Internet Engineering Task Force
Internet-Draft
Obsoletes: 2908 (if approved)
Intended status: Informational
Expires: April 28, 2011

P. Savola
CSC/FUNET

October 25, 2010

Overview of the Internet Multicast Addressing Architecture
draft-ietf-mboned-addrarch-07.txt

Abstract

The lack of up-to-date documentation on IP multicast address allocation and assignment procedures has caused a great deal of confusion. To clarify the situation, this memo describes the allocation and assignment techniques and mechanisms currently (as of this writing) in use.

Status of this Memo

Copyright Notice

Table of Contents

1. Introduction

Good, up-to-date documentation of IP multicast is close to non-existent. Particularly, this is an issue with multicast address allocations (to networks and sites) and assignments (to hosts and applications). This problem is stressed by the fact that there exists confusing or misleading documentation on the subject [RFC2908]. The consequence is that those who wish to learn about IP multicast and how the addressing works do not get a clear view of the current situation.

The aim of this document is to provide a brief overview of multicast addressing and allocation techniques. The term ’addressing architecture’ refers to the set of addressing mechanisms and methods in an informal manner.

It is important to note that Source-specific Multicast (SSM) [RFC4607] does not have these addressing problems because SSM group addresses have only local significance; hence, this document focuses on the Any Source Multicast (ASM) model.

This memo obsoletes and re-classifies to Historic RFC 2908, and re-classifies to Historic RFCs 2776 and 2909.

1.1. Terminology: Allocation or Assignment

Almost all multicast documents and many other RFCs (such as DHCPv4 [RFC2131] and DHCPv6 [RFC3315]) have used the terms address "allocation" and "assignment" interchangeably. However, the operator and address management communities use these terms for two conceptually different processes.

In unicast operations, address allocations refer to leasing a large block of addresses from Internet Assigned Numbers Authority (IANA) to a Regional Internet Registry (RIR) or from RIR to a Local Internet Registry (LIR) possibly through a National Internet Registry (NIR). Address assignments, on the other hand, are the leases of smaller address blocks or even single addresses to the end-user sites or end-users themselves.

Therefore, in this memo, we will separate the two different functions: "allocation" describes how larger blocks of addresses are obtained by the network operators, and "assignment" describes how applications, nodes or sets of nodes obtain a multicast address for their use.

2. Multicast Address Allocation

Multicast address allocation, i.e., how a network operator might be able to obtain a larger block of addresses, can be handled in a number of ways as described below.

Note that these are all only pertinent to ASM -- SSM requires no address block allocation because the group address has only local significance (however, we discuss the address assignment inside the node in Section 3.2).

2.1. Derived Allocation

Derived allocations take the unicast prefix or some other properties of the network (e.g., an autonomous system (AS) number) to determine unique multicast address allocations.

2.1.1. GLOP Allocation

GLOP address allocation [RFC3180] inserts the 16-bit public AS number in the middle of the IPv4 multicast prefix 233.0.0.0/8, so that each AS number can get a /24 worth of multicast addresses. While this is sufficient for multicast testing or small scale use, it might not be sufficient in all cases for extensive multicast use.

A minor operational debugging issue with GLOP addresses is that the connection between the AS and the prefix is not apparent from the prefix when the AS number is greater than 255, but has to be calculated (e.g., from [RFC3180], AS 5662 maps to 233.22.30.0/24). A usage issue is that GLOP addresses are not tied to any prefix but to routing domains, so they cannot be used or calculated automatically.

GLOP mapping is not available with 4-byte AS numbers [RFC4893]. Unicast-prefix-based Allocation or an IANA allocation from "AD-HOC Block III" (the previous so-called "eGLOP" block) could be used instead as needed.

The GLOP allocation algorithm has not been defined for IPv6 multicast because the unicast-prefix-based allocation (described below) addresses the same need in a simpler fashion.

2.1.2. Unicast-prefix-based Allocation

RFC 3306 [RFC3306] describes a mechanism which embeds up to 64 high-order bits of an IPv6 unicast address in the prefix part of the IPv6 multicast address, leaving at least 32 bits of group-id space available after the prefix mapping.

A similar IPv4 mapping is described in [RFC6034], but it provides a limited number of addresses (e.g., 1 per an IPv4 /24 block).

The IPv6 unicast-prefix-based allocations are an extremely useful way to allow each network operator, even each subnet, to obtain multicast addresses easily, through an easy computation. Further, as the IPv6 multicast header also includes the scope value [RFC4291], multicast groups of smaller scope can also be used with the same mapping.

The IPv6 Embedded RP technique [RFC3956], used with Protocol Independent Multicast - Sparse Mode (PIM-SM), further leverages the unicast-prefix-based allocations, by embedding the unicast prefix and interface identifier of the PIM-SM Rendezvous Point (RP) in the prefix. This provides all the necessary information needed to the routing systems to run the group in either inter- or intra-domain operation. A difference from RFC 3306 is, however, that the hosts cannot calculate their "multicast prefix" automatically, as the prefix depends on the decisions of the operator setting up the RP, but instead requires an assignment method.

All the IPv6 unicast-prefix-based allocation techniques provide sufficient amount of multicast address space for network operators.

2.2. Administratively Scoped Allocation

Administratively scoped multicast address allocation [RFC2365] is provided by two different means: under 239.0.0.0/8 in IPv4 or by 4-bit encoding in the IPv6 multicast address prefix [RFC4291].

Since IPv6 administratively scoped allocations can be handled with unicast-prefix-based multicast addressing as described in Section 2.1.2, we’ll only discuss IPv4 in this section.

The IPv4 administratively scoped prefix 239.0.0.0/8 is further divided into Local Scope (239.255.0.0/16) and Organization Local Scope (239.192.0.0/14); other parts of the administrative scopes are either reserved for expansion or undefined [RFC2365]. However, RFC 2365 is ambiguous as to whether the enterprises or the IETF are allowed to expand the space.

Topologies which act under a single administration can easily use the scoped multicast addresses for their internal groups. Groups which need to be shared between multiple routing domains (even if not propagated through the Internet) are more problematic and typically need an assignment of a global multicast address because their scope is undefined.

There is a large number of multicast applications (such as "Norton

Ghost") which are restricted either to a link or a site, and it is extremely undesirable to propagate them further (beyond the link or the site). Typically many such applications have been given or have hijacked a static IANA address assignment. Given the fact that assignments to typically locally used applications come from the same range as global applications, implementing proper propagation limiting is challenging. Filtering would be easier if a separate, identifiable range would be used for such assignments in the future; this is an area of further future work.

There has also been work on a protocol to automatically discover multicast scope zones [RFC2776], but it has never been widely implemented or deployed.

2.3. Static IANA Allocation

In some rare cases, organizations may have been able to obtain static multicast address allocations (of up to 256 addresses) directly from IANA. Typically these have been meant as a block of static assignments to multicast applications, as described in Section 3.4.1. If another means of obtaining addresses is available that approach is preferable.

Especially for those operators that only have a 32-bit AS number and need IPv4 addresses, an IANA allocation from "AD-HOC Block III" (the previous so-called "eGLOP" block) is an option [RFC5771].

2.4. Dynamic Allocation

RFC 2908 [RFC2908] proposed three different layers of multicast address allocation and assignment, where layers 3 (inter-domain allocation) and layer 2 (intra-domain allocation) could be applicable here. Multicast Address-Set Claim Protocol (MASC) [RFC2909] is an example of the former, and Multicast Address Allocation Protocol (AAP) [I-D.ietf-malloc-aap] (abandoned in 2000 due lack of interest and technical problems) is an example of the latter.

Both of the proposed allocation protocols were quite complex, and have never been deployed or seriously implemented.

It can be concluded that dynamic multicast address allocation protocols provide no benefit beyond GLOP/unicast-prefix-based mechanisms and have been abandoned.

3. Multicast Address Assignment

There are a number of possible ways for an application, node or set of nodes to learn a multicast address as described below.

Any IPv6 address assignment method should be aware of the guidelines for the assignment of group-IDs for IPv6 multicast addresses [RFC3307].

3.1. Derived Assignment

There are significantly fewer options for derived address assignment compared to derived allocation. Derived multicast assignment has only been specified for IPv6 link-scoped multicast [RFC4489], where the EUI64 is embedded in the multicast address, providing a node with unique multicast addresses for link-local ASM communications.

3.2. SSM Assignment inside the Node

While SSM multicast addresses have only local (to the node) significance, there is still a minor issue on how to assign the addresses between the applications running on the same IP address.

This assignment is not considered to be a problem because typically the addresses for these applications are selected manually or statically, but if done using an Application Programming Interface (API), the API could check that the addresses do not conflict prior to assigning one.

3.3. Manually Configured Assignment

With manually configured assignment, a network operator who has a multicast address prefix assigns the multicast group addresses to the requesting nodes using a manual process.

Typically, the user or administrator that wants to use a multicast address for a particular application requests an address from the network operator using phone, email, or similar means, and the network operator provides the user with a multicast address. Then the user/administrator of the node or application manually configures the application to use the assigned multicast address.

This is a relatively simple process; it has been sufficient for certain applications which require manual configuration in any case, or which cannot or do not want to justify a static IANA assignment. The manual assignment works when the number of participants in a group is small, as each participant has to be manually configured.

This is the most commonly used technique when the multicast application does not have a static IANA assignment.

3.4. Static IANA Assignment

In contrast to manually configured assignment, as described above, static IANA assignment refers to getting an assignment for the particular application directly from IANA. There are two main forms of IANA assignment: global and scope-relative. Guidelines for IANA are described in [RFC5771].

3.4.1. Global IANA Assignment

Globally unique address assignment is seen as lucrative because it’s the simplest approach for application developers since they can then hard-code the multicast address. Hard-coding requires no lease of the usable multicast address, and likewise the client applications do not need to perform any kind of service discovery (but depending on hard-coded addresses). However, there is an architectural scaling problem with this approach, as it encourages a "land-grab" of the limited multicast address space.

3.4.2. Scope-relative IANA Assignment

IANA also assigns numbers as an integer offset from the highest address in each IPv4 administrative scope as described in [RFC2365]. For example, the SLPv2 discovery scope-relative offset is "2", so SLPv2 discovery address within IPv4 Local-Scope (239.255.0.0/16) is "239.255.255.253", within the IPv4 Organization Local-Scope (239.192.0.0/14) it is "239.195.255.253", and so on.

Similar scope-relative assignments also exist with IPv6 [RFC2375]. As IPv6 multicast addresses have much more flexible scoping, scope-relative assignments are also applicable to global scopes. The assignment policies are described in [RFC3307].

3.5. Dynamic Assignments

The layer 1 of RFC 2908 [RFC2908] described dynamic assignment from Multicast Address Allocation Servers (MAAS) to applications and nodes, with Multicast Address Dynamic Client Allocation Protocol (MADCAP) [RFC2730] as an example. Since then, other mechanisms have also been proposed (e.g., DHCPv6 assignment [I-D.jdurand-assign-addr-ipv6-multicast-dhcpv6]) but these have not gained traction.

It would be rather straightforward to deploy a dynamic assignment protocol which would lease group addresses based on a multicast prefix to applications wishing to use multicast. However, only few have implemented MADCAP, and it hasn’t been significantly deployed. So, it is not clear if the lack of deployment is due to a currently missing need. Moreover, it is not clear how widely for example the APIs for communication between the multicast application and the MADCAP client operating at the host have been implemented [RFC2771].

An entirely different approach is Session Announcement Protocol (SAP) [RFC2974]. In addition to advertising global multicast sessions, the protocol also has associated ranges of addresses for both IPv4 and IPv6 which can be used by SAP-aware applications to create new groups and new group addresses. Creating a session (and obtaining an address) is a rather tedious process which is why it isn't done all that often. It is also worth noting that the IPv6 SAP address is unroutable in the inter-domain multicast.

A conclusion about dynamic assignment protocols is that:

1. multicast is not significantly attractive in the first place,

2. most applications have a static IANA assignment and thus require no dynamic or manual assignment,

3. those that cannot be easily satisfied with IANA or manual assignment (i.e., where dynamic assignment would be desirable) are rather marginal, or

4. that there are other gaps why dynamic assignments are not seen as a useful approach (for example, issues related to service discovery/rendezvous).

In consequence, more work on rendezvous/service discovery would be needed to make dynamic assignments more useful.

4. Summary and Future Directions

This section summarizes the mechanisms and analysis discussed in this memo, and presents some potential future directions.

4.1. Prefix Allocation

A summary of prefix allocation methods for ASM is shown in Figure 1.

Sect.	Prefix allocation method	IPv4	IPv6
2.1.1	Derived: GLOP	Yes	NoNeed*
2.1.2	Derived: Unicast-prefix-based	No	Yes
2.2	Administratively scoped	Yes	NoNeed*
2.3	Static IANA allocation	Yes**	No
2.4	Dynamic allocation protocols	No	No

* = the need satisfied by IPv6 unicast-prefix-based allocation.
** = mainly using the AD-HOC block III (former "eGLOP")

Figure 1

- Only ASM is affected by the assignment/allocation issues.

- With IPv4, GLOP allocations provide a sufficient IPv4 multicast allocation mechanism for those that have 16-bit AS number. IPv4 unicast-prefix based allocation offers some addresses. IANA is also allocating from the AD-HOC block III (former "eGLOP") with especially 32-bit AS number holders in mind. Administratively scoped allocations provide the opportunity for internal IPv4 allocations.

- With IPv6, unicast-prefix-based addresses and the derivatives provide a good allocation strategy and this also works for scoped multicast addresses.

- Dynamic allocations are too complex and unnecessary a mechanism.

4.2. Address Assignment

A summary of address assignment methods is shown in Figure 2.

Sect.	Address assignment method	IPv4	IPv6
3.1	Derived: link-scope addresses	No	Yes
3.2	SSM (inside the node)	Yes	Yes
3.3	Manual assignment	Yes	Yes
3.4.1	Global IANA/RIR assignment	LastResort	LastResort
3.4.2	Scope-relative IANA assignment	Yes	Yes
3.5	Dynamic assignment protocols	Yes	Yes

Figure 2

- Manually configured assignment is typical today, and works to a sufficient degree in smaller scale.

- Global IANA assignment has been done extensively in the past. Scope-relative IANA assignment is acceptable but the size of the pool is not very high. Inter-domain routing of IPv6 IANA-assigned prefixes is likely going to be challenging and as a result that approach is not very appealing.

- Dynamic assignment, e.g., MADCAP has been implemented, but there is no wide deployment. Therefore, either there are other gaps in the multicast architecture or there is no sufficient demand for it in the first place when manual and static IANA assignments are available. Assignments using SAP also exist but are not common; global SAP assignment is unfeasible with IPv6.

- Derived assignments are only applicable in a fringe case of link-scoped multicast.

4.3. Future Actions

- Multicast address discovery/"rendezvous" needs to be analyzed at more length, and an adequate solution provided. See [I-D.ietf-mboned-addrdisc-problems] and [I-D.ietf-mboned-session-announcement-req] for more.

- The IETF should consider whether to specify more ranges of the IPv4 administratively scoped address space for static allocation for applications which should not be routed over the Internet (such as backup software, etc. -- so that these wouldn’t need to use global addresses which should never leak in any case).

- The IETF should consider its static IANA allocations policy, e.g., "locking it down" to a stricter policy (like "IETF Consensus") and looking at developing the discovery/rendezvous functions, if necessary.

5. Acknowledgements

Tutoring a couple of multicast-related papers, the latest by Kaarle Ritvanen [RITVANEN] convinced the author that updated multicast address assignment/allocation documentation is needed.

Multicast address allocations/assignments were discussed at the MBONED WG session at IETF59 [MBONED-IETF59].

Dave Thaler, James Lingard, and Beau Williamson provided useful feedback for the preliminary version of this memo. Myung-Ki Shin, Jerome Durand, John Kristoff, Dave Price, Spencer Dawkins, and Alfred Hoenes also suggested improvements.

6. IANA Considerations

This memo includes no request to IANA.

IANA considerations in sections 4.1.1 and 4.1.2 of obsoleted and now Historic [RFC2908] were never implemented in IANA registry. No update is necessary.

(RFC-editor: This section may be removed prior to publication; alternatively, the second paragraph may be left intact.)

7. Security Considerations

This memo only describes different approaches to allocating and assigning multicast addresses, and this has no security considerations; the security analysis of the mentioned protocols is out of scope of this memo.

Obviously, especially the dynamic assignment protocols are inherently vulnerable to resource exhaustion attacks, as discussed e.g., in [RFC2730].

8. References

8.1. Normative References

[RFC2365] Meyer, D., "Administratively Scoped IP Multicast", BCP 23, RFC 2365, July 1998.

[RFC3180] Meyer, D. and P. Lothberg, "GLOP Addressing in 233/8", BCP 53, RFC 3180, September 2001.

[RFC3306] Haberman, B. and D. Thaler, "Unicast-Prefix-based IPv6 Multicast Addresses", RFC 3306, August 2002.

[RFC3307] Haberman, B., "Allocation Guidelines for IPv6 Multicast Addresses", RFC 3307, August 2002.

[RFC3956] Savola, P. and B. Haberman, "Embedding the Rendezvous Point (RP) Address in an IPv6 Multicast Address", RFC 3956, November 2004.

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

[RFC4489] Park, J-S., Shin, M-K., and H-J. Kim, "A Method for Generating Link-Scoped IPv6 Multicast Addresses", RFC 4489, April 2006.

[RFC4607] Holbrook, H. and B. Cain, "Source-Specific Multicast for IP", RFC 4607, August 2006.

[RFC5771] Cotton, M., Vegoda, L., and D. Meyer, "IANA Guidelines for IPv4 Multicast Address Assignments", BCP 51, RFC 5771, March 2010.

[RFC6034] Thaler, D., "Unicast-Prefix-Based IPv4 Multicast Addresses", RFC 6034, October 2010.

8.2. Informative References

[I-D.ietf-malloc-aap]
Handley, M. and S. Hanna, "Multicast Address Allocation Protocol (AAP)", June 2000.

[I-D.ietf-mboned-addrdisc-problems]
Savola, P., "Lightweight Multicast Address Discovery Problem Space", draft-ietf-mboned-addrdisc-problems-02 (work in progress), March 2006.

[I-D.ietf-mboned-session-announcement-req]
Asaeda, H. and V. Roca, "Requirements for IP Multicast

Session Announcement", draft-ietf-mboned-session-announcement-req-03 (work in progress), March 2010.

[I-D.jdurand-assign-addr-ipv6-multicast-dhcpv6] Durand, J., "IPv6 multicast address assignment with DHCPv6", draft-jdurand-assign-addr-ipv6-multicast-dhcpv6-01 (work in progress), February 2005.

[MBONED-IETF59] "MBONED WG session at IETF59", <http://www.ietf.org/proceedings/04mar/172.htm>.

[RFC2131] Droms, R., "Dynamic Host Configuration Protocol", RFC 2131, March 1997.

[RFC2375] Hinden, R. and S. Deering, "IPv6 Multicast Address Assignments", RFC 2375, July 1998.

[RFC2730] Hanna, S., Patel, B., and M. Shah, "Multicast Address Dynamic Client Allocation Protocol (MADCAP)", RFC 2730, December 1999.

[RFC2771] Finlayson, R., "An Abstract API for Multicast Address Allocation", RFC 2771, February 2000.

[RFC2776] Handley, M., Thaler, D., and R. Kermode, "Multicast-Scope Zone Announcement Protocol (MZAP)", RFC 2776, February 2000.

[RFC2908] Thaler, D., Handley, M., and D. Estrin, "The Internet Multicast Address Allocation Architecture", RFC 2908, September 2000.

[RFC2909] Radoslavov, P., Estrin, D., Govindan, R., Handley, M., Kumar, S., and D. Thaler, "The Multicast Address-Set Claim (MASC) Protocol", RFC 2909, September 2000.

[RFC2974] Handley, M., Perkins, C., and E. Whelan, "Session Announcement Protocol", RFC 2974, October 2000.

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

[RFC4893] Vohra, Q. and E. Chen, "BGP Support for Four-octet AS Number Space", RFC 4893, May 2007.

[RITVANEN]
Ritvanen, K., "Multicast Routing and Addressing", HUT Report, Seminar on Internetworking, May 2004, <http://www.tml.hut.fi/Studies/T-110.551/2004/papers/>.

Appendix A. Changes

(To be removed prior to publication as an RFC.)

A.1. Changes between -06 and -07

- Update uni-based-mcast and iana updates references to point to RFCs.

A.2. Changes between -05 and -06

- Editorial updates.
- Obsolete only RFC2908; the rest only move to Historic.
- Category is Informational instead of BCP (in line with the routing architecture.
- Move 3171bis and v4-uni-based to Normative references in order to make sure we don’t go forward until they’re resolved.
- Resolve pending issues per IETF75 discussion, in particular major changes to eGLOP and IANA policy discussions.

A.3. Changes between -04 and -05

- Editorial updates. These and the following are from Spencer Dawkins.
- New text explicitly stating that GLOP for v6 is not needed and GLOP for 4byte ASNs isn’t (and likely won’t be) defined.
- Expand reasons for filtering difficulties with global IANA assignments for local apps, and that it would be easier if these were done from the local pool.
- Explicitly mention dynamic allocations protocols’ lack of benefit and abandonment.

A.4. Changes between -03 and -04

- S/scope-relative/administratively scoped/ and expand Static IANA Assignment section to two subsections; mainly from Dave Price.

- Mention the routing challenges of IPv6 IANA assigned prefixes in section 4.2

A.5. Changes between -02 and -03

- Reword architectural implications of Static IANA and editorial improvements; mainly from John Kristoff.

A.6. Changes between -01 and -02

- Mention the mechanisms which haven't been so successful: eGLOP and MZAP.

- Remove the appendices on multicast address discovery (a separate draft now) and IPv4 unicast-prefix-based multicast addressing.

- Add a note on administratively scoped address space and the expansion ambiguity.

- Remove the references to draft-ietf-mboned-ipv6-issues-xx.txt

- Minor editorial cleanups.

Author's Address

Pekka Savola
CSC - Scientific Computing Ltd.
Espoo
Finland

Email: psavola@funet.fi

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: June 4, 2015

G. Bumgardner

December 1, 2014

Automatic Multicast Tunneling
draft-ietf-mboned-auto-multicast-18

Abstract

This document describes Automatic Multicast Tunneling (AMT), a protocol for delivering multicast traffic from sources in a multicast-enabled network to receivers that lack multicast connectivity to the source network. The protocol uses UDP encapsulation and unicast replication to provide this functionality.

The AMT protocol is specifically designed to support rapid deployment by requiring minimal changes to existing network infrastructure.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

The advantages and benefits provided by multicast technologies are well known. There are a number of application areas that are ideal candidates for the use of multicast, including media broadcasting, video conferencing, collaboration, real-time data feeds, data replication, and software updates. Unfortunately, many of these applications lack multicast connectivity to networks that carry traffic generated by multicast sources. The reasons for the lack of connectivity vary, but are primarily the result of service provider policies and network limitations.

Automatic Multicast Tunneling (AMT) is a protocol that uses UDP-based encapsulation to overcome the aforementioned lack of multicast connectivity. AMT enables sites, hosts or applications that do not have native multicast access to a network with multicast connectivity to a source, to request and receive SSM [RFC4607] and ASM [RFC1112] traffic from a network that does provide multicast connectivity to that source.

2. Applicability

This document describes a protocol that may be used to deliver multicast traffic from a multicast enabled network to sites that lack multicast connectivity to the source network. This document does not describe any methods for sourcing multicast traffic from isolated sites as this topic is out of scope.

AMT is not intended to be used as a substitute for native multicast, especially in conditions or environments requiring high traffic flow. AMT uses unicast replication to reach multiple receivers and the bandwidth cost for this replication will be higher than that required if the receivers were reachable via native multicast.

AMT is designed to be deployed at the border of networks possessing native multicast capabilities where access and provisioning can be managed by the AMT service provider.

3. Terminology

3.1. Requirements Notation

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

3.2. Definitions

This document adopts the following definitions for use in describing the protocol:

Downstream:
A downstream interface or connection that faces away from the multicast distribution root or towards multicast receivers.

Upstream:
An upstream interface or connection that faces a multicast distribution root or source.

Non-Broadcast Multi-Access (NMBA):
A non-broadcast multiple-access (NBMA) network or interface is one to which multiple network nodes (hosts or routers) are attached, but where packets are transmitted directly from one node to another node over a virtual circuit or physical link. NBMA networks do not support multicast or broadcast traffic - a node that sources multicast traffic must replicate the multicast packets for separate transmission to each node that has requested the multicast traffic.

Multicast Receiver:
An entity that requests and receives multicast traffic. A receiver may be a router, host, application, or application component. The method by which a receiver transmits group

membership requests and receives multicast traffic varies according to receiver type.

Group Membership Database:
A group membership database describes the current multicast subscription state for an interface or system. See Section 3 in [RFC3376] for a detailed definition.

Reception State:
The multicast subscription state of a pseudo, virtual or physical network interface. Often synonymous with group membership database.

Subscription:
A group or state entry in a group membership database or reception state table. The presence of a subscription entry indicates membership in an IP multicast group.

Group Membership Protocol:
The term "group membership protocol" is used as a generic reference to the Internet Group Management (IGMP) ([RFC1112], [RFC2236], [RFC3376]) or Multicast Listener Discovery ([RFC2710], [RFC3810]) protocols.

Multicast Protocol:
The term "multicast protocol" is used as a generic reference to multicast routing protocols used to join or leave multicast distribution trees such as PIM-SM [RFC4601].

Network Address Translation (NAT):
Network Address Translation is the process of modifying the source IP address and port numbers carried by an IP packet while transiting a network node (See [RFC2663]). Intervening NAT devices may change the source address and port carried by messages sent from an AMT gateway to an AMT relay, possibly producing changes in protocol state and behavior.

Anycast:
A network addressing and routing method in which packets from a single sender are routed to the topologically nearest node in a group of potential receivers all identified by the same destination address. See [RFC4786].

3.3. Abbreviations

AMT - Automatic Multicast Tunneling Protocol.

ASM - Any-Source Multicast.

DoS - Denial-of-Service (attack) and DDoS for distributed-DoS.

IGMP - Internet Group Management Protocol (v1, v2 and v3).

IP - Internet Protocol (v4 and v6).

MAC - Message Authentication Code (or Cookie).

MLD - Multicast Listener Discovery protocol (v1 and v2).

NAT - Network Address Translation (or translation node).

NBMA - Non-Broadcast Multi-Access (network, interface or mode)

SSM - Source-Specific Multicast.

PIM - Protocol Independent Multicast.

4. Protocol Overview

This section provides an informative description of the protocol. A normative description of the protocol and implementation requirements may be found in section Section 5.

4.1. General Architecture

```
   Isolated Site |    Unicast Network    |  Native Multicast
                 |      (Internet)       |
                 |                       |
                 |                       |
                 |   Group Membership    |
      +-------+ ==========================> +-------+ Multicast +------+
      |Gateway|  |                       |  | Relay |<----//----|Source|
      +-------+ <========================== +-------+           +------+
                 |   Multicast Data      |
                 |                       |
                 |                       |
```

Figure 1: Basic AMT Architecture

The AMT protocol employs a client-server model in which a "gateway" sends requests to receive specific multicast traffic to a "relay" which responds by delivering the requested multicast traffic back to the gateway.

Gateways are generally deployed within networks that lack multicast support or lack connectivity to a multicast-enabled network containing multicast sources of interest.

Relays are deployed within multicast-enabled networks that contain, or have connectivity to, multicast sources.

