in Section 8 and assigns the LLA to the OMNI interface. Since MNP-LLAs are uniquely derived from an MNP, no Duplicate Address Detection (DAD) or Multicast Listener Discovery (MLD) messaging is necessary.

- since Temporary ULAs are statistically unique, they can be used without DAD, e.g. for MN-to-MN communications until an MNP-LLA is obtained.

- underlying interfaces on the same L2 link segment as an AR do not require any L3 addresses (i.e., not even link-local) in

environments where communications are coordinated entirely over the OMNI interface.

- as underlying interface properties change (e.g., link quality, cost, availability, etc.), any active interface can be used to update the profiles of multiple additional interfaces in a single message. This allows for timely adaptation and service continuity under dynamically changing conditions.

- coordinating underlying interfaces in this way allows them to be represented in a unified MS profile with provisions for mobility and multilink operations.

- exposing a single virtual interface abstraction to the IPv6 layer allows for multilink operation (including QoS based link selection, packet replication, load balancing, etc.) at L2 while still permitting L3 traffic shaping based on, e.g., DSCP, flow label, etc.

- the OMNI interface allows inter-INET traversal when nodes located in different INETs need to communicate with one another. This mode of operation would not be possible via direct communications over the underlying interfaces themselves.

- the OAL supports lossless and adaptive path MTU mitigations not available for communications directly over the underlying interfaces themselves. The OAL supports "packing" of multiple IP payload packets within a single OAL packet.

- the OAL applies per-packet identification values that allow for link-layer reliability and data origin authentication.

- L3 sees the OMNI interface as a point of connection to the OMNI link; if there are multiple OMNI links (i.e., multiple MS's), L3 will see multiple OMNI interfaces.

- Multiple independent OMNI interfaces can be used for increased fault tolerance through Safety-Based Multilink (SBM), with Performance-Based Multilink (PBM) applied within each interface.

Other opportunities are discussed in [RFC7847]. Note that even when the OMNI virtual interface is present, applications can still access underlying interfaces either through the network protocol stack using an Internet socket or directly using a raw socket. This allows for intra-network (or point-to-point) communications without invoking the OMNI interface and/or OAL. For example, when an IPv6 OMNI interface is configured over an underlying IPv4 interface, applications can still invoke IPv4 intra-network communications as long as the

communicating endpoints are not subject to mobility dynamics. However, the opportunities discussed above are not realized when the architectural layering is bypassed in this way.

Figure 3 depicts the architectural model for a MN with an attached EUN connecting to the MS via multiple independent *NETs. When an underlying interface becomes active, the MN's OMNI interface sends IPv6 ND messages without encapsulation if the first-hop Access Router (AR) is on the same underlying link; otherwise, the interface uses IP-in-IP encapsulation. The IPv6 ND messages traverse the ground domain *NETs until they reach an AR (AR#1, AR#2, ..., AR#n), which then coordinates with an INET Mobility Service Endpoint (MSE#1, MSE#2, ..., MSE#m) and returns an IPv6 ND message response to the MN. The Hop Limit in IPv6 ND messages is not decremented due to encapsulation; hence, the OMNI interface appears to be attached to an ordinary link.

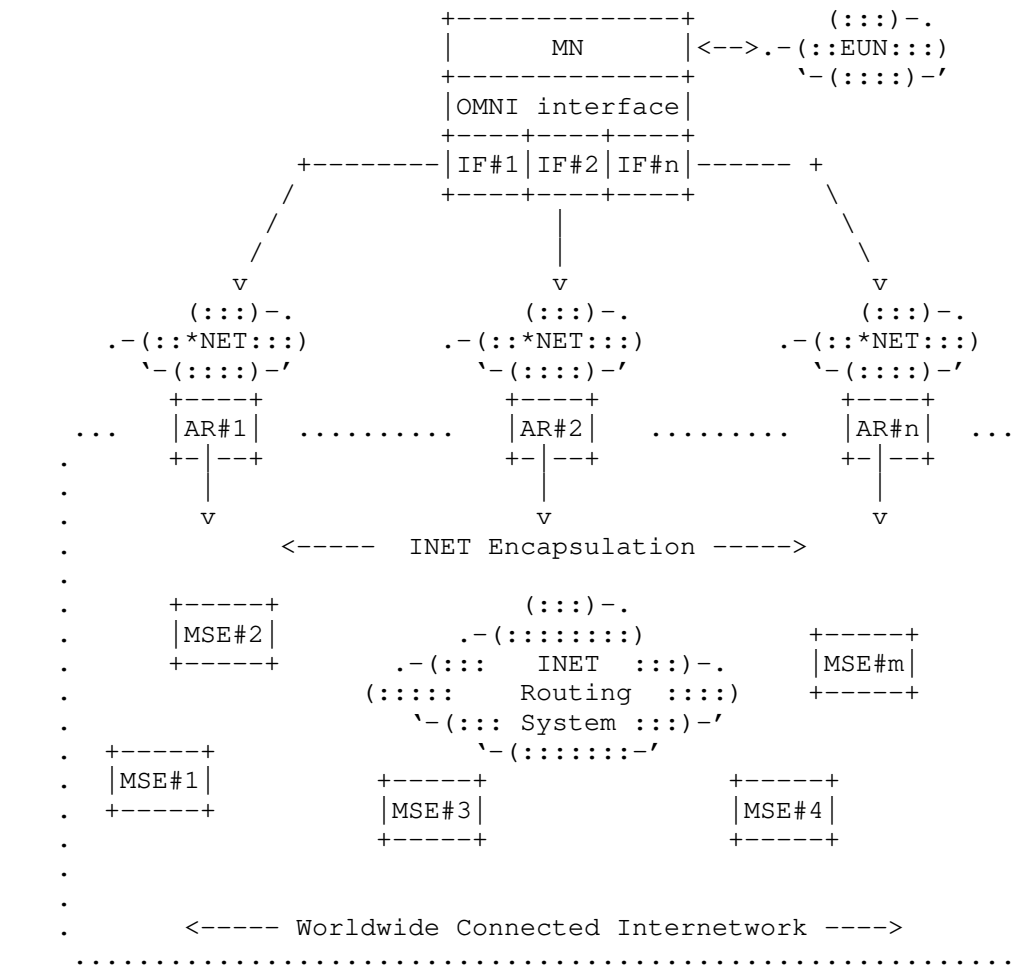


Figure 3: MN/MS Coordination via Multiple *NETs

After the initial IPv6 ND message exchange, the MN (and/or any nodes on its attached EUNs) can send and receive original IP packets over the OMNI interface. OMNI interface multilink services will forward the packets via ARs in the correct underlying *NETs. The AR encapsulates the packets according to the capabilities provided by the MS and forwards them to the next hop within the worldwide connected Internetwork via optimal routes.

5. OMNI Interface Maximum Transmission Unit (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 is configured over one or more underlying interfaces as discussed in Section 4, where the interfaces (and their associated *NET paths) may have diverse MTUs. OMNI interface considerations for accommodating original IP packets of various sizes are discussed in the following sections.

IPv6 underlying interfaces are REQUIRED to configure a minimum MTU of 1280 bytes and a minimum MRU of 1500 bytes [RFC8200]. Therefore, the minimum IPv6 path MTU is 1280 bytes since routers on the path are not permitted to perform network fragmentation even though the destination is required to reassemble more. The network therefore MUST forward original IP packets of at least 1280 bytes without generating an IPv6 Path MTU Discovery (PMTUD) Packet Too Big (PTB) message [RFC8201]. (While the source can apply "source fragmentation" for locally-generated IPv6 packets up to 1500 bytes and larger still if it knows the destination configures a larger MRU, this does not affect the minimum IPv6 path MTU.)

IPv4 underlying interfaces are REQUIRED to configure a minimum MTU of 68 bytes [RFC0791] and a minimum MRU of 576 bytes [RFC0791][RFC1122]. Therefore, when the Don’t Fragment (DF) bit in the IPv4 header is set to 0 the minimum IPv4 path MTU is 576 bytes since routers on the path support network fragmentation and the destination is required to reassemble at least that much. The OMNI interface therefore MUST set DF to 0 in the IPv4 encapsulation headers of carrier packets that are no larger than 576 bytes, and SHOULD set DF to 1 in larger carrier packets. (Note: even if the encapsulation source has a way to determine that the encapsulation destination configures an MRU larger than 576 bytes, it should not assume a larger minimum IPv4 path MTU without careful consideration of the issues discussed in Section 6.9.)

The OMNI interface configures an MTU and MRU of 9180 bytes [RFC2492]; the size is therefore not a reflection of the underlying interface or *NET path MTUs, but rather determines the largest original IP packet the OAL (and/or underlying interface) can forward or reassemble. For each OAL destination (i.e., for each OMNI link neighbor), the OAL source may discover "hard" or "soft" Reassembly Limit values smaller than the MRU based on receipt of IPv6 ND messages with OMNI Reassembly Limit sub-options (see: Section 12.1.11). The OMNI interface employs the OAL as an encapsulation sublayer service to transform original IP packets into OAL packets/fragments, and the OAL

in turn uses *NET encapsulation to forward carrier packets over the underlying interfaces (see: Section 6).

6. The OMNI Adaptation Layer (OAL)

When an OMNI interface forwards an original IP packet from the network layer for transmission over one or more underlying interfaces, the OMNI Adaptation Layer (OAL) acting as the OAL source drops the packet and returns a PTB message if the packet exceeds the MRU and/or the hard Reassembly Limit for the intended OAL destination. Otherwise, the OAL source applies encapsulation to form OAL packets and fragmentation to produce resulting OAL fragments suitable for *NET encapsulation and transmission as carrier packets over underlying interfaces as described in Section 6.1.

These carrier packets travel over one or more underlying networks bridged by OAL intermediate nodes, which re-encapsulate by removing the *NET headers of the first underlying network and appending *NET headers appropriate for the next underlying network in succession. After re-encapsulation by zero or more OAL intermediate nodes, the carrier packets arrive at the OAL destination.

When the OAL destination receives the carrier packets, it discards the *NET headers and reassembles the resulting OAL fragments into an OAL packet as described in Section 6.3. The OAL destination then decapsulates the OAL packet to obtain the original IP packet, which it then delivers to the network layer.

Detailed operations of the OAL are discussed in the following sections.

6.1. OAL Source Encapsulation and Fragmentation

When the network layer forwards an original IP packet into the OMNI interface, the OAL source inserts an IPv6 encapsulation header but does not decrement the Hop Limit/TTL of the original IP packet since encapsulation occurs at a layer below IP forwarding [RFC2473]. The OAL source copies the "Type of Service/Traffic Class" [RFC2983], "Flow Label"[RFC6438] (for IPv6) and "Congestion Experienced" [RFC3168] values in the original packet's IP header into the corresponding fields in the OAL header. The OAL source finally sets the OAL header IPv6 Hop Limit to a small value (e.g., 16) large enough to allow forwarding over a small number of OMNI link segments and sets the Payload Length to the length of the original IP packet.

The OAL next selects source and destination addresses for the IPv6 header of the resulting OAL packet. MN OMNI interfaces set the OAL IPv6 header source address to a Unique Local Address (ULA) based on

the Mobile Network Prefix (MNP-ULA), while AR and MSE OMNI interfaces set the source address to an Administrative ULA (ADM-ULA) (see: Section 9). When a MN OMNI interface does not (yet) have an MNP-ULA, it can use a Temporary ULA and/or Host Identity Tag (HIT) instead (see: Section 22).

When the OAL source forwards an original IP packet toward a final destination via an ANET underlying interface, it sets the OAL IPv6 header source address to its own ULA and sets the destination to either the Administrative ULA (ADM-ULA) of the ANET peer or the Mobile Network Prefix ULA (MNP-ULA) corresponding to the final destination (see below). The OAL source then fragments the OAL packet if necessary, encapsulates the OAL fragments in any ANET headers and sends the resulting carrier packets to the ANET peer which either reassembles before forwarding if the OAL destination is its own ULA or forwards the fragments toward the true OAL destination without first reassembling otherwise.

When the OAL source forwards an original IP packet toward a final destination via an INET underlying interface, it sets the OAL IPv6 header source address to its own ULA and sets the destination to the ULA of an OAL destination node on the final *NET segment. The OAL source then fragments the OAL packet if necessary, encapsulates the OAL fragments in any *NET headers and sends the resulting carrier packets toward the OAL destination on the final segment OMNI node which reassembles before forwarding the original IP packets toward the final destination.

Following OAL IPv6 encapsulation and address selection, the OAL source next appends a 2 octet trailing Checksum (initialized to 0) at the end of the original IP packet while incrementing the OAL header IPv6 Payload Length field to reflect the addition of the trailer. The format of the resulting OAL packet following encapsulation is shown in Figure 4:

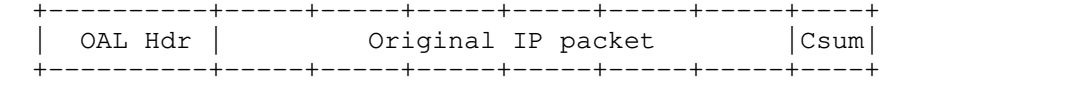


Figure 4: OAL Packet Before Fragmentation

The OAL source next selects a 32-bit Identification value for the packet, beginning with an unpredictable value for the initial OAL packet per [RFC7739] and monotonically incrementing for each successive OAL packet until a new initial value is chosen.

The OAL source then calculates the 2's complement (mod 256) Fletcher's checksum [CKSUM][RFC2328][RFC0905] over the entire OAL

packet beginning with a pseudo-header of the IPv6 header similar to that found in Section 8.1 of [RFC8200]. The OAL IPv6 pseudo-header is formed as shown in Figure 5:

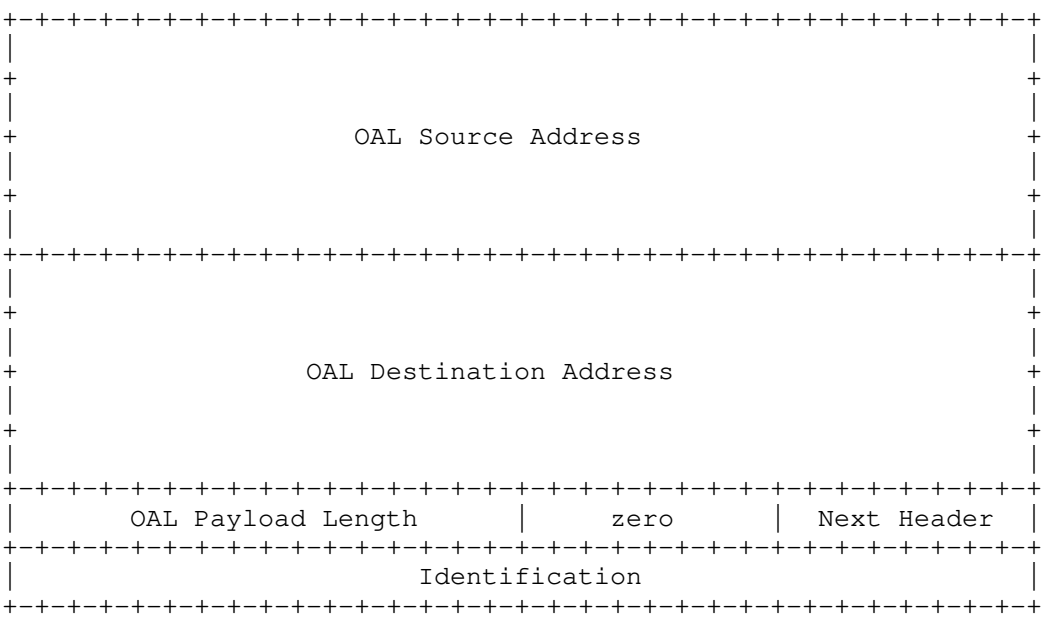


Figure 5: OAL IPv6 Pseudo-Header

The OAL source then inserts a single OMNI Routing Header (ORH) if necessary (see: [I-D.templin-intarea-6706bis]) while incrementing Payload Length to reflect the addition of the ORH (note that the late addition of the ORH is not covered by the trailing checksum).

The OAL source next fragments the OAL packet if necessary while assuming the IPv4 minimum path MTU (i.e., 576 bytes) as the worst case for OAL fragmentation regardless of the underlying interface IP protocol version since IPv6/IPv4 protocol translation and/or IPv6-in-IPv4 encapsulation may occur in any *NET path. By always assuming the IPv4 minimum even for IPv6 underlying interfaces, the OAL source may produce smaller fragments with additional encapsulation overhead but will always interoperate and never run the risk of loss due to an MTU restriction or due to presenting an underlying interface with a carrier packet that exceeds its MRU. Additionally, the OAL path could traverse multiple *NET "segments" with intermediate OAL forwarding nodes performing re-encapsulation where the *NET encapsulation of the previous segment is replaced by the *NET

encapsulation of the next segment which may be based on a different IP protocol version and/or encapsulation sizes.

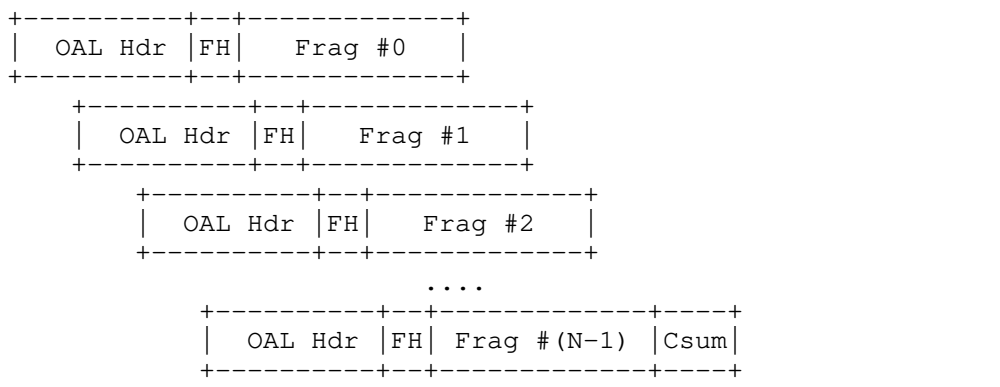
The OAL source therefore assumes a default minimum path MTU of 576 bytes at each *NET segment for the purpose of generating OAL fragments for *NET encapsulation and transmission as carrier packets. In the worst case, each successive *NET segment may re-encapsulate with either a 20 byte IPv4 or 40 byte IPv6 header, an 8 byte UDP header and in some cases an IP security encapsulation (40 bytes maximum assumed). Any *NET segment may also insert a maximum-length (40 byte) ORH as an extension to the existing 40 byte OAL IPv6 header plus 8 byte Fragment Header if an ORH was not already present. Assuming therefore an absolute worst case of (40 + 40 + 8) = 88 bytes for *NET encapsulation plus (40 + 40 + 8) = 88 bytes for OAL encapsulation leaves (576 - 88 - 88) = 400 bytes to accommodate a portion of the original IP packet/fragment. The OAL source therefore sets a minimum Maximum Payload Size (MPS) of 400 bytes as the basis for the minimum-sized OAL fragment that can be assured of traversing all segments without loss due to an MTU/MRU restriction. The Maximum Fragment Size (MFS) for OAL fragmentation is therefore determined by the MPS plus the size of the OAL encapsulation headers. (Note that the OAL source includes the 2 octet trailer as part of the payload during fragmentation, and the OAL destination regards it as ordinary payload until reassembly and checksum verification are complete.)

The OAL source SHOULD maintain "path MPS" values for individual OAL destinations initialized to the minimum MPS and increased to larger values (up to the OMNI interface MTU) if better information is known or discovered. For example, when *NET peers share a common underlying link or a fixed path with a known larger MTU, the OAL source can base path MPS on this larger size (i.e., instead of 576 bytes) as long as the *NET peer reassembles before re-encapsulating and forwarding (while re-fragmenting if necessary). Also, if the OAL source has a way of knowing the maximum *NET encapsulation size for all segments along the path it may be able to increase path MPS to reserve additional room for payload data. The OAL source must include the uncompressed OAL header size in its path MPS calculation, since a full header could be included at any time.

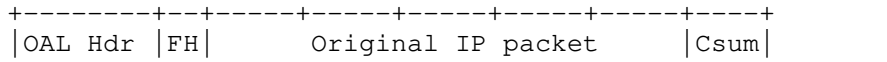
The OAL source can also actively probe individual OAL destinations to discover larger path MPS values using packetization layer probes per [RFC4821][RFC8899], but care must be taken to avoid setting static values for dynamically changing paths leading to black holes. The probe involves sending an OAL packet larger than the current path MPS and receiving a small acknowledgement message in response (with the possible receipt of link-layer error message in case the probe was lost). For this purpose, the OAL source can send an NS message with one or more OMNI options with large PadN sub-options (see:

Section 12) in order to receive a small NA response from the OAL destination. While observing the minimum MPS will always result in robust and secure behavior, the OAL source should optimize path MPS values when more efficient utilization may result in better performance (e.g. for wireless aviation data links).

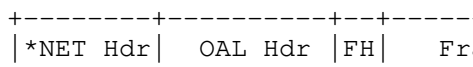
When the OAL source performs fragmentation, it SHOULD produce the minimum number of non-overlapping fragments under current MPS constraints, where each non-final fragment MUST be of equal length at least as large as the minimum MPS, while the final fragment MAY be of different length. The OAL source also converts all original IP packets no larger than the current MPS into "atomic fragments" by including a Fragment Header with Fragment Offset and More Fragments both set to 0. The OAL source finally encapsulates the fragments in *NET headers to form carrier packets and forwards them over an underlying interface, while retaining the fragments and their ordinal positions (i.e., as Frag #0, Frag #1, Frag #2, etc.) for a timeout period in case link-layer retransmission is requested. The formats of OAL fragments and carrier packets are shown in Figure 6.



a) OAL fragments after fragmentation
(FH = Fragment Header; Csum appears only in final fragment)



b) An OAL atomic fragment with FH but no fragmentation.



c) OAL carrier packet after *NET encapsulation

Figure 6: OAL Fragments and Carrier Packets

6.2. OAL *NET Encapsulation and Re-Encapsulation

During *NET encapsulation, OAL sources first encapsulate each OAL fragment in a UDP header as the first *NET encapsulation sublayer if NAT traversal, packet filtering middlebox traversal and/or OAL header compression are necessary. The OAL then optionally appends additional encapsulation sublayer headers, then presents the *NET packet to an underlying interface. This layering can be seen in Figure 2.

When a UDP header is included, the OAL source next sets the UDP source port to a constant value that it will use in each successive carrier packet it sends to the next OAL hop. For packets sent to an MSE, the OAL source sets the UDP destination port to 8060, i.e., the IANA-registered port number for AERO. For packets sent to a MN peer, the source sets the UDP destination port to the cached port value for this peer. The OAL source then sets the UDP length to the total length of the OAL fragment in correspondence with the OAL header

Payload Length (i.e., the UDP length and IPv6 Payload Length must agree). The OAL source finally sets the UDP checksum to 0 [RFC6935][RFC6936] since the only fields not already covered by the OAL checksum or underlying *NET CRCs are the Fragment Header fields, and any corruption in those fields will be garbage collected by the reassembly algorithm. The UDP encapsulation header is often used in association with IP encapsulation, but may also be used between neighbors on a shared physical link with a true L2 header format such as for transmission over IEEE 802 Ethernet links. This document therefore requests a new Ether Type code assignment TBD1 in the IANA 'ieee-802-numbers' registry for direct User Datagram Protocol (UDP) encapsulation over IEEE 802 Ethernet links (see: Section 25).

