MBONED Working Group	H. Asaeda	
Internet-Draft	Keio University	
Expires: May 15, 2008	T. Jinmei	
	Toshiba Corporation	
	W. Fenner	
	AT&T Research	
	S. Casner	
	Packet Design, Inc.	
	November 12, 2007	

TOC

Mtrace Version 2: Traceroute Facility for IP Multicast
draft-ietf-mboned-mtrace-v2-00

Status of this Memo

Abstract

This document describes the IP multicast traceroute facility. Unlike unicast traceroute, multicast traceroute requires special implementations on the part of routers. This specification describes the required functionality in multicast routers, as well as how management applications can use the new router functionality.

Table of Contents

---

1. Introduction

TOC

The unicast "traceroute" program allows the tracing of a path from one machine to another. The key mechanism for unicast traceroute is the ICMP TTL exceeded message, which is specifically precluded as a response to multicast packets. On the other hand, the multicast traceroute facility allows the tracing of an IP multicast routing paths. In this document, we specify the new multicast "traceroute" facility to be implemented in multicast routers and accessed by diagnostic programs. The new multicast traceroute, mtrace version 2 or mtrace2, can provide additional information about packet rates and losses that the unicast traceroute cannot, and generally requires fewer packets to be sent.

- To be able to trace the path that a packet would take from some source to some destination.
- To be able to isolate packet loss problems (e.g., congestion).
- To be able to isolate configuration problems (e.g., TTL threshold).
- To minimize packets sent (e.g. no flooding, no implosion).

This document supports both IPv4 and IPv6 multicast traceroute facility. The protocol design, concept, and program behavior are same between IPv4 and IPv6 mtrace2. Regarding the previous IPv4 multicast traceroute, mtrace, the query and response messages for IPv4 mtrace are implemented as IGMP messages (Cain, B., Deering, S., Kouvelas, I., Fenner, B., and A. Thyagarajan, “Internet Group Management Protocol,

Version 3," October 2002.) [refs.IGMPv3]. On the other hand, mtrace2 messages are carried on UDP, whereas the packet formats of IPv4 and IPv6 mtrace2 are different (but similar) because of the different address family.

---

2. Terminology

TOC

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 RFC 2119 (Bradner, S., "Key words for use in RFCs to indicate requirement levels," March 1997.) [refs.KEYWORDS].

Since multicast traceroutes 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 traffic is expected from the specified source and group.

Outgoing interface:
The interface on which traffic is forwarded from the specified source and group toward the destination. It is the interface on which the multicast traceroute Request was received.

Previous-hop router:
The router that is on the link attached to the Incoming Interface and is responsible for forwarding traffic for the specified source and group.

Group state:
It is the state in which a shared-tree protocol (e.g., PIM-SM (Fenner, B., Handley, M., Holbrook, H., and I. Kouvelas, "Protocol Independent Multicast - Sparse Mode (PIM-SM): Protocol Specification (Revised)," August 2006.) [refs.PIMSM]) running on a router chooses the previous-hop router toward the core router (or RP) as its parent router. In this state, source-specific state is not available for the corresponding multicast address on the router.

Source-specific state:
It is the state in which a routing protocol running on a router chooses the path that would be followed for a source-specific join.

---

3. Overview

TOC

Given a multicast distribution tree, tracing from a source to a multicast destination is hard, since you don't know down which branch of the multicast tree the destination lies. This means that you have to flood the whole tree to find the path from one source to one

destination. However, walking up the tree from destination to source is easy, as most existing multicast routing protocols know the previous hop for each source. Tracing from destination to source can involve only routers on the direct path.

The party requesting the traceroute (which need be neither the source nor the destination) sends a traceroute Query packet to the last-hop multicast router for the given destination. The last-hop router turns the Query into a Request packet by adding a response data block containing its interface addresses and packet statistics, and then forwards the Request packet via unicast to the router that it believes is the proper previous hop for the given source and group. Each hop adds its response data to the end of the Request packet, then unicast forwards it to the previous hop. The first hop router (the router that believes that packets from the source originate on one of its directly connected networks) changes the packet type to indicate a Response packet and sends the completed response to the response destination address. The response may be returned before reaching the first hop router if a fatal error condition such as "no route" is encountered along the path.

Multicast traceroute uses any information available to it in the router to attempt to determine a previous hop to forward the trace towards. Multicast routing protocols vary in the type and amount of state they keep; multicast traceroute endeavors to work with all of them by using whatever is available. For example, if a DVMRP router has no active state for a particular source but does have a DVMRP route, it chooses the parent of the DVMRP route as the previous hop. If a PIM-SM router is on the (*,G) tree, it chooses the parent towards the RP as the previous hop. In these cases, no source/group-specific state is available, but the path may still be traced.