4.1.1. Relationship to IGMP and MLD Protocols

AMT relies on the Internet Group Management (IGMP) [RFC3376] and Multicast Listener Discovery (MLD) [RFC3810] protocols to provide the functionality required to manage, communicate, and act on changes in multicast group membership. A gateway or relay implementation does not necessarily require a fully-functional, conforming implementation of IGMP or MLD to adhere to this specification, but the protocol description that appears in this document assumes that this is the case. The minimum functional and behavioral requirements for the IGMP and MLD protocols are described in Section 5.2.1 and Section 5.3.1.

```
               Gateway                          Relay

                 General _____         _____
     ___________  Query |     |       |     | Query  ___________
    |           |<------|     |       |     |<------|           |
    | Host Mode |       | AMT |       | AMT |       |Router Mode|
    | IGMP/MLD  |       |     |  UDP  |     |       | IGMP/MLD  |
    |___________|------>|     |<----->|     |------>|___________|
                 Report |     |       |     | Report
             Leave/Done |     |       |     | Leave/Done
                        |     |       |     |
    IP Multicast <------|     |       |     |<------ IP Multicast
                        |_____|       |_____|
```

Figure 2: Multicast Reception State Managed By IGMP/MLD

A gateway runs the host portion of the IGMP and MLD protocols to generate group membership updates that are sent via AMT messages to a relay. A relay runs the router portion of the IGMP and MLD protocols to process the group membership updates to produce the required changes in multicast forwarding state. A relay uses AMT messages to send incoming multicast IP datagrams to gateways according to their current group membership state.

The primary function of AMT is to provide the handshaking, encapsulation and decapsulation required to transport the IGMP and MLD messages and multicast IP datagrams between the gateways and relays. The IGMP and MLD messages that are exchanged between gateways and relays are encapsulated as complete IP datagrams within AMT control messages. Multicast IP datagrams are replicated and encapsulated in AMT data messages. All AMT messages are sent via unicast UDP/IP.

4.1.2. Gateways

The downstream side of a gateway services one or more receivers - the gateway accepts group membership requests from receivers and forwards requested multicast traffic back to those receivers. The gateway functionality may be directly implemented in the host requesting the multicast service or within an application running on a host.

The upstream side of a gateway connects to relays. A gateway sends encapsulated IGMP and MLD messages to a relay to indicate an interest in receiving specific multicast traffic.

4.1.2.1. Architecture

Each gateway possesses a logical pseudo-interface:

```
     join/leave ---+                   +----------+
                   |                   |          |
                   V      IGMPv3/MLDv2 |          |
              +---------+ General Query|          |   AMT
              |IGMP/MLD |<-------------|   AMT    | Messages +------+
              |Host Mode|              | Gateway  |<-------->|UDP/IP|
              |Protocol |------------->|Pseudo I/F|          +------+
              +---------+   IGMP/MLD   |          |             ^
                             Report    |          |             |
                           Leave/Done  |          |             V
    IP Multicast <---------------------|          |           +---+
                                       +----------+           |I/F|
                                                              +---+
```

Figure 3: AMT Gateway Pseudo-Interface

The pseudo-interface is conceptually a network interface on which the gateway executes the host portion of the IPv4/IGMP (v2 or v3) and IPv6/MLD (v1 or v2) protocols. The multicast reception state of the pseudo-interface is manipulated using the IGMP or MLD service interface. The IGMP and MLD host protocols produce IP datagrams containing group membership messages that the gateway will send to the relay. The IGMP and MLD protocols also supply the retransmission and timing behavior required for protocol robustness.

All AMT encapsulation, decapsulation and relay interaction is assumed to occur within the pseudo-interface.

A gateway host or application may create separate interfaces for IPv4/IGMP and IPv6/MLD. A gateway host or application may also require additional pseudo-interfaces for each source or domain-specific relay address.

Within this document, the term "gateway" may be used as a generic reference to an entity executing the gateway protocol, a gateway pseudo-interface, or a gateway device that has one or more interfaces connected to a unicast inter-network and one or more AMT gateway pseudo-interfaces.

The following diagram illustrates how an existing host IP stack implementation might be used to provide AMT gateway functionality to a multicast application:

```
           +-----------------------------------------------------+
           |Host                                                 |
           |    ______________________________________           |
           |   |                                      |          |
           |   |    ___________________________       |          |
           |   |   |                           |      |          |
           |   |   |                           v      |          |
           |   |   |        +-----------+  +--------------+      |
           |   |   |        |Application|  |  AMT Daemon  |      |
           |   |   |        +-----------+  +--------------+      |
           |   |   | join/leave |   ^ data        ^ AMT          |
           |   |   |            |   |             |              |
           |   |   |       +----|---|-------------|-+            |
           |   |   |       |  __|   |_________    | |            |
           |   |   |       | |                |   | |            |
           |   |   |       | |       Sockets  |   | |            |
           |   |   |       +-|------+-------+-|---|-+            |
           |   |   |       | | IGMP |  TCP  | |UDP| |            |
           |   |   |       +-|------+-------+-|---|-+            |
           |   |   |       | | ^       IP     |   | |            |
           |   |   |       | | |  ____________|   | |            |
           |   |   |       | | | |                | |            |
           |   |   |       +-|-|-|----------------|-+            |
           |   |   |         | | |                |              |
           |   |   | IP(IGMP)| | |IP(UDP(data))   |IP(UDP(AMT))  |
           |   |   |         v | |                v              |
           |   |   |     +-----------+          +---+            |
           |   |   |     |Virtual I/F|          |I/F|            |
           |   |   |     +-----------+          +---+            |
           |   |   |         |   ^                ^              |
           |   |   | IP(IGMP)|   |IP(UDP(data))   |              |
           |   |   |_________|   |IP(IGMP)        |              |
           |   |                 |                |              |
           |   |_________________|                |              |
           |                                      |              |
           +--------------------------------------|--------------+
                                                  v
                                              AMT Relay
```

Figure 4: Virtual Interface Implementation Example

In this example, the host IP stack uses a virtual network interface to interact with a gateway pseudo-interface implementation.

4.1.2.2. Use-Cases

Use-cases for gateway functionality include:

IGMP/MLD Proxy
An IGMP/MLD proxy that runs AMT on an upstream interface and router-mode IGMP/MLD on downstream interfaces to provide host access to multicast traffic via the IGMP and MLD protocols.

Virtual Network Interface
A virtual network interface or pseudo network device driver that runs AMT on a physical network interface to provide socket layer access to multicast traffic via the IGMP/MLD service interface provided by the host IP stack.

Application
An application or application component that implements and executes IGMP/MLD and AMT internally to gain access to multicast traffic.

4.1.3. Relays

The downstream side of a relay services gateways - the relay accepts encapsulated IGMP and MLD group membership messages from gateways and encapsulates and forwards the requested multicast traffic back to those gateways.

The upstream side of a relay communicates with a native multicast infrastructure - the relay sends join and prune/leave requests towards multicast sources and accepts requested multicast traffic from those sources.

4.1.3.1. Architecture

Each relay possesses a logical pseudo-interface:

```
                                    +------------------------------+
                  +--------+        | Multicast Control Plane      |
                  |        |IGMP/MLD|                              |
                  |        | Query* | +------------+  +----------+ |
                  |        |<---//----|IGMPv3/MLDv2|  |Multicast | |
           AMT    |        |        | |Router Mode |->|Routing   |<->
+------+ Messages | AMT    |----//--->|Protocol    |  |Protocol  | |
|UDP/IP|<-------->| Relay  |IGMP/MLD| +------------+  +----------+ |
+------+          | Pseudo | Report |      |               |       |
   ^              | I/F    | Leave/ +------|---------------|-------+
   |              |        |  Done         |               |
   |              |        |               v               |
   V              |        | IP        +-----------+       |
 +---+            |        | Multicast |Multicast  |<------+
 |I/F|            |        |<---//-----|Forwarding |
 +---+            +--------+           |Plane      |<--- IP Multicast
                                       +-----------+
```

* Queries, if generated, are consumed by the pseudo-interface.

Figure 5: AMT Relay Pseudo-Interface (Router-Based)

The pseudo-interface is conceptually a network interface on which the relay runs the router portion of the IPv4/IGMPv3 and IPv6/MLDv2 protocols. Relays do not send unsolicited IGMPv3/MLDv2 query messages to gateways so relays must consume or discard any local queries normally generated by IGMPv3 or MLDv2. Note that the protocol mandates the use of IGMPv3 and MLDv2 for query messages. The AMT protocol is primarily intended for use in SSM applications and relies on several values provided by IGMPv3/MLDv2 to control gateway behavior.

A relay maintains group membership state for each gateway connected through the pseudo-interface as well as for the entire pseudo-interface (if multiple gateways are managed via a single interface). Multicast packets received on upstream interfaces on the relay are routed to the pseudo-interface where they are replicated, encapsulated and sent to interested gateways. Changes in the pseudo-interface group membership state may trigger the transmission of multicast protocol requests upstream towards a given source or rendezvous point and cause changes in internal routing/forwarding state.

The relay pseudo-interface is a architectural abstraction used to describe AMT protocol operation. For the purposes of this document, the pseudo-interface is most easily viewed as an interface to a single gateway - encapsulation, decapsulation, and other AMT-specific processing occurs "within" the pseudo-interface while forwarding and replication occur outside of it.

An alternative view is to treat the pseudo-interface as a non-broadcast multi-access (NBMA) network interface whose link layer is the unicast-only network over which AMT messages are exchanged with gateways. Individual gateways are conceptually treated as logical NBMA links on the interface. In this architectural model, group membership tracking, replication and forwarding functions occur in the pseudo-interface.

This document does not specify any particular architectural solution - a relay developer may choose to implement and distribute protocol functionality as required to take advantage of existing relay platform services and architecture.

Within this document, the term "relay" may be used as a generic reference to an entity executing the relay protocol, a relay pseudo-interface, or a relay device that has one or more network interfaces with multicast connectivity to a native multicast infrastructure, zero or more interfaces connected to a unicast inter-network, and one or more relay pseudo-interfaces.

4.1.3.2. Use-Cases

Use-cases for relay functionality include:

Multicast Router
A multicast router that runs AMT on a downstream interface to provide gateway access to multicast traffic. A "relay router" uses a multicast routing protocol (e.g. PIM-SM RFC4601 [RFC4601]) to construct a forwarding path for multicast traffic by sending join and prune messages to neighboring routers to join or leave multicast distribution trees for a given SSM source or ASM rendezvous point.

IGMP/MLD Proxy Router
An IGMP/MLD proxy that runs AMT on a downstream interface and host-mode IGMPv3/MLDv2 on a upstream interface. This "relay proxy" sends group membership reports to a local, multicast-enabled router to join and leave specific SSM or ASM groups.

4.1.4. Deployment

The AMT protocol calls for a relay deployment model that uses anycast addressing [RFC1546][RFC4291] to pair gateways with relays.

Under this approach, one or more relays advertise a route for the same IP address prefix. To find a relay with which to communicate, a gateway sends a message to an anycast IP address within that prefix. This message is routed to the topologically-nearest relay that has advertised the prefix. The relay that receives the message responds by sending its unicast address back to the gateway. The gateway uses this address as the destination address for any messages it subsequently sends to the relay.

The use of anycast addressing provides the following benefits:

- Relays may be deployed at multiple locations within a single multicast-enabled network. Relays might be installed "near" gateways to reduce bandwidth requirements, latency and limit the number of gateways that might be serviced by a single relay.

- Relays may be added or removed at any time thereby allowing staged deployment, scaling and hot-swapping - the relay discovery process will always return the nearest operational relay.

- Relays may take themselves offline when they exhaust resources required to service additional gateways. Existing gateway connections may be preserved, but new gateway requests would be routed to the next-nearest relay.

4.1.4.1. Public Versus Private

Ideally, the AMT protocol would provide a universal solution for connecting receivers to multicast sources - that any gateway could be used to access any globally advertised multicast source via publicly-accessible, widely-deployed relays. Unfortunately, today’s Internet does not yet allow this, because many relays will lack native multicast access to sources even though they may be globally accessible via unicast.

In these cases, a provider may deploy relays within their own source network to allow for multicast distribution within that network. Gateways that use these relays must use a provider-specific relay discovery mechanism or a private anycast address to obtain access to these relays.

4.1.4.2. Congestion Considerations

AMT relies on UDP to provide best-effort delivery of multicast data to gateways. Neither AMT or the UDP protocol provide the congestion control mechanisms required to regulate the flow of data messages passing through a network. While congestion remediation might be provided by multicast receiver applications via multicast group selection or upstream reporting mechanisms, there are no means by which to ensure such mechanisms are employed. To limit the possible congestion across a network or wider Internet, AMT service providers are expected to deploy AMT relays near the provider’s network border and its interface with edge routers. The provider must limit relay address advertisements to those edges to prevent distant gateways from being able to access a relay and potentially generate flows that consume or exceed the capacity of intervening links.

4.1.5. Discovery

To execute the gateway portion of the protocol, a gateway requires a unicast IP address of an operational relay. This address may be obtained using a number of methods - it may be statically assigned or dynamically chosen via some form of relay discovery process.

As described in the previous section, the AMT protocol provides a relay discovery method that relies on anycast addressing. Gateways are not required to use AMT relay discovery, but all relay implementations must support it.

The AMT protocol uses the following terminology when describing the discovery process:

Relay Discovery Address Prefix:
The anycast address prefix used to route discovery messages to a relay.

Relay Discovery Address:
The anycast destination address used when sending discovery messages.

Relay Address:
The unicast IP address obtained as a result of the discovery process.

4.1.5.1. Relay Discovery Address Selection

The selection of an anycast Relay Discovery Address may be source-dependent, as a relay located via relay discovery must have multicast connectivity to a desired source.

Similarly, the selection of a unicast Relay Address may be source-dependent, as a relay contacted by a gateway to supply multicast traffic must have native multicast connectivity to the traffic source

Methods that might be used to perform source-specific or group-specific relay selection are highly implementation-dependent and are not further addressed by this document. Possible approaches include the use of static lookup tables, DNS-based queries, or a provision of a service interface that accepts join requests on (S,G,relay-discovery-address) or (S,G,relay-address) tuples.

4.1.5.2. IANA-Assigned Relay Discovery Address Prefix

IANA has assigned an address prefix for use in advertising and discovering publicly accessible relays.

A relay discovery address is constructed from the address prefix by setting the low-order octet of the prefix address to 1 (for both IPv4 and IPv6).

Public relays must advertise a route to the address prefix (e.g. via BGP [RFC4271]) and configure an interface to respond to the relay discovery address.

The IANA address assignments are discussed in Section 7.

4.2. General Operation

4.2.1. Message Sequences

The AMT protocol defines the following messages for control and encapsulation. These messages are exchanged as UDP/IP datagrams, one message per datagram.

Relay Discovery:
Sent by gateways to solicit a Relay Advertisement from any relay. Used to find a relay with which to communicate.

Relay Advertisement:
Sent by relays as a response to a Relay Discovery message. Used to deliver a relay address to a gateway.

Request:
Sent by gateways to solicit a Membership Query message from a relay.

Membership Query:
Sent by relays as a response to a Request message. Used to deliver an encapsulated IGMPv3 or MLDv2 query message to the gateway.

Membership Update:
Sent by gateways to deliver an encapsulated IGMP or MLD report/leave/done message to a relay.

Multicast Data:
Sent by relays to deliver an encapsulated IP multicast datagram or datagram fragment to a gateway.

Teardown:
Sent by gateways to stop the delivery of Multicast Data messages requested in an earlier Membership Update message.

The following sections describe how these messages are exchanged to execute the protocol.

4.2.1.1. Relay Discovery Sequence

```
                 Gateway               Relay
                 -------               -----
                    :                    :
                    |                    |
                [1] |Relay Discovery     |
                    |------------------->|
                    |                    |
                    | Relay Advertisement| [2]
                    |<-------------------|
                [3] |                    |
                    :                    :
```

Figure 6: AMT Relay Discovery Sequence

The following sequence describes how the Relay Discovery and Relay Advertisement messages are used to find a relay with which to communicate:

1. The gateway sends a Relay Discovery message containing a random nonce to the Relay Discovery Address. If the Relay Discovery Address is an anycast address, the message is routed to topologically-nearest network node that advertises that address.

2. The node receiving the Relay Discovery message sends a Relay Advertisement message back to the source of the Relay Discovery message. The message carries a copy of the nonce contained in the Relay Discovery message and the unicast IP address of a relay.

3. When the gateway receives the Relay Advertisement message it verifies that the nonce matches the one sent in the Relay Discovery message, and if it does, uses the relay address carried by the Relay Advertisement as the destination address for subsequent AMT messages.

Note that the responder need not be a relay - the responder may obtain a relay address by some other means and return the result in the Relay Advertisement (i.e., the responder is a load-balancer or broker).

4.2.1.2. Membership Update Sequence

There exists a significant difference between normal IGMP and MLD behavior and that required by AMT. An IGMP/MLD router acting as a querier normally transmits query messages on a network interface to construct and refresh group membership state for the connected network. These query messages are multicast to all IGMP/MLD enabled hosts on the network. Each host responds by multicasting report messages that describe their current multicast reception state.

However, AMT does not allow relays to send unsolicited query messages to gateways, as the set of active gateways may be unknown to the relay and potentially quite large. Instead, AMT requires each gateway to periodically send a message to a relay to solicit a general-query response. A gateway accomplishes this by sending a Request message to a relay. The relay responds by sending Membership Query message back to the gateway. The Membership Query message carries an encapsulated general query that is processed by the IGMP or MLD protocol implementation on the gateway to produce a membership/listener report. Each time the gateway receives a Membership Query message it starts a timer whose expiration will trigger the start of a new Request->Membership Query message exchange. This timer-driven sequence is used to mimic the transmission of a periodic general query by an IGMP/MLD router. This query cycle may continue indefinitely once started by sending the initial Request message.

A membership update occurs when an IGMP or MLD report, leave or done message is passed to the gateway pseudo-interface. These messages may be produced as a result of the aforementioned general-query processing or as a result of receiver interaction with the IGMP/MLD service interface. Each report is encapsulated and sent to the relay after the gateway has successfully established communication with the relay via a Request and Membership Query message exchange. If a report is passed to the pseudo-interface before the gateway has received a Membership Query message from the relay, the gateway may discard the report or queue the report for delivery after a Membership Query is received. Subsequent IGMP/MLD report/leave/done messages that are passed to the pseudo-interface are immediately encapsulated and transmitted to the relay.

```
           IGMP/MLD             Pseudo-I/F              Relay
           --------             ----------              -----
              :                     :                     :
              |                     |       Request       |
              |                    1|-------------------->|
              |                     |  Membership Query   |2
    Query     |                     |       Q(0,{})       |
    Timer     |         Start      3|<--------------------|
     (QT)<--------------------------|                     |
              |        Q(0,{})      |                     |
              |<--------------------|                     |
             4|         R({})       |  Membership Update  |
              |-------------------->|5       R({})        |
              |                     |====================>|6a
    Join(S,G) :                     :                     :
   ()-------->|7 R({G:ALLOW({S})})  |  Membership Update  |
              |-------------------->|8  R({G:ALLOW({S})}) |
              |                     |====================>|9a  Join(S,G)
              |                     |                     |---------->()
              :                     :                     :
              |         ------------|---------------------|------------
              |        |            |                     |            |
              |        |            |    Multicast Data   |  IP(S,G)   |
              |        |            |       IP(S,G)     10|<--------() |
              |        |  IP(S,G) 11|<====================|            |
              |        | ()<--------|                     |            |
              |        |            |                     |            |
              :         ------------:---------------------:------------
              |       Expired       |                     |
     (QT)-------------------------->|12      Request      |
              |                    1|-------------------->|
              |                     |  Membership Query   |2
              |                     |       Q(0,{})       |
              |        Start       3|<--------------------|
     (QT)<--------------------------|                     |
              |       Q(0,{})       |                     |
              |<--------------------|                     |
             4| R({G:INCLUDE({S})}) |  Membership Update  |
              |-------------------->|5 R({G:INCLUDE({S})})|
              |                     |====================>|6b
   Leave(S,G) :                     :                     :
   ()-------->|7 R({G:BLOCK({S})})  |  Membership Update  |
              |-------------------->|8  R({G:BLOCK({S})}) |
              |                     |====================>|9b Prune(S,G)
              |                     |                     |---------->()
              :                     :                     :
```

Figure 7: Membership Update Sequence (IGMPv3/MLDv2 Example)

The following sequence describes how the Request, Membership Query, and Membership Update messages are used to report current group membership state or changes in group membership state:

1. A gateway sends a Request message to the relay that contains a random nonce and a flag indicating whether the relay should return an IGMPv3 or MLDv2 general query.

2. When the relay receives a Request message, it generates a message authentication code (MAC), typically, by computing a hash digest from message source IP address, source UDP port, request nonce and a private secret. The relay then sends a Membership Query message to the gateway that contains the request nonce, the MAC, and an IGMPv3 or MLDv2 general query.

3. When the gateway receives a Membership Query message, it verifies that the request nonce matches the one sent in the last Request, and if it does, the gateway saves the request nonce and MAC for use in sending subsequent Membership Update messages. The gateway starts a timer whose expiration will trigger the transmission of a new Request message and extracts the encapsulated general query message for processing by the IGMP or MLD protocol. The query timer duration is specified by the relay in the Querier’s Query Interval Code (QQIC) field in the IGMPv3 or MLDv2 general query. The QQIC field is defined in Section 4.1.7 of [RFC3376] and Section 5.1.9 of [RFC3810]).

4. The gateway’s IGMP or MLD protocol implementation processes the general query to produce a current-state report.

5. When an IGMP or MLD report is passed to the pseudo-interface, the gateway encapsulates the report in a Membership Update message and sends it to the relay. The request nonce and MAC fields in the Membership Update are assigned the values from the last Membership Query message received for the corresponding group membership protocol (IGMPv3 or MLDv2).

6. When the relay receives a Membership Update message, it computes a MAC from the message source IP address, source UDP port, request nonce and a private secret. The relay accepts the Membership Update message if the received MAC matches the computed MAC, otherwise the message is ignored. If the message is accepted, the relay may proceed to allocate, refresh, or modify tunnel state. This includes making any group membership, routing and forwarding state changes and issuing any upstream protocol requests required to satisfy the state change. The diagram illustrates two scenarios:

A. The gateway has not previously reported any group subscriptions and the report does not contain any group subscriptions, so the relay takes no action.

B. The gateway has previously reported a group subscription so the current-state report lists all current subscriptions. The relay responds by refreshing tunnel or group state and resetting any related timers.

7. A receiver indicates to the gateway that it wishes to join (allow) or leave (block) specific multicast traffic. This request is typically made using some form IGMP/MLD service interface (as described in Section 2 of [RFC3376] or Section 3 of [RFC3810]). The IGMP/MLD protocol responds by generating an IGMP or MLD state-change message.

8. When an IGMP or MLD report/leave/done message is passed to the pseudo-interface, the gateway encapsulates the message in a Membership Update message and sends it to the relay. The request nonce and MAC fields in the Membership Update are assigned the values from the last Membership Query message received for the corresponding group membership protocol (IGMP or MLD).

   The IGMP and MLD protocols may generate multiple messages to provide robustness against packet loss - each of these must be encapsulated in a new Membership Update message and sent to the relay. The Querier Robustness Variable (QRV) field in the last IGMP/MLD query delivered to the IGMP/MLD protocol is typically used to specify the number of repetitions (i.e., the host adopts the QRV value as its own Robustness Variable value). The QRV field is defined in Section 4.1.6 in [RFC3376] and Section 5.1.8 in [RFC3810].

9. When the relay receives a Membership Update message, it again computes a MAC from the message source IP address, source UDP port, request nonce and a private secret. The relay accepts the Membership Update message if the received MAC matches the computed MAC, otherwise the message is ignored. If the message is accepted, the relay processes the encapsulated IGMP/MLD and allocates, modifies or deletes tunnel state accordingly. This includes making any group membership, routing and forwarding state changes and issuing any upstream protocol requests required to satisfy the state change. The diagram illustrates two scenarios:

   A. The gateway wishes to add a group subscription.

B. The gateway wishes to delete a previously reported group subscription.

10. Multicast datagrams transmitted from a source travel through the native multicast infrastructure to the relay. When the relay receives a multicast IP datagram that carries a source and destination address for which a gateway has expressed an interest in receiving (via the Membership Update message), it encapsulates the datagram into a Multicast Data message and sends it to the gateway using the source IP address and UDP port carried by the Membership Update message as the destination address.

11. When the gateway receives a Multicast Data message, it extracts the multicast packet from the message and passes it on to the appropriate receivers.

12. When the query timer expires the gateway sends a new Request message to the relay to start a new membership update cycle.

The MAC-based source-authentication mechanism described above provides a simple defense against malicious attempts to exhaust relay resources via source-address spoofing. Flooding a relay with spoofed Request or Membership Update messages may consume computational resources and network bandwidth, but will not result in the allocation of state because the Request message is stateless and spoofed Membership Update messages will fail source-authentication and be rejected by the relay.

A relay will only allocate new tunnel state if the IGMP/MLD report carried by the Membership Update message creates one or more group subscriptions.

A relay deallocates tunnel state after one of the following events; the gateway sends a Membership Update message containing a report that results in the deletion of all remaining group subscriptions, the IGMP/MLD state expires (due to lack of refresh by the gateway), or the relay receives a valid Teardown message from the gateway (See Section 4.2.1.3).

A gateway that accepts or reports group subscriptions for both IPv4 and IPv6 addresses will send separate Request and Membership Update messages for each protocol (IPv4/IGMP and IPv6/MLD).

4.2.1.3. Teardown Sequence

A gateway sends a Teardown message to a relay to request that it stop delivering Multicast Data messages to a tunnel endpoint created by an earlier Membership Update message. This message is intended to be used following a gateway address change (See Section 4.2.2.1) to stop the transmission of undeliverable or duplicate multicast data messages. Gateway support for the Teardown message is optional - gateways are not required to send them and may instead rely on group membership to expire on the relay.

```
                      Gateway                  Relay
                      -------                  -----
                         :        Request        :
                     [1] |           N           |
                         |---------------------->|
                         |    Membership Query   | [2]
                         |    N,MAC,gADDR,gPORT  |
                         |<======================|
                     [3] |   Membership Update   |
                         |   ({G:INCLUDE({S})})  |
                         |======================>|
                         |                       |
    ---------------------:-----------------------:---------------------
   |                     |                       |                     |
   |                     |    *Multicast Data    |  *IP Packet(S,G)    |
   |                     |      gADDR,gPORT      |<-----------------() |
   |    *IP Packet(S,G)  |<======================|                     |
   | ()<-----------------|                       |                     |
   |                     |                       |                     |
    ---------------------:-----------------------:---------------------
                         ~                       ~
                         ~        Request        ~
                     [4] |           N’          |
                         |---------------------->|
                         |   Membership Query    | [5]
                         | N’,MAC’,gADDR’,gPORT’ |
                         |<======================|
                     [6] |                       |
                         |       Teardown        |
                         |   N,MAC,gADDR,gPORT   |
                         |---------------------->|
                         |                       | [7]
                         |   Membership Update   |
                         |  ({G:INCLUDE({S})})   |
                         |======================>|
                         |                       |
    ---------------------:-----------------------:---------------------
   |                     |                       |                     |
   |                     |    *Multicast Data    |  *IP Packet(S,G)    |
   |                     |     gADDR’,gPORT’     |<-----------------() |
   |    *IP Packet (S,G) |<======================|                     |
   | ()<-----------------|                       |                     |
   |                     |                       |                     |
    ---------------------:-----------------------:---------------------
                         |                       |
                         :                       :
```

Figure 8: Teardown Message Sequence (IGMPv3/MLDv2 Example)

The following sequence describes how the Membership Query and Teardown message are used to detect an address change and stop the delivery of Multicast Data messages to an address:

1. A gateway sends a Request message containing a random nonce to the relay.

2. The relay sends a Membership Query message to the gateway that contains the source IP address (gADDR) and source UDP port (gPORT) values from the Request message. These values will be used to identify the tunnel should one be created by a subsequent Membership Update message.

3. When the gateway receives a Membership Query message that carries the gateway address fields, it compares the gateway IP address and port number values with those received in the previous Membership Query (if any). If these values do not match, this indicates that the Request message arrived at the relay carrying a different source address than the one sent previously. At this point in the sequence, no change in source address or port has occurred.

4. The gateway sends a new Request message to the relay. However, this Request message arrives at the relay carrying a different source address than that of the previous Request due to some change in network interface, address assignment, network topology or NAT mapping.