For *NET encapsulations, the OAL source next copies the "Type of Service/Traffic Class" [RFC2983], "Congestion Experienced" [RFC3168] and "Flow Label" [RFC6438] (for IPv6) values in the OAL IPv6 header into the corresponding fields in the *NET IP header. For carrier packets undergoing re-encapsulation, OAL intermediate nodes instead copy these values from the previous hop *NET encapsulation header into both the OAL IPv6 header and the next hop *NET encapsulation header, i.e., the IP values are transferred between *NET encapsulation headers and *not* copied from the OAL header. During re-encapsulation, the intermediate node decrements the OAL IPv6 header Hop Limit and discards the carrier packet if the value reaches 0.

Following *NET encapsulation/re-encapsulation, the OAL source sends the resulting carrier packets over one or more underlying interfaces. The underlying interfaces often connect directly to physical media on the local platform (e.g., a laptop computer with WiFi, etc.), but in some configurations the physical media may be hosted on a separate Local Area Network (LAN) node. In that case, the OMNI interface can establish a Layer-2 VLAN or a point-to-point tunnel (at a layer below the underlying interface) to the node hosting the physical media. The OMNI interface may also apply encapsulation at the underlying interface layer (e.g., as for a tunnel virtual interface) such that carrier packets would appear "double-encapsulated" on the LAN; the node hosting the physical media in turn removes the LAN encapsulation prior to transmission or inserts it following reception. Finally, the underlying interface must monitor the node hosting the physical media (e.g., through periodic keepalives) so that it can convey up/down/status information to the OMNI interface.

6.3. OAL Destination Decapsulation and Reassembly

When an OMNI interface receives a carrier packet from an underlying interface, the OAL destination discards the *NET encapsulation headers and examines the OAL header of the enclosed OAL fragment. If

the OAL fragment is addressed to a different node, the OAL destination re-encapsulates and forwards as discussed below. If the OAL fragment is addressed to itself, the OAL destination creates or updates a checklist for this (Source, Destination, Identification)-tuple to track the fragments already received (i.e., by examining the Payload Length, Fragment Offset, More Fragments and Identification values supplied by the OAL source). The OAL destination verifies that all non-final OAL fragments are of equal length no less than the minimum MPS and that no fragments overlap or leave "holes", while dropping any non-conforming fragments. The OAL destination records each conforming OAL fragment's ordinal position based on the OAL header Payload Length and Fragment Offset values (i.e., as Frag #0, Frag #1, Frag #2, etc.) and admits each fragment into the reassembly cache.

When reassembly is complete, the OAL destination removes the ORH if present while decrementing Payload Length to reflect the removal of the ORH. The OAL destination next verifies the resulting OAL packet's checksum and discards the packet if the checksum is incorrect. If the OAL packet was accepted, the OAL destination then removes the OAL header/trailer, then delivers the original IP packet to the network layer. Note that link layers include a CRC-32 integrity check which provides effective hop-by-hop error detection in the underlying network for payload sizes up to the OMNI interface MTU [CRC], but that some hops may traverse intermediate layers such as tunnels over IPv4 that do not include integrity checks. The trailing Fletcher checksum therefore allows the OAL destination to detect OAL packet splicing errors due to reassembly misassociations and/or to verify integrity for OAL packets whose fragments may have traversed unprotected underlying network hops [CKSUM]. The Fletcher algorithm also provides diversity with respect to both lower layer CRCs and upper layer Internet checksums as part of a complimentary multi-layer integrity assurance architecture.

6.4. OAL Header Compression

When the OAL source and destination are on the same *NET segment, no ORH is needed and carrier packet header compression is possible. When the OAL source and destination exchange initial IPv6 ND messages as discussed in the following Sections, each caches the observed *NET UDP source port and source IP (or L2) address associated with the OAL IPv6 source address found in the full-length OAL IPv6 header. After the initial IPv6 ND message exchange, the OAL source can begin applying OAL Header Compression to significantly reduce the encapsulation overhead required in each carrier packet.

When the OAL source determines that header compression state has been established (i.e., following the IPv6 ND message exchange), it can

begin sending OAL fragments with significant portions of the IPv6 header and Fragment Header omitted thereby reducing the amount of encapsulation overhead. For OAL first-fragments (including atomic fragments), the OMNI Compressed Header - Type 0 (OCH-0) is used and formatted as shown in Figure 7:

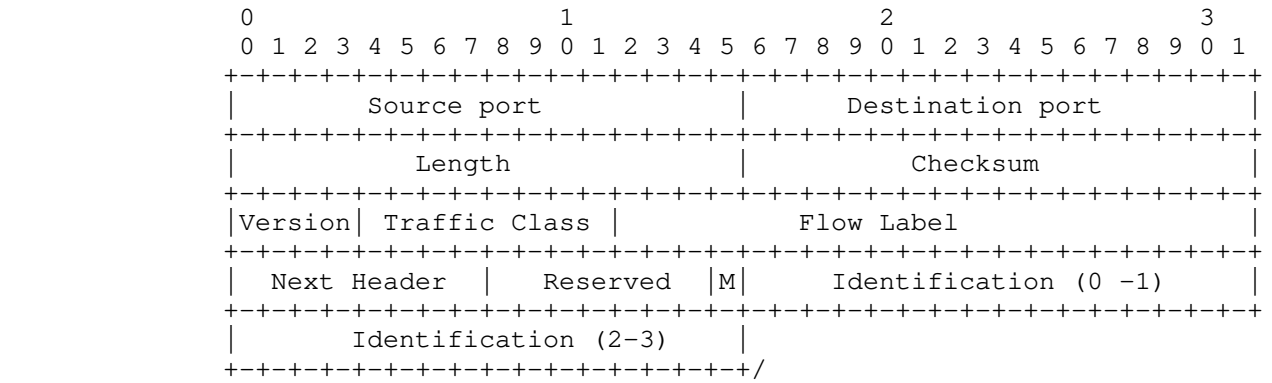


Figure 7: OMNI Compressed Header - Type 0 (OCH-0)

In this format, the UDP header appears in its entirety in the first 8 octets, then followed by the first 4 octets of the IPv6 header with the remainder omitted. (The IPv6 Version field is set to the value 0 to distinguish this header from a true IP protocol version number and from OCH-1 - see below.) The compressed IPv6 header is then followed by a compressed IPv6 Fragment Header with the Fragment Offset field and two Reserved bits omitted (since these fields always encode the value 0 in first-fragments), and with the More Fragments (M) bit relocated to the least significant bit of the first Reserved field. The OCH-0 header is then followed by the OAL fragment body, and the UDP length field is reduced by 38 octets (i.e., the difference in length between full-length IPv6 and Fragment Headers and the length of the compressed headers).

For OAL non-first fragments (i.e., those with non-zero Fragment Offsets), the OMNI Compressed Header - Type 1 (OCH-1) is used and formatted as shown in Figure 8:

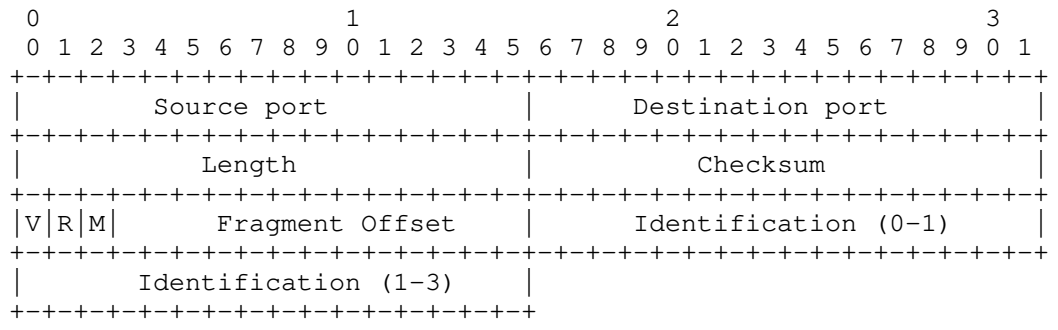


Figure 8: OMNI Compressed Header - Type 1 (OCH-1)

In this format, the UDP header appears in its entirety in the first 8 octets, but all IPv6 header fields except for the most significant Version (V) bit are omitted. (The V bit is set to the value 1 to distinguish this header from a true IP protocol version number and from OCH-0.) The V bit is followed by a single Reserved (R) bit and the More Fragments (M) bit in a compressed IPv6 Fragment Header with the Next Header and first Reserved fields omitted. The OCH-1 header is then followed by the OAL fragment body, and the UDP length field is reduced by 42 octets (i.e., the difference in length between full-length IPv6 and Fragment Headers and the length of the compressed headers).

When the OAL destination receives a carrier packet with an OCH, it first determines the OAL IPv6 source and destination addresses by examining the UDP source port and L2 source address, then determines the length by examining the UDP length. The OAL destination then examines the (V)ersion field immediately following the UDP header. If the (4-bit) Version field encodes the value 0, the OAL destination processes the remainder of the header as an OCH-0, then reconstitutes the full-sized IPv6 and Fragment Headers and adds this OAL fragment to the reassembly buffer if necessary. If the (1-bit) V bit encodes the value 1, the OAL destination instead processes the remainder of the header as an OCH-1, then reconstitutes the full-sized IPv6 and Fragment Headers and adds this OAL fragment to the reassembly buffer. Note that, since the OCH-1 does not include Traffic Class, Flow Label or Next Header information, the OAL destination writes the value 0 into these fields when it reconstitutes the full headers. These values will be correctly populated during reassembly after an OAL first fragment with an OCH-0 or uncompressed OAL header arrives.

6.5. OAL Fragment Identification Window Maintenance

As noted above, the OAL source establishes a window of 32-bit Identifications beginning with an unpredictable value for the initial message [RFC7739] and monotonically incrementing for each successive OAL packet until a new initial value is chosen. The OAL source asserts the starting value by including it as the Identification in an IPv6 ND NS/RS messages. When the OAL destination receives the IPv6 ND message, it resets the Identification window for this OAL source to the new value coded in the message's OAL header and expects future OAL fragments received from this OAL source to include sequential Identification values (subject to loss and reordering) until the neighbor reachable time expires or the OAL source sends a new IPv6 ND message.

For example, if the OAL destination receives an NS/RS message with Identification 0x12345678, it resets the window for this OAL source to begin with 0x12345678 and examines the Identification values in subsequent OAL fragments received from this OAL source. If the Identification values of subsequent OAL fragments fall within the window of (0x12345678 + N) the OAL destination accepts the fragment; otherwise, it silently drops the fragment (where "N" represents the maximum number of fragments expected before the neighbor reachable time expires).

While monitoring the current window, the OAL destination must accept new NS/RS Identification values even if outside the current window. The new Identification value resets the OAL destination's window start, and the window processing continues from this new starting point while allowing a period of overlap in case OAL fragments with Identification values from a previous window are still in flight. Note also that unsolicited NA messages must include Identification values within the current window, and therefore do not reset the current window.

This implies that an IPv6 ND message used to reset the Identification window should fit within a single OAL fragment (i.e., within current MPS constraints), since a fragmented IPv6 ND message with an out-of-window Identification value could be part of a DoS attack. While larger IPv6 ND messages (up to the OMNI interface MTU) can certainly be subject to OAL fragmentation, their Identification should be within the current window maintained by the OAL destination to increase the likelihood that they will be accepted.

6.6. OAL Fragment Retransmission

When the OAL source sends carrier packets with OAL fragments to an OAL destination, the source caches them for a timeout period in case retransmission may be necessary. (The timeout duration is an implementation matter, and may be influenced by factors such as packet arrival rates, OAL source/destination round trip times, etc.) The OAL destination in turn maintains a checklist for the (Source, Destination, Identification)-tuple of each new OAL fragment received and notes the ordinal positions of fragments already received (i.e., as Frag #0, Frag #1, Frag #2, etc.).

If the OAL destination notices some OAL fragments missing after most other fragments within the same Identification window have already arrived, it may send an IPv6 ND unsolicited Neighbor Advertisement (uNA) message to the OAL source that originated the fragments to report loss. The OAL destination creates a uNA message with an OMNI option containing an authentication sub-option to provide authentication (if the OAL source is on an open Internetwork) followed by a Fragmentation Report sub-option that includes a list of (Identification, Bitmap)-tuples for OAL fragments received and missing from this OAL source (see: Section 12). The OAL destination signs the message if an authentication sub-option is included, performs OAL encapsulation (with the its own address as the OAL source and the source address of the message that prompted the uNA as the OAL destination) and sends the message to the OAL source.

When the OAL source receives the uNA message, it authenticates the message using authentication sub-option (if present) then examines the Fragmentation Report. For each (Source, Destination, Identification)-tuple, the OAL source determines whether it still holds the original OAL fragments in its cache and retransmits any for which the Bitmap indicated a loss event. For example, if the Bitmap indicates that the ordinal OAL fragments Frag #3, Frag #7, Frag #10 and Frag #13 from the same OAL packet are missing the OAL source retransmits these fragments only and no others.

Note that the goal of this service is to provide a light-weight link-layer Automatic Repeat Request (ARQ) capability in the spirit of Section 8.1 of [RFC3819]. Rather than provide true end-to-end reliability, however, the service provides timely link-layer retransmissions that may improve packet delivery ratios and avoid some delays inherent in true end-to-end services.

6.7. OAL MTU Feedback Messaging

When the OMNI interface forwards original IP packets from the network layer, it invokes the OAL and returns internally-generated ICMPv4 Fragmentation Needed [RFC1191] or ICMPv6 Path MTU Discovery (PMTUD) Packet Too Big (PTB) [RFC8201] messages as necessary. This document refers to both of these ICMPv4/ICMPv6 message types simply as "PTBs", and introduces a distinction between PTB "hard" and "soft" errors as discussed below.

Ordinary PTB messages with ICMPv4 header "unused" field or ICMPv6 header Code field value 0 are hard errors that always indicate that a packet has been dropped due to a real MTU restriction. In particular, the OAL source drops the packet and returns a PTB hard error if the packet exceeds the OAL destination MRU. However, the OMNI interface can also forward large original IP packets via OAL encapsulation and fragmentation while at the same time returning PTB soft error messages (subject to rate limiting) if it deems the original IP packet too large according to factors such as link performance characteristics, reassembly congestion, etc. This ensures that the path MTU is adaptive and reflects the current path used for a given data flow. The OMNI interface can therefore continuously forward packets without loss while returning PTB soft error messages recommending a smaller size if necessary. Original sources that receive the soft errors in turn reduce the size of the packets they send (i.e., the same as for hard errors), but can soon resume sending larger packets if the soft errors subside.

An OAL source sends PTB soft error messages by setting the ICMPv4 header "unused" field or ICMPv6 header Code field to the value 1 if a original IP packet was deemed lost (e.g., due to reassembly timeout) or to the value 2 otherwise. The OAL source sets the PTB destination address to the original IP packet source, and sets the source address to one of its OMNI interface unicast/anycast addresses that is routable from the perspective of the original source. The OAL source then sets the MTU field to a value smaller than the original packet size but no smaller than 576 for ICMPv4 or 1280 for ICMPv6, writes the leading portion of the original IP packet into the "packet in error" field, and returns the PTB soft error to the original source. When the original source receives the PTB soft error, it temporarily reduces the size of the packets it sends the same as for hard errors but may seek to increase future packet sizes dynamically while no further soft errors are arriving. (If the original source does not recognize the soft error code, it regards the PTB the same as a hard error but should heed the retransmission advice given in [RFC8201] suggesting retransmission based on normal packetization layer retransmission timers.) This document therefore updates [RFC1191][RFC4443] and [RFC8201]. Furthermore, packetization layer

probing strategies [RFC4821][RFC8899] must be aware that PTB hard or soft errors may arrive at any time, i.e., even following a successful probe (this is the same consideration as for an ordinary path fluctuation following a successful probe).

An OAL destination may experience reassembly cache congestion, and can return uNA messages to the OAL source that originated the fragments (subject to rate limiting) to advertise reduced hard/soft Reassembly Limits and/or to report individual reassembly failures. The OAL destination creates a uNA message with an OMNI option containing an authentication message sub-option (if the OAL source is on an open Internetwork) followed optionally by at most one hard and one soft Reassembly Limit sub-options with reduced hard/soft values, and with one of them optionally including the leading portion an OAL first fragment containing the header of an original IP packet whose source must be notified (see: Section 12). The OAL destination encapsulates as much of the OAL first fragment (beginning with the OAL header) as will fit in the "OAL First Fragment" field of sub-option without causing the entire uNA message to exceed the minimum MPS, signs the message if an authentication sub-option is included, performs OAL encapsulation (with the its own address as the OAL source and the source address of the message that prompted the uNA as the OAL destination) and sends the message to the OAL source.

When the OAL source receives the uNA message, it records the new hard/soft Reassembly Limit values for this OAL destination if the OMNI option includes Reassembly Limit sub-options. If a hard or soft Reassembly Limit sub-option includes an OAL First Fragment, the OAL source next sends a corresponding network layer PTB hard or soft error to the original source to recommend a smaller size. For hard errors, the OAL source sets the PTB Code field to 0. For soft errors, the OAL source sets the PTB Code field to 1 if the L flag in the Reassembly Limit sub-option is 1; otherwise, the OAL source sets the Code field to 2. The OAL source crafts the PTB by extracting the leading portion of the original IP packet from the OAL First Fragment field (i.e., not including the OAL header) and writes it in the "packet in error" field of a PTB with destination set to the original IP packet source and source set to one of its OMNI interface unicast/anycast addresses that is routable from the perspective of the original source. For future transmissions, if the original IP packet is larger than the hard Reassembly Limit for this OAL destination the OAL source drops the packet and returns a PTB hard error with MTU set to the hard Reassembly Limit. If the packet is no larger than the current hard Reassembly Limit but larger than the current soft limit, the OAL source can also return PTB soft errors (subject to rate limiting) with Code set to 2 and MTU set to the current soft limit while still forwarding the packet to the OMNI destination.

Original sources that receive PTB soft errors can dynamically tune the size of the original IP packets they to send to produce the best possible throughput and latency, with the understanding that these parameters may change over time due to factors such as congestion, mobility, network path changes, etc. The receipt or absence of soft errors should be seen as hints of when increasing or decreasing packet sizes may be beneficial. The OMNI interface supports continuous transmission and reception of packets of various sizes in the face of dynamically changing network conditions. Moreover, since PTB soft errors do not indicate a hard limit, original sources that receive soft errors can begin sending larger packets without waiting for the recommended 10 minutes specified for PTB hard errors [RFC1191][RFC8201]. The OMNI interface therefore provides an adaptive service that accommodates MTU diversity especially well-suited for dynamic multilink environments.

6.8. OAL Requirements

In light of the above, OAL sources, destinations and intermediate nodes observe the following normative requirements:

- OAL sources MUST NOT send OAL fragments including original IP packets larger than the OMNI interface MTU or the OAL destination hard Reassembly Limit, i.e., whether or not fragmentation is needed.

- OAL sources MUST NOT perform OAL fragmentation for original IP packets smaller than the minimum MPS minus the trailer size, and MUST produce non-final fragments that contain equal-length payloads no smaller than the minimum MPS when performing fragmentation.

- OAL sources MUST NOT send OAL fragments that include any extension headers other than a single ORH and a single Fragment Header.

- OAL intermediate nodes SHOULD and OAL destinations MUST unconditionally drop OAL packets/fragments including original IP packets larger than the OMNI interface MRU and/or OAL destination hard Reassembly Limit, i.e., whether or not reassembly was needed.

- OAL intermediate nodes SHOULD and OAL destinations MUST unconditionally drop any non-final OAL fragments containing a payload smaller than the minimum MPS.

- OAL intermediate nodes SHOULD and OAL destinations MUST unconditionally drop OAL fragments that include any extension headers other than a single ORH and a single Fragment Header.

- o OAL destination nodes MUST drop any new OAL non-final fragments of different length than other non-final fragments that have already been received, and MUST drop any new OAL fragments with Offset and Payload length that would overlap with other fragments and/or leave too-small holes between fragments that have already been received.

Note: Under the minimum MPS, ordinary 1500 byte original IP packets would require at most 4 OAL fragments, with each non-final fragment containing 400 payload bytes and the final fragment containing 302 payload bytes (i.e., the final 300 bytes of the original IP packet plus the 2 octet trailer). Likewise, maximum-length 9180 byte original IP packets would require at most 23 fragments. For all packet sizes, the likelihood of successful reassembly may improve when the OMNI interface sends all fragments of the same fragmented OAL packet consecutively over the same underlying interface. Finally, an assured minimum/path MPS allows continuous operation over all paths including those that traverse bridged L2 media with dissimilar MTUs.

Note: Certain legacy network hardware of the past millennium was unable to accept packet "bursts" resulting from an IP fragmentation event - even to the point that the hardware would reset itself when presented with a burst. This does not seem to be a common problem in the modern era, where fragmentation and reassembly can be readily demonstrated at line rate (e.g., using tools such as 'iperf3') even over fast links on average hardware platforms. Even so, the OAL source could impose an inter-fragment delay while the OAL destination is reporting reassembly congestion (see: Section 6.7) and decrease the delay when reassembly congestion subsides.

6.9. OAL Fragmentation Security Implications

As discussed in Section 3.7 of [RFC8900], there are four basic threats concerning IPv6 fragmentation; each of which is addressed by effective mitigations as follows:

1. Overlapping fragment attacks - reassembly of overlapping fragments is forbidden by [RFC8200]; therefore, this threat does not apply to the OAL.

2. Resource exhaustion attacks - this threat is mitigated by providing a sufficiently large OAL reassembly cache and instituting "fast discard" of incomplete reassemblies that may be part of a buffer exhaustion attack. The reassembly cache should be sufficiently large so that a sustained attack does not cause excessive loss of good reassemblies but not so large that (timer-based) data structure management becomes computationally

expensive. The cache should also be indexed based on the arrival underlying interface such that congestion experienced over a first underlying interface does not cause discard of incomplete reassemblies for uncongested underlying interfaces.

3. Attacks based on predictable fragment identification values - this threat is mitigated by selecting a suitably random ID value per [RFC7739]. Additionally, inclusion of the OAL checksum would make it very difficult for an attacker who could somehow predict a fragment identification value to inject malicious fragments resulting in undetected reassemblies of bad data.

4. Evasion of Network Intrusion Detection Systems (NIDS) - this threat is mitigated by setting a minimum MPS for OAL fragmentation, which defeats all "tiny fragment"-based attacks.

Additionally, IPv4 fragmentation includes a 16-bit Identification (IP ID) field with only 65535 unique values such that at high data rates the field could wrap and apply to new carrier packets while the fragments of old packets using the same ID are still alive in the network [RFC4963]. However, since the largest carrier packet that will be sent via an IPv4 path with DF = 0 is 576 bytes any IPv4 fragmentation would occur only on links with an IPv4 MTU smaller than this size, and [RFC3819] recommendations suggest that such links will have low data rates. Since IPv6 provides a 32-bit Identification value, IP ID wraparound at high data rates is not a concern for IPv6 fragmentation.

Finally, [RFC6980] documents fragmentation security concerns for large IPv6 ND messages. These concerns are addressed when the OMNI interface employs the OAL instead of directly fragmenting the IPv6 ND message itself. For this reason, OMNI interfaces MUST NOT send IPv6 ND messages larger than the OMNI interface MTU, and MUST employ OAL encapsulation and fragmentation for IPv6 ND messages larger than the current MPS for this OAL destination.