5. The relay again responds by sending a Membership Query message to the gateway that contains the new source IP address (gADDR’) and source UDP port (gPORT’) values from the Request message.

6. When the gateway receives the Membership Query message, it compares the gateway address and port number values against those returned in the previous Membership Query message.

7. If the reported address or port has changed, the gateway sends a Teardown message to the relay that contains the request nonce, MAC, gateway IP address and gateway port number returned in the earlier Membership Query message. The gateway may send the Teardown message multiple times where the number of repetitions is governed by the Querier Robustness Variable (QRV) value contained in the IGMPv3/MLDv2 general query carried by the original Membership Query (See Section 4.1.6 in [RFC3376] and Section 5.1.8 in [RFC3810]). The gateway continues to process the new Membership Query message as usual.

8. When the relay receives a Teardown message, it computes a MAC from the message source IP address, source UDP port, request nonce and a private secret. The relay accepts the Teardown message if the received MAC matches the computed MAC, otherwise the message is ignored. If the message is accepted, the relay makes any group membership, routing and forwarding state changes required to stop the transmission of Multicast Data messages to that address.

4.2.1.4. Timeout and Retransmission

The AMT protocol does not establish any requirements regarding what actions a gateway should take if it fails to receive a response from a relay. A gateway implementation may wait for an indefinite period of time to receive a response, may set a time limit on how long to wait for a response, may retransmit messages should the time limit be reached, may limit the number of retransmissions, or may simply report an error.

For example, a gateway may retransmit a Request message if it fails to receive a Membership Query or expected Multicast Data messages within some time period. If the gateway fails to receive any response to a Request after several retransmissions or within some maximum period of time, it may reenter the relay discovery phase in an attempt to find a new relay. This topic is addressed in more detail in Section 5.2.

4.2.2. Tunneling

From the standpoint of a relay, an AMT "tunnel" is identified by the IP address and UDP port pair used as the destination address for sending encapsulated multicast IP datagrams to a gateway. This address is referred here as the tunnel endpoint address.

A gateway sends a Membership Update message to a relay to add or remove group subscriptions to a tunnel endpoint. The tunnel endpoint is identified by the source IP address and source UDP port carried by the Membership Update message when it arrives at a relay (this address may differ from that carried by the message when it exited the gateway as a result of network address translation).

The Membership Update messages sent by a single gateway host may originate from several source addresses or ports - each unique combination represents a unique tunnel endpoint. A single gateway host may legitimately create and accept traffic on multiple tunnel endpoints, e.g., the gateway may use separate ports for the IPv4/IGMP and IPv6/MLD protocols.

A tunnel is "created" when a gateway sends a Membership Update message containing an IGMP or MLD membership report that creates one or more group subscriptions when none currently existed for that tunnel endpoint address.

A tunnel ceases to exist when all group subscriptions for a tunnel endpoint are deleted. This may occur as a result of the following events:

- The gateway sends an IGMP or MLD report, leave or done message to the relay that deletes the last group subscription linked to the tunnel endpoint.

- The gateway sends a Teardown message to the relay that causes it to delete any and all subscriptions bound to the tunnel endpoint.

- The relay stops receiving updates from the gateway until such time that per-group or per-tunnel timers expire, causing the relay to delete the subscriptions.

The tunneling approach described above conceptually transforms a unicast-only inter-network into an NBMA link layer, over which multicast traffic may be delivered. Each relay, plus the set of all gateways using the relay, together may be thought of as being on a separate logical NBMA link, where the "link layer" address is a UDP/IP address-port pair provided by the Membership Update message.

4.2.2.1. Address Roaming

As described above, each time a relay receives a Membership Update message from a new source address-port pair, the group subscriptions described by that message apply to the tunnel endpoint identified by that address.

This can cause problems for a gateway if the address carried by the messages it sends to a relay changes unexpectedly. These changes may cause the relay to transmit duplicate, undeliverable or unrequested traffic back towards the gateway or an intermediate device. This may create congestion and have negative consequences for the gateway, its network, or multicast receivers, and in some cases, may also produce a significant amount of ICMP traffic directed back towards the relay by a NAT, router or gateway host.

There are several scenarios in which the address carried by messages sent by a gateway may change without that gateway’s knowledge, as for example, when:

- The message originates from a different interface on a gateway that possesses multiple interfaces.

- The DHCP assignment for a gateway interface changes.

- The gateway roams to a different wireless network.

- The address mapping applied by an intervening network-translation-device (NAT) changes as a result of mapping expiration or routing changes in a multi-homed network.

In the case where the address change occurs between the transmission of a Request message and subsequent Membership Update messages, the relay will simply ignore any Membership Update messages from the new address because MAC authentication will fail (see Section 4.2.1.2). The relay may continue to transmit previously requested traffic, but no duplication will occur, i.e., the possibility for the delivery of duplicate traffic does not arise until a Request message is received from the new address.

The protocol provides a method for a gateway to detect an address change and explicitly request that the relay stop sending traffic to a previous address. This process involves the Membership Query and Teardown messages and is described in Section 4.2.1.3.

4.2.2.2. Network Address Translation

The messages sent by a gateway to a relay may be subject to network address translation (NAT) - the source IP address and UDP port carried by an IP packet sent by the gateway may be modified multiple times before arriving at the relay. In the most restrictive form of NAT, the NAT device will create a new mapping for each combination of source and destination IP address and UDP port. In this case, bi-directional communication can only be conducted by sending outgoing packets to the source address and port carried by the last incoming packet.

```
       Membership Update                 Membership Update
       src: iADDR:iPORT                  src: eADDR:ePORT
       dst: rADDR:rPORT                  dst: rADDR:rPORT
                          +---------+
                          |   NAT   |
   +---------+           +-----------+          +---------+
   |         |---------->|           |--------->|         |
   | Gateway |           |  Mapping  |          |  Relay  |
   |         |<----------|           |<---------|         |
   +---------+           +-----------+          +---------+
                          |         |
                          +---------+
       Multicast Data                    Multicast Data
       src: rADDR:rPORT                  src: rADDR:rPORT
       dst: iADDR:iPORT                  dst: eADDR:ePORT
```

Figure 9: Network Address Translation in AMT

AMT provides automatic NAT traversal by using the source IP address and UDP port carried by the Membership Update message as received at the relay as the destination address for any Multicast Data messages the relay sends back as a result.

The NAT mapping created by a Membership Update message will eventually expire unless it is refreshed by a passing message. This refresh will occur each time the gateway performs the periodic update required to refresh group state within the relay (See Section 4.2.1.2).

4.2.2.3. UDP Encapsulation

```
                Gateway                              Relay

           IP:IGMP                                       IP:IGMP
              |    AMT:IP:IGMP               AMT:IP:IGMP    |
              |         |                         |         |
              |         |   IP:UDP:AMT:IP:IGMP    |         |
    _______   |   ___   |   ______   |   ______   |   ___   |   _______
   |IGMP|IP|  v  |AMT|  v  |UDP|IP|  v  |IP|UDP|  v  |AMT|  v  |IP|IGMP|
   |    |  |     |   |     |   |  |     |  |   |     |   |     |  |    |
   |    |<------------------------------------------------------->|    |
   |____|  |     |   |     |   |  |     |  |   |     |   |     |  |____|
   |       |<--------------------------------------------------|       |
   |_______|  ^  |___|  ^  |___|__|  ^  |__|___|  ^  |___|  ^  |_______|
              |         |            |            |         |
             IP      AMT:IP    IP:UDP:AMT:IP    AMT:IP      IP
```

Figure 10: AMT Encapsulation

The IGMP and MLD messages used in AMT are exchanged as complete IP datagrams. These IP datagrams are encapsulated in AMT messages that are transmitted using UDP. The same holds true for multicast traffic - each multicast IP datagram or datagram fragment that arrives at the relay is encapsulated in an AMT message and transmitted to one or more gateways via UDP.

The IP protocol of the encapsulated packets need not match the IP protocol used to send the AMT messages. AMT messages sent via IPv4 may carry IPv6/MLD packets and AMT messages sent via IPv6 may carry IPv4/IGMP packets.

The checksum field contained in the UDP header of the messages requires special consideration. Of primary concern is the cost of computing a checksum on each replicated multicast packet after it is encapsulated for delivery to a gateway. Many routing/forwarding platforms do not possess the capability to compute checksums on UDP encapsulated packets as they may not have access to the entire datagram.

To avoid placing an undue burden on the relay platform, the protocol specifically allows zero-valued UDP checksums on the multicast data messages. This is not an issue in UDP over IPv4 as the UDP checksum field may be set to zero. However, this is a problem for UDP over IPv6 as that protocol requires a valid, non-zero checksum in UDP datagrams [RFC2460]. Messages sent over IPv6 with a UDP checksum of zero may fail to reach the gateway. This is a well known issue for UDP-based tunneling protocols that is described [RFC6936]. A recommended solution is described in [RFC6935].

4.2.2.4. UDP Fragmentation

Naive encapsulation of a multicast IP datagrams within an AMT data messages may produce UDP datagrams that might require fragmentation if their size exceeds the MTU of network path between the relay and a gateway. Many multicast applications, especially those related to media streaming, are designed to deliver independent data samples in separate packets, without fragmentation, to ensure some number of complete samples can be delivered even in the presence of packet loss. To prevent or reduce undesirable fragmentation, the AMT protocol describes specific procedures for handling multicast datagrams whose encapsulation might exceed the path MTU. These procedures are described in Section 5.3.3.6.

5. Protocol Description

This section provides a normative description of the AMT protocol.

5.1. Protocol Messages

The AMT protocol defines seven message types for control and encapsulation. These messages are assigned the following names and numeric identifiers:

Message Type	Message Name
1	Relay Discovery
2	Relay Advertisement
3	Request
4	Membership Query
5	Membership Update
6	Multicast Data
7	Teardown

These messages are exchanged as IPv4 or IPv6 UDP datagrams.

5.1.1. Relay Discovery

A Relay Discovery message is used to solicit a response from a relay in the form of a Relay Advertisement message.

The UDP/IP datagram containing this message MUST carry a valid, non-zero UDP checksum and carry the following IP address and UDP port values:

Source IP Address - The IP address of the gateway interface on which the gateway will listen for a relay response. Note: The value of this field may be changed as a result of network address translation before arriving at the relay.

Source UDP Port - The UDP port number on which the gateway will listen for a relay response. Note: The value of this field may be changed as a result of network address translation before arriving at the relay.

Destination IP Address - An anycast or unicast IP address, i.e., the Relay Discovery Address advertised by a relay.

Destination UDP Port - The IANA-assigned AMT port number (See Section 7.2).

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  V=0  |Type=1 |     Reserved                                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Discovery Nonce                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 11: Relay Discovery Message Format

5.1.1.1. Version (V)

The protocol version number for this message is 0.

5.1.1.2. Type

The type number for this message is 1.

5.1.1.3. Reserved

Reserved bits that MUST be set to zero by the gateway and ignored by the relay.

5.1.1.4. Discovery Nonce

A 32-bit random value generated by the gateway and echoed by the relay in a Relay Advertisement message. This value is used by the gateway to correlate Relay Advertisement messages with Relay Discovery messages. Discovery nonce generation is described in Section 5.2.3.4.5.

5.1.2. Relay Advertisement

The Relay Advertisement message is used to supply a gateway with a unicast IP address of a relay. A relay sends this message to a gateway when it receives a Relay Discovery message from that gateway.

The UDP/IP datagram containing this message MUST carry a valid, non-zero UDP checksum and carry the following IP address and UDP port values:

Source IP Address - The destination IP address carried by the Relay Discovery message (i.e., the Relay Discovery Address advertised by the relay).

Source UDP Port - The destination UDP port carried by the Relay Discovery message (i.e., the IANA-assigned AMT port number).

Destination IP Address - The source IP address carried by the Relay Discovery message. Note: The value of this field may be changed as a result of network address translation before arriving at the gateway.

Destination UDP Port - The source UDP port carried by the Relay Discovery message. Note: The value of this field may be changed as a result of network address translation before arriving at the gateway.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |  V=0  |Type=2 |                   Reserved                    |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                        Discovery Nonce                        |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                                                               |
   ~                  Relay Address (IPv4 or IPv6)                 ~
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 12: Relay Advertisement Message Format

5.1.2.1. Version (V)

The protocol version number for this message is 0.

5.1.2.2. Type

The type number for this message is 2.

5.1.2.3. Reserved

Reserved bits that MUST be set to zero by the relay and ignored by the gateway.

5.1.2.4. Discovery Nonce

A 32-bit value copied from the Discovery Nonce field (Section 5.1.1.4) contained in the Relay Discovery message. The gateway uses this value to match a Relay Advertisement to a Relay Discovery message.

5.1.2.5. Relay Address

The unicast IPv4 or IPv6 address of the relay. A gateway uses the length of the UDP datagram containing the Relay Advertisement message to determine the address family; i.e., length - 8 = 4 (IPv4) or 16 (IPv6). The relay returns an IP address for the protocol used to send the Relay Discovery message, i.e., an IPv4 relay address for an IPv4 discovery address or an IPv6 relay address for an IPv6 discovery address.

5.1.3. Request

A gateway sends a Request message to a relay to solicit a Membership Query response.

The successful delivery of this message marks the start of the first stage in the three-way handshake used to create or update state within a relay.

The UDP/IP datagram containing this message MUST carry a valid, non-zero UDP checksum and carry the following IP address and UDP port values:

Source IP Address - The IP address of the gateway interface on which the gateway will listen for a response from the relay. Note: The value of this field may be changed as a result of network address translation before arriving at the relay.

Source UDP Port - The UDP port number on which the gateway will listen for a response from the relay. Note: The value of this field may be changed as a result of network address translation before arriving at the relay.

Destination IP Address - The unicast IP address of the relay.

Destination UDP Port - The IANA-assigned AMT port number.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  V=0  |Type=3 |   Reserved  |P|           Reserved              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          Request Nonce                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 13: Request Message Format

5.1.3.1. Version (V)

The protocol version number for this message is 0.

5.1.3.2. Type

The type number for this message is 3.

5.1.3.3. Reserved

Reserved bits that MUST be set to zero by the gateway and ignored by the relay.

5.1.3.4. P Flag

The "P" flag is set to indicate which group membership protocol the gateway wishes the relay to use in the Membership Query response:

Value Meaning

0 The relay MUST respond with a Membership Query message that contains an IPv4 packet carrying an IGMPv3 general query message.

1 The relay MUST respond with a Membership Query message that contains an IPv6 packet carrying an MLDv2 general query message.

5.1.3.5. Request Nonce

A 32-bit random value generated by the gateway and echoed by the relay in a Membership Query message. This value is used by the relay to compute the Response MAC value and is used by the gateway to correlate Membership Query messages with Request messages. Request nonce generation is described in Section 5.2.3.5.6.

5.1.4. Membership Query

A relay sends a Membership Query message to a gateway to solicit a Membership Update response, but only after receiving a Request message from the gateway.

The successful delivery of this message to a gateway marks the start of the second-stage in the three-way handshake used to create or update tunnel state within a relay.

The UDP/IP datagram containing this message MUST carry a valid, non-zero UDP checksum and carry the following IP address and UDP port values:

Source IP Address - The destination IP address carried by the Request message (i.e., the unicast IP address of the relay).

Source UDP Port - The destination UDP port carried by the Request message (i.e., the IANA-assigned AMT port number).

Destination IP Address - The source IP address carried by the Request message. Note: The value of this field may be changed as a result of network address translation before arriving at the gateway.

Destination UDP Port - The source UDP port carried by the Request message. Note: The value of this field may be changed as a result of network address translation before arriving at the gateway.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |  V=0  |Type=4 | Reserved  |L|G|         Response MAC          |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                               +
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                         Request Nonce                         |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                                                               |
   |               Encapsulated General Query Message              |
   ~                 IPv4:IGMPv3(Membership Query)                 ~
   |                  IPv6:MLDv2(Listener Query)                   |
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |     Gateway Port Number       |                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                               +
   |                                                               |
   +                                                               +
   |                Gateway IP Address (IPv4 or IPv6)              |
   +                                                               +
   |                                                               |
   +                               +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 14: Membership Query Message Format

5.1.4.1. Version (V)

The protocol version number for this message is 0.

5.1.4.2. Type

The type number for this message is 4.

5.1.4.3. Reserved

Reserved bits that MUST be set to zero by the relay and ignored by the gateway.

5.1.4.4. Limit (L) Flag

A 1-bit flag set to 1 to indicate that the relay is NOT accepting Membership Update messages from new gateway tunnel endpoints and that it will ignore any that are. A value of 0 has no special significance - the relay may or may not be accepting Membership Update messages from new gateway tunnel endpoints. A gateway checks this flag before attempting to create new group subscription state on the relay to determine whether it should restart relay discovery. A gateway that has already created group subscriptions on the relay may ignore this flag. Support for this flag is RECOMMENDED.

5.1.4.5. Gateway Address (G) Flag

A 1-bit flag set to 0 to indicate that the message does NOT carry the Gateway Port and Gateway IP Address fields, and 1 to indicate that it does. A relay implementation that supports the optional teardown procedure (See Section 5.3.3.5) SHOULD set this flag and the Gateway Address field values. If a relay sets this flag, it MUST also include the Gateway Address fields in the message. A gateway implementation that does not support the optional teardown procedure (See Section 5.2.3.7) MAY ignore this flag and the Gateway Address fields if they are present.

5.1.4.6. Response MAC

A 48-bit source authentication value generated by the relay as described in Section 5.3.5. The gateway echoes this value in subsequent Membership Update messages to allow the relay to verify that the sender of a Membership Update message was the intended receiver of a Membership Query sent by the relay.

5.1.4.7. Request Nonce

A 32-bit value copied from the Request Nonce field (Section 5.1.3.5) carried by a Request message. The relay will have included this value in the Response MAC computation. The gateway echoes this value in subsequent Membership Update messages. The gateway also uses this value to match a Membership Query to a Request message.

5.1.4.8. Encapsulated General Query Message

An IP-encapsulated IGMP or MLD message generated by the relay. This field will contain one of the following IP datagrams:

IPv4:IGMPv3 Membership Query

IPv6:MLDv2 Listener Query

The source address carried by the query message should be set as described in Section 5.3.3.3.

The Querier’s Query Interval Code (QQIC) field in the general query is used by a relay to specify the time offset a gateway should use to schedule a new three-way handshake to refresh the group membership state within the relay (current time + Query Interval). The QQIC field is defined in Section 4.1.7 in [RFC3376] and Section 5.1.9 in [RFC3810].

The Querier’s Robustness Variable (QRV) field in the general query is used by a relay to specify the number of times a gateway should retransmit unsolicited membership reports, encapsulated within Membership Update messages, and optionally, the number of times to send a Teardown message. The QRV field is defined in Section 4.1.6 in [RFC3376] and Section 5.1.8 in [RFC3810].

5.1.4.9. Gateway Address Fields

The Gateway Port Number and Gateway Address fields are present in the Membership Query message if, and only if, the "G" flag is set.

A gateway need not parse the encapsulated IP datagram to determine the position of these fields within the UDP datagram containing the Membership Query message - if the G-flag is set, the gateway may simply subtract the total length of the fields (18 bytes) from the total length of the UDP datagram to obtain the offset.

5.1.4.9.1. Gateway Port Number

A 16-bit UDP port containing a UDP port value.

The Relay sets this field to the value of the UDP source port of the Request message that triggered the Query message.

5.1.4.9.2. Gateway IP Address

A 16-byte IP address that, when combined with the value contained in the Gateway Port Number field, forms the gateway endpoint address that the relay will use to identify the tunnel instance, if any, created by a subsequent Membership Update message. This field may contain an IPv6 address or an IPv4 address stored as an IPv4-compatible IPv6 address, where the IPv4 address is prefixed with 96 bits set to zero (See [RFC4291]). This address must match that used by the relay to compute the value stored in the Response MAC field.

5.1.5. Membership Update

A gateway sends a Membership Update message to a relay to report a change in group membership state, or to report the current group membership state in response to receiving a Membership Query message. The gateway encapsulates the IGMP or MLD message as an IP datagram within a Membership Update message and sends it to the relay, where it may (see below) be decapsulated and processed by the relay to update group membership and forwarding state.

A gateway cannot send a Membership Update message until a receives a Membership Query from a relay because the gateway must copy the Request Nonce and Response MAC values carried by a Membership Query into any subsequent Membership Update messages it sends back to that relay. These values are used by the relay to verify that the sender of the Membership Update message was the recipient of the Membership Query message from which these values were copied.

The successful delivery of this message to the relay marks the start of the final stage in the three-way handshake. This stage concludes when the relay successfully verifies that sender of the Membership Update message was the recipient of a Membership Query message sent earlier. At this point, the relay may proceed to process the encapsulated IGMP or MLD message to create or update group membership and forwarding state on behalf of the gateway.

The UDP/IP datagram containing this message MUST carry a valid, non-zero UDP checksum and carry the following IP address and UDP port values:

Source IP Address - The IP address of the gateway interface on which the gateway will listen for Multicast Data messages from the relay. The address must be the same address used to send the initial Request message or the message will be ignored. Note: The value of this field may be changed as a result of network address translation before arriving at the relay.

Source UDP Port - The UDP port number on which the gateway will listen for Multicast Data messages from the relay. This port must be the same port used to send the initial Request message or the message will be ignored. Note: The value of this field may be changed as a result of network address translation before arriving at the relay.

Destination IP Address - The unicast IP address of the relay.

Destination UDP Port - The IANA-assigned AMT port number.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |  V=0  |Type=5 |  Reserved     |        Response MAC           |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                               +
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                         Request Nonce                         |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                                                               |
   |         Encapsulated Group Membership Update Message          |
   ~           IPv4:IGMP(Membership Report|Leave Group)            ~
   |            IPv6:MLD(Listener Report|Listener Done)            |
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 15: Membership Update Message Format

5.1.5.1. Version (V)

The protocol version number for this message is 0.

5.1.5.2. Type

The type number for this message is 5.

5.1.5.3. Reserved

Reserved bits that MUST be set to zero by the gateway and ignored by the relay.

5.1.5.4. Response MAC

A 48-bit value copied from the Response MAC field (Section 5.1.4.6) in a Membership Query message. Used by the relay to perform source authentication.

5.1.5.5. Request Nonce

A 32-bit value copied from the Request Nonce field in a Request or Membership Query message. Used by the relay to perform source authentication.

5.1.5.6. Encapsulated Group Membership Update Message

An IP-encapsulated IGMP or MLD message produced by the host-mode IGMP or MLD protocol running on a gateway pseudo-interface. This field will contain of one of the following IP datagrams:

IPv4:IGMPv2 Membership Report

IPv4:IGMPv2 Leave Group

IPv4:IGMPv3 Membership Report

IPv6:MLDv1 Multicast Listener Report

IPv6:MLDv1 Multicast Listener Done

IPv6:MLDv2 Multicast Listener Report

The source address carried by the message should be set as described in Section 5.2.1.

5.1.6. Multicast Data

A relay sends a Multicast Data message to deliver an multicast IP datagram or datagram fragment to a gateway.

The checksum field in the UDP header of this message MAY contain a value of zero when sent over IPv4 but SHOULD, if possible, contain a valid, non-zero value when sent over IPv6 (See Section 4.2.2.3).

The UDP/IP datagram containing this message MUST carry the following IP address and UDP port values:

Source IP Address - The unicast IP address of the relay.

Source UDP Port - The IANA-assigned AMT port number.

Destination IP Address - A tunnel endpoint IP address, i.e., the source IP address carried by the Membership Update message sent by a gateway to indicate an interest in receiving the multicast packet. Note: The value of this field may be changed as a result of network address translation before arriving at the gateway.