6.10. OAL Super-Packets

By default, the OAL source includes a 40-byte IPv6 encapsulation header for each original IP packet during OAL encapsulation. The OAL source also calculates and appends a 2 octet trailing Fletcher checksum then performs fragmentation such that a copy of the 40-byte IPv6 header plus an 8-byte IPv6 Fragment Header is included in each OAL fragment (when an ORH is added, the OAL encapsulation headers become larger still). However, these encapsulations may represent excessive overhead in some environments. OAL header compression can dramatically reduce the amount of encapsulation overhead, however a complimentary technique known as "packing" (see:

[I-D.ietf-intarea-tunnels]) is also supported so that multiple original IP packets and/or control messages can be included within a single OAL "super-packet".

When the OAL source has multiple original IP packets to send to the same OAL destination with total length no larger than the OAL destination MRU, it can concatenate them into a super-packet encapsulated in a single OAL header and trailing checksum. Within the OAL super-packet, the IP header of the first original IP packet (iHa) followed by its data (iDa) is concatenated immediately following the OAL header, then the IP header of the next original packet (iHb) followed by its data (iDb) is concatenated immediately following the first original packet, etc. with the trailing checksum included last. The OAL super-packet format is transposed from [I-D.ietf-intarea-tunnels] and shown in Figure 9:

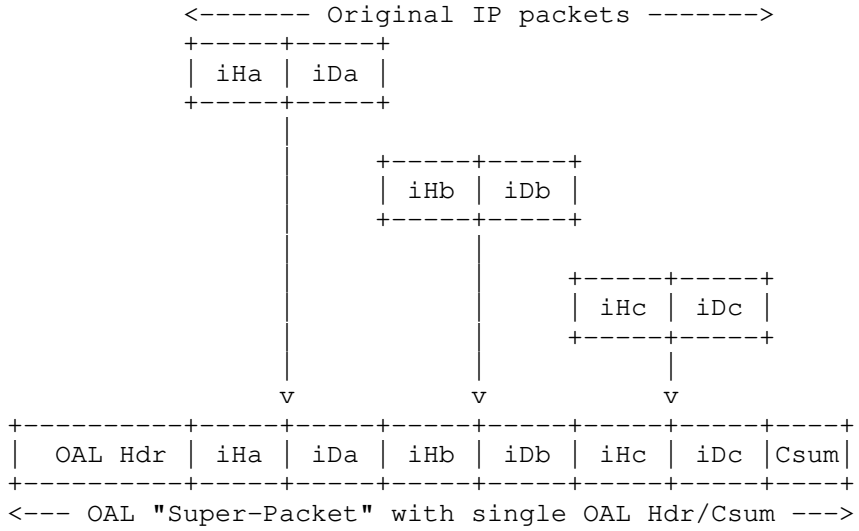


Figure 9: OAL Super-Packet Format

When the OAL source prepares a super-packet, it applies OAL fragmentation and *NET encapsulation then sends the carrier packets to the OAL destination. When the OAL destination receives the super-packet it reassembles if necessary, verifies and removes the trailing checksum, then regards the remaining OAL header Payload Length as the sum of the lengths of all payload packets. The OAL destination then selectively extracts each original IP packet (e.g., by setting pointers into the super-packet buffer and maintaining a reference count, by copying each packet into a separate buffer, etc.) and forwards each packet to the network layer. During extraction, the OAL determines the IP protocol version of each successive original IP

packet 'j' by examining the four most-significant bits of iH(j), and determines the length of the packet by examining the rest of iH(j) according to the IP protocol version.

Note that OMNI interfaces must take care to avoid processing super-packet payload elements that would subvert security. Specifically, if a super-packet contains a mix of data and control payload packets (which could include critical security codes), the node MUST NOT process the data packets before processing the control packets

7. Frame Format

The OMNI interface forwards original IP packets from the network layer by first invoking the OAL to create OAL packets/fragments if necessary, then including any *NET encapsulations and finally engaging the native frame format of the underlying interface. For example, for Ethernet-compatible interfaces the frame format is specified in [RFC2464], for aeronautical radio interfaces the frame format is specified in standards such as ICAO Doc 9776 (VDL Mode 2 Technical Manual), for various forms of tunnels the frame format is found in the appropriate tunneling specification, etc.

See Figure 2 for a map of the various *NET layering combinations possible. For any layering combination, the final layer (e.g., UDP, IP, Ethernet, etc.) must have an assigned number and frame format representation that is compatible with the selected underlying interface.

8. Link-Local Addresses (LLAs)

OMNI nodes are assigned OMNI interface IPv6 Link-Local Addresses (LLAs) through pre-service administrative actions. "MNP-LLAs" embed the MNP assigned to the mobile node, while "ADM-LLAs" include an administratively-unique ID that is guaranteed to be unique on the link. LLAs are configured as follows:

- IPv6 MNP-LLAs encode the most-significant 64 bits of a MNP within the least-significant 64 bits of the IPv6 link-local prefix fe80::/64, i.e., in the LLA "interface identifier" portion. The prefix length for the LLA is determined by adding 64 to the MNP prefix length. For example, for the MNP 2001:db8:1000:2000::/56 the corresponding MNP-LLA is fe80::2001:db8:1000:2000/120. Non-MNP routes are also represented the same as for MNP-LLAs, but include a GUA prefix that is not properly covered by the MSP.

- IPv4-compatible MNP-LLAs are constructed as fe80::ffff:[IPv4], i.e., the interface identifier consists of 16 '0' bits, followed by 16 '1' bits, followed by a 32bit IPv4 address/prefix. The

prefix length for the LLA is determined by adding 96 to the MNP prefix length. For example, the IPv4-Compatible MN OMNI LLA for 192.0.2.0/24 is fe80::ffff:192.0.2.0/120 (also written as fe80::ffff:c000:0200/120).

- o ADM-LLAs are assigned to ARs and MSEs and MUST be managed for uniqueness. The lower 32 bits of the LLA includes a unique integer "MSID" value between 0x00000001 and 0xfeffffff, e.g., as in fe80::1, fe80::2, fe80::3, etc., fe80::feffffff. The ADM-LLA prefix length is determined by adding 96 to the MSID prefix length. For example, if the prefix length for MSID 0x10012001 is 16 then the ADM-LLA prefix length is set to 112 and the LLA is written as fe80::1001:2001/112. The "zero" address for each ADM-LLA prefix is the Subnet-Router anycast address for that prefix [RFC4291]; for example, the Subnet-Router anycast address for fe80::1001:2001/112 is simply fe80::1001:2000. The MSID range 0xff000000 through 0xffffffff is reserved for future use.

Since the prefix 0000::/8 is "Reserved by the IETF" [RFC4291], no MNPs can be allocated from that block ensuring that there is no possibility for overlap between the different MNP- and ADM-LLA constructs discussed above.

Since MNP-LLAs are based on the distribution of administratively assured unique MNPs, and since ADM-LLAs are guaranteed unique through administrative assignment, OMNI interfaces set the autoconfiguration variable DupAddrDetectTransmits to 0 [RFC4862].

Note: If future protocol extensions relax the 64-bit boundary in IPv6 addressing, the additional prefix bits of an MNP could be encoded in bits 16 through 63 of the MNP-LLA. (The most-significant 64 bits would therefore still be in bits 64-127, and the remaining bits would appear in bits 16 through 48.) However, the analysis provided in [RFC7421] suggests that the 64-bit boundary will remain in the IPv6 architecture for the foreseeable future.

Note: Even though this document honors the 64-bit boundary in IPv6 addressing, it specifies prefix lengths longer than /64 for routing purposes. This effectively extends IPv6 routing determination into the interface identifier portion of the IPv6 address, but it does not redefine the 64-bit boundary. Modern routing protocol implementations honor IPv6 prefixes of all lengths, up to and including /128.

9. Unique-Local Addresses (ULAs)

OMNI domains use IPv6 Unique-Local Addresses (ULAs) as the source and destination addresses in OAL packet IPv6 encapsulation headers. ULAs are only routable within the scope of a an OMNI domain, and are derived from the IPv6 Unique Local Address prefix fc00::/7 followed by the L bit set to 1 (i.e., as fd00::/8) followed by a 40-bit pseudo-random Global ID to produce the prefix [ULA]::/48, which is then followed by a 16-bit Subnet ID then finally followed by a 64 bit Interface ID as specified in Section 3 of [RFC4193]. All nodes in the same OMNI domain configure the same 40-bit Global ID as the OMNI domain identifier. The statistic uniqueness of the 40-bit pseudo-random Global ID allows different OMNI domains to be joined together in the future without requiring renumbering.

Each OMNI link instance is identified by a value between 0x0000 and 0xfeff in bits 48-63 of [ULA]::/48; the values 0xff00 through 0xfffe are reserved for future use, and the value 0xffff denotes the presence of a Temporary ULA (see below). For example, OMNI ULAs associated with instance 0 are configured from the prefix [ULA]:0000::/64, instance 1 from [ULA]:0001::/64, instance 2 from [ULA]:0002::/64, etc. ULAs and their associated prefix lengths are configured in correspondence with LLAs through stateless prefix translation where "MNP-ULAs" are assigned in correspondence to MNP-LLAs and "ADM-ULAs" are assigned in correspondence to ADM-LLAs. For example, for OMNI link instance [ULA]:1010::/64:

- the MNP-ULA corresponding to the MNP-LLA fe80::2001:db8:1:2 with a 56-bit MNP length is derived by copying the lower 64 bits of the LLA into the lower 64 bits of the ULA as [ULA]:1010:2001:db8:1:2/120 (where, the ULA prefix length becomes 64 plus the IPv6 MNP length).

- the MNP-ULA corresponding to fe80::ffff:192.0.2.0 with a 28-bit MNP length is derived by simply writing the LLA interface ID into the lower 64 bits as [ULA]:1010:0:ffff:192.0.2.0/124 (where, the ULA prefix length is 64 plus 32 plus the IPv4 MNP length).

- the ADM-ULA corresponding to fe80::1000/112 is simply [ULA]:1010::1000/112.

- the ADM-ULA corresponding to fe80::/128 is simply [ULA]:1010::/128.

- etc.

Each OMNI interface assigns the Anycast ADM-ULA specific to the OMNI link instance. For example, the OMNI interface connected to instance

3 assigns the Anycast address [ULA]:0003::/128. Routers that configure OMNI interfaces advertise the OMNI service prefix (e.g., [ULA]:0003::/64) into the local routing system so that applications can direct traffic according to SBM requirements.

The ULA presents an IPv6 address format that is routable within the OMNI routing system and can be used to convey link-scoped IPv6 ND messages across multiple hops using IPv6 encapsulation [RFC2473]. The OMNI link extends across one or more underling Internetworks to include all ARs and MSEs. All MNs are also considered to be connected to the OMNI link, however OAL encapsulation is omitted whenever possible to conserve bandwidth (see: Section 14).

Each OMNI link can be subdivided into "segments" that often correspond to different administrative domains or physical partitions. OMNI nodes can use IPv6 Segment Routing [RFC8402] when necessary to support efficient forwarding to destinations located in other OMNI link segments. A full discussion of Segment Routing over the OMNI link appears in [I-D.templin-intarea-6706bis].

Temporary ULAs are constructed per [RFC8981] based on the prefix [ULA]:ffff::/64 and used by MNs when they have no other addresses. Temporary ULAs can be used for MN-to-MN communications outside the context of any supporting OMNI link infrastructure, and can also be used as an initial address while the MN is in the process of procuring an MNP. Temporary ULAs are not routable within the OMNI routing system, and are therefore useful only for OMNI link "edge" communications. Temporary ULAs employ optimistic DAD principles [RFC4429] since they are probabilistically unique.

Note: IPv6 ULAs taken from the prefix fc00::/7 followed by the L bit set to 0 (i.e., as fc00::/8) are never used for OMNI OAL addressing, however the range could be used for MSP and MNP addressing under certain limiting conditions (see: Section 10).

10. Global Unicast Addresses (GUAs)

OMNI domains use IP Global Unicast Address (GUA) prefixes [RFC4291] as Mobility Service Prefixes (MSPs) from which Mobile Network Prefixes (MNP) are delegated to Mobile Nodes (MNs). Fixed correspondent node networks reachable from the OMNI domain are represented by non-MNP GUA prefixes that are not derived from the MSP, but are treated in all other ways the same as for MNPs.

For IPv6, GUA prefixes are assigned by IANA [IPV6-GUA] and/or an associated regional assigned numbers authority such that the OMNI domain can be interconnected to the global IPv6 Internet without causing inconsistencies in the routing system. An OMNI domain could

instead use ULAs with the 'L' bit set to 0 (i.e., from the prefix fc00::/8)[RFC4193], however this would require IPv6 NAT if the domain were ever connected to the global IPv6 Internet.

For IPv4, GUA prefixes are assigned by IANA [IPV4-GUA] and/or an associated regional assigned numbers authority such that the OMNI domain can be interconnected to the global IPv4 Internet without causing routing inconsistencies. An OMNI domain could instead use private IPv4 prefixes (e.g., 10.0.0.0/8, etc.) [RFC3330], however this would require IPv4 NAT if the domain were ever connected to the global IPv4 Internet.

11. Node Identification

OMNI MNs and MSEs that connect over open Internetworks include a unique node identification value for themselves in the OMNI options of their IPv6 ND messages (see: Section 12.1.13). One useful identification value alternative is the Host Identity Tag (HIT) as specified in [RFC7401], while Hierarchical HITs (HHITs) [I-D.ietf-drip-rid] may provide a better alternative in certain domains such as the Unmanned (Air) Traffic Management (UTM) service for Unmanned Air Systems (UAS). Another alternative is the Universally Unique IDentifier (UUID) [RFC4122] which can be self-generated by a node without supporting infrastructure with very low probability of collision.

When a MN is truly outside the context of any infrastructure, it may have no MNP information at all. In that case, the MN can use an IPv6 temporary ULA or (H)HIT as an IPv6 source/destination address for sustained communications in Vehicle-to-Vehicle (V2V) and (multihop) Vehicle-to-Infrastructure (V2I) scenarios. The MN can also propagate the ULA/(H)HIT into the multihop routing tables of (collective) Mobile/Vehicular Ad-hoc Networks (MANETs/VANETs) using only the vehicles themselves as communications relays.

When a MN connects to ARs over (non-multihop) protected-spectrum ANETs, an alternate form of node identification (e.g., MAC address, serial number, airframe identification value, VIN, etc.) may be sufficient. The MN can then include OMNI "Node Identification" sub-options (see: Section 12.1.13) in IPv6 ND messages should the need to transmit identification information over the network arise.

12. Address Mapping - Unicast

OMNI interfaces maintain a neighbor cache for tracking per-neighbor state and use the link-local address format specified in Section 8. OMNI interface IPv6 Neighbor Discovery (ND) [RFC4861] messages sent over physical underlying interfaces without encapsulation observe the

native underlying interface Source/Target Link-Layer Address Option (S/TLLAO) format (e.g., for Ethernet the S/TLLAO is specified in [RFC2464]). OMNI interface IPv6 ND messages sent over underlying interfaces via encapsulation do not include S/TLLAOs which were intended for encoding physical L2 media address formats and not encapsulation IP addresses. Furthermore, S/TLLAOs are not intended for encoding additional interface attributes needed for multilink coordination. Hence, this document does not define an S/TLLAO format but instead defines a new option type termed the "OMNI option" designed for these purposes.

MNs such as aircraft typically have many wireless data link types (e.g. satellite-based, cellular, terrestrial, air-to-air directional, etc.) with diverse performance, cost and availability properties. The OMNI interface would therefore appear to have multiple L2 connections, and may include information for multiple underlying interfaces in a single IPv6 ND message exchange. OMNI interfaces use an IPv6 ND option called the OMNI option formatted as shown in Figure 10:

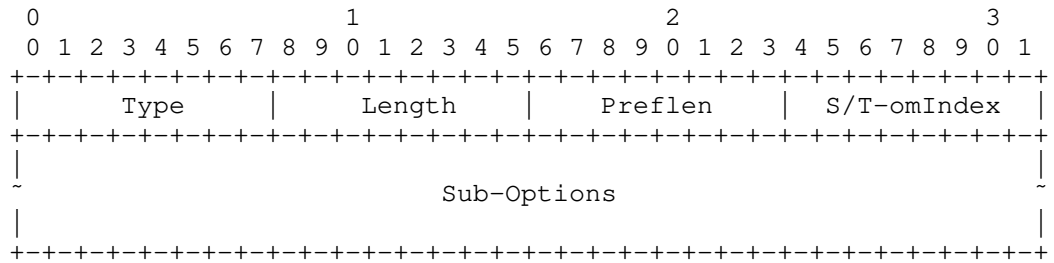


Figure 10: OMNI Option Format

In this format:

- Type is set to TBD2.
- Length is set to the number of 8 octet blocks in the option. The value 0 is invalid, while the values 1 through 255 (i.e., 8 through 2040 octets, respectively) indicate the total length of the OMNI option.
- Preflen is an 8 bit field that determines the length of prefix associated with an LLA. Values 0 through 128 specify a valid prefix length (all other values are invalid). For IPv6 ND messages sent from a MN to the MS, Preflen applies to the IPv6 source LLA and provides the length that the MN is requesting or asserting to the MS. For IPv6 ND messages sent from the MS to the MN, Preflen applies to the IPv6 destination LLA and indicates the

length that the MS is granting to the MN. For IPv6 ND messages sent between MS endpoints, Preflen provides the length associated with the source/target MN that is subject of the ND message.

- S/T-omIndex is an 8 bit field corresponds to the omIndex value for source or target underlying interface used to convey this IPv6 ND message. OMNI interfaces MUST number each distinct underlying interface with an omIndex value between '1' and '255' that represents a MN-specific 8-bit mapping for the actual ifIndex value assigned by network management [RFC2863] (the omIndex value '0' is reserved for use by the MS). For RS and NS messages, S/T-omIndex corresponds to the source underlying interface the message originated from. For RA and NA messages, S/T-omIndex corresponds to the target underlying interface that the message is destined to. (For NS messages used for Neighbor Unreachability Detection (NUD), S/T-omIndex instead identifies the neighbor's underlying interface to be used as the target interface to return the NA.)

- Sub-Options is a Variable-length field, of length such that the complete OMNI Option is an integer multiple of 8 octets long. Contains one or more Sub-Options, as described in Section 12.1.

The OMNI option may appear in any IPv6 ND message type; it is processed by interfaces that recognize the option and ignored by all other interfaces. If multiple OMNI option instances appear in the same IPv6 ND message, the interface processes the Preflen and S/T-omIndex fields in the first instance and ignores those fields in all other instances. The interface processes the Sub-Options of all OMNI option instances in the same IPv6 ND message in the consecutive order in which they appear.

The OMNI option(s) in each IPv6 ND message may include full or partial information for the neighbor. The union of the information in the most recently received OMNI options is therefore retained, and the information is aged/removed in conjunction with the corresponding neighbor cache entry.

12.1. Sub-Options

Each OMNI option includes zero or more Sub-Options. Each consecutive Sub-Option is concatenated immediately after its predecessor. All Sub-Options except Pad1 (see below) are in type-length-value (TLV) encoded in the following format:

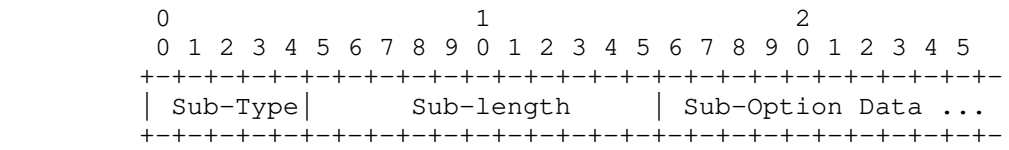


Figure 11: Sub-Option Format

- Sub-Type is a 5-bit field that encodes the Sub-Option type. Sub-Options defined in this document are:

Sub-Option Name	Sub-Type
Pad1	0
PadN	1
Interface Attributes (Type 1)	2
Interface Attributes (Type 2)	3
Traffic Selector	4
MS-Register	5
MS-Release	6
Geo Coordinates	7
DHCPv6 Message	8
HIP Message	9
Reassembly Limit	10
Fragmentation Report	11
Node Identification	12
Sub-Type Extension	30

Figure 12

  Sub-Types 13-29 are available for future assignment for major protocol functions. Sub-Type 31 is reserved by IANA.

- Sub-Length is an 11-bit field that encodes the length of the Sub-Option Data ranging from 0 to 2034 octets.

- Sub-Option Data is a block of data with format determined by Sub-Type and length determined by Sub-Length.

During transmission, the OMNI interface codes Sub-Type and Sub-Length together in network byte order in 2 consecutive octets, where Sub-Option Data may be up to 2034 octets in length. This allows ample space for coding large objects (e.g., ASCII strings, domain names, protocol messages, security codes, etc.), while a single OMNI option is limited to 2040 octets the same as for any IPv6 ND option. If the Sub-Options to be coded would cause an OMNI option to exceed 2040 octets, the OMNI interface codes any remaining Sub-Options in additional OMNI option instances in the intended order of processing in the same IPv6 ND message. Implementations must therefore observe

size limitations, and must refrain from sending IPv6 ND messages larger than the OMNI interface MTU. If the available OMNI information would cause a single IPv6 ND message to exceed the OMNI interface MTU, the OMNI interface codes as much as possible in a first IPv6 ND message and codes the remainder in additional IPv6 ND messages.

During reception, the OMNI interface processes each OMNI option Sub-Option while skipping over and ignoring any unrecognized Sub-Options. The OMNI interface processes the Sub-Options of all OMNI option instances in the consecutive order in which they appear in the IPv6 ND message, beginning with the first instance and continuing through any additional instances to the end of the message. If a Sub-Option length would cause processing to exceed the OMNI option total length, the OMNI interface accepts any Sub-Options already processed and ignores the final Sub-Option. The interface then processes any remaining OMNI options in the same fashion to the end of the IPv6 ND message.

Note: large objects that exceed the Sub-Option Data limit of 2034 octets are not supported under the current specification; if this proves to be limiting in practice, future specifications may define support for fragmenting large objects across multiple OMNI options within the same IPv6 ND message.

The following Sub-Option types and formats are defined in this document:

12.1.1. Pad1

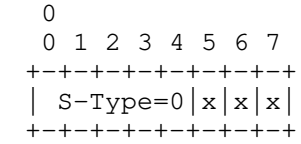


Figure 13: Pad1

- Sub-Type is set to 0. If multiple instances appear in OMNI options of the same message all are processed.

- Sub-Type is followed by 3 'x' bits, set to any value on transmission (typically all-zeros) and ignored on receipt. Pad1 therefore consists of 1 octet with the most significant 5 bits set to 0, and with no Sub-Length or Sub-Option Data fields following.

12.1.2. PadN

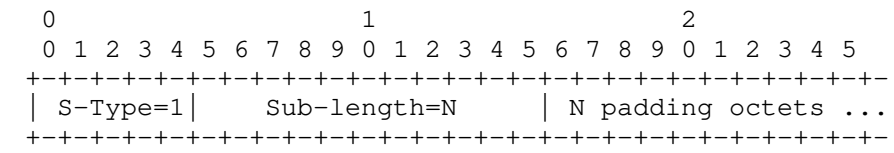


Figure 14: PadN

- Sub-Type is set to 1. If multiple instances appear in OMNI options of the same message all are processed.

- Sub-Length is set to N (from 0 to 2034) that encodes the number of padding octets that follow.

- Sub-Option Data consists of N octets, set to any value on transmission (typically all-zeros) and ignored on receipt.

12.1.3. Interface Attributes (Type 1)

The Interface Attributes (Type 1) sub-option provides a basic set of attributes for underlying interfaces. Interface Attributes (Type 1) is deprecated throughout the rest of this specification, and Interface Attributes (Type 2) (see: Section 12.1.4) are indicated wherever the term "Interface Attributes" appears without an associated Type designation.

Nodes SHOULD NOT include Interface Attributes (Type 1) sub-options in IPv6 ND messages they send, and MUST ignore any in IPv6 ND messages they receive. If an Interface Attributes (Type 1) is included, it must have the following format:

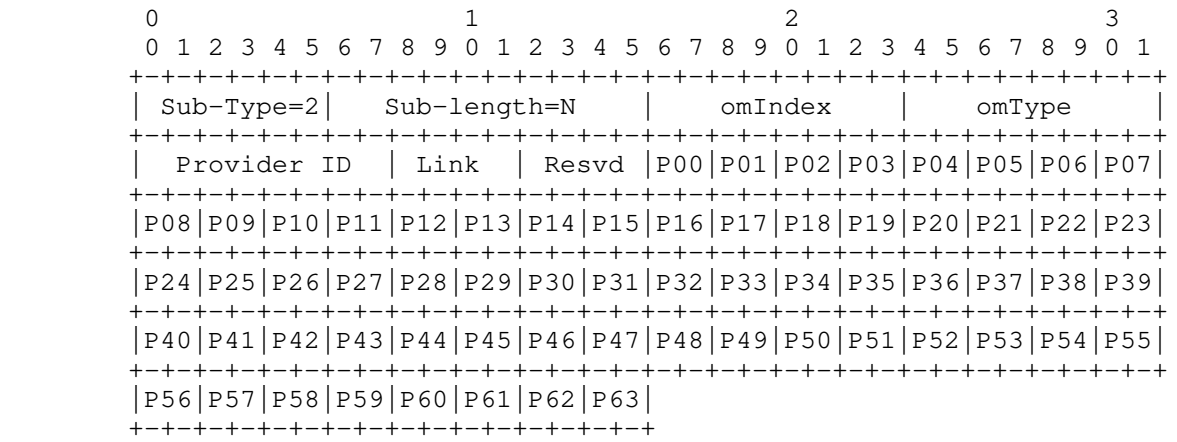


Figure 15: Interface Attributes (Type 1)

- Sub-Type is set to 2. If multiple instances with different omIndex values appear in OMNI option of the same message all are processed; if multiple instances with the same omIndex value appear, the first is processed and all others are ignored

- Sub-Length is set to N (from 4 to 2034) that encodes the number of Sub-Option Data octets that follow.

- omIndex is a 1-octet field containing a value from 0 to 255 identifying the underlying interface for which the attributes apply.

- omType is a 1-octet field containing a value from 0 to 255 corresponding to the underlying interface identified by omIndex.

- Provider ID is a 1-octet field containing a value from 0 to 255 corresponding to the underlying interface identified by omIndex.

- Link encodes a 4-bit link metric. The value '0' means the link is DOWN, and the remaining values mean the link is UP with metric ranging from '1' ("lowest") to '15' ("highest").

- Resvd is reserved for future use. Set to 0 on transmission and ignored on reception.

- A 16-octet ""Preferences" field immediately follows 'Resvd', with values P[00] through P[63] corresponding to the 64 Differentiated Service Code Point (DSCP) values [RFC2474]. Each 2-bit P[*] field is set to the value '0' ("disabled"), '1' ("low"), '2' ("medium")

or '3' ("high") to indicate a QoS preference for underlying interface selection purposes.

12.1.4. Interface Attributes (Type 2)

The Interface Attributes (Type 2) sub-option provides L2 forwarding information for the multilink conceptual sending algorithm discussed in Section 14. The L2 information is used for selecting among potentially multiple candidate underlying interfaces that can be used to forward carrier packets to the neighbor based on factors such as DSCP preferences and link quality. Interface Attributes (Type 2) further includes link-layer address information to be used for either OAL encapsulation or direct UDP/IP encapsulation (when OAL encapsulation can be avoided).

Interface Attributes (Type 2) are the sole Interface Attributes format in this specification that all OMNI nodes must honor. Wherever the term "Interface Attributes" occurs throughout this specification without a "Type" designation, the format given below is indicated:

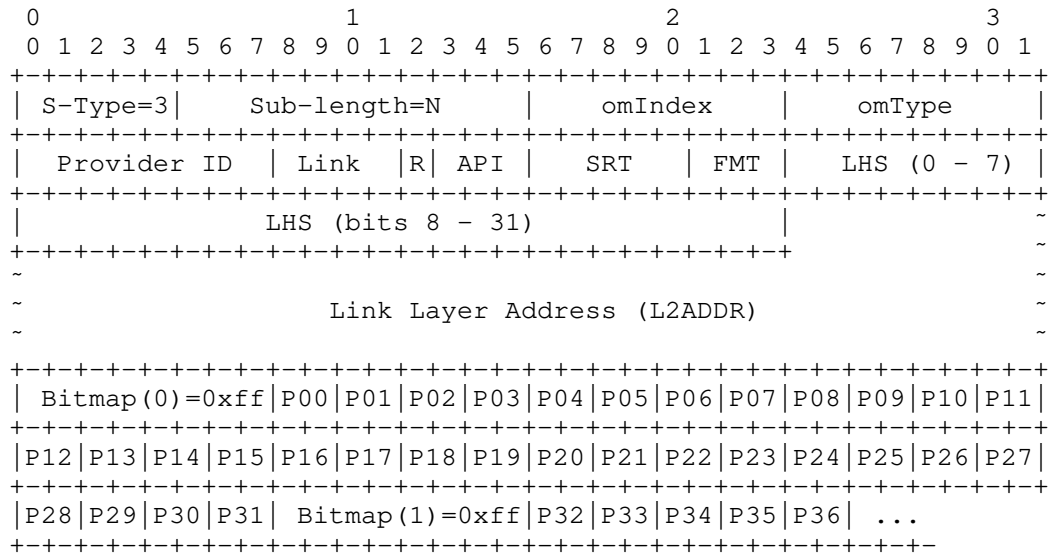


Figure 16: Interface Attributes (Type 2)

- Sub-Type is set to 3. If multiple instances with different omIndex values appear in OMNI options of the same message all are processed; if multiple instances with the same omIndex value appear, the first is processed and all others are ignored.

- Sub-Length is set to N (from 4 to 2034) that encodes the number of Sub-Option Data octets that follow. The 'omIndex', 'omType', 'Provider ID', 'Link', 'R' and 'API' fields are always present; hence, the remainder of the Sub-Option Data is limited to 2030 octets.

- Sub-Option Data contains an "Interface Attributes (Type 2)" option encoded as follows:

  * omIndex is set to an 8-bit integer value corresponding to a specific underlying interface the same as specified above for the OMNI option S/T-omIndex field. The OMNI options of a same message may include multiple Interface Attributes Sub-Options, with each distinct omIndex value pertaining to a different underlying interface. The OMNI option will often include an Interface Attributes Sub-Option with the same omIndex value that appears in the S/T-omIndex. In that case, the actual encapsulation address of the received IPv6 ND message should be compared with the L2ADDR encoded in the Sub-Option (see below); if the addresses are different (or, if L2ADDR is absent) the presence of a NAT is assumed.

  * omType is set to an 8-bit integer value corresponding to the underlying interface identified by omIndex. The value represents an OMNI interface-specific 8-bit mapping for the actual IANA ifType value registered in the 'IANAifType-MIB' registry [http://www.iana.org].

  * Provider ID is set to an OMNI interface-specific 8-bit ID value for the network service provider associated with this omIndex.

  * Link encodes a 4-bit link metric. The value '0' means the link is DOWN, and the remaining values mean the link is UP with metric ranging from '1' ("lowest") to '15' ("highest").

  * R is reserved for future use.

  * API - a 3-bit "Address/Preferences/Indexed" code that determines the contents of the remainder of the sub-option as follows:

    + When the most significant bit (i.e., "Address") is set to 1, the SRT, FMT, LHS and L2ADDR fields are included immediately following the API code; else, they are omitted.

    + When the next most significant bit (i.e., "Preferences") is set to 1, a preferences block is included next; else, it is omitted. (Note that if "Address" is set the preferences

block immediately follows L2ADDR; else, it immediately follows the API code.)

+ When a preferences block is present and the least significant bit (i.e., "Indexed") is set to 0, the block is encoded in "Simplex" form as shown in Figure 15; else it is encoded in "Indexed" form as discussed below.

* When API indicates that an "Address" is included, the following fields appear in consecutive order (else, they are omitted):

  + SRT - a 5-bit Segment Routing Topology prefix length value that (when added to 96) determines the prefix length to apply to the ULA formed from concatenating [ULA*]::/96 with the 32 bit LHS MSID value that follows. For example, the value 16 corresponds to the prefix length 112.

  + FMT - a 3-bit "Framework/Mode/Type" code corresponding to the included Link Layer Address as follows:

    - When the most significant bit (i.e., "Framework") is set to 1, L2ADDR is the INET encapsulation address for the Source/Target Client itself; otherwise L2ADDR is the address of the Proxy/Server named in the LHS.

    - When the next most significant bit (i.e., "Mode") is set to 1, the Framework node is (likely) located behind an INET Network Address Translator (NAT); otherwise, it is on the open INET.

    - When the least significant bit (i.e., "Type") is set to 0, L2ADDR includes a UDP Port Number followed by an IPv4 address; otherwise, it includes a UDP Port Number followed by an IPv6 address.

  + LHS - the 32 bit MSID of the Last Hop Proxy/Server on the path to the target. When SRT and LHS are both set to 0, the LHS is considered unspecified in this IPv6 ND message. When SRT is set to 0 and LHS is non-zero, the prefix length is set to 128. SRT and LHS together provide guidance to the OMNI interface forwarding algorithm. Specifically, if SRT/LHS is located in the local OMNI link segment then the OMNI interface can encapsulate according to FMT/L2ADDR (following any necessary NAT traversal messaging); else, it must forward according to the OMNI link spanning tree. See [I-D.templin-intarea-6706bis] for further discussion.

+ Link Layer Address (L2ADDR) - Formatted according to FMT, and identifies the link-layer address (i.e., the encapsulation address) of the source/target. The UDP Port Number appears in the first 2 octets and the IP address appears in the next 4 octets for IPv4 or 16 octets for IPv6. 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 FMT/L2ADDR to determine the encapsulation address for forwarding when SRT/LHS is located in the local OMNI link segment. Note that if the target is behind a NAT, L2ADDR will contain the mapped INET address stored in the NAT; otherwise, L2ADDR will contain the native INET information of the target itself.

* When API indicates that "Preferences" are included, a preferences block appears as the remainder of the Sub-Option as a series of Bitmaps and P[*] values. In "Simplex" form, the index for each singleton Bitmap octet is inferred from its sequential position (i.e., 0, 1, 2, ...) as shown in Figure 16. In "Indexed" form, each Bitmap is preceded by an Index octet that encodes a value "i" = (0 - 255) as the index for its companion Bitmap as follows:

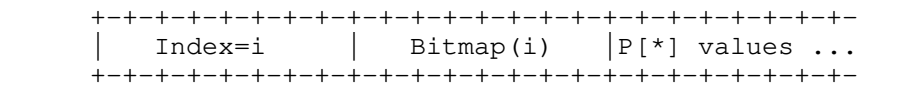


Figure 17

* The preferences consist of a first (simplex/indexed) Bitmap (i.e., "Bitmap(i)") followed by 0-8 single-octet blocks of 2-bit P[*] values, followed by a second Bitmap (i), followed by 0-8 blocks of P[*] values, etc. Reading from bit 0 to bit 7, the bits of each Bitmap(i) that are set to '1'' indicate the P[*] blocks from the range P[(i*32)] through P[(i*32) + 31] that follow; if any Bitmap(i) bits are '0', then the corresponding P[*] block is instead omitted. For example, if Bitmap(0) contains 0xff then the block with P[00]-P[03], followed by the block with P[04]-P[07], etc., and ending with the block with P[28]-P[31] are included (as shown in Figure 15). The next Bitmap(i) is then consulted with its bits indicating which P[*] blocks follow, etc. out to the end of the Sub-Option.

* Each 2-bit P[*] field is set to the value '0' ("disabled"), '1' ("low"), '2' ("medium") or '3' ("high") to indicate a QoS preference for underlying interface selection purposes. Not

all P[*] values need to be included in the OMNI option of each IPv6 ND message received. Any P[*] values represented in an earlier OMNI option but omitted in the current OMNI option remain unchanged. Any P[*] values not yet represented in any OMNI option default to "medium".

* The first 16 P[*] blocks correspond to the 64 Differentiated Service Code Point (DSCP) values P[00] - P[63] [RFC2474]. Any additional P[*] blocks that follow correspond to "pseudo-DSCP" traffic classifier values P[64], P[65], P[66], etc. See Appendix A for further discussion and examples.

12.1.5. Traffic Selector

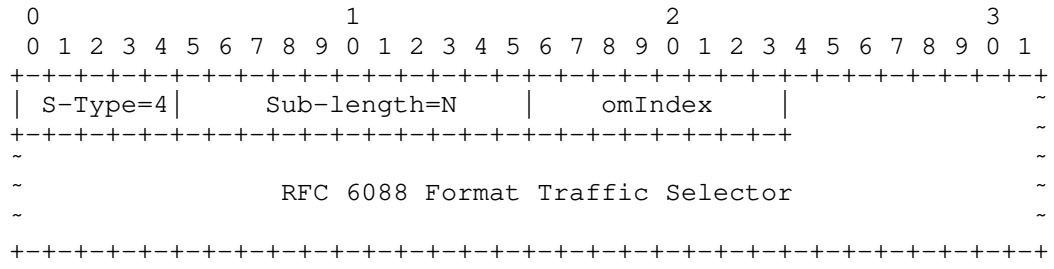


Figure 18: Traffic Selector

- Sub-Type is set to 4. If multiple instances appear in OMNI options of the same message all are processed, i.e., even if the same omIndex value appears multiple times.

- Sub-Length is set to N (from 1 to 2034) that encodes the number of Sub-Option Data octets that follow.

- Sub-Option Data contains a 1 octet omIndex encoded exactly as specified in Section 12.1.3, followed by an N-1 octet traffic selector formatted per [RFC6088] beginning with the "TS Format" field. The largest traffic selector for a given omIndex is therefore 2033 octets.

12.1.6. MS-Register

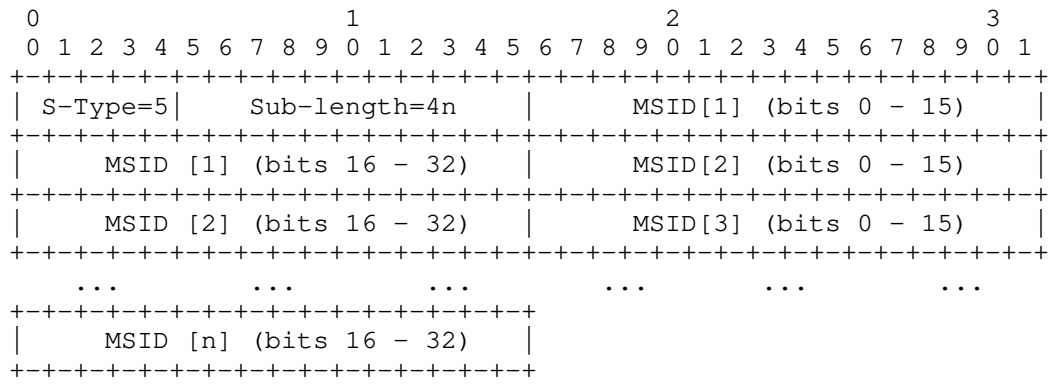


Figure 19: MS-Register Sub-option

- Sub-Type is set to 5. If multiple instances appear in OMNI options of the same message all are processed. Only the first MAX_MSID values processed (whether in a single instance or multiple) are retained and all other MSIDs are ignored.

- Sub-Length is set to 4n, with 508 as the maximum value for n. The length of the Sub-Option Data section is therefore limited to 2032 octets.

- A list of n 4 octet MSIDs is included in the following 4n octets. The Anycast MSID value '0' in an RS message MS-Register sub-option requests the recipient to return the MSID of a nearby MSE in a corresponding RA response.

12.1.7. MS-Release

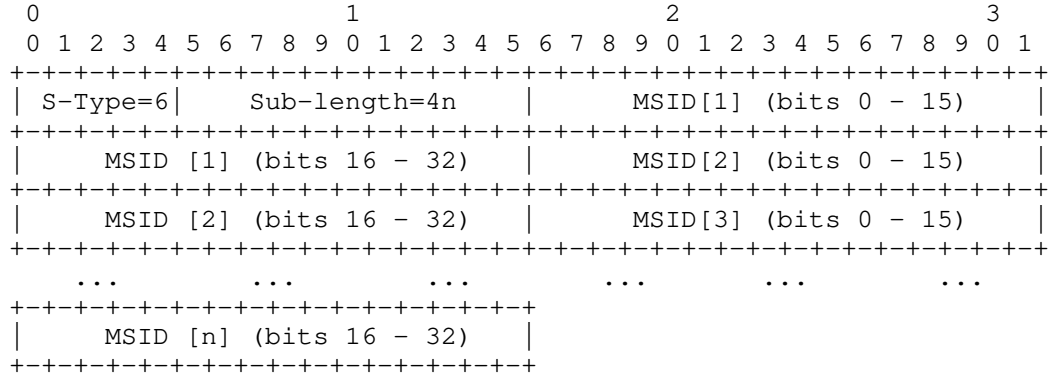


Figure 20: MS-Release Sub-option

- Sub-Type is set to 6. If multiple instances appear in OMNI options of the same message all are processed. Only the first MAX_MSID values processed (whether in a single instance or multiple) are retained and all other MSIDs are ignored.

- Sub-Length is set to 4n, with 508 as the maximum value for n. The length of the Sub-Option Data section is therefore limited to 2032 octets.

- A list of n 4 octet MSIDs is included in the following 4n octets. The Anycast MSID value '0' is ignored in MS-Release sub-options, i.e., only non-zero values are processed.

12.1.8. Geo Coordinates

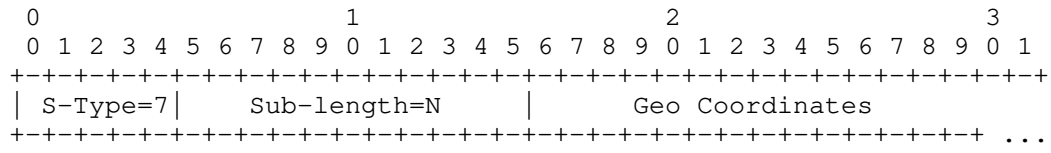


Figure 21: Geo Coordinates Sub-option

- Sub-Type is set to 7. If multiple instances appear in OMNI options of the same message the first is processed and all others are ignored.

- Sub-Length is set to N (from 0 to 2034) that encodes the number of Sub-Option Data octets that follow.

- A set of Geo Coordinates of maximum length 2034 octets. Format(s) to be specified in future documents; should include Latitude/ Longitude, plus any additional attributes such as altitude, heading, speed, etc.

12.1.9. Dynamic Host Configuration Protocol for IPv6 (DHCPv6) Message

The Dynamic Host Configuration Protocol for IPv6 (DHCPv6) sub-option may be included in the OMNI options of RS messages sent by MNs and RA messages returned by MSEs. ARs that act as proxys to forward RS/RA messages between MNs and MSEs also forward DHCPv6 sub-options unchanged and do not process DHCPv6 sub-options themselves. Note that DHCPv6 message sub-option integrity is protected by the Checksum included in the IPv6 ND message header.

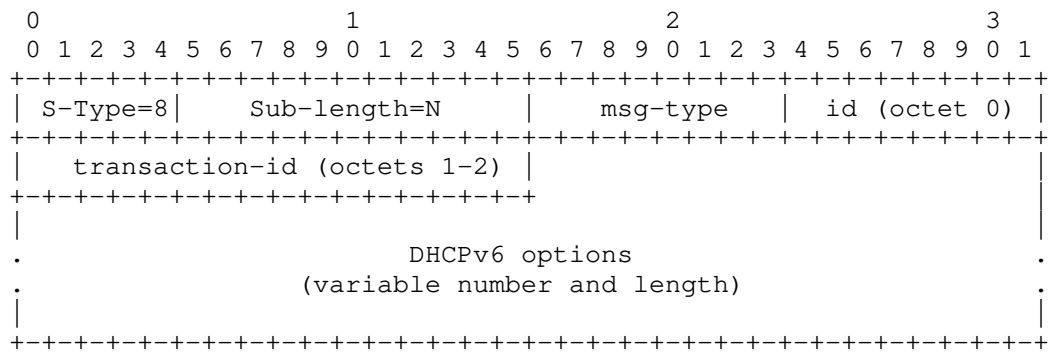


Figure 22: DHCPv6 Message Sub-option

- Sub-Type is set to 8. If multiple instances appear in OMNI options of the same message the first is processed and all others are ignored.

- Sub-Length is set to N (from 4 to 2034) that encodes the number of Sub-Option Data octets that follow. The 'msg-type' and 'transaction-id' fields are always present; hence, the length of the DHCPv6 options is restricted to 2030 octets.

- 'msg-type' and 'transaction-id' are coded according to Section 8 of [RFC8415].

- A set of DHCPv6 options coded according to Section 21 of [RFC8415] follows.

12.1.10. Host Identity Protocol (HIP) Message

The Host Identity Protocol (HIP) Message sub-option may be included in the OMNI options of RS messages sent by MNs and RA messages returned by ARs. ARs that act as proxys authenticate and remove HIP messages in RS messages they forward from a MN to an MSE. ARs that act as proxys insert and sign HIP messages in the RA messages they forward from an MSE to a MN.

The HIP message sub-option may also be included in any IPv6 ND message that may traverse an open Internetwork, i.e., where link-layer authentication is not already assured by lower layers.