Destination UDP Port - A tunnel endpoint UDP port, i.e., the source UDP port carried by the Membership Update message sent by a gateway to indicate an interest in receiving the multicast packet. Note: The value of this field may be changed as a result of network address translation before arriving at the gateway.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |  V=0  |Type=6 |    Reserved   |                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                               +
   |                                                               |
   ~                     IP Multicast Packet                       ~
   |                                                               |
   +                - - - - - - - - - - - - - - - - - - - - - - - -+
   |               :               :               :               :
   +-+-+-+-+-+-+-+-+- - - - - - - - - - - - - - - - - - - - - - - -
```

Figure 16: Multicast Data Message Format

5.1.6.1. Version (V)

The protocol version number for this message is 0.

5.1.6.2. Type

The type number for this message is 6.

5.1.6.3. Reserved

Bits that MUST be set to zero by the relay and ignored by the gateway.

5.1.6.4. IP Multicast Data

A complete IPv4 or IPv6 multicast datagram or datagram fragment.

5.1.7. Teardown

A gateway sends a Teardown message to a relay to request that it stop sending Multicast Data messages to a tunnel endpoint created by an earlier Membership Update message. A gateway sends this message when it detects that a Request message sent to the relay carries an address that differs from that carried by a previous Request message. The gateway uses the Gateway IP Address and Gateway Port Number Fields in the Membership Query message to detect these address changes.

To provide backwards compatibility with early implementations of the AMT protocol, support for this message and associated procedures is considered OPTIONAL - gateways are not required to send this message and relays are not required to act upon it.

The UDP/IP datagram containing this message MUST carry a valid, non-zero UDP checksum and carry the following IP address and UDP port values:

Source IP Address - The IP address of the gateway interface used to send the message. This address may differ from that used to send earlier messages. Note: The value of this field may be changed as a result of network address translation before arriving at the relay.

Source UDP Port - The UDP port number. This port number may differ from that used to send earlier messages. Note: The value of this field may be changed as a result of network address translation before arriving at the relay.

Destination IP Address - The unicast IP address of the relay.

Destination UDP Port - The IANA-assigned AMT port number.

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |  V=0  |Type=7 |  Reserved     |         Response MAC          |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                               +
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                         Request Nonce                         |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |     Gateway Port Number       |                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                               +
   |                                                               |
   +                                                               +
   |              Gateway IP Address (IPv4 or IPv6)                |
   +                                                               +
   |                                                               |
   +                               +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 17: Membership Teardown Message Format

5.1.7.1. Version (V)

The protocol version number for this message is 0.

5.1.7.2. Type

The type number for this message is 7.

5.1.7.3. Reserved

Reserved bits that MUST be set to zero by the gateway and ignored by the relay.

5.1.7.4. Response MAC

A 48-bit value copied from the Response MAC field (Section 5.1.4.6) in the last Membership Query message the relay sent to the gateway endpoint address of the tunnel to be torn down. The gateway endpoint address is provided by the Gateway IP Address and Gateway Port Number fields carried by the Membership Query message. The relay validates the Teardown message by comparing this value with one computed from the Gateway IP Address, Gateway Port Number, Request Nonce fields and a private secret (just as it does in the Membership Update message).

5.1.7.5. Request Nonce

A 32-bit value copied from the Request Nonce field (Section 5.1.4.7) in the last Membership Query message the relay sent to the gateway endpoint address of the tunnel to be torn down. The gateway endpoint address is provided by the Gateway IP Address and Gateway Port Number fields carried by the Membership Query message. This value must match that used by the relay to compute the value stored in the Response MAC field.

5.1.7.6. Gateway Port Number

A 16-bit UDP port number that, when combined with the value contained in the Gateway IP Address field, forms the tunnel endpoint address that the relay will use to identify the tunnel instance to tear down. The relay provides this value to the gateway using the Gateway Port Number field (Section 5.1.4.9.1) in a Membership Query message. This port number must match that used by the relay to compute the value stored in the Response MAC field.

5.1.7.7. Gateway IP Address

A 16-byte IP address that, when combined with the value contained in the Gateway Port Number field, forms the tunnel endpoint address that the relay will used to identify the tunnel instance to tear down. The relay provides this value to the gateway using the Gateway IP Address field (Section 5.1.4.9.2) in a Membership Query message. This field may contain an IPv6 address or an IPv4 address stored as an IPv4-compatible IPv6 address, where the IPv4 address is prefixed with 96 bits set to zero (See [RFC4291]). This address must match that used by the relay to compute the value stored in the Response MAC field.

5.2. Gateway Operation

The following sections describe gateway implementation requirements. A non-normative discussion of gateway operation may be found in Section 4.2.

5.2.1. IP/IGMP/MLD Protocol Requirements

Gateway operation requires a subset of host mode IPv4/IGMP and IPv6/ MLD functionality to provide group membership tracking, general query processing, and report generation. A gateway MAY use IGMPv2 (ASM), IGMPv3 (ASM and SSM), MLDv1 (ASM) or MLDv2 (ASM and SSM).

An application with embedded gateway functionality must provide its own implementation of this subset of the IPv4/IGMP and IPv6/MLD protocols. The service interface used to manipulate group membership state need not match that described in the IGMP and MLD specifications, but the actions taken as a result SHOULD be similar to those described in Section 5.1 of [RFC3376] and Section 6.1 of [RFC3810]. The gateway application will likely need to implement many of the same functions as a host IP stack, including checksum verification, dispatching, datagram filtering and forwarding, and IP encapsulation/decapsulation.

The encapsulated IGMP datagrams generated by a gateway MUST conform to the descriptions found in Section 4 of [RFC3376]. These datagrams MUST possess the IP headers, header options and header values called for in [RFC3376], with the following exception; a gateway MAY use any source address value in an IGMP report datagram including the "unspecified" address (all octets are zero). This exception is made because a gateway pseudo-interface might not possess a valid IPv4 address, and even if an address has been assigned to the interface, that address might not be a valid link-local source address on any relay interface. It is for this reason that a relay must accept encapsulated IGMP reports regardless of the source address they carry. See Section 5.3.1.

The encapsulated MLD messages generated by a gateway MUST conform to the description found in Section 5 of [RFC3810]. These datagrams MUST possess the IP headers, header options and header values called for in [RFC3810], with the following exception; a gateway MAY use any source address value in an MLD report datagram including the "unspecified" address (all octets are zero). This exception is made because a gateway pseudo-interface might not possess a valid IPv6 address, and even if an address has been assigned to the interface, that address might not be a valid link-local source address on any relay interface. As with IGMP, it is for this reason that a relay must accept encapsulated MLD reports regardless of the source address they carry. See Section 5.3.1.

The gateway IGMP/MLD implementation SHOULD retransmit unsolicited membership state-change reports and merge new state change reports with pending reports as described in Section 5.1 of [RFC3376] and Section 6.1 of [RFC3810]. The number of retransmissions is specified by the relay in the Querier’s Robustness Variable (QRV) field in the last general query forwarded by the pseudo-interface. See Section 4.1.6 in [RFC3376] and Section 5.1.8 in [RFC3810].

The gateway IGMP/MLD implementation SHOULD handle general query messages as described in Section 5.2 of [RFC3376] and Section 6.2 of [RFC3810], but MAY ignore the Max Resp Code field value and generate a current state report without any delay.

An IPv4 gateway implementation MUST accept IPv4 datagrams that carry the general query variant of the IGMPv3 Membership Query message, as described in Section 4 of [RFC3376]. The gateway MUST accept the IGMP datagram regardless of the IP source address carried by that datagram.

An IPv6 gateway implementation MUST accept IPv6 datagrams that carry the general query variant of the MLDv2 Multicast Listener Query message, as described in Section 5 of [RFC3810]. The gateway MUST accept the MLD datagram regardless of the IP source address carried by that datagram.

5.2.2. Pseudo-Interface Configuration

A gateway host may possess or create multiple gateway pseudo-interfaces, each with a unique configuration that describes a binding to a specific IP protocol, relay address, relay discovery address or upstream network interface.

5.2.2.1. Relay Discovery Address

If a gateway implementation uses AMT relay discovery to obtain a relay address, it must first be supplied with a relay discovery address. The relay discovery address may be an anycast or unicast address. A gateway implementation may rely on a static address assignment or some form of dynamic address discovery. This specification does not require that a gateway implementation use any particular method to obtain a relay discovery address - an implementation may employ any method that returns a suitable relay discovery address.

5.2.2.2. Relay Address

Before a gateway implementation can execute the AMT protocol to request and receive multicast traffic, it must be supplied with a unicast relay address. A gateway implementation may rely on static address assignment or support some form of dynamic address discovery. This specification does not require the use of any particular method to obtain a relay address - an implementation may employ any method that returns a suitable relay address.

5.2.2.3. Upstream Interface Selection

A gateway host that possesses multiple network interfaces or addresses may allow for an explicit selection of the interface to use when communicating with a relay. The selection might be made to satisfy connectivity, tunneling or IP protocol requirements.

5.2.2.4. Optional Retransmission Parameters

A gateway implementation that supports retransmission MAY require the following information:

Discovery Timeout
Initial time to wait for a response to a Relay Discovery message.

Maximum Relay Discovery Retransmission Count
Maximum number of Relay Discovery retransmissions to allow before terminating relay discovery and reporting an error.

Request Timeout
Initial time to wait for a response to a Request message.

Maximum Request Retransmission Count
Maximum number of Request retransmissions to allow before abandoning a relay and restarting relay discovery or reporting an error.

Maximum Retries Count For "Destination Unreachable"
The maximum number of times a gateway should attempt to send the same Request or Membership Update message after receiving an ICMP "Destination Unreachable".

5.2.3. Gateway Service

In the following descriptions, a gateway pseudo interface is treated as a passive entity managed by a gateway service. The gateway pseudo-interface provides the state and the gateway service provides the processing. The term "gateway" is used when describing service behavior with respect to a single pseudo-interface.

5.2.3.1. Startup

When a gateway pseudo-interface is started, the gateway service begins listening for AMT messages sent to the UDP endpoint(s) associated with the pseudo-interface and for any locally-generated IGMP/MLD messages passed to the pseudo-interface. The handling of these messages is described below.

When the pseudo-interface is enabled, the gateway service MAY:

- Optionally execute the relay discovery procedure described in Section 5.2.3.4.

- Optionally execute the membership query procedure described in Section 5.2.3.5 to start the periodic membership update cycle.

5.2.3.2. Handling AMT Messages

A gateway MUST ignore any datagram it receives that cannot be interpreted as a Relay Advertisement, Membership Query, or Multicast Data message. The handling of Relay Advertisement, Membership Query, and Multicast Data messages is addressed in the sections that follow.

A gateway that conforms to this specification MUST ignore any message with a Version field value other than zero.

While listening for AMT messages, a gateway may be notified that an ICMP Destination Unreachable message was received as a result of an AMT message transmission. Handling of ICMP Destination Unreachable messages is described in Section 5.2.3.9.

5.2.3.3. Handling Multicast Data Messages

A gateway may receive Multicast Data messages after it sends a Membership Update message to a relay that adds a group subscription. The gateway may continue to receive Multicast Data messages long after the gateway sends a Membership Update message that deletes existing group subscriptions. The gateway MUST be prepared to receive these messages at any time, but MAY ignore them or discard their contents if the gateway no longer has any interest in receiving the multicast datagrams contained within them.

A gateway MUST ignore a Multicast Data message if it fails to satisfy any of the following requirements:

- The source IP address and UDP port carried by the Multicast Data message MUST be equal to the destination IP address and UDP port carried by the matching Membership Update message (i.e., the current relay address).

- The destination address carried by the encapsulated IP datagram MUST fall within the multicast address allocation assigned to the relevant IP protocol, i.e., 224.0.0.0/4 for IPv4 and FF00::/8 for IPv6.

The gateway extracts the encapsulated IP datagram and forwards it to the local IP protocol implementation for checksum verification, fragmented datagram reassembly, source and group filtering, and transport-layer protocol processing.

Because AMT uses UDP encapsulation to deliver multicast datagrams to gateways, it qualifies as a tunneling protocol subject to the limitations described in [RFC6936]. If supported, a gateway SHOULD employ the solution described in [RFC6936] to ensure that the local IP stack does not discard IPv6 datagrams with zero checksums. If Multicast Data message datagrams are processed directly within the gateway (instead of the host IP stack), the gateway MUST NOT discard any of these datagrams because they carry a UDP checksum of zero.

5.2.3.4. Relay Discovery Procedure

This section describes gateway requirements related to the relay discovery message sequence described in Section 4.2.1.1.

5.2.3.4.1. Starting Relay Discovery

A gateway may start or restart the relay discovery procedure in response to the following events:

- When a gateway pseudo-interface is started (enabled).

- When the gateway wishes to report a group subscription when none currently exist.

- Before sending the next Request message in a membership update cycle, i.e., each time the query timer expires (see below).

- After the gateway fails to receive a response to a Request message.

- After the gateway receives a Membership Query message with the L-flag set to 1.

5.2.3.4.2. Sending a Relay Discovery Message

A gateway sends a Relay Discovery message to a relay to start the relay discovery process.

The gateway MUST send the Relay Discovery message using the current Relay Discovery Address and IANA-assigned AMT port number as the destination. The Discovery Nonce value in the Relay Discovery message MUST be computed as described in Section 5.2.3.4.5.

The gateway MUST save a copy of Relay Discovery message or save the Discovery Nonce value for possible retransmission and verification of a Relay Advertisement response.

When a gateway sends a Relay Discovery message, it may be notified that an ICMP Destination Unreachable message was received as a result of an earlier AMT message transmission. Handling of ICMP Destination Unreachable messages is described in Section 5.2.3.9.

5.2.3.4.3. Waiting for a Relay Advertisement Message

A gateway MAY retransmit a Relay Discovery message if it does not receive a matching Relay Advertisement message within some timeout period. If the gateway retransmits the message multiple times, the timeout period SHOULD be adjusted to provide an random exponential back-off. The RECOMMENDED timeout is a random value in the range [initial_timeout, MIN(initial_timeout * 2^retry_count, maximum_timeout)], with a RECOMMENDED initial_timeout of 1 second and a RECOMMENDED maximum_timeout of 120 seconds (which is the recommended minimum NAT mapping timeout described in [RFC4787]).

5.2.3.4.4. Handling a Relay Advertisement Message

When a gateway receives a Relay Advertisement message it must first determine whether it should accept or ignore the message. A gateway MUST ignore a Relay Advertisement message if it fails to satisfy any of the following requirements:

- The gateway MUST be waiting for a Relay Advertisement message.

- The Discovery Nonce value contained in the Relay Advertisement message MUST equal to the Discovery Nonce value contained in the Relay Discovery message.

- The source IP address and UDP port of the Relay Advertisement message MUST equal to the destination IP address and UDP port of the matching Relay Discovery message.

Once a gateway receives a Relay Advertisement response to a Relay Discovery message, it SHOULD ignore any other Relay Advertisements that arrive on the AMT interface until it sends a new Relay Discovery message.

If a gateway executes the relay discovery procedure at the start of each membership update cycle and the relay address returned in the latest Relay Advertisement message differs from the address returned in a previous Relay Advertisement message, then the gateway SHOULD send a Teardown message (if supported) to the old relay address, using information from the last Membership Query message received from that relay, as described in Section 5.2.3.7. This behavior is illustrated in the following diagram.