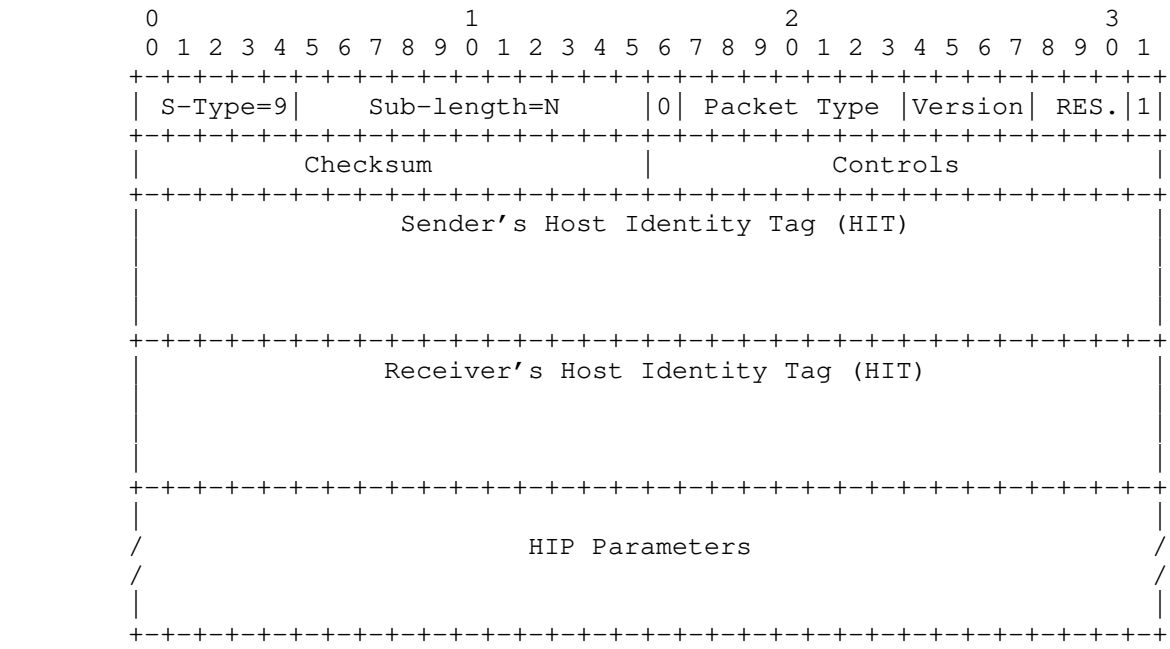


Figure 23: HIP Message Sub-option

- Sub-Type is set to 9. If multiple instances appear in OMNI options of the same message the first is processed and all others are ignored.

- Sub-Length is set to N, i.e., the length of the option in octets beginning immediately following the Sub-Length field and extending to the end of the HIP parameters. The length of the entire HIP message is therefore restricted to 2034 octets.

- The HIP message is coded exactly as specified in Section 5 of [RFC7401], except that the OMNI "Sub-Type" and "Sub-Length" fields replace the first 2 octets of the HIP message header (i.e., the Next Header and Header Length fields). Note that, since the IPv6 ND message header already includes a Checksum, the HIP message Checksum field is set to 0 on transmission and ignored on reception. (The Checksum field is still included to retain the [RFC7401] message format.)

12.1.11. Reassembly Limit

The Reassembly Limit sub-option may be included in the OMNI options of IPv6 ND messages. The message consists of a 14-bit Reassembly Limit value, followed by two flag bits (H, L) optionally followed by

an (N-2)-octet leading portion of an OAL First Fragment that triggered the message.

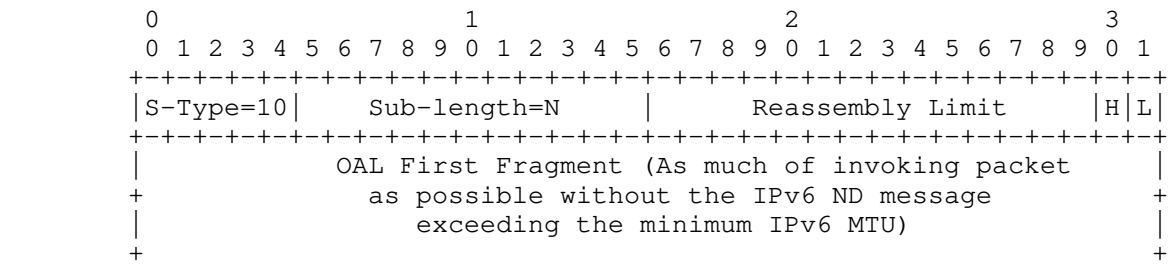


Figure 24: Reassembly Limit

- Sub-Type is set to 10. If multiple instances appear in OMNI options of the same message the first occurring "hard" and "soft" Reassembly Limit values are accepted, and any additional Reassembly Limit values are ignored.

- Sub-Length is set to 2 if no OAL First Fragment is included, or to a value N greater than 2 if an OAL First Fragment is included.

- A 14-bit Reassembly Limit follows, and includes a value between 1500 and 9180. If any other value is included, the sub-option is ignored. The value indicates the hard or soft limit for original IP packets that the source of the message is currently willing to reassemble; the source may increase or decrease the hard or soft limit at any time through the transmission of new IPv6 ND messages. Until the first IPv6 ND message with a Reassembly Limit sub-option arrives, OMNI nodes assume initial default hard/soft limits of 9180 bytes (I.e., the OMNI interface MRU). After IPv6 ND messages with Reassembly Limit sub-options arrive, the OMNI node retains the most recent hard/soft limit values until new IPv6 ND messages with different values arrive.

- The 'H' flag is set to 1 if the Reassembly Limit is a "Hard" limit, and set to 0 if the Reassembly Limit is a "Soft" limit.

- The 'L' flag is set to 1 if an OAL First Fragment corresponding to a reassembly loss event was included; otherwise set to 0.

- If N is greater than 2, the remainder of the Reassembly Limit sub-option encodes the leading portion of an OAL First Fragment that prompted this IPv6 ND message. The first fragment is included beginning with the OAL IPv6 header, and continuing with as much of the fragment payload as possible without causing the IPv6 ND message to exceed the minimum IPv6 MTU. (Note that only the OAL

First Fragment is consulted regardless of its size, and without waiting for additional fragments.)

12.1.12. Fragmentation Report

The Fragmentation Report may be included in the OMNI options of uNA messages sent from an OAL destination to an OAL source. The message consists of (N / 8)-many (Identification, Bitmap)-tuples which include the Identification values of OAL fragments received plus a Bitmap marking the ordinal positions of individual fragments received and fragments missing.

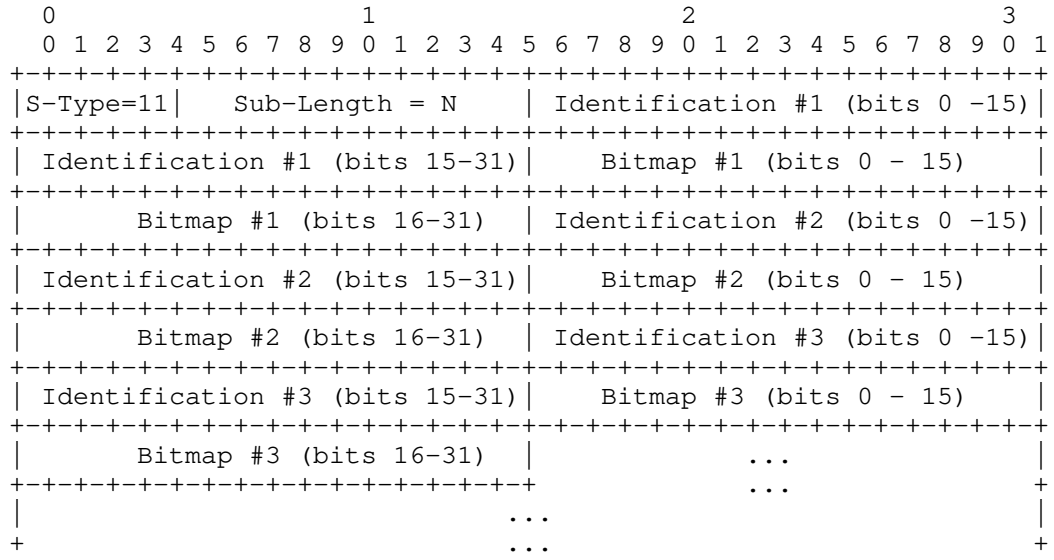


Figure 25: Fragmentation Report

- Sub-Type is set to 11. If multiple instances appear in OMNI options of the same message all are processed.

- Sub-Length is set to N, i.e., the length of the option in octets beginning immediately following the Sub-Length field and extending to the end of the ICMPv6 error message body. N must be an integral multiple of 8 octets; otherwise, the sub-option is ignored. The length of the entire sub-option should not cause the entire IPv6 ND message to exceed the minimum MPS.

- Identification (i) includes the IPv6 Identification value found in the Fragment Header of a received OAL fragment. (Only those Identification values included represent fragments for which loss was unambiguously observed; any Identification values not included

correspond to fragments that were either received in their entirety or are still in transit.)

- Bitmap (i) includes an ordinal checklist of fragments, with each bit set to 1 for a fragment received or 0 for a fragment missing. For example, for a 20-fragment fragmented OAL packet with ordinal fragments #3, #10, #13 and #17 missing and all other fragments received, the bitmap would encode:

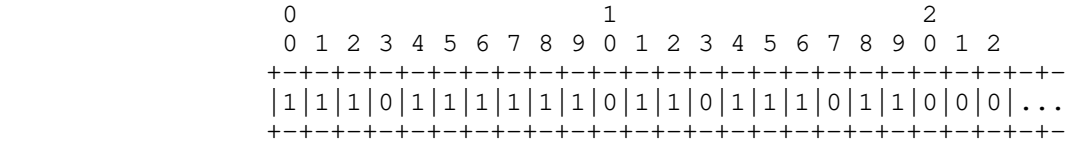


Figure 26

(Note that loss of an OAL atomic fragment is indicated by a Bitmap(i) with all bits set to 0.)

12.1.13. Node Identification

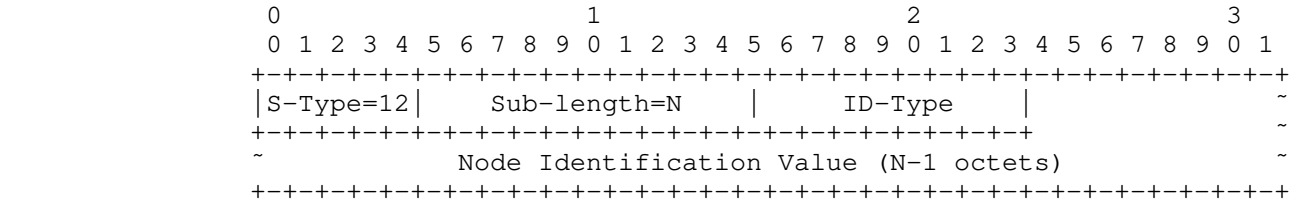


Figure 27: Node Identification

- Sub-Type is set to 12. If multiple instances appear in OMNI options of the same IPv6 ND message the first instance of a specific ID-Type is processed and all other instances of the same ID-Type are ignored. (Note therefore that it is possible for a single IPv6 ND message to convey multiple Node Identifications - each having a different ID-Type.)

- Sub-Length is set to N (from 1 to 2034) that encodes the number of Sub-Option Data octets that follow. The ID-Type field is always present; hence, the maximum Node Identification Value length is 2033 octets.

- ID-Type is a 1 octet field that encodes the type of the Node Identification Value. The following ID-Type values are currently defined:

* 0 - Universally Unique IDentifier (UUID) [RFC4122]. Indicates that Node Identification Value contains a 16 octet UUID.

* 1 - Host Identity Tag (HIT) [RFC7401]. Indicates that Node Identification Value contains a 16 octet HIT.

* 2 - Hierarchical HIT (HHIT) [I-D.ietf-drip-rid]. Indicates that Node Identification Value contains a 16 octet HHIT.

* 3 - Network Access Identifier (NAI) [RFC7542]. Indicates that Node Identification Value contains an N-1 octet NAI.

* 4 - Fully-Qualified Domain Name (FQDN) [RFC1035]. Indicates that Node Identification Value contains an N-1 octet FQDN.

* 5 - 252 - Unassigned.

* 253-254 - Reserved for experimentation, as recommended in [RFC3692].

* 255 - reserved by IANA.

o Node Identification Value is an (N - 1) octet field encoded according to the appropriate the "ID-Type" reference above.

When a Node Identification Value is needed for DHCPv6 messaging purposes, it is encoded as a DHCP Unique IDentifier (DUID) using the "DUID-EN for OMNI" format with enterprise number 45282 (see: Section 25) as shown in Figure 28:

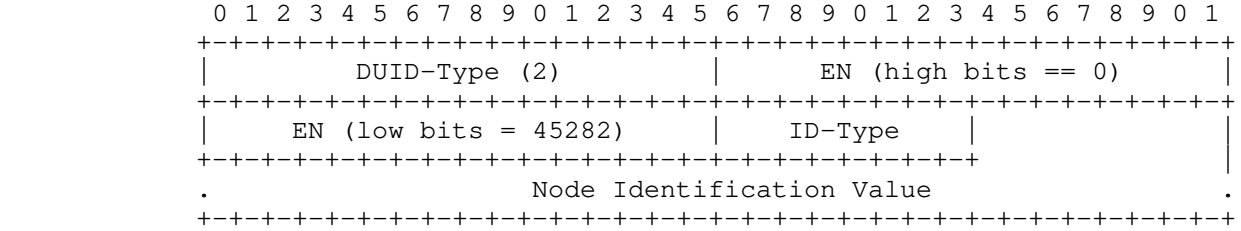


Figure 28: DUID-EN for OMNI Format

In this format, the ID-Type and Node Identification Value fields are coded exactly as in Figure 27 following the 6 octet DUID-EN header, and the entire "DUID-EN for OMNI" is included in a DHCPv6 message per [RFC8415].

12.1.14. Sub-Type Extension

Since the Sub-Type field is only 5 bits in length, future specifications of major protocol functions may exhaust the remaining Sub-Type values available for assignment. This document therefore defines Sub-Type 30 as an "extension", meaning that the actual sub-option type is determined by examining a 1 octet "Extension-Type" field immediately following the Sub-Length field. The Sub-Type Extension is formatted as shown in Figure 29:

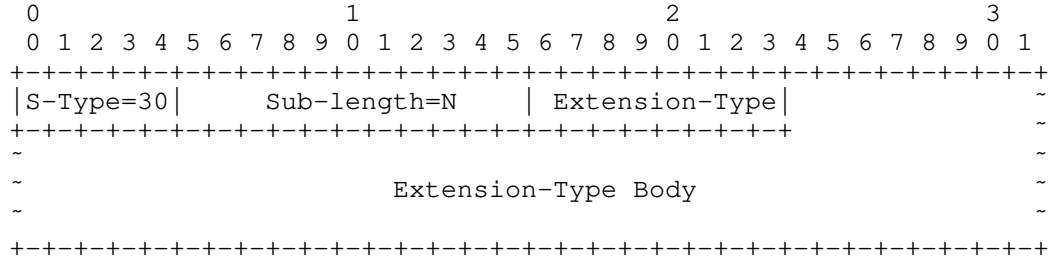


Figure 29: Sub-Type Extension

- Sub-Type is set to 30. If multiple instances appear in OMNI options of the same message all are processed, where each individual extension defines its own policy for processing multiple of that type.

- Sub-Length is set to N (from 1 to 2034) that encodes the number of Sub-Option Data octets that follow. The Extension-Type field is always present; hence, the maximum Extension-Type Body length is 2033 octets.

- Extension-Type contains a 1 octet Sub-Type Extension value between 0 and 255.

- Extension-Type Body contains an N-1 octet block with format defined by the given extension specification.

Extension-Type values 2 through 252 are available for assignment by future specifications, which must also define the format of the Extension-Type Body and its processing rules. Extension-Type values 253 and 254 are reserved for experimentation, as recommended in [RFC3692], and value 255 is reserved by IANA. Extension-Type values 0 and 1 are defined in the following subsections:

12.1.14.1. RFC4380 UDP/IP Header Option

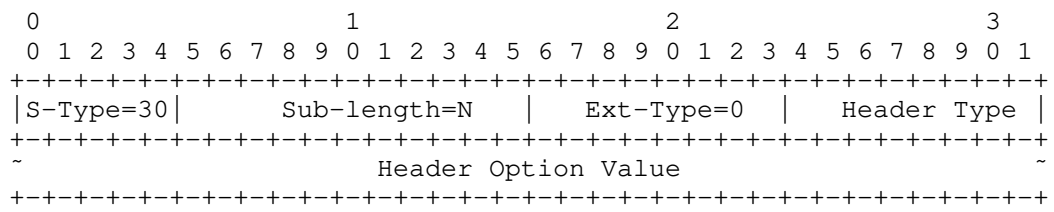


Figure 30: RFC4380 UDP/IP Header Option (Extension-Type 0)

- Sub-Type is set to 30.
- Sub-Length is set to N (from 2 to 2034) that encodes the number of Sub-Option Data octets that follow. The Extension-Type and Header Type fields are always present; hence, the maximum-length Header Option Value is 2032 octets.
- Extension-Type is set to 0. Each instance encodes exactly one header option per Section 5.1.1 of [RFC4380], with the leading '0' octet omitted and the following octet coded as Header Type. If multiple instances of the same Header Type appear in OMNI options of the same message the first instance is processed and all others are ignored.
- Header Type and Header Option Value are coded exactly as specified in Section 5.1.1 of [RFC4380]; the following types are currently defined:
  - 0 - Origin Indication (IPv4) - value coded per Section 5.1.1 of [RFC4380].
  - 1 - Authentication Encapsulation - value coded per Section 5.1.1 of [RFC4380].
  - 2 - Origin Indication (IPv6) - value coded per Section 5.1.1 of [RFC4380], except that the address is a 16-octet IPv6 address instead of a 4-octet IPv4 address.
- Header Type values 3 through 252 are available for assignment by future specifications, which must also define the format of the Header Option Value and its processing rules. Header Type values 253 and 254 are reserved for experimentation, as recommended in [RFC3692], and value 255 is Reserved by IANA.

12.1.14.2. RFC6081 UDP/IP Trailer Option

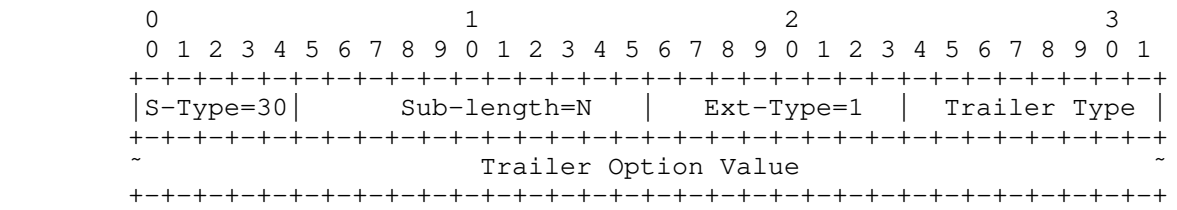


Figure 31: RFC6081 UDP/IP Trailer Option (Extension-Type 1)

- Sub-Type is set to 30.
- Sub-Length is set to N (from 2 to 2034) that encodes the number of Sub-Option Data octets that follow. The Extension-Type and Trailer Type fields are always present; hence, the maximum-length Trailer Option Value is 2032 octets.
- Extension-Type is set to 1. Each instance encodes exactly one trailer option per Section 4 of [RFC6081]. If multiple instances of the same trailer type appear in OMNI options of the same message the first instance is processed and all others ignored.
- Trailer Type and Trailer Option Value are coded exactly as specified in Section 4 of [RFC6081]; the following Trailer Types are currently defined:
  - 0 - Unassigned
  - 1 - Nonce Trailer - value coded per Section 4.2 of [RFC6081].
  - 2 - Unassigned
  - 3 - Alternate Address Trailer (IPv4) - value coded per Section 4.3 of [RFC6081].
  - 4 - Neighbor Discovery Option Trailer - value coded per Section 4.4 of [RFC6081].
  - 5 - Random Port Trailer - value coded per Section 4.5 of [RFC6081].
  - 6 - Alternate Address Trailer (IPv6) - value coded per Section 4.3 of [RFC6081], except that each address is a 16-octet IPv6 address instead of a 4-octet IPv4 address.

- o Trailer Type values 7 through 252 are available for assignment by future specifications, which must also define the format of the Trailer Option Value and its processing rules. Trailer Type values 253 and 254 are reserved for experimentation, as recommended in [RFC3692], and value 255 is Reserved by IANA.

13. Address Mapping - Multicast

The multicast address mapping of the native underlying interface applies. The mobile router on board the MN also serves as an IGMP/ MLD Proxy for its EUNs and/or hosted applications per [RFC4605] while using the L2 address of the AR as the L2 address for all multicast packets.

The MN uses Multicast Listener Discovery (MLDv2) [RFC3810] to coordinate with the AR, and *NET L2 elements use MLD snooping [RFC4541].

14. Multilink Conceptual Sending Algorithm

The MN's IPv6 layer selects the outbound OMNI interface according to SBM considerations when forwarding original IP packets from local or EUN applications to external correspondents. Each OMNI interface maintains a neighbor cache the same as for any IPv6 interface, but with additional state for multilink coordination. Each OMNI interface maintains default routes via ARs discovered as discussed in Section 15, and may configure more-specific routes discovered through means outside the scope of this specification.

After an original IP packet enters the OMNI interface, one or more outbound underlying interfaces are selected based on PBM traffic attributes, and one or more neighbor underlying interfaces are selected based on the receipt of Interface Attributes sub-options in IPv6 ND messages (see: Figure 15). Underlying interface selection for the nodes own local interfaces are based on attributes such as DSCP, application port number, cost, performance, message size, etc. OMNI interface multilink selections could also be configured to perform replication across multiple underlying interfaces for increased reliability at the expense of packet duplication. The set of all Interface Attributes received in IPv6 ND messages determines the multilink forwarding profile for selecting the neighbor's underlying interfaces.

When the OMNI interface sends an original IP packet over a selected outbound underlying interface, the OAL employs encapsulation and fragmentation as discussed in Section 5, then performs *NET encapsulation as determined by the L2 address information received in Interface Attributes. The OAL also performs encapsulation when the

nearest AR is located multiple hops away as discussed in Section 15.1. (Note that the OAL MAY employ packing when multiple original IP packets and/or control messages are available for forwarding to the same OAL destination.)

OMNI interface multilink service designers MUST observe the BCP guidance in Section 15 [RFC3819] in terms of implications for reordering when original IP packets from the same flow may be spread across multiple underlying interfaces having diverse properties.

14.1. Multiple OMNI Interfaces

MNs may connect to multiple independent OMNI links concurrently in support of SBM. Each OMNI interface is distinguished by its Anycast ULA (e.g., [ULA]:0002::, [ULA]:1000::, [ULA]:7345::, etc.). The MN configures a separate OMNI interface for each link so that multiple interfaces (e.g., omni0, omni1, omni2, etc.) are exposed to the IPv6 layer. A different Anycast ULA is assigned to each interface, and the MN injects the service prefixes for the OMNI link instances into the EUN routing system.

Applications in EUNs can use Segment Routing to select the desired OMNI interface based on SBM considerations. The Anycast ULA is written into an original IP packet's IPv6 destination address, and the actual destination (along with any additional intermediate hops) is written into the Segment Routing Header. Standard IP routing directs the packet to the MN's mobile router entity, and the Anycast ULA identifies the OMNI interface to be used for transmission to the next hop. When the MN receives the packet, it replaces the IPv6 destination address with the next hop found in the routing header and transmits the message over the OMNI interface identified by the Anycast ULA.

Multiple distinct OMNI links can therefore be used to support fault tolerance, load balancing, reliability, etc. The architectural model is similar to Layer 2 Virtual Local Area Networks (VLANs).

14.2. MN<->AR Traffic Loop Prevention

After an AR has registered an MNP for a MN (see: Section 15), the AR will forward packets destined to an address within the MNP to the MN. The MN will under normal circumstances then forward the packet to the correct destination within its internal networks.

If at some later time the MN loses state (e.g., after a reboot), it may begin returning packets destined to an MNP address to the AR as its default router. The AR therefore must drop any packets

originating from the MN and destined to an address within the MN's registered MNP. To do so, the AR institutes the following check:

- o if the IP destination address belongs to a neighbor on the same OMNI interface, and if the link-layer source address is the same as one of the neighbor's link-layer addresses, drop the packet.