```
                     Gateway              Relay-1
                     -------              -------
                        :                    :
     Query      Expired |                    |
     Timer (QT)-------->|                    |
                        |  Relay Discovery   |
                        |------------------->|
                        |                    |
                        | Relay Advertisement|
                        |<-------------------|
                        |                    |
                        |       Request      |
                        |------------------->|
                        |                    |
                        |  Membership Query  |
                        |<===================|
                  Start |                    |
           (QT)<--------| Membership Update  |
                        |===================>|
                        |                    |
                        ~                    ~             Relay-2
                Expired |                    |             -------
           (QT)-------->|                    |                :
                        |  Relay Discovery   |                |
                        |------------------------------------>|
                        |                    |                |
                        | Relay Advertisement|                |
                        |<------------------------------------|
                        |                    |                |
                        |      Teardown      |                |
                        |------------------->|                |
                        |                    |                |
                        |       Request      |                |
                        |------------------------------------>|
                        |                    |                |
                        |  Membership Query  |                |
                        |<====================================|
                  Start |                    |                |
           (QT)<--------| Membership Update  |                |
                        |====================================>|
                        |                    |                |
                        :                    :                :
```

Figure 18: Teardown After Relay Address Change

5.2.3.4.5. Discovery Nonce Generation

The discovery nonce MUST be a random, non-zero, 32-bit value, and if possible, SHOULD be computed using a cryptographically secure pseudo random number generator. A new nonce SHOULD be generated each time the gateway restarts the relay discovery process. The same nonce SHOULD be used when retransmitting a Relay Discovery message.

5.2.3.5. Membership Query Procedure

This section describes gateway requirements related to the membership update message sequence described in Section 4.2.1.2.

5.2.3.5.1. Starting the Membership Update Cycle

A gateway may send a Request message to start a membership update cycle (following the optional relay discovery procedure) in response to the following events:

- o When the gateway pseudo-interface is activated.

- o When the gateway wishes to report a group subscription when none currently exist.

Starting the membership update cycle when a gateway pseudo-interface is started provides several benefits:

- o Better performance by allowing state-change reports to be sent as they are generated, thus minimizing the time to join.

- o More robustness by relying on unsolicited state-change reports to update group membership state rather than the current-state reports generated by the membership update cycle. Unsolicited state-change reports are typically retransmitted multiple times while current-state reports are not.

- o Simplified implementation by eliminating any need to queue IGMP/MLD messages for delivery after a Membership Query is received, since the IGMP/MLD state-change messages may be sent as they are generated.

However, this approach places an additional load on relays as a gateway will send periodic requests even when it has no multicast subscriptions. To reduce load on a relay, a gateway SHOULD only send a Membership Update message while it has active group subscriptions. A relay will still need to compute a Response MAC for each Request, but will not be required to recompute it a second time to authenticate a Membership Update message that contains no subscriptions.

5.2.3.5.2. Sending a Request Message

A gateway sends a Request message to a relay to solicit a Membership Query response and start the membership update cycle.

A gateway constructs a Request message containing a Request Nonce value computed as described in Section 5.2.3.5.6. The gateway MUST set the "P" flag in the Request message to identify the protocol the gateway wishes the relay to use for the general query response.

A gateway MUST send a Request message using the current Relay Address and IANA-assigned AMT port number as the destination.

A gateway MUST save a copy of the Request message or save the Request Nonce and P-flag values for possible retransmission and verification of a Membership Query response.

When a gateway sends a Request message, it may be notified that an ICMP Destination Unreachable message was received as a result of an earlier AMT message transmission. Handling of ICMP Destination Unreachable messages is described in Section 5.2.3.9.

5.2.3.5.3. Waiting for a Membership Query Message

A gateway MAY retransmit a Request message if it does not receive a matching Membership Query message within some timeout period. If the gateway retransmits the message multiple times, the timeout period SHOULD be adjusted to provide an random exponential back-off. The RECOMMENDED timeout is a random value in the range [initial_timeout, MIN(initial_timeout * 2^retry_count, maximum_timeout)], with a RECOMMENDED initial_timeout of 1 second and a RECOMMENDED maximum_timeout of 120 seconds (which is the recommended minimum NAT mapping timeout described in [RFC4787]).

If a gateway that uses relay discovery does not receive a Membership Query within a specified time period or after a specified number of retries, the gateway SHOULD stop waiting for a Membership Query message and restart relay discovery to locate another relay.

5.2.3.5.4. Handling a Membership Query Message

When a gateway receives a Membership Query message it must first determine whether it should accept or ignore the message. A gateway MUST ignore a Membership Query message, or the encapsulated IP datagram within it, if the message fails to satisfy any of the following requirements:

- The gateway MUST be waiting for a Membership Query message.

- The Request Nonce value contained in the Membership Query MUST equal the Request Nonce value contained in the Request message.

- The source IP address and UDP port of the Membership Query MUST equal the destination IP address and UDP port of the matching Request message (i.e., the current relay address).

- The encapsulated IP datagram MUST carry an IGMPv3 or MLDv2 message. The protocol MUST match the protocol identified by the "P" flag in the Request message.

- The IGMPv3 or MLDv2 message MUST be a general query message.

- The total length of the encapsulated IP datagram as computed from the lengths contained in the datagram header(s) MUST NOT exceed the available field length within the Membership Query message.

Once a gateway receives a Membership Query response to a Request message, it SHOULD ignore any other Membership Query messages that arrive on the AMT interface until it sends a new Request message.

The gateway MUST save the Membership Query message, or the Request Nonce, Response MAC, Gateway IP Address and Gateway Port Number fields for use in sending subsequent Membership Update and Teardown messages.

The gateway extracts the encapsulated IP datagram and forwards it to the local IP protocol implementation for checksum verification and dispatching to the IGMP or MLD implementation running on the pseudo-interface. The gateway MUST NOT forward any octets that might exist between the encapsulated IP datagram and the end of the message or Gateway Address fields.

The MLD protocol specification indicates that senders should use a link-local source IP address in message datagrams. This requirement must be relaxed for AMT because gateways and relays do not normally share a common subnet. For this reason, a gateway implementation MUST accept MLD (and IGMP) query message datagrams regardless of the source IP address they carry. This may require additional processing on the part of the gateway that might be avoided if the relay and gateway use the IPv4 and IPv6 addresses allocated for use in AMT encapsulated control packets as described in Section 5.2.1.

The gateway MUST start a timer that will trigger the next iteration of the membership update cycle by executing the membership query procedure. The gateway SHOULD compute the timer duration from the Querier’s Query Interval Code carried by the general-query. A gateway MAY use a smaller timer duration if required to refresh a NAT mapping that would otherwise timeout. A gateway MAY use a larger timer duration if it has no group subscriptions to report.

If the gateway supports the Teardown message and the G-flag is set in the Membership Query message, the gateway MUST compare the Gateway IP Address and Gateway Port Number on the new Membership Query message with the values carried by the previous Membership Query message. If either value has changed the gateway MUST send a Teardown message to the relay as described in Section 5.2.3.7.

If the L-flag is set in the Membership Query message, the relay is reporting that it is NOT accepting Membership Update messages that create new tunnel endpoints and will simply ignore any that do. If the L-flag is set and the gateway is not currently reporting any group subscriptions to the relay, the gateway SHOULD stop sending periodic Request messages and restart the relay discovery procedure (if discovery is enabled) to find a new relay with which to communicate. The gateway MAY continue to send updates even if the L-flag is set, if it has previously reported group subscriptions to the relay, one or more subscriptions still exist and the gateway endpoint address has not changed since the last Membership Query was received (see previous paragraph).

5.2.3.5.5. Handling Query Timer Expiration

When the query timer (started in the previous step) expires, the gateway should execute the membership query procedure again to continue the membership update cycle.

5.2.3.5.6. Request Nonce Generation

The request nonce MUST be a random value, and if possible, SHOULD be computed using a cryptographically secure pseudo random number generator. A new nonce MUST be generated each time the gateway starts the membership query process. The same nonce SHOULD be used when retransmitting a Request message.

5.2.3.6. Membership Update Procedure

This section describes gateway requirements related to the membership update message sequence described in Section 4.2.1.2.

The membership update process is primarily driven by the host-mode IGMP or MLD protocol implementation running on the gateway pseudo-interface. The IGMP and MLD protocols produce current-state reports in response to general queries generated by the pseudo-interface via AMT and produce state-change reports in response to receiver requests made using the IGMP or MLD service interface.

5.2.3.6.1. Handling an IGMP/MLD IP Datagram

The gateway pseudo-interface MUST accept the following IP datagrams from the IPv4/IGMP and IPv6/MLD protocols running on the pseudo-interface:

- IPv4 datagrams that carry an IGMPv2, or IGMPv3 Membership Report or an IGMPv2 Leave Group message as described in Section 4 of [RFC3376].

- IPv6 datagrams that carry an MLDv1 or MLDv2 Multicast Listener Report or an MLDv1 Multicast Listener Done message as described in Section 5 of [RFC3810].

The gateway must be prepared to receive these messages any time the pseudo-interface is running. The gateway MUST ignore any datagrams not listed above.

A gateway that waits to start a membership update cycle until after it receives a datagram containing an IGMP/MLD state-change message MAY:

- Discard IGMP or MLD datagrams until it receives a Membership Query message, at which time it processes the Membership Query message as normal to eventually produce a current-state report on the pseudo-interface which describes the end state (RECOMMENDED).

- Insert IGMP or MLD datagrams into a queue for transmission after it receives a Membership Query message.

If and when a gateway receives a Membership Query message (for IGMP or MLD) it sends any queued or incoming IGMP or MLD datagrams to the relay as described in the next section.

5.2.3.6.2. Sending a Membership Update Message

A gateway cannot send a Membership Update message to a relay until it has received a Membership Query message from a relay. If the gateway has not yet located a relay with which to communicate, it MUST first execute the relay discovery procedure described in Section 5.2.3.4 to obtain a relay address. If the gateway has a relay address, but has not yet received a Membership Query message, it MUST first execute the membership query procedure described in Section 5.2.3.5 to obtain a Request Nonce and Response MAC that can be used to send a Membership Update message.

Once a gateway possesses a valid Relay Address, Request Nonce and Response MAC, it may encapsulate the IP datagram containing the IGMP/MLD message into a Membership Update message. The gateway MUST copy the Request Nonce and Response MAC values from the last Membership Query received from the relay into the corresponding fields in the Membership Update. The gateway MUST send the Membership Update message using the Relay Address and IANA-assigned AMT port number as the destination.

When a gateway sends a Membership Update message, it may be notified that an ICMP Destination Unreachable message was received as a result of an earlier AMT message transmission. Handling of ICMP Destination Unreachable messages is described in Section 5.2.3.9.

5.2.3.7. Teardown Procedure

This section describes gateway requirements related to the teardown message sequence described in Section 4.2.1.3.

Gateway support for the Teardown message is RECOMMENDED.

A gateway that supports Teardown SHOULD make use of Teardown functionality if it receives a Membership Query message from a relay that has the "G" flag set to indicate that it contains valid gateway address fields.

5.2.3.7.1. Handling a Membership Query Message

As described in Section 5.2.3.5.4, if a gateway supports the Teardown message, has reported active group subscriptions, and receives a Membership Query message with the "G" flag set, the gateway MUST compare the Gateway IP Address and Gateway Port Number on the new Membership Query message with the values carried by the previous Membership Query message. If either value has changed the gateway MUST send a Teardown message as described in the next section.

5.2.3.7.2. Sending a Teardown Message

A gateway sends a Teardown message to a relay to request that it stop delivering Multicast Data messages to the gateway and delete any group memberships created by the gateway.

When a gateway constructs a Teardown message, it MUST copy the Request Nonce, Response MAC, Gateway IP Address and Gateway Port Number fields from the Membership Query message that provided the Response MAC for the last Membership Update message sent, into the corresponding fields of the Teardown message.

A gateway MUST send the Teardown message using the Relay Address and IANA-assigned AMT port number as the destination. A gateway MAY send the Teardown message multiple times for robustness. The gateway SHOULD use the Querier’s Robustness Variable (QRV) field contained in the query encapsulated within the last Membership Query to set the limit on the number of retransmissions (See Section 4.1.6 in [RFC3376] and Section 5.1.7 in [RFC3810]). If the gateway sends the Teardown message multiple times, it SHOULD insert a delay between each transmission using the timing algorithm employed in IGMP/MLD for transmitting unsolicited state-change reports. The RECOMMENDED default delay value is 1 second.

When a gateway sends a Teardown message, it may be notified that an ICMP Destination Unreachable message was received as a result of an earlier AMT message transmission. Handling of ICMP Destination Unreachable messages is described in Section 5.2.3.9.

5.2.3.8. Shutdown

When a gateway pseudo-interface is stopped and the gateway has existing group subscriptions, the gateway SHOULD either:

- Send a Teardown message to the relay as described in Section 5.2.3.7, but only if the gateway supports the Teardown message, and the current relay is returning gateway address fields in Membership Query messages, or

- Send a Membership Update message to the relay that will delete existing group subscriptions.

5.2.3.9. Handling ICMP Destination Unreachable Responses

A gateway may receive an ICMP "Destination Unreachable" message [RFC0792] after sending an AMT message. Whether the gateway is notified that an ICMP message was received is highly dependent on firewall and gateway IP stack behavior and gateway implementation.

If the reception of an ICMP Destination Unreachable message is reported to the gateway while waiting to receive an AMT message, the gateway may respond as follows, depending on platform capabilities and which outgoing message triggered the ICMP response:

1. The gateway MAY simply abandon the current relay and restart relay discovery (if used). This is the least desirable approach as it does not allow for transient network changes.

2. If the last message sent was a Relay Discovery or Request message, the gateway MAY simply ignore the ICMP response and continue waiting for incoming AMT messages. If the gateway is configured to retransmit Relay Discovery or Request messages, the normal retransmission behavior for those messages is preserved to prevent the gateway from prematurely abandoning a relay.

3. If the last message sent was a Membership Update message, the gateway MAY start a new membership update and associated Request retransmission cycle.

If the reception of an ICMP Destination Unreachable message is reported to the gateway when attempting to transmit a new AMT message, the gateway may respond as follows, depending on platform capabilities and which outgoing message triggered the ICMP response:

1. The gateway MAY simply abandon the current relay and restart relay discovery (if used). This is the least desirable approach as it does not allow for transient network changes.

2. If the last message sent was a Relay Discovery, Request or Teardown message, the gateway MAY attempt to transmit the new message. If the gateway is configured to retransmit Relay Discovery, Request or Teardown messages, the normal retransmission behavior for those messages is preserved to prevent the gateway from prematurely abandoning a relay.

3. If the last message sent was a Membership Update message, the gateway SHOULD start a new membership update and associated Request retransmission cycle.

5.3. Relay Operation

The following sections describe relay implementation requirements. A non-normative discussion of relay operation may be found in Section 4.2.

5.3.1. IP/IGMP/MLD Protocol Requirements

A relay requires a subset of router-mode IGMP and MLD functionality to provide group membership tracking and report processing.

A relay accessible via IPv4 MUST support IPv4/IGMPv3 and MAY support IPv6/MLDv2. A relay accessible via IPv6 MUST support IPv6/MLDv2 and MAY support IPv4/IGMPv3.

A relay MUST apply the forwarding rules described in Section 6.3 of [RFC3376] and Section 7.3 of [RFC3810].

A relay MUST handle incoming reports as described in Section 6.4 of [RFC3376] and Section 7.4 of [RFC3810] with the exception that actions that lead to queries MAY be modified to eliminate query generation. A relay MUST accept IGMP and MLD report datagrams regardless of the IP source address carried by those datagrams.

All other aspects of IGMP/MLD router behavior, such as the handling of queries, querier election, etc., are not used or required for relay operation.

5.3.2. Startup

If a relay is deployed for anycast discovery, the relay MUST advertise an anycast Relay Discovery Address Prefix into the unicast routing system of the anycast domain. An address within that prefix, i.e., a Relay Discovery Address, MUST be assigned to a relay interface.

A unicast IPv4 and/or IPv6 address MUST be assigned to the relay interface that will be used to send and receive AMT control and data messages. This address or addresses are returned in Relay Advertisement messages.

The remaining details of relay "startup" are highly implementation-dependent and are not addressed in this document.

5.3.3. Running

When a relay is started, it begins listening for AMT messages on the interface to which the unicast Relay Address(es) has been assigned, i.e., the address returned in Relay Advertisement messages.

5.3.3.1. Handling AMT Messages

A relay MUST ignore any message other than a Relay Discovery, Request, Membership Update or Teardown message. The handling of Relay Discovery, Request, Membership Update, and Teardown messages is addressed in the sections that follow.

Support for the Teardown message is OPTIONAL. If a relay does not support the Teardown message, it MUST also ignore this message.

A relay that conforms to this specification MUST ignore any message with a Version field value other than zero.

5.3.3.2. Handling a Relay Discovery Message

This section describes relay requirements related to the relay discovery message sequence described in Section 4.2.1.1.

A relay MUST accept and respond to Relay Discovery messages sent to an anycast relay discovery address or the unicast relay address. If a relay receives a Relay Discovery message sent to its unicast address, it MUST respond just as it would if the message had been sent to its anycast discovery address.

When a relay receives a Relay Discovery message it responds by sending a Relay Advertisement message back to the source of the Relay Discovery message. The relay MUST use the source IP address and UDP port of the Relay Discovery message as the destination IP address and UDP port. The relay MUST use the destination IP address and UDP port of the Relay Discovery as the source IP address and UDP port to ensure successful NAT traversal.

The relay MUST copy the value contained in the Discovery Nonce field of the Relay Discovery message into the Discovery Nonce field in the Relay Advertisement message.

If the Relay Discovery message was received as an IPv4 datagram, the relay MUST return an IPv4 address in the Relay Address field of the Relay Advertisement message. If the Relay Discovery message was received as an IPv6 datagram, the relay MUST return an IPv6 address in the Relay Address field.

5.3.3.3. Handling a Request Message

This section describes relay requirements related to the membership query portion of the message sequence described in Section 4.2.1.2.

When a relay receives a Request message it responds by sending a Membership Query message back to the source of the Request message.

The relay MUST use the source IP address and UDP port of the Request message as the destination IP address and UDP port for the Membership Query message. The source IP address and UDP port carried by the Membership Query MUST match the destination IP address and UDP port of the Request to ensure successful NAT traversal.

The relay MUST return the value contained in the Request Nonce field of the Request message in the Request Nonce field of the Membership

Query message. The relay MUST compute a MAC value, as described in Section 5.3.5, and return that value in the Response MAC field of the Membership Query message.

If a relay supports the Teardown message, it MUST set the G-flag in the Membership Query message and return the source IP address and UDP port carried by the Request message in the corresponding Gateway IP Address and Gateway Port Number fields. If the relay does not support the Teardown message it SHOULD NOT set these fields as this may cause the gateway to generate unnecessary Teardown messages.

If the P-flag in the Request message is 0, the relay MUST return an IPv4-encapsulated IGMPv3 general query in the Membership Query message. If the P-flag is 1, the relay MUST return an IPv6-encapsulated MLDv2 general query in the Membership Query message.

If the relay is not accepting Membership Update messages that create new tunnel endpoints due to resource limitations, it SHOULD set the L-flag in the Membership Query message to notify the gateway of this state. Support for the L-flag is OPTIONAL. See Section 5.3.3.8.

The encapsulated IGMPv3 general query datagrams generated by a relay MUST conform to the descriptions found in Section 4.1 of [RFC3376]. These datagrams MUST possess the IP headers, header options and header values called for in [RFC3376], with the following exception; a relay MAY use any source IP address for an IGMP general query datagram including the "unspecified" address (all octets are zero). This exception is made because any source address that a relay might normally send may not be a valid link-local address on any gateway interface. It is for this reason that a gateway must accept encapsulated IGMP queries regardless of the source address they carry. See Section 5.2.1.

The encapsulated MLDv2 general query datagrams generated by a relay MUST conform to the descriptions found in Section 5.1 of [RFC3810]. These datagrams MUST possess the IP headers, header options and header values called for in [RFC3810], with the following exception; a relay MAY use any source IP address for an MLD general query datagram including the "unspecified" address (all octets are zero). This exception is made because any source address that a relay might normally send may not be a valid link-local address on any gateway interface. As with IGMP, it is for this reason that a gateway must accept encapsulated MLD queries regardless of the source address they carry. See Section 5.2.1.

A relay MUST set the Querier’s Query Interval Code (QQIC) field in the general query to supply the gateway with a suggested time duration to use for the membership query timer. The QQIC field is defined in Section 4.1.7 in [RFC3376] and Section 5.1.9 in [RFC3810]. A relay MAY adjust this value to affect the rate at which the Request messages are sent from a gateway. However, a gateway is allowed to use a shorter duration than specified in the QQIC field, so a relay may be limited in its ability to spread out Requests coming from a gateway.

A relay MUST set the Querier’s Robustness Variable (QRV) field in the general query to a non-zero value. This value SHOULD be greater than one. If a gateway retransmits membership state change messages, it will retransmit them (robustness variable - 1) times. The QRV field is defined in Section 4.1.6 in [RFC3376] and Section 5.1.8 in [RFC3810].

A relay SHOULD set the Maximum Response Code field in the general query to a value of 1 to trigger an immediate response from the gateway (some host IGMP/MLD implementations may not accept a value of zero). A relay SHOULD NOT use the IGMPv3/MLDv2 Query Response Interval variable, if available, to generate the Maximum Response Code field value as the Query Response Interval variable is used in setting the duration of group state timers and must not be set to such a small value. The Maximum Response Code field is defined in Section 4.1.1 in [RFC3376] and Section 5.1.3 in [RFC3810]. See Section 5.3.3.7.

5.3.3.4. Handling a Membership Update Message

This section describes relay requirements related to the membership update portion of the message sequence described in Section 4.2.1.2.

When a relay receives a Membership Update message it must first determine whether it should accept or ignore the message. A relay MUST NOT make any changes to group membership and forwarding state if the message fails to satisfy any of the following requirements:

o The IP datagram encapsulated within the message MUST be one of the following:

  * IPv4 datagram carrying an IGMPv2 or IGMPv3 Membership Report message.

  * IPv4 datagram carrying an IGMPv2 Leave Group message.

  * IPv6 datagram carrying an MLDv1 or MLDv2 Multicast Listener Report message.

  * IPv6 datagram carrying MLDv1 Multicast Listener Done message.

- The encapsulated IP datagram MUST satisfy the IP header requirements for the IGMP or MLD message type as described in Section 4 of [RFC3376], Section 2 of [RFC2236], Section 5 of [RFC3810], and Section 3 of [RFC2710], with the following exception - a relay MUST accept an IGMP or MLD message regardless of the IP source address carried by the datagram.

- The total length of the encapsulated IP datagram as computed from the lengths contained in the datagram header(s) MUST NOT exceed the available field length within the Membership Update message.

- The computed checksums for the encapsulated IP datagram and its payload MUST match the values contained therein. Checksum computation and verification varies by protocol; See [RFC0791] for IPv4, [RFC3376] for IGMPv3, and [RFC4443] for MLD (ICMPv6).

- If processing of the encapsulated IGMP or MLD message would result in an allocation of new state or a modification of existing state, the relay MUST authenticate the source of the Membership message by verifying that the value contained in the Response MAC field equals the MAC value computed from the fields in the Membership Update message datagram. If a time-varying private secret is used in the computation of a Response MAC, the relay MUST retain the previous version of the private secret for use in authenticating Membership Updates sent during the subsequent query interval. If the first attempt at Response MAC authentication fails, the relay MUST attempt to authenticate the Response MAC using the previous private secret value unless 2*query_interval time has elapsed since the private secret change. See Section 5.3.5.

A relay MAY skip source authentication to reduce the computational cost of handling Membership Update messages if the relay can make a trivial determination that the IGMP/MLD message carried by the Membership Update message will produce no changes in group membership or forwarding state. The relay does not need to compute and compare MAC values if it finds there are no group subscriptions for the source of the Membership Update message and either of the following is true:

- The encapsulated IP datagram is an IGMPv3 Membership Report or MLDv2 Multicast Listener Report message that contains no group records. This may often be the case for gateways that continuously repeat the membership update cycle even though they have no group subscriptions to report.

- The encapsulated IP datagram is an IGMPv2 Leave Group or MLDv1 Multicast Listener Done message.

The IGMP and MLD protocol specifications indicate that senders SHOULD use a link-local source IP address in message datagrams. This requirement must be relaxed for AMT because gateways and relays do not share a common subnet. For this reason, a relay implementation MUST accept IGMP and MLD datagrams regardless of the source IP address they carry.

Once a relay has determined that the Membership Update message is valid, it processes the encapsulated IGMP or MLD membership message to update group membership state and communicates with the multicast protocol to update forwarding state and possibly send multicast protocol messages towards upstream routers. The relay MUST ignore any octets that might exist between the encapsulated IP datagram and the end of the Membership Update message.

As described in Section 4.2.2, a relay uses the source IP address and source UDP port carried by a Membership Update messages to identify a tunnel endpoint. A relay uses the tunnel endpoint as the destination address for any Multicast Data messages it sends as a result of the group membership and forwarding state created by processing the IGMP/MLD messages contained in Membership Update messages received from the endpoint.

If a Membership Update message originates from a new endpoint, the relay MUST determine whether it can accept updates from a new endpoint. If a relay has been configured with a limit on the total number of endpoints, or a limit on the total number of endpoints for a given source address, then the relay MAY ignore the Membership Update message and possibly withdraw any Relay Discovery Address Prefix announcement that it might have made. See Section 5.3.3.8.

A relay MUST maintain some form of group membership database for each endpoint. The per-endpoint databases are used update a forwarding table containing entries that map an (*,G) or (S,G) subscription to a list of tunnel endpoints.

A relay MUST maintain some form of group membership database representing a merger of the group membership databases of all endpoints. The merged group membership database is used to update upstream multicast forwarding state.

A relay MUST maintain a forwarding table that maps each unique (*,G) and (S,G) subscription to a list of tunnel endpoints. A relay uses this forwarding table to provide the destination address when performing UDP/IP encapsulation of the incoming multicast IP datagrams to form Multicast Data messages.

If a group filter mode for a group entry on a tunnel endpoint is EXCLUDE, the relay SHOULD NOT forward datagrams that originate from sources in the filter source list unless the relay architecture does not readily support source filtering. A relay MAY ignore the source list if necessary because gateways are expected to do their own source filtering.

5.3.3.5. Handling a Teardown Message

This section describes relay requirements related to the teardown message sequence described in Section 4.2.1.3.

When a relay (that supports the Teardown message) receives a Teardown message, it MUST first authenticate the source of the Teardown message by verifying that the Response MAC carried by the Teardown message is equal to a MAC value computed from the fields carried by the Teardown message. The method used to compute the MAC differs from that used to generate and validate the Membership Query and Membership Update messages in that the source IP address and source UDP port number used to compute the MAC are taken from the Gateway IP Address and Gateway Port Number field in the Teardown message rather than from the IP and UDP headers in the datagram that carries the Teardown message. The MAC computation is described Section 5.3.5. A relay MUST ignore a Teardown message If the computed MAC does not equal the value of the Response MAC field.

If a relay determines that a Teardown message is authentic, it MUST immediately stop transmitting Multicast Data messages to the endpoint identified by the Gateway IP Address and Gateway Port Number fields in the message. The relay MUST eventually delete any group membership and forwarding state associated with the endpoint, but MAY delay doing so to allow a gateway to recreate group membership state on a new endpoint and thereby avoid making unnecessary (temporary) changes in upstream routing/forwarding state.

The state changes made by a relay when processing a Teardown message MUST be identical to those that would be made as if the relay had received an IGMP/MLD report that would cause the IGMP or MLD protocol to delete all existing group records in the group membership database associated with the endpoint. The processing of the Teardown message should trigger or mimic the normal interaction between IGMP or MLD and a multicast protocol to produce required changes in forwarding state and possibly send prune/leave messages towards upstream routers.

5.3.3.6. Handling Multicast IP Datagrams

When a multicast IP datagram is forwarded to the relay pseudo-interface, the relay MUST, for each gateway that has expressed an interest in receiving the datagram, encapsulate the IP datagram into a Multicast Data message or messages and send that message or messages to the gateway. This process is highly implementation dependent, but conceptually requires the following steps:

- Use the IP datagram source and destination address to look up the appropriate (*,G) or (S,G) entry in the endpoint forwarding table created for the pseudo-interface as a result of IGMP/MLD processing.

- Possibly replicate the datagram for each gateway endpoint listed for that (*,G) or (S,G) entry.

- If the multicast IP datagram size exceeds the Tunnel MTU as determined according to the procedure described in Section 5.3.3.6.1, the relay must execute the procedure described in Section 5.3.3.6.2.

- Encapsulate and transmit the IP datagram according to the procedure described in Section 5.3.3.6.3.

The relay pseudo-interface MUST ignore any other IP datagrams forwarded to the pseudo-interface.

5.3.3.6.1. Path and Tunnel MTU

A relay MUST compute a Tunnel MTU (TMTU) value for each AMT tunnel that originates on the relay. A relay will use the TMTU value to determine whether an incoming multicast IP datagram can be delivered downstream in a Membership Data message without fragmentation. A relay MUST compute the TMTU by subtracting the size of the Membership Data message headers (IP, UDP, and AMT) from the current Path MTU (PMTU) associated with each AMT tunnel. The relay MUST maintain a PMTU value on a per-tunnel or per-relay basis. A relay MUST support one or both of the following methods for determining the PMTU value:

- The relay MAY provide a configuration option that establishes a fixed PMTU that will be applied to all AMT tunnels originating at the relay.

- The relay MAY dynamically adjust PMTU value(s) in response to receipt of ICMP/ICMPv6 "Datagram Too Big" messages as described in [RFC1191] and [RFC1981].

If a relay supports dynamic adjustment of per-tunnel or per-relay PMTU values in response to ICMP messages, the relay MUST provide a configuration option that disables this feature and also provide a configuration option that establishes a minimum PMTU for all tunnels. These configuration options may be used to mitigate certain types of denial of service attacks (See (Section 6)). When dynamic PMTU adjustments are disabled, the PMTU for all tunnels MUST default to the Link MTU (first-hop) on the downstream interface.

5.3.3.6.2. MTU Filtering Procedure

This section defines procedures that a relay must execute when it receives a multicast datagram whose size is greater than the Tunnel MTU of the tunnel or tunnels through which it must be delivered.

5.3.3.6.2.1. IPv4 Multicast IP Datagrams

If the DF bit in the multicast datagram header is set to 1 (Don’t Fragment), the relay MUST discard the packet and, if the datagram originated from an SSM source, send an ICMPv4 [RFC0792] Destination Unreachable message to the source, with type equal to 4 (fragmentation needed and DF set). The ICMP Destination Unreachable message MUST contain an next-hop MTU (as specified by [RFC1191]) and the relay MUST set the next-hop MTU to the TMTU associated with the tunnel or tunnels. If the DF bit in the multicast datagram header is set to 0 (May Fragment), the relay MUST fragment the datagram and encapsulate each fragment within Multicast Data messages for transmission through the tunnel or tunnels. This ensures that gateways will receive complete, non-fragmented Multicast Data messages, containing fragmented multicast datagram payloads. The relay SHOULD avoid generating a separate ICMP message for each tunnel, but instead send a single ICMP message with a Next-hop MTU equal to the smallest TMTU of all tunnels to which the datagram was to be forwarded.

5.3.3.6.2.2. IPv6 Multicast IP Datagrams

The relay MUST discard the packet and, if the datagram originated from an SSM source, send an ICMPv6 [RFC4443] Packet Too Big message to the payload source. The MTU specified in the Packet Too Big message MUST be equal to the TMTU associated with the tunnel or tunnels. The relay SHOULD avoid generating a separate ICMPv6 message for each tunnel, but instead send a single ICMPv6 message with a Next-hop MTU equal to the smallest TMTU of all tunnels to which the datagram was to be forwarded.

5.3.3.6.3. Encapsulation Procedure

A relay encapsulates a multicast IP datagram in a UDP/IP Membership Data message, using the tunnel endpoint UDP/IP address as the destination address and the unicast relay address and IANA-assigned AMT port number as the source UDP/IP address. To ensure successful NAT traversal, the source address and port MUST match the destination address and port carried by the Membership Update message sent by the gateway to create the forwarding table entry.

If possible, the relay SHOULD compute a valid, non-zero checksum for the UDP datagram carrying the Multicast Data message. See Section 4.2.2.3.

The following sections describe additional requirements related to the IP protocol of the tunnel and that of the multicast IP datagram.

5.3.3.6.3.1. Tunneling over IPv4

When a relay delivers an IPv4 payload over an IPv4 tunnel, and the DF Bit in the payload header is set to 1 (Don’t Fragment), the relay MUST set the DF bit in the Multicast Data IP header to 1. When a relay delivers an IPv4 payload over an IPv4 tunnel, and the DF Bit in the payload header is set to 0 (May Fragment), by default, the relay MUST set the DF bit in the Multicast Data IP header to 1. However, a relay MAY provide a configuration option that allows the DF bit to be copied from the payload header to the Multicast Data IP header to allow downstream fragmentation of the Multicast Data message. When a relay delivers an IPv6 payload over an IPv4 tunnel, the relay MUST set the DF bit in the Multicast Data IP header to 1. The relay MUST NOT transmit a Multicast Data message with an IP header in which the MF (More Fragments) bit is set to 1.

5.3.3.6.3.2. Tunneling over IPv6

When a tunneling over IPv6, a relay MUST NOT emit a Multicast Data message datagram containing an IPv6 fragment header.

5.3.3.6.4. Handling Destination Unreachable Messages

If a relay receives a sequence of ICMP or ICMPv6 messages of type "Destination Unreachable" in response to transmission of a sequence of AMT Multicast Data messages to a gateway, the relay SHOULD discontinue sending messages to that gateway and shutdown the tunnel for that gateway (Handling of ICMP "Destination Unreachable" messages with code 4, "fragmentation required" is covered in Section 5.3.3.6.1). If a relay provides this capability, it MUST provide a configuration option that indicates what number of sequential "Destination Unreachable" messages can be received and ignored before the relay will automatically shutdown a tunnel.

5.3.3.7. State Timers

A relay MUST maintain a timer or timers whose expiration will trigger the removal of any group subscriptions and forwarding state previously created for a gateway endpoint should the gateway fail to refresh the group membership state within a specified time interval.

A relay MAY use a variant of the IGMPv3/MLDv2 state management protocol described in Section 6 of [RFC3376] or Section 7 of [RFC3810], or may maintain a per-endpoint timer to trigger the deletion of group membership state.

If a per-endpoint timer is used, the relay MUST restart this timer each time it receives a new Membership Update message from the gateway endpoint.

The endpoint timer duration MAY be computed from tunable IGMP/MLD variables as follows:

((Robustness_Variable) * (Query_Interval)) + Query_Response_Interval

If IGMP/MLD default values are used for these variables, the gateway will timeout after 125s * 2 + 10s = 260s. The timer duration MUST be greater than the query interval suggested in the last Membership Query message sent to the gateway endpoint.

Regardless of the timers used (IGMPv3/MLDv2 or endpoint), the Query_Response_Interval value SHOULD be greater than or equal to 10s to allow for packet loss and round-trip time in the Request/ Membership Query message exchange.

5.3.3.8. Relay Resource Management

A relay may be configured with various service limits to ensure a minimum level of performance for gateways that connect to it.

If a relay has determined that it has reached or exceeded maximum allowable capacity or has otherwise exhausted resources required to support additional gateways, it SHOULD withdraw any Relay Discovery Address Prefix it has advertised into the unicast internetwork and SHOULD set the L-flag in any Membership Query messages it returns to gateways while in this state.

If the relay receives an update from a gateway that adds group membership or forwarding state for an endpoint that has already reached maximum allowable state entries, the relay SHOULD continue to accept updates from the gateway but ignore any group membership/forwarding state additions requested by that gateway.

If the relay receives an update from a gateway that would create a new tunnel endpoint for a source IP address that has already reached the maximum allowable number of endpoints (maximum UDP ports), it should simply ignore the Membership Update.

5.3.4. Shutdown

The following steps should be treated as an abstract description of the shutdown procedure for a relay:

- Withdraw the Relay Discovery Address Prefix advertisement (if used).

- Stop listening for Relay Discovery messages.

- Stop listening for control messages from gateways.

- Stop sending data messages to gateways.

- Delete all AMT group membership and forwarding state created on the relay, coordinating with the multicast routing protocol to update the group membership state on upstream interfaces as required.

5.3.5. Response MAC Generation

A Response MAC value is computed by the relay. A Response MAC computation is required in the following situations:

- To generate a Response MAC value from a Request message for inclusion in a Membership Query message.

- To generate a Response MAC value from a Membership Update message for use in authenticating the Response MAC carried within that message.

- To generate a Response MAC value from a Teardown message to authenticate the Response MAC carried within that message.

Gateways treat the Response MAC field as an opaque value, so a relay implementation may generate the MAC using any method available to it. The RECOMMENDED method for computing the Response MAC is to compute a cryptographically-secure hash or keyed-hash digest from the following values:

- The Source IP address of the message (or Teardown Gateway IP Address field)

- The Source UDP port of the message (or Teardown Gateway Port Number field)

- The Request Nonce contained in the message.

- A private secret or key known only to the relay.

5.3.6. Private Secret Generation

If the relay implementation uses a private secret (or key) to compute the Response MAC value, the relay SHOULD periodically compute a new private secret. The RECOMMENDED maximum interval is 2 hours. A relay MUST retain the prior secret for use in verifying MAC values that were sent to gateways just prior to the use of the new secret.

6. Security Considerations

AMT is not intended to be a strongly secured protocol. In general, the protocol provides the same level of security and robustness as is provided by the UDP, IGMP and MLD protocols on which it relies. The lack of strong security features can largely be attributed to the desire to make the protocol light-weight by minimizing the state and computation required to service a single gateway, thereby allowing a relay to service a larger number of gateways.

Many of the threats and vectors described in [RFC3552] may be employed against the protocol to launch various types of denial-of-service attacks that can affect the functioning of gateways or their ability to locate and communicate with a relay. These scenarios are described below.

As is the case for UDP, IGMP and MLD, the AMT protocol provides no mechanisms for ensuring message delivery or integrity. The protocol does not provide confidentiality - multicast groups, sources and streams requested by a gateway are sent in the clear.

The protocol does use a three-way handshake to provide trivial source authentication for state allocation and updates (see below). The protocol also requires gateways and relays to ignore malformed messages and those messages that do not carry expected address values or protocol payload types or content.

6.1. Relays

The three-way handshake provided by the membership update message sequence (See (Section 4.2.1.2)) provides a defense against source-spoofing-based resource-exhaustion attacks on a relay by requiring source authentication before state allocation. However, attackers may still attempt to flood a relay with Request and Membership Update messages to force the relay to make the MAC authentication computations in an effort to consume computational resources. Implementations may choose to limit the frequency with which a relay responds to Request messages sent from a single IP address or IP address and UDP port pair, but support for this functionality is not required. The three-way handshake provides no defense against an eavesdropping or man-in-the-middle attacker.

Attackers that execute the gateway protocol may consume relay resources by instantiating a large number of tunnels or joining a large number of multicast streams. A relay implementation should provide a mechanism for limiting the number of tunnels (Multicast Data message destinations) that can be created for a single gateway source address. Relays should also provide a means for limiting the number of joins per tunnel instance as a defense against these attacks.

Relays may withdraw their AMT anycast prefix advertisement when they reach configured maximum capacity or exhaust required resources. This behavior allows gateways to use the relay discovery process to find the next topologically-nearest relay that has advertised the prefix. This behavior also allows a successful resource exhaustion attack to propagate from one relay to the next until all relays reachable using the anycast address have effectively been taken offline. This behavior may also be used to acquire the unicast addresses for individual relays which can then be used to launch a DDoS attack on all of the relays without using the relay discovery process. To prevent wider disruption of AMT-based distribution network, relay anycast address advertisements can be limited to specific administrative routing domains. This will isolate such attacks to a single domain.

The Path and Tunnel MTU adjustment (discovery) procedure described in Section 5.3.3.6.1 is vulnerable to two denial of service attacks (see Section 8 of [RFC1191] for details). Both attacks are based upon on a malicious party sending forged ICMPv4 Destination Unreachable or ICMPv6 Packet Too Big messages to a host. In the first attack, the forged message indicates an inordinately small Path MTU. In the second attack, the forged message indicates an inordinately large Path MTU. In both cases, throughput is adversely affected. In order to mitigate such attacks, relay implementations MUST include a configuration option to disable Path MTU adjustments on AMT tunnels.

6.2. Gateways

A passive eavesdropper may launch a denial-of-service attack on a gateway by capturing a Membership Query or Membership Update message and using the request nonce and message authentication code carried by the captured message to send a spoofed a Membership Update or Teardown message to the relay. The spoofed messages may be used to modify or destroy group membership state associated with the gateway, thereby changing or interrupting the multicast traffic flows.

A passive eavesdropper may also spoof Multicast Data messages in an attempt to overload the gateway or disrupt or supplant existing traffic flows. A properly implemented gateway will filter Multicast Data messages that do not originate from the expected relay address and should filter non-multicast packets and multicast IP packets whose group or source addresses are not included in the current reception state for the gateway pseudo-interface.

An active eavesdropper may launch a man-in-the-middle attack in which messages normally exchanged between a gateway and relay are intercepted, modified, spoofed or discarded by the attacker. The attacker may deny access to, modify or replace requested multicast traffic. The AMT protocol provides no means for detecting or defending against a man-in-the-middle attack - any such functionality must be provided by multicast receiver applications through independent detection and validation of incoming multicast datagrams.

The anycast discovery technique for finding relays (see Section 4.1.4) introduces a risk that a rogue router or a rogue AS could introduce a bogus route to a specific Relay Discovery Address prefix, and thus divert or absorb Relay Discovery messages sent by gateways. Network managers must guarantee the integrity of their routing to a particular Relay Discovery Address prefix in much the same way that they guarantee the integrity of all other routes.

6.3. Encapsulated IP Packets

An attacker forging or modifying a Membership Query or Membership Update message may attempt to embed something other than an IGMP or MLD message within the encapsulated IP packet carried by these messages in an effort to introduce these into the recipient’s IP stack. A properly implemented gateway or relay will ignore any such messages - and may further choose to ignore Membership Query messages that do not contain a IGMP/MLD general queries or Membership Update messages that do not contain IGMP/MLD membership reports.

Properly implemented gateways and relays will also filter encapsulated IP packets that appear corrupted or truncated by verifying packet length and checksums.

7. IANA Considerations

7.1. IPv4 and IPv6 Anycast Prefix Allocation

The following unicast prefixes have been assigned to provide anycast routing of relay discovery messages to public AMT Relays as described in Section 4.1.4.

7.1.1. IPv4

We suggest that IANA assign an x.x.x.x/24 from the IPv4 Recovered Address Space Registry, but any /24 which has been unassigned and unadvertised for at least twelve months is acceptable. The block should be registered as follows:

Attribute	Value
Address Block	x.x.x.x./24
Name	AMT
RFC	[TBD]
Allocation Date	[TBD]
Termination Date	N/A
Source	True
Destination	True
Forwardable	True
Global	True
Reserved-by-Protocol	False

7.1.2. IPv6

IANA should register the following special-purpose address block for IPv6 anycast AMT relay discovery.

Attribute	Value
Address Block	2001:0003::/32
Name	AMT
RFC	[TBD]
Allocation Date	[TBD]
Termination Date	N/A
Source	True
Destination	True
Forwardable	True
Global	True
Reserved-by-Protocol	False

7.2. UDP Port Number

The UDP port number 2268 has been reserved with IANA for use in the implementation and deployment of AMT. The protocol described by this document continues to use this port number according to the intent of the original request. IANA should assign this port number to AMT upon acceptance of this I-D.

8. Contributors

The following people provided significant contributions to the design of the protocol and earlier versions of this specification:

Amit Aggarwal
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052-6399
USA
Email: amitag@microsoft.com

Thomas Morin
Orange
2, avenue Pierre Marzin
Lannion 22300
France
Email: thomas.morin@orange.com

Dirk Ooms
OneSparrow
Robert Molsstraat 11; 2018 Antwerp
Belgium
EMail: dirk@onesparrow.com

Tom Pusateri
!j
Wake Forest, NC
USA
Email: pusateri@bangj.com

Dave Thaler
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052-6399
USA
Email: dthaler@microsoft.com

9. Acknowledgments

The authors would like to thank the following individuals for their suggestions, comments, and corrections:

Mark Altom
Toerless Eckert
Marshall Eubanks
Gorry Fairhurst
Dino Farinacci
Lenny Giuliano
Andy Huang
Tom Imburgia
Patricia McCrink
Han Nguyen
Doug Nortz
Pekka Savola
Robert Sayko
Greg Shepherd
Steve Simlo
Mohit Talwar
Lorenzo Vicisano
Kurt Windisch
John Zwiebel

The anycast discovery mechanism described in this document is based on similar work done by the NGTrans WG for obtaining automatic IPv6 connectivity without explicit tunnels ("6to4"). Tony Ballardie provided helpful discussion that inspired this document.

Juniper Networks was instrumental in funding several versions of this draft as well as an open source implementation.

10. References

10.1. Normative References

[RFC3376] Cain, B., Deering, S., Kouvelas, I., Fenner, B., and A. Thyagarajan, "Internet Group Management Protocol, Version 3", RFC 3376, October 2002.

[RFC3810] Vida, R. and L. Costa, "Multicast Listener Discovery Version 2 (MLDv2) for IPv6", RFC 3810, June 2004.

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

[RFC4607] Holbrook, H. and B. Cain, "Source-Specific Multicast for IP", RFC 4607, August 2006.

[RFC4787] Audet, F. and C. Jennings, "Network Address Translation (NAT) Behavioral Requirements for Unicast UDP", BCP 127, RFC 4787, January 2007.

10.2. Informative References

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

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

[RFC1112] Deering, S., "Host extensions for IP multicasting", STD 5, RFC 1112, August 1989.

[RFC1191] Mogul, J. and S. Deering, "Path MTU discovery", RFC 1191, November 1990.

[RFC1546] Partridge, C., Mendez, T., and W. Milliken, "Host Anycasting Service", RFC 1546, November 1993.

[RFC1981] McCann, J., Deering, S., and J. Mogul, "Path MTU Discovery for IP version 6", RFC 1981, August 1996.

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

[RFC2236] Fenner, W., "Internet Group Management Protocol, Version 2", RFC 2236, November 1997.

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

[RFC2663] Srisuresh, P. and M. Holdrege, "IP Network Address Translator (NAT) Terminology and Considerations", RFC 2663, August 1999.

[RFC2710] Deering, S., Fenner, W., and B. Haberman, "Multicast Listener Discovery (MLD) for IPv6", RFC 2710, October 1999.

[RFC3552] Rescorla, E. and B. Korver, "Guidelines for Writing RFC Text on Security Considerations", BCP 72, RFC 3552, July 2003.

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

[RFC4443] Conta, A., Deering, S., and M. Gupta, "Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification", RFC 4443, March 2006.

[RFC4601] Fenner, B., Handley, M., Holbrook, H., and I. Kouvelas, "Protocol Independent Multicast - Sparse Mode (PIM-SM): Protocol Specification (Revised)", RFC 4601, August 2006.

[RFC4786] Abley, J. and K. Lindqvist, "Operation of Anycast Services", BCP 126, RFC 4786, December 2006.

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

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

Author’s Address

Gregory Bumgardner

Phone: +1 541 343 6790
Email: gbumgard@gmail.com

mboned
Internet-Draft
Intended status: Informational
Expires: February 25, 2011

T. Hayashi,
H. Satou,
H. Ohta
NTT
H.He
Nortel
S. Vaidya
Cisco Systems, Inc.
August 24, 2010

Requirements for Multicast AAA coordinated between Content Provider(s) and Network Service Provider(s)
draft-ietf-mboned-maccnt-req-10

Abstract

This memo presents requirements in the area of accounting and access control for IP multicasting. The scope of the requirements is limited to cases where Authentication, Accounting and Authorization (AAA) functions are coordinated between Content Provider(s) and Network Service Provider(s).

In order to describe the new requirements of a multi-entity Content Deliver System(CDS) using multicast, the memo presents three basic business models: 1) the Content Provider and the Network Provider are the same entity, 2) the Content Provider(s) and the Network Provider(s) are separate entities and users are not directly billed, and 3) the Content Provider(s) and the Network Provider(s) are separate entities and users are billed based on content consumption or subscriptions. The requirements of these three models are listed and evaluated as to which aspects are already supported by existing technologies and which aspects are not.

General requirements for accounting and admission control capabilities including quality-of-service (QoS) related issues are listed and the constituent logical functional components are presented.

This memo assumes that the capabilities can be realized by integrating AAA functionalities with a multicast CDS system, with IGMP/MLD at the edge of the network.

Status of this Memo

1. Introduction

Broadband access networks such as ADSL (Asymmetric Digital Subscriber Line) or FTTH (Fiber to the Home) have been deployed widely in recent years. Content Delivery Service (CDS) is expected to be a major application provided through broadband access networks. Because many services such as television broadcasting require huge bandwidth (e.g., 6Mbit/s) and processing power at the content server(s), IP multicast is used as an efficient delivery mechanism for CDS.

A single entity may design and be responsible for a system that covers the various common high-level requirements of a multicasting CDS such as 1) content serving, 2) the infrastructure to multicast it, 3) network and content access control mechanisms. For cases in which the business model includes the direct billing of users, the single provider of both content and network services has sufficient data in its control to bill users based on their content consumption. Furthermore it is possible to tie access to the network and QoS based on a user’s contract status. Therefore current technologies support the single entity case.

Often, however, the content provision and network provision roles are split between separate entities. Commonly, Content Providers (CP) do not build and maintain their own multicast network infrastructure as this is not their primary business area. Instead, CPs often purchase transport and management services from network service providers. This memo lists the requirements of a business model in which the NSP provides CDS using multicast as one such contractible service.

The direct revenue source for the multiple entity provider is a defining aspect of the business model which often has implications on requirements for the technologies that support the system. There are cases such as the the advertising-based model where billing end-users is not done and therefore accounting of content consumption can be anonymous and/or in aggegrate. In these cases the requirements of the business model for accounting for billing purposes are already supported by existing technologies. However, the NSP can not guarantee high quality transmission on a per-content basis with existing technologies.

There is also the business model in which the individual user of multicasted contents is the source of revenue for both consumed content and network resources. In this model the NSP wants to receive the appropriate fees for multicast services and the NSP undertakes collecting bills as a proxy for the CPs. The NSP may provide high quality service by admission control. Current standards do not fully support this model and this memo will list the requirements which need to be supported.

2. Definitions and Abbreviations

2.1. Definitions

Authentication: action for identifying a user as a genuine one.

Authorization: action for giving permission for a user to access content or the network.

Eligible user: Users may be eligible (permitted) to access resources because of the attributes they have (e.g., delivery may require possession of the correct password or digital certificate), their equipment has (e.g., content may only be eligible to players that can decode H.264 or 3GPP streams), their access network has (e.g., HDTV content may only be eligible to users with 10 Mbps or faster access line), or because of where they are in network topology (e.g., HDTV content may not be eligible for users across congested links) or in actual geography (e.g., content may only be licensed for distribution to certain countries), and, of course, a mix of attributes may be required for eligibility or ineligibility.

User: In this document user refers to a requester and a recipient of multicast data, termed a viewer in CDS.

User-based accounting: actions for grasping each user’s behavior, when she/he starts/stops to receive a channel, which channel she/he receives, etc.

2.2. Abbreviations

AAA: Authentication, Accounting and Authorization

ASM: Any-Source Multicast

CDS: Content Delivery Service

CP: Content Provider

IGMP: Internet Group Management Protocol

MLD: Multicast Listener Discovery

NSP: Network Service Provider

SSM: Source Specific Multicast

QoS: Quality of Service

3. Current Business Models

3.1. Single entity model where CP and NSP are the same entity

One existing business model is that of a single entity responsible for both content and network service provision which bills its users based on content provision. (See figure below.)