15. Router Discovery and Prefix Registration

MNs interface with the MS by sending RS messages with OMNI options under the assumption that one or more AR on the *NET will process the message and respond. The MN then configures default routes for the OMNI interface via the discovered ARs as the next hop. The manner in which the *NET ensures AR coordination is link-specific and outside the scope of this document (however, considerations for *NETs that do not provide ARs that recognize the OMNI option are discussed in Section 20).

For each underlying interface, the MN sends an RS message with an OMNI option to coordinate with MSEs identified by MSID values. Example MSID discovery methods are given in [RFC5214] and include data link login parameters, name service lookups, static configuration, a static "hosts" file, etc. The MN can also send an RS with an MS-Register sub-option that includes the Anycast MSID value '0', i.e., instead of or in addition to any non-zero MSIDs. When the AR receives an RS with a MSID '0', it selects a nearby MSE (which may be itself) and returns an RA with the selected MSID in an MS-Register sub-option. The AR selects only a single wildcard MSE (i.e., even if the RS MS-Register sub-option included multiple '0' MSIDs) while also soliciting the MSEs corresponding to any non-zero MSIDs.

MNs configure OMNI interfaces that observe the properties discussed in the previous section. The OMNI interface and its underlying interfaces are said to be in either the "UP" or "DOWN" state according to administrative actions in conjunction with the interface connectivity status. An OMNI interface transitions to UP or DOWN through administrative action and/or through state transitions of the underlying interfaces. When a first underlying interface transitions to UP, the OMNI interface also transitions to UP. When all underlying interfaces transition to DOWN, the OMNI interface also transitions to DOWN.

When an OMNI interface transitions to UP, the MN sends RS messages to register its MNP and an initial set of underlying interfaces that are also UP. The MN sends additional RS messages to refresh lifetimes and to register/deregister underlying interfaces as they transition to UP or DOWN. The MN's OMNI interface sends initial RS messages

over an UP underlying interface with its MNP-LLA as the source and with destination set to link-scoped All-Routers multicast (ff02::2) [RFC4291]. The OMNI interface includes an OMNI option per Section 12 with a Preflen assertion, Interface Attributes appropriate for underlying interfaces, MS-Register/Release sub-options containing MSID values, Reassembly Limits, an authentication sub-option and with any other necessary OMNI sub-options (e.g., a Node Identification sub-option as an identity for the MN). The OMNI interface then sets the S/T-omIndex field to the index of the underlying interface over which the RS message is sent.

The OMNI interface then sends the RS over the underlying interface using OAL encapsulation and fragmentation if necessary. If OAL encapsulation is used for RS messages sent over an INET interface, the entire RS message must appear within a single carrier packet so that it can be authenticated without requiring reassembly. The OMNI interface selects an unpredictable initial Identification value per Section 6.5, sets the OAL source address to the ULA corresponding to the RS source and sets the OAL destination to site-scoped All-Routers multicast (ff05::2) then sends the message.

ARs process IPv6 ND messages with OMNI options and act as an MSE themselves and/or as a proxy for other MSEs. ARs receive RS messages and create a neighbor cache entry for the MN, then coordinate with any MSEs named in the Register/Release lists in a manner outside the scope of this document. When an MSE processes the OMNI information, it first validates the prefix registration information then injects/withdraws the MNP in the routing/mapping system and caches/discards the new Preflen, MNP and Interface Attributes. The MSE then informs the AR of registration success/failure, and the AR returns an RA message to the MN with an OMNI option per Section 12.

The AR's OMNI interface returns the RA message via the same underlying interface of the MN over which the RS was received, and with destination address set to the MNP-LLA (i.e., unicast), with source address set to its own LLA, and with an OMNI option with S/T-omIndex set to the value included in the RS. The OMNI option also includes a Preflen confirmation, Interface Attributes, MS-Register/Release and any other necessary OMNI sub-options (e.g., a Node Identification sub-option as an identity for the AR). The RA also includes any information for the link, including RA Cur Hop Limit, M and O flags, Router Lifetime, Reachable Time and Retrans Timer values, and includes any necessary options such as:

- PIOs with (A; L=0) that include MSPs for the link [RFC8028].

- RIOs [RFC4191] with more-specific routes.

- an MTU option that specifies the maximum acceptable packet size for this underlying interface.

The OMNI interface then sends the RA, using OAL encapsulation/ fragmentation with the same Identification value that appeared in the RS message OAL header. The OMNI interface sets the OAL source address to the ULA corresponding to the RA source and sets the OAL destination to the ULA corresponding to the RA destination. The AR MAY also send periodic and/or event-driven unsolicited RA messages per [RFC4861]. In that case, the S/T-omIndex field in the OMNI option of the unsolicited RA message identifies the target underlying interface of the destination MN.

The AR can combine the information from multiple MSEs into one or more "aggregate" RAs sent to the MN in order conserve *NET bandwidth. Each aggregate RA includes an OMNI option with MS-Register/Release sub-options with the MSEs represented by the aggregate. If an aggregate is sent, the RA message contents must consistently represent the combined information advertised by all represented MSEs. Note that since the AR uses its own ADM-LLA as the RA source address, the MN determines the addresses of the represented MSEs by examining the MS-Register/Release OMNI sub-options.

When the MN receives the RA message, it creates an OMNI interface neighbor cache entry for each MSID that has confirmed MNP registration via the L2 address of this AR. If the MN connects to multiple *NETs, it records the additional L2 AR addresses in each MSID neighbor cache entry (i.e., as multilink neighbors). The MN then configures a default route via the MSE that returned the RA message, and assigns the Subnet Router Anycast address corresponding to the MNP (e.g., 2001:db8:1:2::) to the OMNI interface. The MN then manages its underlying interfaces according to their states as follows:

- When an underlying interface transitions to UP, the MN sends an RS over the underlying interface with an OMNI option. The OMNI option contains at least one Interface Attribute sub-option with values specific to this underlying interface, and may contain additional Interface Attributes specific to other underlying interfaces. The option also includes any MS-Register/Release sub-options.

- When an underlying interface transitions to DOWN, the MN sends an RS or unsolicited NA message over any UP underlying interface with an OMNI option containing an Interface Attribute sub-option for the DOWN underlying interface with Link set to '0'. The MN sends an RS when an acknowledgement is required, or an unsolicited NA when reliability is not thought to be a concern (e.g., if

redundant transmissions are sent on multiple underlying interfaces).

- When the Router Lifetime for a specific AR nears expiration, the MN sends an RS over the underlying interface to receive a fresh RA. If no RA is received, the MN can send RS messages to an alternate MSID in case the current MSID has failed. If no RS messages are received even after trying to contact alternate MSIDs, the MN marks the underlying interface as DOWN.

- When a MN wishes to release from one or more current MSIDs, it sends an RS or unsolicited NA message over any UP underlying interfaces with an OMNI option with a Release MSID. Each MSID then withdraws the MNP from the routing/mapping system and informs the AR that the release was successful.

- When all of a MNs underlying interfaces have transitioned to DOWN (or if the prefix registration lifetime expires), any associated MSEs withdraw the MNP the same as if they had received a message with a release indication.

The MN is responsible for retrying each RS exchange up to MAX_RTR_SOLICITATIONS times separated by RTR_SOLICITATION_INTERVAL seconds until an RA is received. If no RA is received over an UP underlying interface (i.e., even after attempting to contact alternate MSEs), the MN declares this underlying interface as DOWN.

The IPv6 layer sees the OMNI interface as an ordinary IPv6 interface. Therefore, when the IPv6 layer sends an RS message the OMNI interface returns an internally-generated RA message as though the message originated from an IPv6 router. The internally-generated RA message contains configuration information that is consistent with the information received from the RAs generated by the MS. Whether the OMNI interface IPv6 ND messaging process is initiated from the receipt of an RS message from the IPv6 layer is an implementation matter. Some implementations may elect to defer the IPv6 ND messaging process until an RS is received from the IPv6 layer, while others may elect to initiate the process proactively. Still other deployments may elect to administratively disable the ordinary RS/RA messaging used by the IPv6 layer over the OMNI interface, since they are not required to drive the internal RS/RA processing. (Note that this same logic applies to IPv4 implementations that employ ICMP-based Router Discovery per [RFC1256].)

Note: The Router Lifetime value in RA messages indicates the time before which the MN must send another RS message over this underlying interface (e.g., 600 seconds), however that timescale may be significantly longer than the lifetime the MS has committed to retain

the prefix registration (e.g., REACHABLETIME seconds). ARs are therefore responsible for keeping MS state alive on a shorter timescale than the MN is required to do on its own behalf.

Note: On multicast-capable underlying interfaces, MNs should send periodic unsolicited multicast NA messages and ARs should send periodic unsolicited multicast RA messages as "beacons" that can be heard by other nodes on the link. If a node fails to receive a beacon after a timeout value specific to the link, it can initiate a unicast exchange to test reachability.

Note: if an AR acting as a proxy forwards a MN's RS message to another node acting as an MSE using UDP/IP encapsulation, it must use a distinct UDP source port number for each MN. This allows the MSE to distinguish different MNs behind the same AR at the link-layer, whereas the link-layer addresses would otherwise be indistinguishable.

Note: when an AR acting as an MSE returns an RA to an INET Client, it includes an OMNI option with an Interface Attributes sub-option with omIndex set to 0 and with SRT, FMT, LHS and L2ADDR information for its INET interface. This provides the Client with partition prefix context regarding the local OMNI link segment.

15.1. Router Discovery in IP Multihop and IPv4-Only Networks

On some *NETs, a MN may be located multiple IP hops away from the nearest AR. Forwarding through IP multihop *NETs is conducted through the application of a routing protocol (e.g., a MANET/VANET routing protocol over omni-directional wireless interfaces, an inter-domain routing protocol in an enterprise network, etc.). These *NETs could be either IPv6-enabled or IPv4-only, while IPv4-only *NETs could be either multicast-capable or unicast-only (note that for IPv4-only *NETs the following procedures apply for both single-hop and multihop cases).

A MN located potentially multiple *NET hops away from the nearest AR prepares an RS message with source address set to its MNP-LLA (or to the unspecified address (::) if it does not yet have an MNP-LLA), and with destination set to link-scoped All-Routers multicast the same as discussed above. The OMNI interface then employs OAL encapsulation and fragmentation, and sets the OAL source address to the ULA corresponding to the RS source (or to a Temporary ULA if the RS source was the unspecified address (::)) and sets the OAL destination to site-scoped All-Routers multicast (ff05::2). For IPv6-enabled *NETs, the MN then encapsulates the message in UDP/IPv6 headers with source address set to the underlying interface address (or to the ULA that would be used for OAL encapsulation if the underlying interface

does not yet have an address) and sets the destination to either a unicast or anycast address of an AR. For IPv4-only *NETs, the MN instead encapsulates the RS message in UDP/IPv4 headers with source address set to the IPv4 address of the underlying interface and with destination address set to either the unicast IPv4 address of an AR [RFC5214] or an IPv4 anycast address reserved for OMNI. The MN then sends the encapsulated RS message via the *NET interface, where it will be forwarded by zero or more intermediate *NET hops.

When an intermediate *NET hop that participates in the routing protocol receives the encapsulated RS, it forwards the message according to its routing tables (note that an intermediate node could be a fixed infrastructure element or another MN). This process repeats iteratively until the RS message is received by a penultimate *NET hop within single-hop communications range of an AR, which forwards the message to the AR.

When the AR receives the message, it decapsulates the RS (while performing OAL reassembly, if necessary) and coordinates with the MS the same as for an ordinary link-local RS, since the network layer Hop Limit will not have been decremented by the multihop forwarding process. The AR then prepares an RA message with source address set to its own ADM-LLA and destination address set to the LLA of the original MN. The AR then performs OAL encapsulation and fragmentation, with OAL source set to its own ADM-ULA and destination set to the ULA corresponding to the RA source. The AR then encapsulates the message in UDP/IPv4 or UDP/IPv6 headers with source address set to its own address and with destination set to the encapsulation source of the RS.

The AR then forwards the message to an *NET node within communications range, which forwards the message according to its routing tables to an intermediate node. The multihop forwarding process within the *NET continues repetitively until the message is delivered to the original MN, which decapsulates the message and performs autoconfiguration the same as if it had received the RA directly from the AR as an on-link neighbor.

Note: An alternate approach to multihop forwarding via IPv6 encapsulation would be for the MN and AR to statelessly translate the IPv6 LLAs into ULAs and forward the RS/RA messages without encapsulation. This would violate the [RFC4861] requirement that certain IPv6 ND messages must use link-local addresses and must not be accepted if received with Hop Limit less than 255. This document therefore mandates encapsulation since the overhead is nominal considering the infrequent nature and small size of IPv6 ND messages. Future documents may consider encapsulation avoidance through translation while updating [RFC4861].

Note: An alternate approach to multihop forwarding via IPv4 encapsulation would be to employ IPv6/IPv4 protocol translation. However, for IPv6 ND messages the LLAs would be truncated due to translation and the OMNI Router and Prefix Discovery services would not be able to function. The use of IPv4 encapsulation is therefore indicated.

Note: An IPv4 anycast address for OMNI in IPv4 networks could be part of a new IPv4 /24 prefix allocation, but this may be difficult to obtain given IPv4 address exhaustion. An alternative would be to re-purpose the prefix 192.88.99.0 which has been set aside from its former use by [RFC7526].

15.2. MS-Register and MS-Release List Processing

OMNI links maintain a constant value "MAX_MSID" selected to provide MNs with an acceptable level of MSE redundancy while minimizing control message amplification. It is RECOMMENDED that MAX_MSID be set to the default value 5; if a different value is chosen, it should be set uniformly by all nodes on the OMNI link.

When a MN sends an RS message with an OMNI option via an underlying interface to an AR, the MN must convey its knowledge of its currently-associated MSEs. Initially, the MN will have no associated MSEs and should therefore include an MS-Register sub-option with the single "anycast" MSID value 0 which requests the AR to select and assign an MSE. The AR will then return an RA message with source address set to the ADM-LLA of the selected MSE.

As the MN activates additional underlying interfaces, it can optionally include an MS-Register sub-option with MSID value 0, or with non-zero MSIDs for MSEs discovered from previous RS/RA exchanges. The MN will thus eventually begin to learn and manage its currently active set of MSEs, and can register with new MSEs or release from former MSEs with each successive RS/RA exchange. As the MN's MSE constituency grows, it alone is responsible for including or omitting MSIDs in the MS-Register/Release lists it sends in RS messages. The inclusion or omission of MSIDs determines the MN's interface to the MS and defines the manner in which MSEs will respond. The only limiting factor is that the MN should include no more than MAX_MSID values in each list per each IPv6 ND message, and should avoid duplication of entries in each list unless it wants to increase likelihood of control message delivery.

When an AR receives an RS message sent by a MN with an OMNI option, the option will contain zero or more MS-Register and MS-Release sub-options containing MSIDs. After processing the OMNI option, the AR will have a list of zero or more MS-Register MSIDs and a list of zero

or more of MS-Release MSIDs. The AR then processes the lists as follows:

- For each list, retain the first MAX_MSID values in the list and discard any additional MSIDs (i.e., even if there are duplicates within a list).

- Next, for each MSID in the MS-Register list, remove all matching MSIDs from the MS-Release list.

- Next, proceed as follows:

  * If the AR's own MSID or the value 0 appears in the MS-Register list, send an RA message directly back to the MN and send a proxy copy of the RS message to each additional MSID in the MS-Register list with the MS-Register/Release lists omitted. Then, send an unsolicited NA (uNA) message to each MSID in the MS-Release list with the MS-Register/Release lists omitted and with an OMNI option with S/T-omIndex set to 0.

  * Otherwise, send a proxy copy of the RS message to each additional MSID in the MS-Register list with the MS-Register list omitted. For the first MSID, include the original MS-Release list; for all other MSIDs, omit the MS-Release list.

Each proxy copy of the RS message will include an OMNI option and OAL encapsulation header with the ADM-ULA of the AR as the source and the ADM-ULA of the Register MSE as the destination. When the Register MSE receives the proxy RS message, if the message includes an MS-Release list the MSE sends a uNA message to each additional MSID in the Release list with an OMNI option with S/T-omIndex set to 0. The Register MSE then sends an RA message back to the (Proxy) AR wrapped in an OAL encapsulation header with source and destination addresses reversed, and with RA destination set to the MNP-LLA of the MN. When the AR receives this RA message, it sends a proxy copy of the RA to the MN.

Each uNA message (whether sent by the first-hop AR or by a Register MSE) will include an OMNI option and an OAL encapsulation header with the ADM-ULA of the Register MSE as the source and the ADM-ULA of the Release MSE as the destination. The uNA informs the Release MSE that its previous relationship with the MN has been released and that the source of the uNA message is now registered. The Release MSE must then note that the subject MN of the uNA message is now "departed", and forward any subsequent packets destined to the MN to the Register MSE.

Note that it is not an error for the MS-Register/Release lists to include duplicate entries. If duplicates occur within a list, the AR will generate multiple proxy RS and/or uNA messages - one for each copy of the duplicate entries.

15.3. DHCPv6-based Prefix Registration

When a MN is not pre-provisioned with an MNP-LLA (or, when the MN requires additional MNP delegations), it requests the MSE to select MNPs on its behalf and set up the correct routing state within the MS. The DHCPv6 service [RFC8415] supports this requirement.

When an MN needs to have the MSE select MNPs, it sends an RS message with source set to the unspecified address (::) if it has no MNP_LLAs. If the MN requires only a single MNP delegation, it can then include a Node Identification sub-option in the OMNI option and set Preflen to the length of the desired MNP. If the MN requires multiple MNP delegations and/or more complex DHCPv6 services, it instead includes a DHCPv6 Message sub-option containing a Client Identifier, one or more IA_PD options and a Rapid Commit option then sets the 'msg-type' field to "Solicit", and includes a 3 octet 'transaction-id'. The MN then sets the RS destination to All-Routers multicast and sends the message using OAL encapsulation and fragmentation if necessary as discussed above.

When the MSE receives the RS message, it performs OAL reassembly if necessary. Next, if the RS source is the unspecified address (::) and/or the OMNI option includes a DHCPv6 message sub-option, the MSE acts as a "Proxy DHCPv6 Client" in a message exchange with the locally-resident DHCPv6 server. If the RS did not contain a DHCPv6 message sub-option, the MSE generates a DHCPv6 Solicit message on behalf of the MN using an IA_PD option with the prefix length set to the OMNI header Preflen value and with a Client Identifier formed from the OMNI option Node Identification sub-option; otherwise, the MSE uses the DHCPv6 Solicit message contained in the OMNI option. The MSE then sends the DHCPv6 message to the DHCPv6 Server, which delegates MNPs and returns a DHCPv6 Reply message with PD parameters. (If the MSE wishes to defer creation of MN state until the DHCPv6 Reply is received, it can instead act as a Lightweight DHCPv6 Relay Agent per [RFC6221] by encapsulating the DHCPv6 message in a Relay-forward/reply exchange with Relay Message and Interface ID options. In the process, the MSE packs any state information needed to return an RA to the MN in the Relay-forward Interface ID option so that the information will be echoed back in the Relay-reply.)

When the MSE receives the DHCPv6 Reply, it adds routes to the routing system and creates MNP-LLAs based on the delegated MNPs. The MSE then sends an RA back to the MN with the DHCPv6 Reply message

included in an OMNI DHCPv6 message sub-option if and only if the RS message had included an explicit DHCPv6 Solicit. If the RS message source was the unspecified address (::), the MSE includes one of the (newly-created) MNP-LLAs as the RA destination address and sets the OMNI option Preflen accordingly; otherwise, the MSE includes the RS source address as the RA destination address. The MSE then sets the RA source address to its own ADM-LLA then performs OAL encapsulation and fragmentation and sends the RA to the MN. When the MN receives the RA, it reassembles and discards the OAL encapsulation, then creates a default route, assigns Subnet Router Anycast addresses and uses the RA destination address as its primary MNP-LLA. The MN will then use this primary MNP-LLA as the source address of any IPv6 ND messages it sends as long as it retains ownership of the MNP.

Note: After a MN performs a DHCPv6-based prefix registration exchange with a first MSE, it would need to repeat the exchange with each additional MSE it registers with. In that case, the MN supplies the MNP delegation information received from the first MSE when it engages the additional MSEs.

16. Secure Redirection

If the *NET link model is multiple access, the AR is responsible for assuring that address duplication cannot corrupt the neighbor caches of other nodes on the link. When the MN sends an RS message on a multiple access *NET link, the AR verifies that the MN is authorized to use the address and returns an RA with a non-zero Router Lifetime only if the MN is authorized.

After verifying MN authorization and returning an RA, the AR MAY return IPv6 ND Redirect messages to direct MNs located on the same *NET link to exchange packets directly without transiting the AR. In that case, the MNs can exchange packets according to their unicast L2 addresses discovered from the Redirect message instead of using the dogleg path through the AR. In some *NET links, however, such direct communications may be undesirable and continued use of the dogleg path through the AR may provide better performance. In that case, the AR can refrain from sending Redirects, and/or MNs can ignore them.

17. AR and MSE Resilience

*NETs SHOULD deploy ARs in Virtual Router Redundancy Protocol (VRRP) [RFC5798] configurations so that service continuity is maintained even if one or more ARs fail. Using VRRP, the MN is unaware which of the (redundant) ARs is currently providing service, and any service discontinuity will be limited to the failover time supported by VRRP. Widely deployed public domain implementations of VRRP are available.

MSEs SHOULD use high availability clustering services so that multiple redundant systems can provide coordinated response to failures. As with VRRP, widely deployed public domain implementations of high availability clustering services are available. Note that special-purpose and expensive dedicated hardware is not necessary, and public domain implementations can be used even between lightweight virtual machines in cloud deployments.

18. Detecting and Responding to MSE Failures

In environments where fast recovery from MSE failure is required, ARs SHOULD use proactive Neighbor Unreachability Detection (NUD) in a manner that parallels Bidirectional Forwarding Detection (BFD) [RFC5880] to track MSE reachability. ARs can then quickly detect and react to failures so that cached information is re-established through alternate paths. Proactive NUD control messaging is carried only over well-connected ground domain networks (i.e., and not low-end *NET links such as aeronautical radios) and can therefore be tuned for rapid response.

ARs perform proactive NUD for MSEs for which there are currently active MNs on the *NET. If an MSE fails, ARs can quickly inform MNs of the outage by sending multicast RA messages on the *NET interface. The AR sends RA messages to MNs via the *NET interface with an OMNI option with a Release ID for the failed MSE, and with destination address set to All-Nodes multicast (ff02::1) [RFC4291].

The AR SHOULD send MAX_FINAL_RTR_ADVERTISEMENTS RA messages separated by small delays [RFC4861]. Any MNs on the *NET interface that have been using the (now defunct) MSE will receive the RA messages and associate with a new MSE.

19. Transition Considerations

When a MN connects to an *NET link for the first time, it sends an RS message with an OMNI option. If the first hop AR recognizes the option, it returns an RA with its ADM-LLA as the source, the MNP-LLA as the destination and with an OMNI option included. The MN then engages the AR according to the OMNI link model specified above. If the first hop AR is a legacy IPv6 router, however, it instead returns an RA message with no OMNI option and with a non-OMNI unicast source LLA as specified in [RFC4861]. In that case, the MN engages the *NET according to the legacy IPv6 link model and without the OMNI extensions specified in this document.