```
          +-----------------------------------------------------+
          |              +---------+                            |
          |              | Content |                            |
          |              | Server  |                            |
          |              +----+----+                            |
          |                   |                                 |
          | CP+NSP    +-------+-------+                         |
          |           | Provider Edge |                         |
          |           +-------+-------+                         |
          |                   |                                 |
          |                   |                                 |
          |           +-------------+                           |
          |           | User Edge   |                           |
          |           +--+---+---+--+                           |
          |             /    |    \                             |
          +----------- / --- | --- \ ---------------------------+
                      /      |      \
                     /       |       \ <- user/network interface
                    /        |        \
         +---------++  +-----+----+   ++---------+
         |Client #A |  |Client #B |   |Client #C |
         +----------+  +----------+   +----------+
           User A       User B         User C
```

Example of CDS network configuration

Figure 1

In this model the network can query a content-policy-enabled AAA server within its own domain at the time a user requests content. The network can provide the AAA server with information such as user identity, device identity, the requested content (channel), geographic information, method of network connection, etc. that might be required for the content provision authorization decision. It is therefore possible to configure a network to deny network access based on the content policy decision.

In this model there are no issues of mapping user identities between different entity domains. The provider has access to the information on which user accessed from which point on what device. Furthermore as network provider they can record not only when a user joined or left a certain channel, but also if packets were actually delivered. Moreover, there are no inter-entity security and privacy concerns between the CP and NSP.

The single entity network service and content provider also knows the content schedules for various channels. This is important not only for time and content-sensitive authorization decisions but also for providing meaningful billing details to end users.

3.2. Multiple entity model without direct content-based billing

An additional model for delivering contents over a CDS is the advertising-based model where billing end-users is not done. In this model the four different roles may be filled by separate entities: Content Provider (CP), Network Service Provider (NSP), user clients, and advertising sponsors. In the general case of this business model, insofar as the advertiser does not require user-based metrics the accounting of content consumption can be anonymous and/or in aggregrate and can be off-line from the multicast-with-AAA CDS system itself. Therefore this model does not require any new standards to provide user-based accounting for a multi-entity CDS using multicast with AAA. (Providing this data in near real-time and inline would entail further requirements which can be dealt with in a separate memo if necessary.)

A more complex version of this business model is conceivable in which a CP may require a user to enter into a subscription contract, even when the user does not get billed for content consumption. For example, a CP may value individual data because it allows it to supply the advertisers with rich, user-segmented data and charge a higher premium. In that case the requirements of the next section "CDS with direct billing of the end user" are generally applicable because of the need to link the user data which the CP has to the actual viewing (or stream downloading) data that the NSP has.

4. Proposed Model: Multity-entity CDS

In this model the networks for CDS contain three different types of entities: Content Provider (CP), Network Service Provider (NSP), and user clients. An NSP owns the network resources (infrastructure). It accommodates content providers on one side and accommodates user clients on the other side. NSP provides the network for CDS to two entities (i.e., CPs and user clients). A CP provides content to each user through the network of NSPs and charges users for content. NSPs are responsible for delivering the content to user clients, and for controlling the network resources. A NSP charges a user or a CP for network usage. A NSP may charge users for content as a proxy of the CP.

```
+-------------+  +-------------+  +-------------+
| CP          |  | CP          |  | CP          |
|          #1 |  |          #2 |  |          #3 |
| +---------+ |  | +---------+ |  | +---------+ |
| | Content | |  | | Content | |  | | Content | |
| | Server  | |  | | Server  | |  | | Server  | |
| +-------+-+ |  | +----+----+ |  | +-+-------+ |
+----------\--+  +------|------+  +--/----------+
            \           |           /
             \          |          / <- network/network
              \         |         /     interface
+------------- \ ------ | ------ / ----+
|               \       |       /      |
|   NSP         +-+-----+-----+-+      |
|               | Provider Edge |      |
|               +-------+-------+      |
|                       |              |
|                       |              |
|             +--+------+---+          |
|             | User Edge   |          |
|             +--+---+---+--+          |
|               /    |    \            |
+------------- / --- | --- \ ----------+
              /      |      \
             /       |       \ <- user/network interface
            /        |        \
 +---------++  +-----+----+   ++---------+
 |Client #A |  |Client #B |   |client #C |
 +----------+  +----------+   +----------+
 User A        User B         User C
```

Example of CDS network configuration

Figure 2

The CP provides detailed channel information (e.g., Time table of each channel) to the information server which is either managed by the NSP or CP. An end-user client gets the information from the information server. In this model, multicasting is used in the NSP’s CDS network, and there are two different contracts. One is the contract between the NSP and the user which permits the user to access the basic network resources of the NSP. Another contract is between the CP and user to permit the user to subscribe to multicast content. Because the CP and NSP are different entities, and the NSP generally does not allow a CP to control (operate) the network resources of the NSP, user authorization needs to be done by the CP and NSP independently. Since there is no direct connection to the user/network interface, the CP cannot control the user/network interface. A user may want to move to another place, or may want to change her/his device (client) any time without interrupting her/his reception of services.

4.1. Information Required by Entities to Support the Proposed Business Model

User identification and Authentication:

The network should be able to identify and authenticate each user when they attempt to access the service requesting content. This user identification is required for:

authorization for content consumption eligibility

user tracking for billing based on actual content consumption and network resource usage

With current protocols (IGMP/MLD), the sender cannot distinguish which receivers (end hosts) are actually receiving the information. The sender must rely on the information from the multicasting routers. This can be complicated if the sender and routers are maintained by different entities. Furthermore, the current user associated with receiver must be identified.

User Authorization:

The network, at its option, should be able to authorize a user’s access to content or a multicast group, so as to meet any demands by a CP to prevent content access by ineligible users.

Sharing Programming data:

NSP needs a mechanism to receive channel programming data from the CP in order to provide the information to the user at channel selection time and also for somehow logging or recording what programming content has been streamed to the user. In some cases the CP may contract the NSP to bill the user as a proxy for the CP. In this case there needs to be a mechanism for supplying the user-based viewing history with human-meaningful channel data to the end-user.

Content usage information by user:

For billing and auditing purposes the CP needs the NSP to provide it with detailed per-user usage behavior indicating what content was consumed from when to when. There needs to be a mechanism to supply the user-based viewing history from the NSP to the CP. If the CP is selling on an on-demand model, or tiered subscription basis or supplies some sort of online account statement this history needs to be fed back to the CP in near real-time. To assemble such data on user behavior, it is necessary to precisely log information such as who (host/user) is accessing what content at what time (join action) until what time (leave action). The result of the access-control decision (e.g. results of authorization) would also be valuable information. The desired degree of logging precisions would depend on the application used.

Notification to Users of the Result of the Join Request:

It should be possible to provide information to the user about the status of his/her join request(granted/denied/other). Such information can be used to give meaningful feedback to the user.

5. Admission Control for Multicasting

In order to guarantee certain QoS it is important for network providers (at their option) to be able to protect their network resources from being wasted, (either maliciously or accidentally). The NSP should be able to apply appropriate access controlling actions based on user eligibility status:

The network should be able to apply necessary access controlling actions when an eligible user requests an action (such as a join or a leave.)

The network should be able to reject any action requested from an ineligible user.

In order to maintain a predefined QoS level, depending on the NSP’s policy, a user edge should be able to control the number of streams it serves to a user, and total bandwidth consumed to that user. For example if the number of streams being served to a certain user has reached the limit defined by the NSP’s policy, then the user edge should not accept a subsequent "join" until one of the existing streams is terminated. Similarly, if the NSP is controlling by per-user bandwidth consumption, then a subsequent "join" should not be accepted if delivery of the requested stream would push the consumed bandwidth over the NSP policy-defined limit.

The network may need to control the combined bandwidth for all channels at the physical port of the edge router or switch so that these given physical entities are not overflowed with traffic. This entails being able to control the number of channels delivered, the bandwidth for each channel and the combined bandwidth for all channels.

6. Reauthorization/ deauthorization requirements

A mechanism for periodic reauthorization of users who have already joined a channel stream should be supported. The reauthorization could be an authorization check based on the NSP’s eligibility requirements and/or could involve the NSP querying the CP for reauthorization of a user.

A mechanism for deauthorization should be supported for cases in which a user is deemed ineligible by the NSP and/or CP at the time of a reauthorization check. If a NSP revokes authorization for the network for a user it should force a leave, and record details of the leave (including the time and reason for the forced leave.) If a CP revokes authorization to content for a user the CP signals to the NSP to cease streaming to that user. An example usage case for deauthorizing a user is one where a user has a subscription or has paid for a certain amount of content and has reached that limit. In some models, it is conceivable that a CP could communicate the parameters for de-authorization to the NSP at the time of the original join’s authorization so as to make NSP->CP reauthorization requests unnecessary.

7. Performance requirements

Channel Join Latency and Leave Latency

Commercial implementations of IP multicasting are likely to have strict requirements in terms of user experience. Join latency is the time between when a user sends a "join" request and when the requested data streaming first reaches the user. Leave latency is the time between when a user sends a "leave" signal and when the network stops streaming to the user. Leave and Join latencies impact the acceptable user experience for fast channel surfing. In an IP-TV application, users are not going to be receptive to a slow response time when changing channels. If there are policies for controlling the number of simultaneous streams a user may access then channel surfing will be determined by the join and leave latencies. Furthermore, leave affects resource consumption: with a low "leave latency" network providers could minimize streaming content when there are no audiences. It is important that any overhead for authentication, authorization, and access-control be minimized at the times of joining and leaving multicast channels so as to achieve join and leave latencies acceptable in terms of user experience. For example this is important in an IP-TV application, because users are not going to be receptive to a slow response time when changing channels.

8. Concomitant requirements

Scalability

Solutions that are used for AAA and QoS enabled IP multicasting should scale enough to support the needs of content providers and network operators. NSP’s multicast access and QoS policies should be manageable for large scale users. (e.g. millions of users, thousands of edge-routers)

Service and Terminal Portability:

Depending on the service, networks should allow for a user to receive a service from different places and/or with a different terminal device.

Deployable as Alternative to Unicast

IP Multicasting would ideally be available as an alternative to IP unicasting when the "on-demand" nature of unicasting is not required. Therefore interfaces to multicasting should allow for easy integration into CDS systems that support unicasting. Especially equivalent interfaces for authorization, access control and accounting capabilities should be provided.

Support of ASM and SSM

Both ASM (G), and SSM (S,G) should be supported as multicast models.

Support for Tunneled Multicast

The AAA requirements specified in this document should apply to both end-to-end native multicast and to tunnel-enabled multicast, such as AMT multicast: [I-D.ietf-mboned-auto-multicast]

Small Impact on the Existing Products

Impact on the existing products (e.g., protocols, software, etc.) should be as minimal as possible. Ideally the NSP should be able to use the same infrastructure (such as access control) to support commercial multicast services for the so called "triple play" services: voice (VoIP), video, and broadband Internet access services. When a CP requires the NSP to provide a level of QoS surpassing "best effort" delivery or to provide special services (e.g., to limited users with specific attributes), certain parameters of the CDS may be defined by a contractual relation between the NSP and the CP. However, just as for best-effort unicast, multicast allows for content sourced by CPs without a contractual relation with the NSP. Therefore, solutions addressing the requirements defined in this memo should not make obsolete multicasting that does not include AAA features. NSPs may offer tiered services, with higher QOS, accounting, authentication, etc., depending on contractual relation with the CPs. It is therefore important that Multicast AAA and QoS functions be as modular and flexible as possible.

Multicast Replication

The above requirements should also apply if multicast replication is being done on an access-node (e.g. DSLAMs or OLTs).

9. Constituent Logical Functional Components

Below is a diagram of a AAA enabled multicasting network, including the logical components within the various entities.

```
         +-------------------------------+
         | user                          |
         |+- - - - - - - - - - - - - - -+|
         || CPE                         ||
         ||                             ||
         |+- - - - - | - - - - - - - - -+|
         +-----------|-------------------+
         |
         -------|------ IFa
         |
         +-----------|-----------------------+
         |  NSP      |                       |
         |           |                       |
         |+- - - - - |- - _+   + - - - - - + |
         ||        |   | |   |           | |
         |    +------|-+ |       +--------+  |
         ||   | AN     | | |   | | MACF  || |
         |    |        | |       |        |  |
         ||   +------|-+ | |   | +---|----+| |
         |           |   |           |    |  |
         |           |   | |     IFd----- |  |
         |           |   |  IFb      |    |  |
         ||   +------|---+ | | | +---|----+| |
         |    |          |---|---| mAAA   |  |
         ||   | NAS      | | | | |(MACF *)|| | * optional
         |    +----------+ |     +--------+  |
         ||+- - - - - - - -+ - - |- - - - -+ |
         +-----------------------|-----------+
         |
         -------|------ IFc
         |
         +-----------------------|-------+
         | CP               +---------+  |
         |                  |  CP-AAA |  |
         |                  +---------+  |
         +-------------------------------+
```

AAA enabled multicasting network with admission control

Figure 3

The user entity includes the CPE (Customer Premise Equipment) which connects the receiver (s).

The NSP (Network Service Provider) includes the transport system and a logical element for multicast AAA functionality. The TS (transport system) is comprised of the access node and NAS (Network Access Server) An AN (Access Node) may be connected directly to mAAA or a

NAS relays AAA information between an AN and a mAAA. Descriptions of AN and its interfaces are out of the scope for this memo. The multicast AAA function may be provided by a mAAA which may include the function that downloads Join access control lists to the NAS (this function is referred to as the conditional access policy control function.)

Interface between mAAA and NAS

The interface between mAAA and the NAS is labeled IFb in Figure 3. Over IFb the NAS sends an access request to the NSP-mAAA and the mAAA replies. The mAAA may push conditional access policy to the NAS.

CP-AAA

The content provider may have its own AAA server which has the authority over access policy for its contents.

Interface between user and NSP

The interface between the user and the NSP is labeled IFa in Figure 3. Over IFa the user makes a multicasting request to the NSP. The NSP may in return forward multicast traffic depending on the NSP and CP’s policy decisions.

Interface between NSP and CP

The interface between the NSP and CP is labeled IFc. Over IFc the NSP requests to the CP-AAA for access to contents and the CP replies. CP may also send conditional access policy over this interface for AAA-proxying.

The NSP may also include a component that provides network resource management (e.g. QoS management), as described in section 5, "Admission Control for Multicasting". Resource management and admission control is provided by MACF (Multicast Admission Control Function). This means that, before replying to the user’s multicast request, the mAAA queries the MACF for a network resource access decision over the interface IFd. The MACF is responsible for allocating network resources for forwarding multicast traffic. MACF also receives Leave information from NAS so that MACF releases corresponding reserved resources.

10. Acknowledgments

The authors of this draft would like to express their appreciation to Christian Jacquenet of France Telecom whose contributions to the "AAA

Framework for Multicasting" [draft-ietf-mboned-multiaaa-framework] largely influenced this draft; Pekka Savola of Netcore Ltd.; Daniel Alvarez, and Toerless Eckert of Cisco Systems; Sam Sambasivan of AT&T; Sanjay Wadhwa, Greg Shepherd, and Leonard Giuliano of Juniper; Tom Anschutz and Steven Wright of BellSouth; Nicolai Leymann of T-Systems; Bill Atwood of Concordia University; Carlos Garcia Braschi of Telefonica Empresas; Mark Altom, Andy Huang, Tom Imburgia, Han Nguyen, Doug Nortz of ATT Labs; Marshall Eubanks in his role as mboned WG chair; Ron Bonica in his role as Director as the Operations and Management Area; Stephen Rife of Digital Garage and David Meyer in his former role as mboned WG chair as well as their thanks to the participants of the MBONED WG in general.

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

11. IANA Considerations

This memo does not raise any IANA consideration issues.

12. Security Considerations

Accounting capabilities can be used to enhance the security of multicast networks by excluding ineligible clients from the networks.

These requirements are not meant to address encryption issues. Any solution meeting these requirements should allow for the implementation of encryption such as MSEC on the multicast data.

13. Privacy considerations

Any solution which meets these requirements should weigh the benefits of user-based accounting with the privacy considerations of the user. For example solutions are encouraged when applicable to consider encryption of the content data between the content provider and the user in such a way that the Network Provider does not know the contents of the channel.

14. Conclusion

This memo describes general requirements for providing AAA and QoS enabled IP multicasting services in multi-entity models. A few models are evaluated with regard to their support by current technologies. The "multi-entity CDS with direct billing of the end user" model is presented and requirements for information sharing between entities and requirements for admission control to enable guaranteeing of QoS are derived. Performance requirements and concomitant requirements are also presented.

15. References

15.1. Normative References

[RFC2975] Aboba, B., Arkko, J., and D. Harrington, "Introduction to Accounting Management", RFC 2975, October 2000.

[RFC3376] Cain, B., Deering, S., Kouvelas, I., Fenner, B., and A. Thyagarajan, "Internet Group Management Protocol, Version 3", RFC 3376, October 2002.

[RFC3810] Vida, R. and L. Costa, "Multicast Listener Discovery Version 2 (MLDv2) for IPv6", RFC 3810, June 2004.

15.2. Informative References

[I-D.ietf-mboned-auto-multicast]
Thaler, D., Talwar, M., Aggarwal, A., Vicisano, L., and T. Pusateri, "Automatic IP Multicast Without Explicit Tunnels (AMT)", draft-ietf-mboned-auto-multicast-09 (work in progress), June 2008.

Authors’ Addresses

Tsunemasa Hayashi
Nippon Telegraph and Telephone Corporation
1-1 Hikarino’oka
Yokosuka-shi, Kanagawa 239-0847
Japan

Phone: +81 46 859 8790
Email: hayashi.tsunemasa@lab.ntt.co.jp

Hiroaki Satou
Nippon Telegraph and Telephone Corporation
3-9-11 Midoricho
Musashino-shi, Tokyo 180-8585
Japan

Phone: +81 422 59 4683
Email: satou.hiroaki@lab.ntt.co.jp

Hiroshi Ohta
Nippon Telegraph and Telephone Corporation
3-9-11 Midoricho
Musashino-shi, Tokyo 180-8585
Japan

Phone: +81 422 59 3617
Email: ohta.hiroshi@lab.ntt.co.jp

Haixiang He
Nortel
600 Technology Park Drive
Billerica, MA 01801
USA

Phone: +1 978 288 7482
Email: haixiang@nortel.com

Susheela Vaidya
Cisco Systems, Inc.
170 W. Tasman Drive
San Jose, CA 95134
USA

Phone: +1 408 525 1952
Email: svaidya@cisco.com

Copyright and License Notice

MBONED Working Group
Internet-Draft
Intended status: Standards Track
Expires: February 1, 2019

H. Asaeda
NICT
K. Meyer

W. Lee, Ed.

July 31, 2018

Mtrace Version 2: Traceroute Facility for IP Multicast
draft-ietf-mboned-mtrace-v2-26

Abstract

This document describes the IP multicast traceroute facility, named Mtrace version 2 (Mtrace2). Unlike unicast traceroute, Mtrace2 requires special implementations on the part of routers. This specification describes the required functionality in multicast routers, as well as how an Mtrace2 client invokes a query and receives a reply.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

Given a multicast distribution tree, tracing hop-by-hop downstream from a multicast source to a given multicast receiver is difficult because there is no efficient and deterministic way to determine the branch of the multicast routing tree on which that receiver lies. On the other hand, walking up the tree from a receiver to a source is easy, as most existing multicast routing protocols know the upstream router for each source. Tracing from a receiver to a source can involve only the routers on the direct path.

This document specifies the multicast traceroute facility named Mtrace version 2 or Mtrace2 which allows the tracing of an IP multicast routing path. Mtrace2 is usually initiated from an Mtrace2 client by sending an Mtrace2 Query to a Last Hop Router (LHR) or to a Rendezvous Point (RP). The RP is a special router where sources and receivers meet in Protocol Independent Multicast - Sparse Mode (PIM-SM) [5]. From the LHR/RP receiving the query, the tracing is directed towards a specified source if a source address is specified and source specific state exists on the receiving router. If no source address is specified or if no source specific state exists on a receiving LHR, the tracing is directed toward the RP for the specified group address. Moreover, Mtrace2 provides additional information such as the packet rates and losses, as well as other diagnostic information. Mtrace2 is primarily intended for the following purposes:

- To trace the path that a packet would take from a source to a receiver.

- To isolate packet loss problems (e.g., congestion).

- To isolate configuration problems (e.g., Time to live (TTL) threshold).

Figure 1 shows a typical case on how Mtrace2 is used. First-hop router (FHR) represents the first-hop router, LHR represents the last-hop router (LHR), and the arrow lines represent the Mtrace2 messages that are sent from one node to another. The numbers before the Mtrace2 messages represent the sequence of the messages that would happen. Source, Receiver and Mtrace2 client are typically hosts.

```
                  2. Request                 2. Request
                    +----+                    +----+
                    |    |                    |    |
                    v    |                    v    |
   +--------+    +-----+                        +-----+    +----------+
   | Source |----| FHR |----- The Internet -----| LHR |----| Receiver |
   +--------+    +-----+            |           +-----+    +----------+
                     \              |             ^
                      \             |            /
                       \            |           /
                        \           |          /
                3. Reply \          |         / 1. Query
                          \         |        /
                           \        |       /
                            \  +---------+ /
                             v | Mtrace2 |/
                               | client  |
                               +---------+
```

Figure 1

When an Mtrace2 client initiates a multicast trace, it sends an Mtrace2 Query packet to an LHR or RP for a multicast group and, optionally, a source address. The LHR/RP turns the Query packet into a Request. The Request message type enables each of the upstream routers processing the message to apply different packet and message validation rules than those required for handling of a Query message. The LHR/RP then appends a standard response block containing its interface addresses and packet statistics to the Request packet, then forwards the packet towards the source/RP. The Request packet is either unicasted to its upstream router towards the source/RP, or multicasted to the group if the upstream router’s IP address is not known. In a similar fashion, each router along the path to the source/RP appends a standard response block to the end of the Request packet before forwarding it to its upstream router. When the FHR receives the Request packet, it appends its own standard response block, turns the Request packet into a Reply, and unicasts the Reply back to the Mtrace2 client.

The Mtrace2 Reply may be returned before reaching the FHR under some circumstances. This can happen if a Request packet is received at an RP or gateway, or when any of several types of error or exception conditions occur which prevent sending of a request to the next upstream router.

The Mtrace2 client waits for the Mtrace2 Reply message and displays the results. When not receiving an Mtrace2 Reply message due to network congestion, a broken router (see Section 5.6), or a non-responding router (see Section 5.7), the Mtrace2 client may resend another Mtrace2 Query with a lower hop count (see Section 3.2.1), and repeat the process until it receives an Mtrace2 Reply message. The details are Mtrace2 client specific and outside the scope of this document.

Note that when a router’s control plane and forwarding plane are out of sync, the Mtrace2 Requests might be forwarded based on the control states instead. In this case, the traced path might not represent the real path the data packets would follow.

Mtrace2 supports both IPv4 and IPv6. Unlike the previous version of Mtrace, which implements its query and response as Internet Group Management Protocol (IGMP) messages [8], all Mtrace2 messages are UDP-based. Although the packet formats of IPv4 and IPv6 Mtrace2 are different because of the address families, the syntax between them is similar.

This document describes the base specification of Mtrace2 that can serve as a basis for future proposals such as Mtrace2 for Automatic Multicast Tunneling (AMT) [9] and Mtrace2 for Multicast in MPLS/BGP IP VPNs (MVPN) [10]. They are therefore out of the scope of this document.

2. Terminology

In this document, the key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" are to be interpreted as described in RFC 2119 [1], and indicate requirement levels for compliant Mtrace2 implementations.

2.1. Definitions

Since Mtrace2 Queries and Requests flow in the opposite direction to the data flow, we refer to "upstream" and "downstream" with respect to data, unless explicitly specified.

Incoming interface
The interface on which data is expected to arrive from the specified source and group.

Outgoing interface
This is one of the interfaces to which data from the source or RP is expected to be transmitted for the specified source and group. It is also the interface on which the Mtrace2 Request was received.

Upstream router
The router, connecting to the Incoming interface of the current router, which is responsible for forwarding data for the specified source and group to the current router.

First-hop router (FHR)
The router that is directly connected to the source the Mtrace2 Query specifies.

Last-hop router (LHR)
A router that is directly connected to a receiver. It is also the router that receives the Mtrace2 Query from an Mtrace2 client.

Group state
The state a shared-tree protocol, such as PIM-SM [5], uses to choose the upstream router towards the RP for the specified group. In this state, source-specific state is not available for the corresponding group address on the router.

Source-specific state
The state that is used to choose the path towards the source for the specified source and group.

ALL-[protocol]-ROUTERS group
Link-local multicast address for multicast routers to communicate with their adjacent routers that are running the same routing protocol. For instance, the IPv4 'ALL-PIM-ROUTERS' group is '224.0.0.13', and the IPv6 'ALL-PIM-ROUTERS' group is 'ff02::d' [5].

3. Packet Formats

This section describes the details of the packet formats for Mtrace2 messages.

All Mtrace2 messages are encoded in the Type/Length/Value (TLV) format (see Section 3.1). The first TLV of a message is a message header TLV specifying the type of message and additional context information required for processing of the message and for parsing of subsequent TLVs in the message. Subsequent TLVs in a message, referred to as Blocks, are appended after the header TLV to provide additional information associated with the message. If an implementation receives an unknown TLV type for any TLV in a message, it SHOULD ignore and silently discard the entire packet. If the length of a TLV exceeds the available space in the containing packet, the implementation MUST ignore and silently discard the TLV and any remaining portion of the containing packet.

All Mtrace2 messages are UDP packets. For IPv4, Mtrace2 Query/Request/Reply messages MUST NOT be fragmented. Therefore, Mtrace2 clients and LHRs/RPs MUST set the IP header do-not-fragment (DF) bit for all Mtrace2 messages. For IPv6, the packet size for the Mtrace2 messages MUST NOT exceed 1280 bytes, which is the smallest Maximum Transmission Unit (MTU) for an IPv6 interface [2]. The source port is uniquely selected by the local host operating system. The destination port is the IANA reserved Mtrace2 port number (see Section 8). All Mtrace2 messages MUST have a valid UDP checksum.

Additionally, Mtrace2 supports both IPv4 and IPv6, but not mixed. For example, if an Mtrace2 Query or Request message arrives in as an IPv4 packet, all addresses specified in the Mtrace2 messages MUST be IPv4 as well. Same rule applies to IPv6 Mtrace2 messages.

3.1. Mtrace2 TLV format

```
    0                   1                   2                   3
    0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |     Type      |           Length              |   Value ....  |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Type: 8 bits

Describes the format of the Value field. For all the available types, please see Section 3.2

Length: 16 bits

Length of Type, Length, and Value fields in octets. Minimum length required is 4 octets. The length MUST be a multiple of 4 octets. The maximum TLV length is not defined; however the entire Mtrace2 packet length MUST NOT exceed the available MTU.

Value: variable length

The format is based on the Type value. The length of the value field is Length field minus 3. All reserved fields in the Value field MUST be transmitted as zeros and ignored on receipt.

3.2. Defined TLVs

The following TLV Types are defined:

Code	Type
0x00	Reserved
0x01	Mtrace2 Query
0x02	Mtrace2 Request
0x03	Mtrace2 Reply
0x04	Mtrace2 Standard Response Block
0x05	Mtrace2 Augmented Response Block
0x06	Mtrace2 Extended Query Block

Each Mtrace2 message MUST begin with either a Query, Request or Reply TLV. The first TLV determines the type of each Mtrace2 message. Following a Query TLV, there can be a sequence of optional Extended Query Blocks. In the case of a Request or a Reply TLV, it is then followed by a sequence of Standard Response Blocks, each from a multicast router on the path towards the source or the RP. In the case more information is needed, a Standard Response Block can be followed by one or multiple Augmented Response Blocks.

We will describe each message type in detail in the next few sections.

3.2.1. Mtrace2 Query

An Mtrace2 Query is originated by an Mtrace2 client which sends an Mtrace2 Query message to the LHR. The LHR modifies only the Type field of the Query TLV (to turn it into a "Request") before appending a Standard Response Block and forwarding it upstream. The LHR and intermediate routers handling the Mtrace2 message when tracing upstream MUST NOT modify any other fields within the Query/Request TLV. Additionally, intermediate routers handling the message after the LHR has converted the Query into a Request MUST NOT modify the type field of the Request TLV. If the actual number of hops is not known, an Mtrace2 client could send an initial Query message with a large # Hops (e.g., 0xff), in order to try to trace the full path.

An Mtrace2 Query message is shown as follows:

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |     Type      |           Length              |    # Hops     |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |                                                               |
     |                      Multicast Address                        |
     |                                                               |
     +=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
     |                                                               |
     |                        Source Address                         |
     |                                                               |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |                                                               |
     |                    Mtrace2 Client Address                     |
     |                                                               |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |            Query ID           |         Client Port #         |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2

Length: 16 bits

The length field MUST be either 20 (i.e., 8 plus 3 * 4 (IPv4 addresses)) or 56 (i.e., 8 + 3 * 16 (IPv6 addresses)); if the length is 20, then IPv4 addresses MUST be assumed and if the length is 56, then IPv6 addresses MUST be assumed.

# Hops: 8 bits

This field specifies the maximum number of hops that the Mtrace2 client wants to trace. If there are some error conditions in the middle of the path that prevent an Mtrace2 Reply from being received by the client, the client MAY issue another Mtrace2 Query with a lower number of hops until it receives a Reply.

Multicast Address: 32 bits or 128 bits

This field specifies an IPv4 or IPv6 address, which can be either:

m-1: a multicast group address to be traced; or,

m-2: all 1’s in case of IPv4 or the unspecified address (::) in case of IPv6 if no group-specific information is desired.

Source Address: 32 bits or 128 bits

This field specifies an IPv4 or IPv6 address, which can be either:

s-1: a unicast address of the source to be traced; or, s-2: all 1’s in case of IPv4 or the unspecified address (::) in case of IPv6 if no source-specific information is desired. For example, the client is tracing a (*,g) group state.

Note that it is invalid to have a source-group combination of (s-2, m-2). If a router receives such combination in an Mtrace2 Query, it MUST silently discard the Query.

Mtrace2 Client Address: 32 bits or 128 bits
This field specifies the Mtrace2 client’s IPv4 address or IPv6 global address. This address MUST be a valid unicast address, and therefore, MUST NOT be all 1’s or an unspecified address. The Mtrace2 Reply will be sent to this address.

Query ID: 16 bits
This field is used as a unique identifier for this Mtrace2 Query so that duplicate or delayed Reply messages may be detected.

Client Port #: 16 bits
This field specifies the destination UDP port number for receiving the Mtrace2 Reply packet.

3.2.2. Mtrace2 Request

The Mtrace2 Request TLV is exactly the same as an Mtrace2 Query except for identifying the Type field of 0x02.

When a LHR receives an Mtrace2 Query message, it turns the Query into a Request by changing the Type field of the Query from 0x01 to 0x02. The LHR then appends an Mtrace2 Standard Response Block (see Section 3.2.4) of its own to the Request message before sending it upstream. The upstream routers do the same without changing the Type field until one of them is ready to send a Reply.

3.2.3. Mtrace2 Reply

The Mtrace2 Reply TLV is exactly the same as an Mtrace2 Query except for identifying the Type field of 0x03.

When a FHR or an RP receives an Mtrace2 Request message which is destined to itself, it appends an Mtrace2 Standard Response Block (see Section 3.2.4) of its own to the Request message. Next, it turns the Request message into a Reply by changing the Type field of the Request from 0x02 to 0x03 and by changing the UDP destination port to the port number specified in the Client Port number field in the Request. It then unicasts the Reply message to the Mtrace2 client specified in the Mtrace2 Client Address field.

There are a number of cases in which an intermediate router might return a Reply before a Request reaches the FHR or the RP. See Section 4.1.1, Section 4.2.2, Section 4.3.3, and Section 4.5 for more details.

3.2.4. IPv4 Mtrace2 Standard Response Block

This section describes the message format of an IPv4 Mtrace2 Standard Response Block. The Type field is 0x04.

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |     Type      |            Length             |      MBZ      |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |                      Query Arrival Time                       |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |                  Incoming Interface Address                   |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |                  Outgoing Interface Address                   |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |                   Upstream Router Address                     |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |                                                               |
     .           Input packet count on incoming interface            .
     |                                                               |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |                                                               |
     .           Output packet count on outgoing interface           .
     |                                                               |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |                                                               |
     .      Total number of packets for this source-group pair       .
     |                                                               |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |          Rtg Protocol         |    Multicast Rtg Protocol     |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |    Fwd TTL    |      MBZ      |S|   Src Mask  |Forwarding Code|
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

MBZ: 8 bits
   This field MUST be zeroed on transmission and ignored on reception.

Query Arrival Time: 32 bits
   The Query Arrival Time is a 32-bit Network Time Protocol (NTP) timestamp specifying the arrival time of the Mtrace2 Query or Request packet at this router. The 32-bit form of an NTP

timestamp consists of the middle 32 bits of the full 64-bit form; that is, the low 16 bits of the integer part and the high 16 bits of the fractional part.

The following formula converts from a timespec (fractional part in nanoseconds) to a 32-bit NTP timestamp:

```
query_arrival_time
= ((tv.tv_sec + 32384) << 16) + ((tv.tv_nsec << 7) / 1953125)
```

The constant 32384 is the number of seconds from Jan 1, 1900 to Jan 1, 1970 truncated to 16 bits. ((tv.tv_nsec << 7) / 1953125) is a reduction of ((tv.tv_nsec / 1000000000) << 16).

Note that synchronized clocks are required on the traced routers to estimate propagation and queueing delays between successive hops. Nevertheless, even without this synchronization, an application can still estimate an upper bound on cumulative one way latency by measuring the time between sending a Query and receiving a Reply.

Additionally, Query Arrival Time is useful for measuring the packet rate. For example, suppose that a client issues two queries, and the corresponding requests R1 and R2 arrive at router X at time T1 and T2, then the client would be able to compute the packet rate on router X by using the packet count information stored in the R1 and R2, and the time T1 and T2.

Incoming Interface Address: 32 bits
This field specifies the address of the interface on which packets from the source or the RP are expected to arrive, or 0 if unknown or unnumbered.

Outgoing Interface Address: 32 bits
This field specifies the address of the interface on which packets from the source or the RP are expected to transmit towards the receiver, or 0 if unknown or unnumbered. This is also the address of the interface on which the Mtrace2 Query or Request arrives.

Upstream Router Address: 32 bits
This field specifies the address of the upstream router from which this router expects packets from this source. This MAY be a multicast group (e.g., ALL-[protocol]-ROUTERS group) if the upstream router is not known because of the workings of the multicast routing protocol. However, it MUST be 0 if the incoming interface address is unknown or unnumbered.

Input packet count on incoming interface: 64 bits

This field contains the number of multicast packets received for all groups and sources on the incoming interface, or all 1’s if no count can be reported. This counter may have the same value as ifHCInMulticastPkts from the Interfaces Group MIB (IF-MIB) [12] for this interface.

Output packet count on outgoing interface: 64 bit
This field contains the number of multicast packets that have been transmitted or queued for transmission for all groups and sources on the outgoing interface, or all 1’s if no count can be reported. This counter may have the same value as ifHCOutMulticastPkts from the IF-MIB [12] for this interface.

Total number of packets for this source-group pair: 64 bits
This field counts the number of packets from the specified source forwarded by the router to the specified group, or all 1’s if no count can be reported. If the S bit is set (see below), the count is for the source network, as specified by the Src Mask field (see below). If the S bit is set and the Src Mask field is 127, indicating no source-specific state, the count is for all sources sending to this group. This counter should have the same value as ipMcastRoutePkts from the IP Multicast MIB [13] for this forwarding entry.

Rtg Protocol: 16 bits
This field describes the unicast routing protocol running between this router and the upstream router, and it is used to determine the RPF interface for the specified source or RP. This value should have the same value as ipMcastRouteRtProtocol from the IP Multicast MIB [13] for this entry. If the router is not able to obtain this value, all 0’s must be specified.

Multicast Rtg Protocol: 16 bits
This field describes the multicast routing protocol in use between the router and the upstream router. This value should have the same value as ipMcastRouteProtocol from the IP Multicast MIB [13] for this entry. If the router cannot obtain this value, all 0’s must be specified.

Fwd TTL: 8 bits
This field contains the configured multicast TTL threshold, if any, of the outgoing interface.

S: 1 bit
If this bit is set, it indicates that the packet count for the source-group pair is for the source network, as determined by masking the source address with the Src Mask field.

Src Mask: 7 bits

This field contains the number of 1's in the netmask the router has for the source (i.e. a value of 24 means the netmask is 0xffffff00). If the router is forwarding solely on group state, this field is set to 127 (0x7f).

Forwarding Code: 8 bits

This field contains a forwarding information/error code. Values with the high order bit set (0x80-0xff) are intended for use with conditions that are transitory or automatically recovered. Other forwarding code values indicate a need to fix a problem in the Query or a need to redirect the Query. Section 4.1 and Section 4.2 explain how and when the Forwarding Code is filled. Defined values are as follows:

Value	Name	Description
0x00	NO_ERROR	No error
0x01	WRONG_IF	Mtrace2 Request arrived on an interface to which this router would not forward for the specified group towards the source or RP.
0x02	PRUNE_SENT	This router has sent a prune upstream which applies to the source and group in the Mtrace2 Request.
0x03	PRUNE_RCVD	This router has stopped forwarding for this source and group in response to a request from the downstream router.
0x04	SCOPED	The group is subject to administrative scoping at this router.
0x05	NO_ROUTE	This router has no route for the source or group and no way to determine a potential route.
0x06	WRONG_LAST_HOP	This router is not the proper LHR.
0x07	NOT_FORWARDING	This router is not forwarding this source and group out the outgoing interface for an unspecified reason.
0x08	REACHED_RP	Reached the Rendezvous Point.
0x09	RPF_IF	Mtrace2 Request arrived on the expected RPF interface for this source and group.
0x0A	NO_MULTICAST	Mtrace2 Request arrived on an interface which is not enabled for multicast.
0x0B	INFO_HIDDEN	One or more hops have been hidden from this trace.
0x0C	REACHED_GW	Mtrace2 Request arrived on a gateway (e.g., a NAT or firewall) that hides the information between this router and the Mtrace2 client.
0x0D	UNKNOWN_QUERY	A non-transitive Extended Query Type was received by a router which does not support the type.
0x80	FATAL_ERROR	A fatal error is one where the router may know the upstream router but cannot forward the message to it.
0x81	NO_SPACE	There was not enough room to insert another Standard Response Block in the packet.
0x83	ADMIN_PROHIB	Mtrace2 is administratively prohibited.

3.2.5. IPv6 Mtrace2 Standard Response Block

This section describes the message format of an IPv6 Mtrace2 Standard Response Block. The Type field is also 0x04.

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |     Type      |            Length             |      MBZ      |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                      Query Arrival Time                       |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                     Incoming Interface ID                     |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                     Outgoing Interface ID                     |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                                                               |
    *                         Local Address                         *
    |                                                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                                                               |
    *                         Remote Address                        *
    |                                                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                                                               |
    .           Input packet count on incoming interface            .
    |                                                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                                                               |
    .           Output packet count on outgoing interface           .
    |                                                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |                                                               |
    .      Total number of packets for this source-group pair       .
    |                                                               |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |          Rtg Protocol         |    Multicast Rtg Protocol     |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    |              MBZ 2          |S|Src Prefix Len |Forwarding Code|
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

MBZ: 8 bits
This field MUST be zeroed on transmission and ignored on reception.

Query Arrival Time: 32 bits
Same definition as in IPv4.

Incoming Interface ID: 32 bits
This field specifies the interface ID on which packets from the source or RP are expected to arrive, or 0 if unknown. This ID should be the value taken from InterfaceIndex of the IF-MIB [12] for this interface.

Outgoing Interface ID: 32 bits

This field specifies the interface ID to which packets from the source or RP are expected to transmit, or 0 if unknown. This ID should be the value taken from InterfaceIndex of the IF-MIB [12] for this interface

Local Address: 128 bits

This field specifies a global IPv6 address that uniquely identifies the router. A unique local unicast address [11] SHOULD NOT be used unless the router is only assigned link-local and unique local addresses. If the router is only assigned link-local addresses, its link-local address can be specified in this field.

Remote Address: 128 bits

This field specifies the address of the upstream router, which, in most cases, is a link-local unicast address for the upstream router.

Although a link-local address does not have enough information to identify a node, it is possible to detect the upstream router with the assistance of Incoming Interface ID and the current router address (i.e., Local Address).

Note that this may be a multicast group (e.g., ALL-[protocol]-ROUTERS group) if the upstream router is not known because of the workings of a multicast routing protocol. However, it should be the unspecified address (::) if the incoming interface address is unknown.

Input packet count on incoming interface: 64 bits

Same definition as in IPv4.

Output packet count on outgoing interface: 64 bits

Same definition as in IPv4.

Total number of packets for this source-group pair: 64 bits

Same definition as in IPv4, except if the S bit is set (see below), the count is for the source network, as specified by the Src Prefix Len field. If the S bit is set and the Src Prefix Len field is 255, indicating no source-specific state, the count is for all sources sending to this group. This counter should have the same value as ipMcastRoutePkts from the IP Multicast MIB [13] for this forwarding entry.

Rtg Protocol: 16 bits

Same definition as in IPv4.

Multicast Rtg Protocol: 16 bits

Same definition as in IPv4.

MBZ 2: 15 bits
This field MUST be zeroed on transmission and ignored on reception.

S: 1 bit
Same definition as in IPv4, except the Src Prefix Len field is used to mask the source address.

Src Prefix Len: 8 bits
This field contains the prefix length this router has for the source. If the router is forwarding solely on group state, this field is set to 255 (0xff).

Forwarding Code: 8 bits
Same definition as in IPv4.

3.2.6. Mtrace2 Augmented Response Block

In addition to the Standard Response Block, a multicast router on the traced path can optionally add one or multiple Augmented Response Blocks before sending the Request to its upstream router.

The Augmented Response Block is flexible for various purposes such as providing diagnosis information (see Section 7) and protocol verification. Its Type field is 0x05, and its format is as follows:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |           Length              |      MBZ      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Augmented Response Type    |           Value ....          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

MBZ: 8 bits
This field MUST be zeroed on transmission and ignored on reception.

Augmented Response Type: 16 bits
This field specifies the type of various responses from a multicast router that might need to communicate back to the Mtrace2 client as well as the multicast routers on the traced path.

The Augmented Response Type is defined as follows:

Code	Type
0x0001	# of the returned Standard Response Blocks

When the NO_SPACE error occurs on a router, the router should send the original Mtrace2 Request received from the downstream router as a Reply back to the Mtrace2 client and continue with a new Mtrace2 Request. In the new Request, the router adds a Standard Response Block followed by an Augmented Response Block with 0x01 as the Augmented Response Type, and the number of the returned Mtrace2 Standard Response Blocks as the Value.

Each upstream router recognizes the total number of hops the Request has been traced so far by adding this number and the number of the Standard Response Block in the current Request message.

This document only defines one Augmented Response Type in the Augmented Response Block. The description on how to provide diagnosis information using the Augmented Response Block is out of the scope of this document, and will be addressed in separate documents.

Value: variable length

The format is based on the Augmented Response Type value. The length of the value field is Length field minus 6.

3.2.7. Mtrace2 Extended Query Block

There may be a sequence of optional Extended Query Blocks that follow an Mtrace2 Query to further specify any information needed for the Query. For example, an Mtrace2 client might be interested in tracing the path the specified source and group would take based on a certain topology. In this case, the client can pass in the multi-topology ID as the Value for an Extended Query Type (see below). The Extended Query Type is extensible and the behavior of the new types will be addressed by separate documents.

The Mtrace2 Extended Query Block’s Type field is 0x06, and is formatted as follows:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |           Length              |      MBZ    |T|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Extended Query Type      |           Value ....          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

MBZ: 7 bits
This field MUST be zeroed on transmission and ignored on reception.

T-bit (Transitive Attribute): 1 bit
If the TLV type is unrecognized by the receiving router, then this TLV is either discarded or forwarded along with the Query, depending on the value of this bit. If this bit is set, then the router MUST forward this TLV. If this bit is clear, the router MUST send an Mtrace2 Reply with an UNKNOWN_QUERY error.

Extended Query Type: 16 bits
This field specifies the type of the Extended Query Block.

Value: 16 bits
This field specifies the value of this Extended Query.

4. Router Behavior

This section describes the router behavior in the context of Mtrace2 in detail.

4.1. Receiving Mtrace2 Query

An Mtrace2 Query message is an Mtrace2 message with no response blocks filled in, and uses TLV type of 0x01.

4.1.1. Query Packet Verification

Upon receiving an Mtrace2 Query message, a router MUST examine whether the Multicast Address and the Source Address are a valid combination as specified in Section 3.2.1, and whether the Mtrace2 Client Address is a valid IP unicast address. If either one is invalid, the Query MUST be silently ignored.

Mtrace2 supports a non-local client to the LHR/RP. A router MUST, however, support a mechanism to drop Queries from clients beyond a specified administrative boundary. The potential approaches are described in Section 9.2.

In the case where a local LHR client is required, the router must then examine the Query to see if it is the proper LHR/RP for the destination address in the packet. It is the proper local LHR if it has a multicast-capable interface on the same subnet as the Mtrace2 Client Address and is the router that would forward traffic from the given (S,G) or (*,G) onto that subnet. It is the proper RP if the multicast group address specified in the query is 0 and if the IP header destination address is a valid RP address on this router.

If the router determines that it is not the proper LHR/RP, or it cannot make that determination, it does one of two things depending on whether the Query was received via multicast or unicast. If the Query was received via multicast, then it MUST be silently discarded. If it was received via unicast, the router turns the Query into a Reply message by changing the TLV type to 0x03 and appending a Standard Response Block with a Forwarding Code of WRONG_LAST_HOP. The rest of the fields in the Standard Response Block MUST be zeroed. The router then sends the Reply message to the Mtrace2 Client Address on the Client Port # as specified in the Mtrace2 Query.

Duplicate Query messages as identified by the tuple (Mtrace2 Client Address, Query ID) SHOULD be ignored. This MAY be implemented using a cache of previously processed queries keyed by the Mtrace2 Client Address and Query ID pair. The duration of the cached entries is implementation specific. Duplicate Request messages MUST NOT be ignored in this manner.

4.1.2. Query Normal Processing

When a router receives an Mtrace2 Query and it determines that it is the proper LHR/RP, it turns the Query to a Request by changing the TLV type from 0x01 to 0x02, and performs the steps listed in Section 4.2.

4.2. Receiving Mtrace2 Request

An Mtrace2 Request is an Mtrace2 message that uses TLV type of 0x02. With the exception of the LHR, whose Request was just converted from a Query, each Request received by a router should have at least one Standard Response Block filled in.

4.2.1. Request Packet Verification

If the Mtrace2 Request does not come from an adjacent router, or if the Request is not addressed to this router, or if the Request is addressed to a multicast group which is not a link-scoped group (i.e., 224.0.0.0/24 for IPv4, FFx2::/16 [3] for IPv6), it MUST be silently ignored. The Generalized TTL Security Mechanism (GTSM) [14] SHOULD be used by the router to determine whether the router is adjacent or not. Source verification specified in Section 9.2 is also considered.

If the sum of the number of the Standard Response Blocks in the received Mtrace2 Request and the value of the Augmented Response Type of 0x01, if any, is equal or more than the # Hops in the Mtrace2 Request, it MUST be silently ignored.

4.2.2. Request Normal Processing

When a router receives an Mtrace2 Request message, it performs the following steps. Note that it is possible to have multiple situations covered by the Forwarding Codes. The first one encountered is the one that is reported, i.e. all "note Forwarding Code N" should be interpreted as "if Forwarding Code is not already set, set Forwarding Code to N". Note that in the steps described below the "Outgoing Interface" is the one on which the Mtrace2 Request message arrives.

1. Prepare a Standard Response Block to be appended to the packet, setting all fields to an initial default value of zero.

2. If Mtrace2 is administratively prohibited, note the Forwarding Code of ADMIN_PROHIB and skip to step 4.

3. In the Standard Response Block, fill in the Query Arrival Time, Outgoing Interface Address (for IPv4) or Outgoing Interface ID (for IPv6), Output Packet Count, and Fwd TTL (for IPv4).

4. Attempt to determine the forwarding information for the specified source and group, using the same mechanisms as would be used when a packet is received from the source destined for the group. A state need not be instantiated, it can be a "phantom" state created only for the purpose of the trace, such as "dry-run."

   If using a shared-tree protocol and there is no source-specific state, or if no source-specific information is desired (i.e., all 1’s for IPv4 or unspecified address (::) for IPv6), group state should be used. If there is no group state or no group-specific information is desired, potential source state (i.e., the path that would be followed for a source-specific Join) should be used.