If the *NET link model is multiple access, there must be assurance that address duplication cannot corrupt the neighbor caches of other nodes on the link. When the MN sends an RS message on a multiple

access *NET link with an LLA source address and an OMNI option, ARs that recognize the option ensure that the MN is authorized to use the address and return an RA with a non-zero Router Lifetime only if the MN is authorized. ARs that do not recognize the option instead return an RA that makes no statement about the MN's authorization to use the source address. In that case, the MN should perform Duplicate Address Detection to ensure that it does not interfere with other nodes on the link.

An alternative approach for multiple access *NET links to ensure isolation for MN / AR communications is through L2 address mappings as discussed in Appendix C. This arrangement imparts a (virtual) point-to-point link model over the (physical) multiple access link.

20. OMNI Interfaces on Open Internetworks

OMNI interfaces configured over IPv6-enabled underlying interfaces on an open Internetwork without an OMNI-aware first-hop AR receive RA messages that do not include an OMNI option, while OMNI interfaces configured over IPv4-only underlying interfaces do not receive any (IPv6) RA messages at all (although they may receive IPv4 RA messages [RFC1256]). OMNI interfaces that receive RA messages without an OMNI option configure addresses, on-link prefixes, etc. on the underlying interface that received the RA according to standard IPv6 ND and address resolution conventions [RFC4861] [RFC4862]. OMNI interfaces configured over IPv4-only underlying interfaces configure IPv4 address information on the underlying interfaces using mechanisms such as DHCPv4 [RFC2131].

OMNI interfaces configured over underlying interfaces that connect to an open Internetwork can apply security services such as VPNs to connect to an MSE, or can establish a direct link to an MSE through some other means (see Section 4). In environments where an explicit VPN or direct link may be impractical, OMNI interfaces can instead use UDP/IP encapsulation per [RFC6081][RFC4380] and HIP-based message authentication per [RFC7401].

OMNI interfaces use UDP service port number 8060 (see: Section 25.10 and Section 3.6 of [I-D.templin-intarea-6706bis]) according to the simple UDP/IP encapsulation format specified in [RFC4380] for both IPv4 and IPv6 underlying interfaces. OMNI interfaces do not include the UDP/IP header/trailer extensions specified in [RFC4380][RFC6081], but may include them as OMNI sub-options instead when necessary. Since the OAL includes an integrity check over the OAL packet, OAL sources selectively disable UDP checksums for OAL packets that do not require UDP/IP address integrity, but enable UDP checksums for others including non-OAL packets, IPv6 ND messages used to establish link-layer addresses, etc. If the OAL source discovers that packets with

UDP checksums disabled are being dropped in the path it should enable UDP checksums in future packets. Further considerations for UDP encapsulation checksums are found in [RFC6935][RFC6936].

For "Vehicle-to-Infrastructure (V2I)" coordination, the MN codes an authentication sub-option in an OMNI option of an IPv6 RS message and the AR responds with an authentication sub-option in an OMNI option of an IPv6 RA message. HIP security services can be applied per [RFC7401] using the RS/RA messages as simple "shipping containers" to convey the HIP parameters. Alternatively, a simple Hashed Message Authentication Code (HMAC) can be included in the manner specified in [RFC4380]. For "Vehicle-to-Vehicle (V2V)" coordination, two MNs can coordinate directly with one another with HIP "Initiator/Responder" messages coded in OMNI options of IPv6 NS/NA messages. In that case, a four-message HIP exchange (i.e., two back-to-back NS/NA exchanges) may be necessary for the two MNs to attain mutual authentication.

After establishing a VPN or preparing for UDP/IP encapsulation, OMNI interfaces send control plane messages to interface with the MS, including RS/RA messages used according to Section 15 and NS/NA messages used for route optimization and mobility (see: [I-D.templin-intarea-6706bis]). The control plane messages must be authenticated while data plane messages are delivered the same as for ordinary best-effort traffic with basic source address-based data origin verification. Data plane communications via OMNI interfaces that connect over open Internetworks without an explicit VPN should therefore employ transport- or higher-layer security to ensure integrity and/or confidentiality.

OMNI interfaces configured over open Internetworks are often located behind NATs. The OMNI interface accommodates NAT traversal using UDP/IP encapsulation and the mechanisms discussed in [I-D.templin-intarea-6706bis]. To support NAT determination, ARs include an Origin Indication sub-option in RA messages sent in response to RS messages received from a Client via UDP/IP encapsulation.

Note: Following the initial HIP Initiator/Responder exchange, OMNI interfaces configured over open Internetworks maintain HIP associations through the transmission of IPv6 ND messages that include OMNI options with HIP "Update" and "Notify" messages. OMNI interfaces use the HIP "Update" message when an acknowledgement is required, and use the "Notify" message in unacknowledged isolated IPv6 ND messages (e.g., unsolicited NAs). When HMAC authentication is used instead of HIP, the MN and AR exchange all IPv6 ND messages with HMAC signatures included based on a shared-secret.

Note: ARs that act as proxys on an open Internetwork authenticate and remove authentication OMNI sub-options from IPv6 ND messages they forward from a MN, and insert and sign authentication Origin Indication sub-options in IPv6 ND messages they forward from the network to the MN. Conversely, ARs that act as proxys forward without processing any DHCPv6 information in RS/RA message exchanges between MNs and MSEs. The AR is therefore responsible for MN authentication while the MSE is responsible for registering/ delegating MNPs.

Note: A simpler arrangement is possible when the AR also acts as a MSE itself, i.e., when the proxy and MSE functions are combined on a single physical or logical platform.

21. Time-Varying MNPs

In some use cases, it is desirable, beneficial and efficient for the MN to receive a constant MNP that travels with the MN 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 prefix delegation services discussed in Section 15.3 allows OMNI MNs that desire time-varying MNPs to obtain short-lived prefixes to send RS messages with source set to the unspecified address (::) and/ or with an OMNI option with DHCPv6 Option sub-options. The MN would then be obligated to renumber its internal networks whenever its MNP (and therefore also its OMNI address) changes. This should not present a challenge for MNs with automated network renumbering services, however presents limits for the durations of ongoing sessions that would prefer to use a constant address.

22. (H)HITs and Temporary ULAs

MNs that generate (H)HITs but do not have pre-assigned MNPs can request MNP delegations by issuing IPv6 ND messages that use the (H)HIT instead of a Temporary ULA. In particular, when a MN creates an RS message it can set the source to the unspecified address (::) and destination to All-Routers multicast. The IPv6 ND message includes an OMNI option with a HIP "Initiator" message sub-option, and need not include a Node Identification sub-option since the MN's HIT appears in the HIP message. The MN then encapsulates the message in an IPv6 header with the (H)HIT as the source address and with destination set to either a unicast or anycast ADM-ULA. The MN then sends the message to the AR as specified in Section 15.1.

When the AR receives the message, it notes that the RS source was the unspecified address (::), then examines the RS encapsulation source address to determine that the source is a (H)HIT and not a Temporary ULA. The AR next invokes the DHCPv6 protocol to request an MNP prefix delegation while using the HIT as the Client Identifier, then prepares an RA message with source address set to its own ADM-LLA and destination set to the MNP-LLA corresponding to the delegated MNP. The AR next includes an OMNI option with a HIP "Responder" message and any DHCPv6 prefix delegation parameters. The AR then finally encapsulates the RA in an IPv6 header with source address set to its own ADM-ULA and destination set to the (H)HIT from the RS encapsulation source address, then returns the encapsulated RA to the MN.

MNs can also use (H)HITs and/or Temporary ULAs for direct MN-to-MN communications outside the context of any OMNI link supporting infrastructure. When two MNs encounter one another they can use their (H)HITs and/or Temporary ULAs as original IPv6 packet source and destination addresses to support direct communications. MNs can also inject their (H)HITs and/or Temporary ULAs into a MANET/VANET routing protocol to enable multihop communications. MNs can further exchange IPv6 ND messages (such as NS/NA) using their (H)HITs and/or Temporary ULAs as source and destination addresses. Note that the HIP security protocols for establishing secure neighbor relationships are based on (H)HITs. Temporary ULAs instead use the HMAC authentication service specified in [RFC4380].

Lastly, when MNs are within the coverage range of OMNI link infrastructure a case could be made for injecting (H)HITs and/or Temporary ULAs into the global MS routing system. For example, when the MN sends an RS to a MSE it could include a request to inject the (H)HIT / Temporary ULA into the routing system instead of requesting an MNP prefix delegation. This would potentially enable OMNI link-wide communications using only (H)HITs or Temporary ULAs, and not MNPs. This document notes the opportunity, but makes no recommendation.

23. Address Selection

OMNI MNs use LLAs only for link-scoped communications on the OMNI link. Typically, MNs use LLAs as source/destination IPv6 addresses of IPv6 ND messages, but may also use them for addressing ordinary original IP packets exchanged with an OMNI link neighbor.

OMNI MNs use MNP-ULAs as source/destination IPv6 addresses in the encapsulation headers of OAL packets. OMNI MNs use Temporary ULAs for OAL addressing when an MNP-ULA is not available, or as source/ destination IPv6 addresses for communications within a MANET/VANET

local area. OMNI MNs use HITs instead of Temporary ULAs when operation outside the context of a specific ULA domain and/or source address attestation is necessary.

OMNI MNs use MNP-based GUAs as original IP packet source and destination addresses for communications with Internet destinations when they are within range of OMNI link supporting infrastructure that can inject the MNP into the routing system.

24. Error Messages

An OAL destination or intermediate node may need to return ICMPv6 error messages (e.g., Destination Unreachable, Packet Too Big, Time Exceeded, etc.) [RFC4443] to an OAL source. Since ICMPv6 error messages do not themselves include authentication codes, the OAL includes the ICMPv6 error message as an OMNI sub-option in an IPv6 ND uNA message. The OAL also includes a HIP message sub-option if the uNA needs to travel over an open Internetwork.

25. IANA Considerations

The following IANA actions are requested:

25.1. "IEEE 802 Numbers" Registry

The IANA is instructed to allocate an official Ether Type number TBD1 from the 'ieee-802-numbers' registry for User Datagram Protocol (UDP) encapsulation on Ethernet networks. Guidance is found in [RFC7042].

25.2. "IPv6 Neighbor Discovery Option Formats" Registry

The IANA is instructed to allocate an official Type number TBD2 from the "IPv6 Neighbor Discovery Option Formats" registry for the OMNI option. Implementations set Type to 253 as an interim value [RFC4727].

25.3. "Ethernet Numbers" Registry

The IANA is instructed to allocate one Ethernet unicast address TBD3 (suggested value '00-52-14') in the 'ethernet-numbers' registry under "IANA Unicast 48-bit MAC Addresses" as follows:

Addresses	Usage	Reference
00-52-14	Overlay Multilink Network (OMNI) Interface	[RFCXXXX]

Figure 32: IANA Unicast 48-bit MAC Addresses

25.4. "ICMPv6 Code Fields: Type 2 - Packet Too Big" Registry

The IANA is instructed to assign two new Code values in the "ICMPv6 Code Fields: Type 2 - Packet Too Big" registry. The registry should appear as follows:

Code	Name	Reference
0	PTB Hard Error	[RFC4443]
1	PTB Soft Error (loss)	[RFCXXXX]
2	PTB Soft Error (no loss)	[RFCXXXX]

Figure 33: ICMPv6 Code Fields: Type 2 - Packet Too Big Values

(Note: this registry also to be used to define values for setting the "unused" field of ICMPv4 "Destination Unreachable - Fragmentation Needed" messages.)

25.5. "OMNI Option Sub-Type Values" (New Registry)

The OMNI option defines a 5-bit Sub-Type field, for which IANA is instructed to create and maintain a new registry entitled "OMNI Option Sub-Type Values". Initial values are given below (future assignments are to be made through Standards Action [RFC8126]):

Value	Sub-Type name	Reference
0	Pad1	[RFCXXXX]
1	PadN	[RFCXXXX]
2	Interface Attributes (Type 1)	[RFCXXXX]
3	Interface Attributes (Type 2)	[RFCXXXX]
4	Traffic Selector	[RFCXXXX]
5	MS-Register	[RFCXXXX]
6	MS-Release	[RFCXXXX]
7	Geo Coordinates	[RFCXXXX]
8	DHCPv6 Message	[RFCXXXX]
9	HIP Message	[RFCXXXX]
10	Reassembly Limit	[RFCXXXX]
11	Fragmentation Report	[RFCXXXX]
12	Node Identification	[RFCXXXX]
13-29	Unassigned	
30	Sub-Type Extension	[RFCXXXX]
31	Reserved by IANA	[RFCXXXX]

Figure 34: OMNI Option Sub-Type Values

25.6. "OMNI Node Identification ID-Type Values" (New Registry)

The OMNI Node Identification Sub-Option (see: Section 12.1.13) contains an 8-bit ID-Type field, for which IANA is instructed to create and maintain a new registry entitled "OMNI Node Identification ID-Type Values". Initial values are given below (future assignments are to be made through Expert Review [RFC8126]):

Value	Sub-Type name	Reference
0	UUID	[RFCXXXX]
1	HIT	[RFCXXXX]
2	HHIT	[RFCXXXX]
3	Network Access Identifier	[RFCXXXX]
4	FQDN	[RFCXXXX]
5-252	Unassigned	[RFCXXXX]
253-254	Reserved for Experimentation	[RFCXXXX]
255	Reserved by IANA	[RFCXXXX]

Figure 35: OMNI Node Identification ID-Type Values

25.7. "OMNI Option Sub-Type Extension Values" (New Registry)

The OMNI option defines an 8-bit Extension-Type field for Sub-Type 30 (Sub-Type Extension), for which IANA is instructed to create and maintain a new registry entitled "OMNI Option Sub-Type Extension Values". Initial values are given below (future assignments are to be made through Expert Review [RFC8126]):

Value	Sub-Type name	Reference
0	RFC4380 UDP/IP Header Option	[RFCXXXX]
1	RFC6081 UDP/IP Trailer Option	[RFCXXXX]
2-252	Unassigned	
253-254	Reserved for Experimentation	[RFCXXXX]
255	Reserved by IANA	[RFCXXXX]

Figure 36: OMNI Option Sub-Type Extension Values

25.8. "OMNI RFC4380 UDP/IP Header Option" (New Registry)

The OMNI Sub-Type Extension "RFC4380 UDP/IP Header Option" defines an 8-bit Header Type field, for which IANA is instructed to create and maintain a new registry entitled "OMNI RFC4380 UDP/IP Header Option". Initial registry values are given below (future assignments are to be made through Expert Review [RFC8126]):

Value	Sub-Type name	Reference
0	Origin Indication (IPv4)	[RFC4380]
1	Authentication Encapsulation	[RFC4380]
2	Origin Indication (IPv6)	[RFCXXXX]
3-252	Unassigned	
253-254	Reserved for Experimentation	[RFCXXXX]
255	Reserved by IANA	[RFCXXXX]

Figure 37: OMNI RFC4380 UDP/IP Header Option

25.9. "OMNI RFC6081 UDP/IP Trailer Option" (New Registry)

The OMNI Sub-Type Extension for "RFC6081 UDP/IP Trailer Option" defines an 8-bit Trailer Type field, for which IANA is instructed to create and maintain a new registry entitled "OMNI RFC6081 UDP/IP Trailer Option". Initial registry values are given below (future assignments are to be made through Expert Review [RFC8126]):

Value	Sub-Type name	Reference
0	Unassigned	
1	Nonce	[RFC6081]
2	Unassigned	
3	Alternate Address (IPv4)	[RFC6081]
4	Neighbor Discovery Option	[RFC6081]
5	Random Port	[RFC6081]
6	Alternate Address (IPv6)	[RFCXXXX]
7-252	Unassigned	
253-254	Reserved for Experimentation	[RFCXXXX]
255	Reserved by IANA	[RFCXXXX]

Figure 38: OMNI RFC6081 Trailer Option

25.10. Additional Considerations

The IANA has assigned the UDP port number "8060" for an earlier experimental version of AERO [RFC6706]. This document together with [I-D.templin-intarea-6706bis] 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 an invalid ICMPv6 message type number '0'.) The IANA is therefore instructed to update the reference for UDP port number "8060" from "RFC6706" to "RFCXXXX" (i.e., this document).

The IANA has assigned a 4 octet Private Enterprise Number (PEN) code "45282" in the "enterprise-numbers" registry. This document is the

normative reference for using this code in DHCP Unique IDentifiers based on Enterprise Numbers ("DUID-EN for OMNI Interfaces") (see: Section 11). The IANA is therefore instructed to change the enterprise designation for PEN code "45282" from "LinkUp Networks" to "Overlay Multilink Network Interface (OMNI)".

The IANA has assigned the ifType code "301 - omni - Overlay Multilink Network Interface (OMNI)" in accordance with Section 6 of [RFC8892]. The registration appears under the IANA "Structure of Management Information (SMI) Numbers (MIB Module Registrations) - Interface Types (ifType)" registry.

No further IANA actions are required.

26. Security Considerations

Security considerations for IPv4 [RFC0791], IPv6 [RFC8200] and IPv6 Neighbor Discovery [RFC4861] apply. OMNI interface IPv6 ND messages SHOULD include Nonce and Timestamp options [RFC3971] when transaction confirmation and/or time synchronization is needed. (Note however that when OAL encapsulation is used the (echoed) OAL Identification value can provide sufficient transaction confirmation.)

MN OMNI interfaces configured over secured ANET interfaces inherit the physical and/or link-layer security properties (i.e., "protected spectrum") of the connected ANETs. MN OMNI interfaces configured over open INET interfaces can use symmetric securing services such as VPNs or can by some other means establish a direct link. When a VPN or direct link may be impractical, however, the security services specified in [RFC7401] and/or [RFC4380] can be employed. While the OMNI link protects control plane messaging, applications must still employ end-to-end transport- or higher-layer security services to protect the data plane.

Strong network layer security for control plane messages and forwarding path integrity for data plane messages between MSEs MUST be supported. In one example, the AERO service [I-D.templin-intarea-6706bis] constructs a spanning tree between MSEs and secures the links in the spanning tree with network layer security mechanisms such as IPsec [RFC4301] or Wireguard. Control plane messages are then constrained to travel only over the secured spanning tree paths and are therefore protected from attack or eavesdropping. Since data plane messages can travel over route optimized paths that do not strictly follow the spanning tree, however, end-to-end transport- or higher-layer security services are still required.

Identity-based key verification infrastructure services such as iPSK may be necessary for verifying the identities claimed by MNs. This requirement should be harmonized with the manner in which (H)HITs are attested in a given operational environment.

Security considerations for specific access network interface types are covered under the corresponding IP-over-(foo) specification (e.g., [RFC2464], [RFC2492], etc.).

Security considerations for IPv6 fragmentation and reassembly are discussed in Section 6.9.

27. Implementation Status

AERO/OMNI Release-3.0.2 was tagged on October 15, 2020, and is undergoing internal testing. Additional internal releases expected within the coming months, with first public release expected end of 1H2021.

28. Acknowledgements

The first version of this document was prepared per the consensus decision at the 7th Conference of the International Civil Aviation Organization (ICAO) Working Group-I Mobility Subgroup on March 22, 2019. Consensus to take the document forward to the IETF was reached at the 9th Conference of the Mobility Subgroup on November 22, 2019. Attendees and contributors included: Guray Acar, Danny Bharj, Francois D'Humieres, Pavel Drasil, Nikos Fistas, Giovanni Garofolo, Bernhard Haindl, Vaughn Maiolla, Tom McParland, Victor Moreno, Madhu Niraula, Brent Phillips, Liviu Popescu, Jacky Pouzet, Aloke Roy, Greg Saccone, Robert Segers, Michal Skorepa, Michel Solery, Stephane Tamalet, Fred Templin, Jean-Marc Vacher, Bela Varkonyi, Tony Whyman, Fryderyk Wrobel and Dongsong Zeng.

The following individuals are acknowledged for their useful comments: Stuart Card, Michael Matyas, Robert Moskowitz, Madhu Niraula, Greg Saccone, Stephane Tamalet, Eric Vyncke. Pavel Drasil, Zdenek Jaron and Michal Skorepa are especially recognized for their many helpful ideas and suggestions. Madhuri Madhava Badgandi, Sean Dickson, Don Dillenburg, Joe Dudkowski, Vijayasarathy Rajagopalan, Ron Sackman and Katherine Tran are acknowledged for their hard work on the implementation and technical insights that led to improvements for the spec.

Discussions on the IETF 6man and atn mailing lists during the fall of 2020 suggested additional points to consider. The authors gratefully acknowledge the list members who contributed valuable insights through those discussions. Eric Vyncke and Erik Kline were the

intarea ADs, while Bob Hinden and Ole Troan were the 6man WG chairs at the time the document was developed; they are all gratefully acknowledged for their many helpful insights. Many of the ideas in this document have further built on IETF experiences beginning as early as Y2K, with insights from colleagues including Brian Carpenter, Ralph Droms, Christian Huitema, Thomas Narten, Dave Thaler, Joe Touch, and many others who deserve recognition.

Early observations on IP fragmentation performance implications were noted in the 1986 Digital Equipment Corporation (DEC) "qe reset" investigation, where fragment bursts from NFS UDP traffic triggered hardware resets resulting in communication failures. Jeff Chase, Fred Glover and Chet Juzsczak of the Ultrix Engineering Group led the investigation, and determined that setting a smaller NFS mount block size reduced the amount of fragmentation and suppressed the resets. Early observations on L2 media MTU issues were noted in the 1988 DEC FDDI investigation, where Raj Jain, KK Ramakrishnan and Kathy Wilde represented architectural considerations for FDDI networking in general including FDDI/Ethernet bridging. Jeff Mogul (who led the IETF Path MTU Discovery working group) and other DEC colleagues who supported these early investigations are also acknowledged.

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 Information Technology (BIT) Mobility Vision Lab (MVL) program.

29. References

29.1. Normative References

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

[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>.

[RFC2474] Nichols, K., Blake, S., Baker, F., and D. Black, "Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers", RFC 2474, DOI 10.17487/RFC2474, December 1998, <https://www.rfc-editor.org/info/rfc2474>.

[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>.

[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>.

[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>.

[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>.

[RFC4727] Fenner, B., "Experimental Values In IPv4, IPv6, ICMPv4, ICMPv6, UDP, and TCP Headers", RFC 4727, DOI 10.17487/RFC4727, November 2006, <https://www.rfc-editor.org/info/rfc4727>.

[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>.

[RFC6088] Tsirtsis, G., Giarreta, G., Soliman, H., and N. Montavont, "Traffic Selectors for Flow Bindings", RFC 6088, DOI 10.17487/RFC6088, January 2011, <https://www.rfc-editor.org/info/rfc6088>.

[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>.

[RFC8028] Baker, F. and B. Carpenter, "First-Hop Router Selection by Hosts in a Multi-Prefix Network", RFC 8028, DOI 10.17487/RFC8028, November 2016, <https://www.rfc-editor.org/info/rfc8028>.

[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>.

[RFC8201] McCann, J., Deering, S., Mogul, J., and R. Hinden, Ed., "Path MTU Discovery for IP version 6", STD 87, RFC 8201, DOI 10.17487/RFC8201, July 2017, <https://www.rfc-editor.org/info/rfc8201>.