5. If no forwarding information can be determined, the router notes a Forwarding Code of NO_ROUTE, sets the remaining fields that have not yet been filled in to zero, and then sends an Mtrace2 Reply back to the Mtrace2 client.

6. If a Forwarding Code of ADMIN_PROHIB has been set, skip to step 7. Otherwise, fill in the Incoming Interface Address (or Incoming Interface ID and Local Address for IPv6), Upstream Router Address (or Remote Address for IPv6), Input Packet Count, Total Number of Packets, Routing Protocol, S, and Src Mask (or Src Prefix Len for IPv6) using the forwarding information determined in step 4.

7. If the Outgoing interface is not enabled for multicast, note Forwarding Code of NO_MULTICAST. If the Outgoing interface is the interface from which the router would expect data to arrive from the source, note forwarding code RPF_IF. If the Outgoing interface is not one to which the router would forward data from the source or RP to the group, a Forwarding code of WRONG_IF is noted. In the above three cases, the router will return an Mtrace2 Reply and terminate the trace.

8. If the group is subject to administrative scoping on either the Outgoing or Incoming interfaces, a Forwarding Code of SCOPED is noted.

9. If this router is the RP for the group for a non-source-specific query, note a Forwarding Code of REACHED_RP. The router will send an Mtrace2 Reply and terminate the trace.

10. If this router is directly connected to the specified source or source network on the Incoming interface, it sets the Upstream Router Address (for IPv4) or the Remote Address (for IPv6) of the response block to zero. The router will send an Mtrace2 Reply and terminate the trace.

11. If this router has sent a prune upstream which applies to the source and group in the Mtrace2 Request, it notes a Forwarding Code of PRUNE_SENT. If the router has stopped forwarding downstream in response to a prune sent by the downstream router, it notes a Forwarding Code of PRUNE_RCVD. If the router should normally forward traffic downstream for this source and group but is not, it notes a Forwarding Code of NOT_FORWARDING.

12. If this router is a gateway (e.g., a NAT or firewall) that hides the information between this router and the Mtrace2 client, it notes a Forwarding Code of REACHED_GW. The router continues the processing as described in Section 4.5.

13. If the total number of the Standard Response Blocks, including the newly prepared one, and the value of the Augmented Response Type of 0x01, if any, is less than the # Hops in the Request, the packet is then forwarded to the upstream router as described in Section 4.3; otherwise, the packet is sent as an Mtrace2 Reply to the Mtrace2 client as described in Section 4.4.

4.3. Forwarding Mtrace2 Request

This section describes how an Mtrace2 Request should be forwarded.

4.3.1. Destination Address

If the upstream router for the Mtrace2 Request is known for this request, the Mtrace2 Request is sent to that router. If the Incoming interface is known but the upstream router is not, the Mtrace2 Request is sent to an appropriate multicast address on the Incoming interface. The multicast address SHOULD depend on the multicast routing protocol in use, such as ALL-[protocol]-ROUTERS group. It MUST be a link-scoped group (i.e., 224.0.0.0/24 for IPv4, FF02::/16 for IPv6), and MUST NOT be the all-systems multicast group (224.0.0.1) for IPv4 and All Nodes Address (FF02::1) for IPv6. It MAY also be the all-routers multicast group (224.0.0.2) for IPv4 or All Routers Address (FF02::2) for IPv6 if the routing protocol in use does not define a more appropriate multicast address.

4.3.2. Source Address

An Mtrace2 Request should be sent with the address of the Incoming interface. However, if the Incoming interface is unnumbered, the router can use one of its numbered interface addresses as the source address.

4.3.3. Appending Standard Response Block

An Mtrace2 Request MUST be sent upstream towards the source or the RP after appending a Standard Response Block to the end of the received Mtrace2 Request. The Standard Response Block includes the multicast states and statistics information of the router described in Section 3.2.4.

If appending the Standard Response Block would make the Mtrace2 Request packet longer than the MTU of the Incoming Interface, or, in the case of IPv6, longer than 1280 bytes, the router MUST change the Forwarding Code in the last Standard Response Block of the received Mtrace2 Request into NO_SPACE. The router then turns the Request into a Reply and sends the Reply as described in Section 4.4.

The router will continue with a new Request by copying from the old Request excluding all the response blocks, followed by the previously prepared Standard Response Block, and an Augmented Response Block with Augmented Response Type of 0x01 and the number of the returned Standard Response Blocks as the value. The new Request is then forwarded upstream.

4.4. Sending Mtrace2 Reply

An Mtrace2 Reply MUST be returned to the client by a router if any of the following conditions occur:

1. The total number of the traced routers are equal to the # of hops in the request (including the one just added) plus the number of the returned blocks, if any.

2. Appending the Standard Response Block would make the Mtrace2 Request packet longer than the MTU of the Incoming interface. (In case of IPv6 not more than 1280 bytes; see Section 4.3.3 for additional details on handling of this case.)

3. The request has reached the RP for a non source specific query or has reached the first hop router for a source specific query (see Section 4.2.2, items 9 and 10 for additional details).

4.4.1. Destination Address

An Mtrace2 Reply MUST be sent to the address specified in the Mtrace2 Client Address field in the Mtrace2 Request.

4.4.2. Source Address

An Mtrace2 Reply SHOULD be sent with the address of the router’s Outgoing interface. However, if the Outgoing interface address is unnumbered, the router can use one of its numbered interface addresses as the source address.

4.4.3. Appending Standard Response Block

An Mtrace2 Reply MUST be sent with the prepared Standard Response Block appended at the end of the received Mtrace2 Request except in the case of NO_SPACE forwarding code.

4.5. Proxying Mtrace2 Query

When a gateway (e.g., a NAT or firewall), which needs to block unicast packets to the Mtrace2 client, or hide information between the gateway and the Mtrace2 client, receives an Mtrace2 Query from an adjacent host or Mtrace2 Request from an adjacent router, it appends a Standard Response Block with REACHED_GW as the Forwarding Code. It turns the Query or Request into a Reply, and sends the Reply back to the client.

At the same time, the gateway originates a new Mtrace2 Query message by copying the original Mtrace2 header (the Query or Request without any of the response blocks), and makes the changes as follows:

- sets the RPF interface’s address as the Mtrace2 Client Address;

- uses its own port number as the Client Port #; and,

- decreases # Hops by ((number of the Standard Response Blocks that were just returned in a Reply) - 1). The "-1" in this expression accounts for the additional Standard Response Block appended by the gateway router.

The new Mtrace2 Query message is then sent to the upstream router or to an appropriate multicast address on the RPF interface.

When the gateway receives an Mtrace2 Reply whose Query ID matches the one in the original Mtrace2 header, it MUST relay the Mtrace2 Reply back to the Mtrace2 client by replacing the Reply’s header with the original Mtrace2 header. If the gateway does not receive the corresponding Mtrace2 Reply within the [Mtrace Reply Timeout] period (see Section 5.8.4), then it silently discards the original Mtrace2 Query or Request message, and terminates the trace.

4.6. Hiding Information

Information about a domain’s topology and connectivity may be hidden from the Mtrace2 Requests. The Forwarding Code of INFO_HIDDEN may be used to note that. For example, the incoming interface address and packet count on the ingress router of a domain, and the outgoing interface address and packet count on the egress router of the domain can be specified as all 1’s. Additionally, the source-group packet count (see Section 3.2.4 and Section 3.2.5) within the domain may be all 1’s if it is hidden.

5. Client Behavior

This section describes the behavior of an Mtrace2 client in detail.

5.1. Sending Mtrace2 Query

An Mtrace2 client initiates an Mtrace2 Query by sending the Query to the LHR of interest.

5.1.1. Destination Address

If an Mtrace2 client knows the proper LHR, it unicasts an Mtrace2 Query packet to that router; otherwise, it MAY send the Mtrace2 Query packet to the all-routers multicast group (224.0.0.2) for IPv4 or All Routers Address (FF02::2) for IPv6. This will ensure that the packet is received by the LHR on the subnet.

See also Section 5.4 on determining the LHR.

5.1.2. Source Address

An Mtrace2 Query MUST be sent with the client’s interface address, which is the Mtrace2 Client Address.

5.2. Determining the Path

An Mtrace2 client could send an initial Query messages with a large # Hops, in order to try to trace the full path. If this attempt fails, one strategy is to perform a linear search (as the traditional unicast traceroute program does); set the # Hops field to 1 and try to get a Reply, then 2, and so on. If no Reply is received at a certain hop, this hop is identified as the probable cause of forwarding failures on the path. Nevertheless, the sender may attempt to continue tracing past the non-responding hop by further increasing the hop count in the hopes that further hops may respond. Each of these attempts MUST NOT be initiated before the previous attempt has terminated either because of successful reception of a Reply or because the [Mtrace Reply Timeout] timeout has occurred.

See also Section 5.6 on receiving the results of a trace.

5.3. Collecting Statistics

After a client has determined that it has traced the whole path or as much as it can expect to (see Section 5.8), it might collect statistics by waiting a short time and performing a second trace. If the path is the same in the two traces, statistics can be displayed as described in Section 7.3 and Section 7.4.

5.4. Last Hop Router (LHR)

The Mtrace2 client may not know which is the last-hop router, or that router may be behind a firewall that blocks unicast packets but passes multicast packets. In these cases, the Mtrace2 Request should be multicasted to the all-routers multicast group (224.0.0.2) for IPv4 or All Routers Address (FF02::2) for IPv6. All routers except the correct last-hop router SHOULD ignore any Mtrace2 Request received via multicast.

5.5. First Hop Router (FHR)

The IANA assigned 224.0.1.32 as the default multicast group for old IPv4 mtrace (v1) responses, in order to support mtrace clients that are not unicast reachable from the first-hop router. Mtrace2, however, does not require any IPv4/IPv6 multicast addresses for the Mtrace2 Replies. Every Mtrace2 Reply is sent to the unicast address specified in the Mtrace2 Client Address field of the Mtrace2 Reply.

5.6. Broken Intermediate Router

A broken intermediate router might simply not understand Mtrace2 packets, and drop them. The Mtrace2 client will get no Reply at all as a result. It should then perform a hop-by-hop search by setting the # Hops field until it gets an Mtrace2 Reply. The client may use linear or binary search; however, the latter is likely to be slower because a failure requires waiting for the [Mtrace Reply Timeout] period.

5.7. Non-Supported Router

When a non-supported router receives an Mtrace2 Query or Request message whose destination address is a multicast address, the router will silently discard the message.

When the router receives an Mtrace2 Query which is destined to itself, the router returns an Internet Control Message Protocol (ICMP) port unreachable to the Mtrace2 client. On the other hand, when the router receives an Mtrace2 Request which is destined to itself, the router returns an ICMP port unreachable to its adjacent router from which the Request receives. Therefore, the Mtrace2 client needs to terminate the trace when the [Mtrace Reply Timeout] timeout has occurred, and may then issue another Query with a lower number of # Hops.

5.8. Mtrace2 Termination

When performing an expanding hop-by-hop trace, it is necessary to determine when to stop expanding.

5.8.1. Arriving at Source

A trace can be determined to have arrived at the source if the Incoming Interface of the last router in the trace is non-zero, but the Upstream Router is zero.

5.8.2. Fatal Error

A trace has encountered a fatal error if the last Forwarding Error in the trace has the 0x80 bit set.

5.8.3. No Upstream Router

A trace cannot continue if the last Upstream Router in the trace is set to 0.

5.8.4. Reply Timeout

This document defines the [Mtrace Reply Timeout] value, which is used to time out an Mtrace2 Reply as seen in Section 4.5, Section 5.2, and Section 5.7. The default [Mtrace Reply Timeout] value is 10 (seconds), and can be manually changed on the Mtrace2 client and routers.

5.9. Continuing after an Error

When the NO_SPACE error occurs, as described in Section 4.2, a router will send back an Mtrace2 Reply to the Mtrace2 client, and continue with a new Request (see Section 4.3.3). In this case, the Mtrace2 client may receive multiple Mtrace2 Replies from different routers along the path. When this happens, the client MUST treat them as a single Mtrace2 Reply message by collating the augmented response blocks of subsequent Replies sharing the same query ID, sequencing each cluster of augmented response blocks based on the order in which they are received.

If a trace times out, it is very likely that a router in the middle of the path does not support Mtrace2. That router’s address will be in the Upstream Router field of the last Standard Response Block in the last received Reply. A client may be able to determine (via mrinfo or the Simple Network Management Protocol (SNMP) [11][13]) a list of neighbors of the non-responding router. The neighbors obtained in this way could then be probed (via the multicast MIB [13]) to determine which one is the upstream neighbor (i.e., Reverse Path Forwarding (RPF) neighbor) of the non-responding router. This algorithm can identify the upstream neighbor because, even though there may be multiple neighbors, the non-responding router should only have sent a "join" to the one neighbor corresponding to its selected RPF path. Because of this, only the RPF neighbor should contain the non-responding router as a multicast next hop in its MIB output list for the affected multicast route.

6. Protocol-Specific Considerations

This section describes the Mtrace2 behavior with the presence of different multicast protocols.

6.1. PIM-SM

When an Mtrace2 reaches a PIM-SM RP, and the RP does not forward the trace on, it means that the RP has not performed a source-specific join so there is no more state to trace. However, the path that traffic would use if the RP did perform a source-specific join can be traced by setting the trace destination to the RP, the trace source to the traffic source, and the trace group to 0. This Mtrace2 Query may be unicasted to the RP, and the RP takes the same actions as an LHR.

6.2. Bi-Directional PIM

Bi-directional PIM [6] is a variant of PIM-SM that builds bi-directional shared trees connecting multicast sources and receivers. Along the bi-directional shared trees, multicast data is natively forwarded from the sources to the Rendezvous Point Link (RPL), and from which, to receivers without requiring source-specific state. In contrast to PIM-SM, Bi-directional PIM always has the state to trace.

A Designated Forwarder (DF) for a given Rendezvous Point Address (RPA) is in charge of forwarding downstream traffic onto its link, and forwarding upstream traffic from its link towards the RPL that the RPA belongs to. Hence Mtrace2 Reply reports DF addresses or RPA along the path.

6.3. PIM-DM

Routers running PIM Dense Mode [15] do not know the path packets would take unless traffic is flowing. Without some extra protocol mechanism, this means that in an environment with multiple possible paths with branch points on shared media, Mtrace2 can only trace existing paths, not potential paths. When there are multiple possible paths but the branch points are not on shared media, the upstream router is known, but the LHR may not know that it is the appropriate last hop.

When traffic is flowing, PIM Dense Mode routers know whether or not they are the LHR for the link (because they won or lost an Assert battle) and know who the upstream router is (because it won an Assert battle). Therefore, Mtrace2 is always able to follow the proper path when traffic is flowing.

6.4. IGMP/MLD Proxy

When an IGMP or Multicast Listener Discovery (MLD) Proxy [7] receives an Mtrace2 Query packet on an incoming interface, it notes a WRONG_IF in the Forwarding Code of the last Standard Response Block (see Section 3.2.4), and sends the Mtrace2 Reply back to the Mtrace2 client. On the other hand, when an Mtrace2 Query packet reaches an outgoing interface of the IGMP/MLD proxy, it is forwarded onto its incoming interface towards the upstream router.

7. Problem Diagnosis

This section describes different scenarios Mtrace2 can be used to diagnose the multicast problems.

7.1. Forwarding Inconsistencies

The Forwarding Error code can tell if a group is unexpectedly pruned or administratively scoped.

7.2. TTL or Hop Limit Problems

By taking the maximum of hops from the source and forwarding TTL threshold over all hops, it is possible to discover the TTL or hop limit required for the source to reach the destination.

7.3. Packet Loss

By taking multiple traces, it is possible to find packet loss information by tracking the difference between the output packet count for the specified source-group address pair at a given upstream router and the input packet count on the next hop downstream router. On a point-to-point link, any steadily increasing difference in these counts implies packet loss. Although the packet counts will differ due to Mtrace2 Request propagation delay, the difference should remain essentially constant (except for jitter caused by differences in propagation time among the trace iterations). However, this difference will display a steady increase if packet loss is occurring. On a shared link, the count of input packets can be larger than the number of output packets at the previous hop, due to other routers or hosts on the link injecting packets. This appears as "negative loss" which may mask real packet loss.

In addition to the counts of input and output packets for all multicast traffic on the interfaces, the Standard Response Block includes a count of the packets forwarded by a node for the specified source-group pair. Taking the difference in this count between two traces and then comparing those differences between two hops gives a measure of packet loss just for traffic from the specified source to the specified receiver via the specified group. This measure is not affected by shared links.

On a point-to-point link that is a multicast tunnel, packet loss is usually due to congestion in unicast routers along the path of that tunnel. On native multicast links, loss is more likely in the output queue of one hop, perhaps due to priority dropping, or in the input queue at the next hop. The counters in the Standard Response Block do not allow these cases to be distinguished. Differences in packet counts between the incoming and outgoing interfaces on one node cannot generally be used to measure queue overflow in the node.

7.4. Link Utilization

Again, with two traces, you can divide the difference in the input or output packet counts at some hop by the difference in time stamps from the same hop to obtain the packet rate over the link. If the average packet size is known, then the link utilization can also be estimated to see whether packet loss may be due to the rate limit or the physical capacity on a particular link being exceeded.

7.5. Time Delay

If the routers have synchronized clocks, it is possible to estimate propagation and queuing delay from the differences between the timestamps at successive hops. However, this delay includes control processing overhead, so is not necessarily indicative of the delay that data traffic would experience.

8. IANA Considerations

The following new registries are to be created and maintained under the "Specification Required" registry policy as specified in [4].

8.1. "Mtrace2 Forwarding Codes" Registry

This is an integer in the range 0-255. Assignment of a Forwarding Code requires specification of a value and a name for the Forwarding Code. Initial values for the forwarding codes are given in the table at the end of Section 3.2.4. Additional values (specific to IPv6) may also be specified at the end of Section 3.2.5. Any additions to this registry are required to fully describe the conditions under which the new Forwarding Code is used.

8.2. "Mtrace2 TLV Types" Registry

Assignment of a TLV Type requires specification of an integer value "Code" in the range 0-255 and a name ("Type"). Initial values for the TLV Types are given in the table at the beginning of Section 3.2.

8.3. UDP Destination Port

IANA has assigned UDP user port 33435 (mtrace) for use by this protocol as the Mtrace2 UDP destination port.

9. Security Considerations

This section addresses some of the security considerations related to Mtrace2.

9.1. Addresses in Mtrace2 Header

An Mtrace2 header includes three addresses, source address, multicast address, and Mtrace2 client address. These addresses MUST be congruent with the definition defined in Section 3.2.1 and forwarding Mtrace2 messages having invalid addresses MUST be prohibited. For instance, if Mtrace2 Client Address specified in an Mtrace2 header is a multicast address, then a router that receives the Mtrace2 message MUST silently discard it.

9.2. Verification of Clients and Peers

A router providing Mtrace2 functionality MUST support a source verification mechanism to drop Queries from clients and Requests from peer router or client addresses that are unauthorized or that are beyond a specified administrative boundary. This verification could, for example, be specified via a list of allowed/disallowed client and peer addresses or subnets for a given Mtrace2 message type sent to the Mtrace2 protocol port. If a Query or Request is received from an unauthorized address or one beyond the specified administrative boundary, the Query/Request MUST NOT be processed. The router MAY, however, perform rate limited logging of such events.

The required use of source verification on the participating routers minimizes the possible methods for introduction of spoofed Query/ Request packets that would otherwise enable DoS amplification attacks targeting an authorized "query" host. The source verification mechanisms provide this protection by allowing Query messages from an authorized host address to be received only by the router(s) connected to that host, and only on the interface to which that host is attached. For protection against spoofed Request messages, the source verification mechanisms allow Request messages only from a directly connected routing peer and allow these messages to be received only on the interface to which that peer is attached.

Note that the following vulnerabilities cannot be covered by the source verification methods described here. These methods can, nevertheless, prevent attacks launched from outside the boundaries of a given network as well as from any hosts within the network that are not on the same LAN as an intended authorized query client.

- A server/router "B" other than the server/router "A" that actually "owns" a given IP address could, if it is connected to the same LAN, send an Mtrace2 Query or Request with the source address set to the address for server/router "A". This is not a significant threat, however, if only trusted servers and routers are connected to that LAN.

- A malicious application running on a trusted server or router could send packets that might cause an amplification problem. It is beyond the scope of this document to protect against a DoS attack launched from the same host that is the target of the attack or from another "on path" host, but this is not a likely threat scenario. In addition, routers on the path MAY rate-limit the packets as specified in Section 9.5 and Section 9.6.

9.3. Topology Discovery

Mtrace2 can be used to discover any actively-used topology. If your network topology is a secret, Mtrace2 may be restricted at the border of your domain, using the ADMIN_PROHIB forwarding code.

9.4. Characteristics of Multicast Channel

Mtrace2 can be used to discover what sources are sending to what groups and at what rates. If this information is a secret, Mtrace2 may be restricted at the border of your domain, using the ADMIN_PROHIB forwarding code.

9.5. Limiting Query/Request Rates

A router may limit Mtrace2 Queries and Requests by ignoring some of the consecutive messages. The router MAY randomly ignore the received messages to minimize the processing overhead, i.e., to keep fairness in processing queries, or prevent traffic amplification. The rate limit is left to the router’s implementation.

9.6. Limiting Reply Rates

The proxying and NO_SPACE behaviors may result in one Query returning multiple Reply messages. In order to prevent abuse, the routers in the traced path MAY need to rate-limit the Replies. The rate limit function is left to the router’s implementation.

9.7. Specific Security Concerns

9.7.1. Request and Response Bombardment

A malicious sender could generate invalid and undesirable Mtrace2 traffic to hosts and/or routers on a network by eliciting responses to spoofed or multicast client addresses. This could be done via forged or multicast client/source addresses in Mtrace2 Query or Request messages. The recommended protections against this type of attack are described in Section 9.1, Section 9.2, Section 9.5, and Section 9.6.

9.7.2. Amplification Attack

Because an Mtrace2 Query results in Mtrace2 Request and Mtrace2 Reply messages that are larger than the original message, the potential exists for an amplification attack from a malicious sender. This threat is minimized by restricting the set of addresses from which Mtrace2 messages can be received on a given router as specified in Section 9.2.

In addition, for a router running a PIM protocol (PIM-SM, PIM-DM, PIM Source-Specific Multicast, or Bi-Directional PIM), the router SHOULD drop any Mtrace2 Request or Reply message that is received from an IP address that does not correspond to an authenticated PIM neighbor on the interface from which the packet is received. The intent of this text is to prevent non-router endpoints from injecting Request messages. Implementations of non-PIM protocols SHOULD employ some other mechanism to prevent this attack.

9.7.3. Leaking of Confidential Topology Details

Mtrace2 Queries are a potential mechanism for obtaining confidential topology information for a targeted network. Section 9.2 and Section 9.4 describe required and optional methods for ensuring that information delivered with Mtrace2 messages is not disseminated to unauthorized hosts.

9.7.4. Delivery of False Information (Forged Reply Messages)

Forged Reply messages could potentially provide a host with invalid or incorrect topology information. They could also provide invalid or incorrect information regarding multicast traffic statistics, multicast stream propagation delay between hops, multicast and unicast protocols in use between hops and other information used for analyzing multicast traffic patterns and for troubleshooting multicast traffic problems. This threat is mitigated by the following factors:

- The required source verification of permissible source addresses specified in Section 9.2 eliminates the origination of forged Replies from addresses that have not been authorized to send Mtrace2 messages to routers on a given network. This mechanism can block forged Reply messages sent from any "off path" source.

- To forge a Reply, the sender would need to somehow know (or guess) the associated two byte Query ID for an extant Query and the dynamically allocated source port number. Because "off path" sources can be blocked by a source verification mechanism, the scope of this threat is limited to "on path" attackers.

- The required use of source verification (Section 9.2) and recommended use of PIM neighbor authentication (Section 9.7.2) for messages that are only valid when sent by a multicast routing peer (Request and Reply messages) eliminate the possibility of reception of a forged Reply from an authorized host address that does not belong to a multicast peer router.

- The use of encryption between the source of a Query and the endpoint of the trace would provide a method to protect the values of the Query ID and the dynamically allocated client (source) port (see Section 3.2.1). These are the values needed to create a forged Reply message that would pass validity checks at the querying client. This type of cryptographic protection is not practical, however, because the primary reason for executing an Mtrace2 is that the destination endpoint (and path to that endpoint) are not known by the querying client. While it is not practical to provide cryptographic protection between a client and the Mtrace2 endpoints (destinations), it may be possible to prevent forged responses from "off path" nodes attached to any Mtrace2 transit LAN by devising a scheme to encrypt the critical portions of an Mtrace2 message between each valid sender/receiver pair at each hop to be used for multicast/mtrace transit. The use of encryption protection between nodes is, however, out of the scope of this document.

10. Acknowledgements

This specification started largely as a transcription of Van Jacobson's slides from the 30th IETF, and the implementation in mrouted 3.3 by Ajit Thyagarajan. Van's original slides credit Steve Casner, Steve Deering, Dino Farinacci and Deb Agrawal. The original multicast traceroute client, mtrace (version 1), has been implemented by Ajit Thyagarajan, Steve Casner and Bill Fenner. The idea of the "S" bit to allow statistics for a source subnet is due to Tom Pusateri.

For the Mtrace version 2 specification, the authors would like to give special thanks to Tatsuya Jinmei, Bill Fenner, and Steve Casner. Also, extensive comments were received from David L. Black, Ronald Bonica, Yiqun Cai, Liu Hui, Bharat Joshi, Robert Kebler, John Kristoff, Mankamana Mishra, Heidi Ou, Eric Rescorla, Pekka Savola, Shinsuke Suzuki, Dave Thaler, Achmad Husni Thamrin, Stig Venaas, Cao Wei, and the Mboned working group members.

11. References

11.1. Normative References

[1] Bradner, S., "Key words for use in RFCs to indicate requirement levels", RFC 2119, March 1997.

[2] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 8200, July 2017.

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

[4] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", RFC 8126, June 2017.

[5] Fenner, B., Handley, M., Holbrook, H., Kouvelas, I., Parekh, R., Zhang, Z., and L. Zheng, "Protocol Independent Multicast - Sparse Mode (PIM-SM): Protocol Specification (Revised)", RFC 7761, March 2016.

[6] Handley, M., Kouvelas, I., Speakman, T., and L. Vicisano, "Bidirectional Protocol Independent Multicast (BIDIR-PIM)", RFC 5015, October 2007.

[7] 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, August 2006.

11.2. Informative References

[8] Cain, B., Deering, S., Kouvelas, I., Fenner, B., and A. Thyagarajan, "Internet Group Management Protocol, Version 3", RFC 3376, October 2002.

[9] Bumgardner, G., "Automatic Multicast Tunneling", RFC 7450, February 2015.

[10] Rosen, E. and R. Aggarwal, "Multicast in MPLS/BGP IP VPNs", RFC 6513, February 2012.

[11] Draves, R. and D. Thaler, "Default Router Preferences and More-Specific Routes", RFC 4191, November 2005.

[12] McCloghrie, K. and F. Kastenholz, "The Interfaces Group MIB", RFC 2863, June 2000.

[13] McWalter, D., Thaler, D., and A. Kessler, "IP Multicast MIB", RFC 5132, December 2007.

[14] Gill, V., Heasley, J., Meyer, D., Savola, P., and C. Pignataro, "The Generalized TTL Security Mechanism (GTSM)", RFC 5082, October 2007.

[15] Adams, A., Nicholas, J., and W. Siadak, "Protocol Independent Multicast - Dense Mode (PIM-DM): Protocol Specification (Revised)", RFC 3973, January 2005.

Authors’ Addresses

Hitoshi Asaeda
National Institute of Information and Communications Technology
4-2-1 Nukui-Kitamachi
Koganei, Tokyo 184-8795
Japan

Email: asaeda@nict.go.jp

Kerry Meyer

Email: kerry.meyer@me.com

WeeSan Lee (editor)

Email: weesan@weesan.com