[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>.

29.2. Informative References

[ATN] Maiolla, V., "The OMNI Interface - An IPv6 Air/Ground Interface for Civil Aviation, IETF Liaison Statement #1676, https://datatracker.ietf.org/liaison/1676/", March 2020.

[ATN-IPS] WG-I, ICAO., "ICAO Document 9896 (Manual on the Aeronautical Telecommunication Network (ATN) using Internet Protocol Suite (IPS) Standards and Protocol), Draft Edition 3 (work-in-progress)", December 2020.

[CKSUM] Stone, J., Greenwald, M., Partridge, C., and J. Hughes, "Performance of Checksums and CRC's Over Real Data, IEEE/ACM Transactions on Networking, Vol. 6, No. 5", October 1998.

[CRC] Jain, R., "Error Characteristics of Fiber Distributed Data Interface (FDDI), IEEE Transactions on Communications", August 1990.

[I-D.ietf-drip-rid]
Moskowitz, R., Card, S., Wiethuechter, A., and A. Gurtov, "UAS Remote ID", draft-ietf-drip-rid-06 (work in progress), December 2020.

[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]
Jeong, J., "IPv6 Wireless Access in Vehicular Environments (IPWAVE): Problem Statement and Use Cases", draft-ietf-ipwave-vehicular-networking-19 (work in progress), July 2020.

[I-D.ietf-tsvwg-udp-options]
Touch, J., "Transport Options for UDP", draft-ietf-tsvwg-udp-options-09 (work in progress), November 2020.

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

[I-D.templin-6man-lla-type]
Templin, F., "The IPv6 Link-Local Address Type Field", draft-templin-6man-lla-type-02 (work in progress), November 2020.

[I-D.templin-intarea-6706bis]
Templin, F., "Asymmetric Extended Route Optimization (AERO)", draft-templin-intarea-6706bis-87 (work in progress), January 2021.

[IPV4-GUA]
Postel, J., "IPv4 Address Space Registry, https://www.iana.org/assignments/ipv4-address-space/ipv4-address-space.xhtml", December 2020.

[IPV6-GUA]
Postel, J., "IPv6 Global Unicast Address Assignments, https://www.iana.org/assignments/ipv6-unicast-address-assignments/ipv6-unicast-address-assignments.xhtml", December 2020.

[RFC0905] "ISO Transport Protocol specification ISO DP 8073", RFC 905, DOI 10.17487/RFC0905, April 1984, <https://www.rfc-editor.org/info/rfc905>.

[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>.

[RFC1122] Braden, R., Ed., "Requirements for Internet Hosts - Communication Layers", STD 3, RFC 1122, DOI 10.17487/RFC1122, October 1989, <https://www.rfc-editor.org/info/rfc1122>.

[RFC1191] Mogul, J. and S. Deering, "Path MTU discovery", RFC 1191, DOI 10.17487/RFC1191, November 1990, <https://www.rfc-editor.org/info/rfc1191>.

[RFC1256] Deering, S., Ed., "ICMP Router Discovery Messages", RFC 1256, DOI 10.17487/RFC1256, September 1991, <https://www.rfc-editor.org/info/rfc1256>.

[RFC2131] Droms, R., "Dynamic Host Configuration Protocol", RFC 2131, DOI 10.17487/RFC2131, March 1997, <https://www.rfc-editor.org/info/rfc2131>.

[RFC2225] Laubach, M. and J. Halpern, "Classical IP and ARP over ATM", RFC 2225, DOI 10.17487/RFC2225, April 1998, <https://www.rfc-editor.org/info/rfc2225>.

[RFC2328] Moy, J., "OSPF Version 2", STD 54, RFC 2328, DOI 10.17487/RFC2328, April 1998, <https://www.rfc-editor.org/info/rfc2328>.

[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>.

[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>.

[RFC2492] Armitage, G., Schulter, P., and M. Jork, "IPv6 over ATM Networks", RFC 2492, DOI 10.17487/RFC2492, January 1999, <https://www.rfc-editor.org/info/rfc2492>.

[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>.

[RFC2863] McCloghrie, K. and F. Kastenholz, "The Interfaces Group MIB", RFC 2863, DOI 10.17487/RFC2863, June 2000, <https://www.rfc-editor.org/info/rfc2863>.

[RFC2923] Lahey, K., "TCP Problems with Path MTU Discovery", RFC 2923, DOI 10.17487/RFC2923, September 2000, <https://www.rfc-editor.org/info/rfc2923>.

[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>.

[RFC3692] Narten, T., "Assigning Experimental and Testing Numbers Considered Useful", BCP 82, RFC 3692, DOI 10.17487/RFC3692, January 2004, <https://www.rfc-editor.org/info/rfc3692>.

[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>.

[RFC3819] Karn, P., Ed., Bormann, C., Fairhurst, G., Grossman, D., Ludwig, R., Mahdavi, J., Montenegro, G., Touch, J., and L. Wood, "Advice for Internet Subnetwork Designers", BCP 89, RFC 3819, DOI 10.17487/RFC3819, July 2004, <https://www.rfc-editor.org/info/rfc3819>.

[RFC3879] Huitema, C. and B. Carpenter, "Deprecating Site Local Addresses", RFC 3879, DOI 10.17487/RFC3879, September 2004, <https://www.rfc-editor.org/info/rfc3879>.

[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>.

[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>.

[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>.

[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>.

[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>.

[RFC4429] Moore, N., "Optimistic Duplicate Address Detection (DAD) for IPv6", RFC 4429, DOI 10.17487/RFC4429, April 2006, <https://www.rfc-editor.org/info/rfc4429>.

[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>.

[RFC4821] Mathis, M. and J. Heffner, "Packetization Layer Path MTU Discovery", RFC 4821, DOI 10.17487/RFC4821, March 2007, <https://www.rfc-editor.org/info/rfc4821>.

[RFC4963] Heffner, J., Mathis, M., and B. Chandler, "IPv4 Reassembly Errors at High Data Rates", RFC 4963, DOI 10.17487/RFC4963, July 2007, <https://www.rfc-editor.org/info/rfc4963>.

[RFC5175] Haberman, B., Ed. and R. Hinden, "IPv6 Router Advertisement Flags Option", RFC 5175, DOI 10.17487/RFC5175, March 2008, <https://www.rfc-editor.org/info/rfc5175>.

[RFC5213] Gundavelli, S., Ed., Leung, K., Devarapalli, V., Chowdhury, K., and B. Patil, "Proxy Mobile IPv6", RFC 5213, DOI 10.17487/RFC5213, August 2008, <https://www.rfc-editor.org/info/rfc5213>.

[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>.

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

[RFC5798] Nadas, S., Ed., "Virtual Router Redundancy Protocol (VRRP) Version 3 for IPv4 and IPv6", RFC 5798, DOI 10.17487/RFC5798, March 2010, <https://www.rfc-editor.org/info/rfc5798>.

[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>.

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

[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>.

[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>.

[RFC6543] Gundavelli, S., "Reserved IPv6 Interface Identifier for Proxy Mobile IPv6", RFC 6543, DOI 10.17487/RFC6543, May 2012, <https://www.rfc-editor.org/info/rfc6543>.

[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>.

[RFC6980] Gont, F., "Security Implications of IPv6 Fragmentation with IPv6 Neighbor Discovery", RFC 6980, DOI 10.17487/RFC6980, August 2013, <https://www.rfc-editor.org/info/rfc6980>.

[RFC7042] Eastlake 3rd, D. and J. Abley, "IANA Considerations and IETF Protocol and Documentation Usage for IEEE 802 Parameters", BCP 141, RFC 7042, DOI 10.17487/RFC7042, October 2013, <https://www.rfc-editor.org/info/rfc7042>.

[RFC7084] Singh, H., Beebee, W., Donley, C., and B. Stark, "Basic Requirements for IPv6 Customer Edge Routers", RFC 7084, DOI 10.17487/RFC7084, November 2013, <https://www.rfc-editor.org/info/rfc7084>.

[RFC7421] Carpenter, B., Ed., Chown, T., Gont, F., Jiang, S., Petrescu, A., and A. Yourtchenko, "Analysis of the 64-bit Boundary in IPv6 Addressing", RFC 7421, DOI 10.17487/RFC7421, January 2015, <https://www.rfc-editor.org/info/rfc7421>.

[RFC7526] Troan, O. and B. Carpenter, Ed., "Deprecating the Anycast Prefix for 6to4 Relay Routers", BCP 196, RFC 7526, DOI 10.17487/RFC7526, May 2015, <https://www.rfc-editor.org/info/rfc7526>.

[RFC7542] DeKok, A., "The Network Access Identifier", RFC 7542, DOI 10.17487/RFC7542, May 2015, <https://www.rfc-editor.org/info/rfc7542>.

[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>.

[RFC7847] Melia, T., Ed. and S. Gundavelli, Ed., "Logical-Interface Support for IP Hosts with Multi-Access Support", RFC 7847, DOI 10.17487/RFC7847, May 2016, <https://www.rfc-editor.org/info/rfc7847>.

[RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <https://www.rfc-editor.org/info/rfc8126>.

[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>.

[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>.

[RFC8892] Thaler, D. and D. Romascanu, "Guidelines and Registration Procedures for Interface Types and Tunnel Types", RFC 8892, DOI 10.17487/RFC8892, August 2020, <https://www.rfc-editor.org/info/rfc8892>.

[RFC8899] Fairhurst, G., Jones, T., Tuexen, M., Ruengeler, I., and T. Voelker, "Packetization Layer Path MTU Discovery for Datagram Transports", RFC 8899, DOI 10.17487/RFC8899, September 2020, <https://www.rfc-editor.org/info/rfc8899>.

[RFC8900] Bonica, R., Baker, F., Huston, G., Hinden, R., Troan, O., and F. Gont, "IP Fragmentation Considered Fragile", BCP 230, RFC 8900, DOI 10.17487/RFC8900, September 2020, <https://www.rfc-editor.org/info/rfc8900>.

[RFC8981] Gont, F., Krishnan, S., Narten, T., and R. Draves, "Temporary Address Extensions for Stateless Address Autoconfiguration in IPv6", RFC 8981, DOI 10.17487/RFC8981, February 2021, <https://www.rfc-editor.org/info/rfc8981>.

Appendix A. Interface Attribute Preferences Bitmap Encoding

Adaptation of the OMNI option Interface Attributes Preferences Bitmap encoding to specific Internetworks such as the Aeronautical Telecommunications Network with Internet Protocol Services (ATN/IPS) may include link selection preferences based on other traffic classifiers (e.g., transport port numbers, etc.) in addition to the existing DSCP-based preferences. Nodes on specific Internetworks maintain a map of traffic classifiers to additional P[*] preference fields beyond the first 64. For example, TCP port 22 maps to P[67], TCP port 443 maps to P[70], UDP port 8060 maps to P[76], etc.

Implementations use Simplex or Indexed encoding formats for P[*] encoding in order to encode a given set of traffic classifiers in the most efficient way. Some use cases may be more efficiently coded using Simplex form, while others may be more efficient using Indexed. Once a format is selected for preparation of a single Interface Attribute the same format must be used for the entire Interface Attribute sub-option. Different sub-options may use different formats.

The following figures show coding examples for various Simplex and Indexed formats:

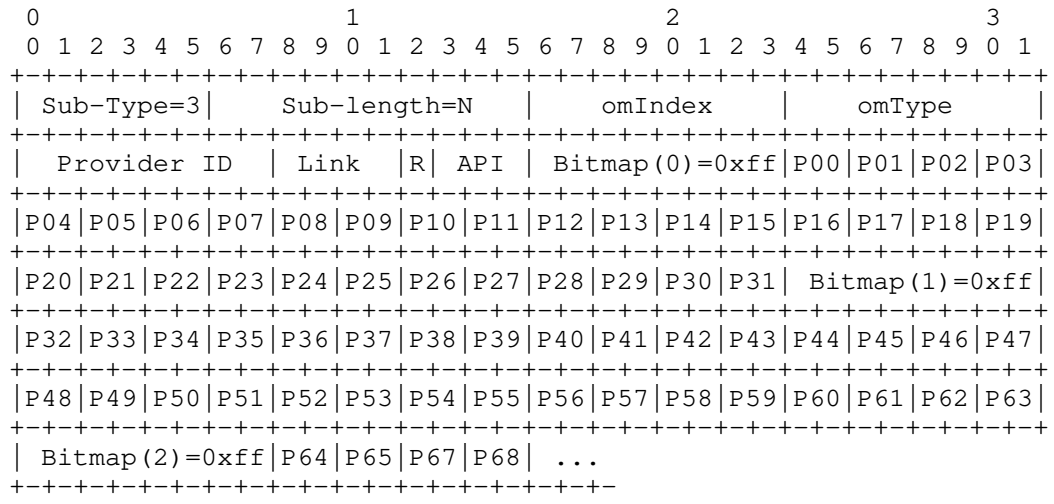


Figure 39: Example 1: Dense Simplex Encoding

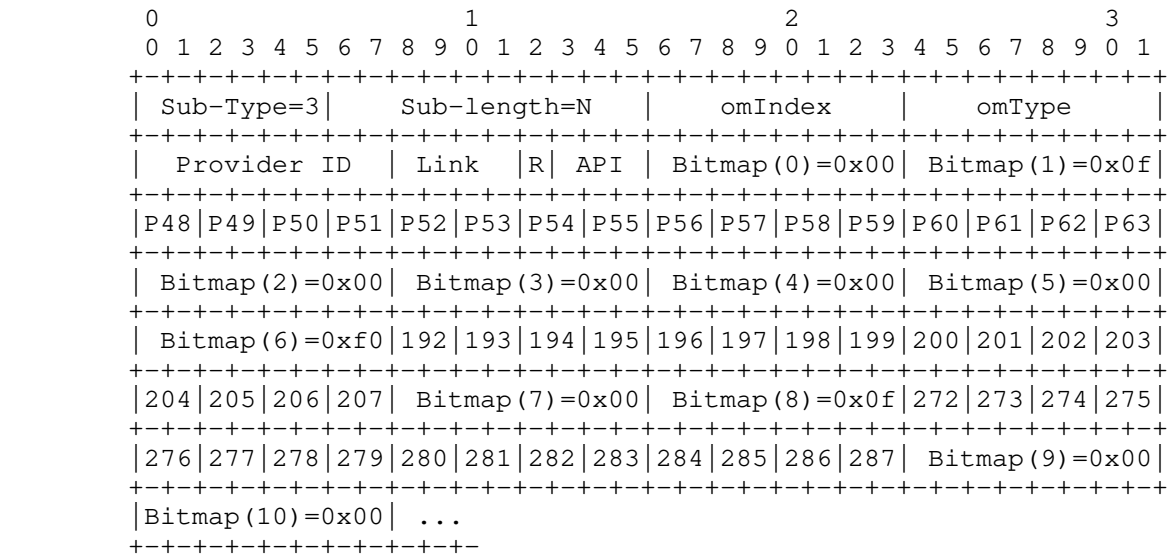


Figure 40: Example 2: Sparse Simplex Encoding

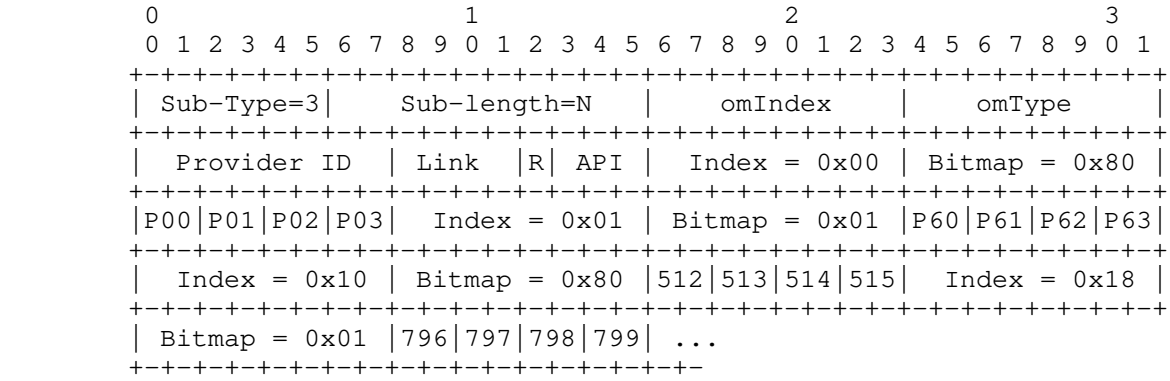


Figure 41: Example 3: Indexed Encoding

Appendix B. VDL Mode 2 Considerations

ICAO Doc 9776 is the "Technical Manual for VHF Data Link Mode 2" (VDLM2) that specifies an essential radio frequency data link service for aircraft and ground stations in worldwide civil aviation air traffic management. The VDLM2 link type is "multicast capable" [RFC4861], but with considerable differences from common multicast links such as Ethernet and IEEE 802.11.

First, the VDLM2 link data rate is only 31.5Kbps - multiple orders of magnitude less than most modern wireless networking gear. Second, due to the low available link bandwidth only VDLM2 ground stations (i.e., and not aircraft) are permitted to send broadcasts, and even so only as compact layer 2 "beacons". Third, aircraft employ the services of ground stations by performing unicast RS/RA exchanges upon receipt of beacons instead of listening for multicast RA messages and/or sending multicast RS messages.

This beacon-oriented unicast RS/RA approach is necessary to conserve the already-scarce available link bandwidth. Moreover, since the numbers of beaconing ground stations operating within a given spatial range must be kept as sparse as possible, it would not be feasible to have different classes of ground stations within the same region observing different protocols. It is therefore highly desirable that all ground stations observe a common language of RS/RA as specified in this document.

Note that links of this nature may benefit from compression techniques that reduce the bandwidth necessary for conveying the same amount of data. The IETF lpwan working group is considering possible alternatives: [https://datatracker.ietf.org/wg/lpwan/documents].

Appendix C. MN / AR Isolation Through L2 Address Mapping

Per [RFC4861], IPv6 ND messages may be sent to either a multicast or unicast link-scoped IPv6 destination address. However, IPv6 ND messaging should be coordinated between the MN and AR only without invoking other nodes on the *NET. This implies that MN / AR control messaging should be isolated and not overheard by other nodes on the link.

To support MN / AR isolation on some *NET links, ARs can maintain an OMNI-specific unicast L2 address ("MSADDR"). For Ethernet-compatible *NETs, this specification reserves one Ethernet unicast address TBD3 (see: Section 25). For non-Ethernet statically-addressed *NETs, MSADDR is reserved per the assigned numbers authority for the *NET addressing space. For still other *NETs, MSADDR may be dynamically discovered through other means, e.g., L2 beacons.

MNs map the L3 addresses of all IPv6 ND messages they send (i.e., both multicast and unicast) to MSADDR instead of to an ordinary unicast or multicast L2 address. In this way, all of the MN's IPv6 ND messages will be received by ARs that are configured to accept packets destined to MSADDR. Note that multiple ARs on the link could be configured to accept packets destined to MSADDR, e.g., as a basis for supporting redundancy.

Therefore, ARs must accept and process packets destined to MSADDR, while all other devices must not process packets destined to MSADDR. This model has well-established operational experience in Proxy Mobile IPv6 (PMIP) [RFC5213][RFC6543].

Appendix D. Change Log

<< RFC Editor - remove prior to publication >>

Differences from draft-templin-6man-omni-interface-35 to draft-templin-6man-omni-interface-36:

- Major clarifications on aspects such as "hard/soft" PTB error messages
- Made generic so that either IP protocol version (IPv4 or IPv6) can be used in the data plane.

Differences from draft-templin-6man-omni-interface-31 to draft-templin-6man-omni-interface-32:

- MTU
- Support for multi-hop ANETS such as ISATAP.

Differences from draft-templin-6man-omni-interface-29 to draft-templin-6man-omni-interface-30:

- Moved link-layer addressing information into the OMNI option on a per-ifIndex basis
- Renamed "ifIndex-tuple" to "Interface Attributes"

Differences from draft-templin-6man-omni-interface-27 to draft-templin-6man-omni-interface-28:

- Updates based on implementation experience.

Differences from draft-templin-6man-omni-interface-25 to draft-templin-6man-omni-interface-26:

- Further clarification on "aggregate" RA messages.
- Expanded Security Considerations to discuss expectations for security in the Mobility Service.

Differences from draft-templin-6man-omni-interface-20 to draft-templin-6man-omni-interface-21:

- Safety-Based Multilink (SBM) and Performance-Based Multilink (PBM).

Differences from draft-templin-6man-omni-interface-18 to draft-templin-6man-omni-interface-19:

- SEND/CGA.

Differences from draft-templin-6man-omni-interface-17 to draft-templin-6man-omni-interface-18:

- Teredo

Differences from draft-templin-6man-omni-interface-14 to draft-templin-6man-omni-interface-15:

- Prefix length discussions removed.

Differences from draft-templin-6man-omni-interface-12 to draft-templin-6man-omni-interface-13:

- Teredo

Differences from draft-templin-6man-omni-interface-11 to draft-templin-6man-omni-interface-12:

- Major simplifications and clarifications on MTU and fragmentation.

- Document now updates RFC4443 and RFC8201.

Differences from draft-templin-6man-omni-interface-10 to draft-templin-6man-omni-interface-11:

- Removed /64 assumption, resulting in new OMNI address format.

Differences from draft-templin-6man-omni-interface-07 to draft-templin-6man-omni-interface-08:

- OMNI MNs in the open Internet

Differences from draft-templin-6man-omni-interface-06 to draft-templin-6man-omni-interface-07:

- Brought back L2 MSADDR mapping text for MN / AR isolation based on L2 addressing.

- Expanded "Transition Considerations".

Differences from draft-templin-6man-omni-interface-05 to draft-templin-6man-omni-interface-06:

- Brought back OMNI option "R" flag, and discussed its use.

Differences from draft-templin-6man-omni-interface-04 to draft-templin-6man-omni-interface-05:

- Transition considerations, and overhaul of RS/RA addressing with the inclusion of MSE addresses within the OMNI option instead of as RS/RA addresses (developed under FAA SE2025 contract number DTFAWA-15-D-00030).

Differences from draft-templin-6man-omni-interface-02 to draft-templin-6man-omni-interface-03:

- Added "advisory PTB messages" under FAA SE2025 contract number DTFAWA-15-D-00030.

Differences from draft-templin-6man-omni-interface-01 to draft-templin-6man-omni-interface-02:

- Removed "Primary" flag and supporting text.

- Clarified that "Router Lifetime" applies to each ANET interface independently, and that the union of all ANET interface Router Lifetimes determines MSE lifetime.

Differences from draft-templin-6man-omni-interface-00 to draft-templin-6man-omni-interface-01:

- "All-MSEs" OMNI LLA defined. Also reserved fe80::ff00:0000/104 for future use (most likely as "pseudo-multicast").

- Non-normative discussion of alternate OMNI LLA construction form made possible if the 64-bit assumption were relaxed.

First draft version (draft-templin-atn-aero-interface-00):

- Draft based on consensus decision of ICAO Working Group I Mobility Subgroup March 22, 2019.

Authors' Addresses

Fred L. Templin (editor)
The Boeing Company
P.O. Box 3707
Seattle, WA 98124
USA

Email: fltemplin@acm.org

Tony Whyman
MWA Ltd c/o Inmarsat Global Ltd
99 City Road
London EC1Y 1AX
England

Email: tony.whyman@mccallumwhyman.com