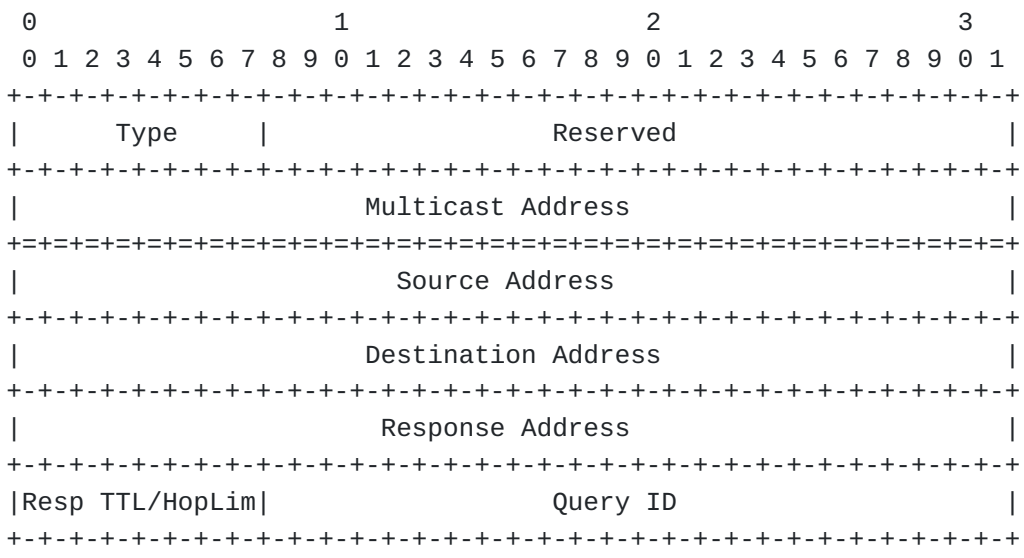
---

4. IPv4 Multicast Traceroute Header

TOC

The mtrace2 message is carried as a UDP packet. The UDP source port is uniquely selected by the local host operating system. The UDP destination port is the IANA reserved mtrace2 port number (see Section 11 (IANA Considerations)). The UDP checksum MUST be valid in mtrace2 control messages.

The IPv4 mtrace2 includes the common packet header as follows. The header is only filled in by the originator of the traceroute Query; intermediate routers MUST NOT modify any of the fields.



4.1. Type: 8 bits

TOC

The UDP type field is defined to be "0x1" for traceroute queries and requests. The UDP type field is changed to "0x2" when the packet is completed and sent as a response from the first hop router to the querier. Two codes are required so that multicast routers won't attempt to process a completed response in those cases where the initial query was issued from a router or the response is sent via multicast.

4.2. # hops: 8 bits

TOC

This field specifies the maximum number of hops that the requester wants to trace. If there is some error condition in the middle of the path that keeps the traceroute request from reaching the first-hop router, this field can be used to perform an expanding-ring search to trace the path to just before the problem.

4.3. Checksum: 16 bits

TOC

The checksum is the 16-bit one's complement of the one's complement sum of the whole UDP message (the entire IP payload) (Braden, B., Borman, D., and C. Partridge, “Computing the Internet Checksum,” September 1988.) [refs.Cksum]. When computing the checksum, the

checksum field is set to zero. When transmitting packets, the checksum MUST be computed and inserted into this field. When receiving packets, the checksum MUST be verified before processing a packet.

4.4. Multicast Address

TOC

This field specifies the multicast address to be traced, or zero if no group-specific information is desired. Note that non-group-specific traceroutes may not be possible with certain multicast routing protocols.

4.5. Source Address

TOC

This field specifies the IP address of the multicast source for the path being traced, or 0xffffffff if no source-specific information is desired. Note that non-source-specific traceroutes may not be possible with certain multicast routing protocols.

4.6. Destination Address

TOC

This field specifies the IP address of the multicast receiver for the path being traced. The trace starts at this destination and proceeds toward the traffic source.

4.7. Response Address

TOC

This field specifies IP address to which the completed traceroute response packet gets sent. It can be a unicast address or a multicast address, as explained in Section 8.2 (Traceroute Request)

4.8. Resp TTL: 8 bits

TOC

This field specifies the TTL at which to multicast the response, if the response address is a multicast address.

4.9. Query ID: 24 bits

TOC

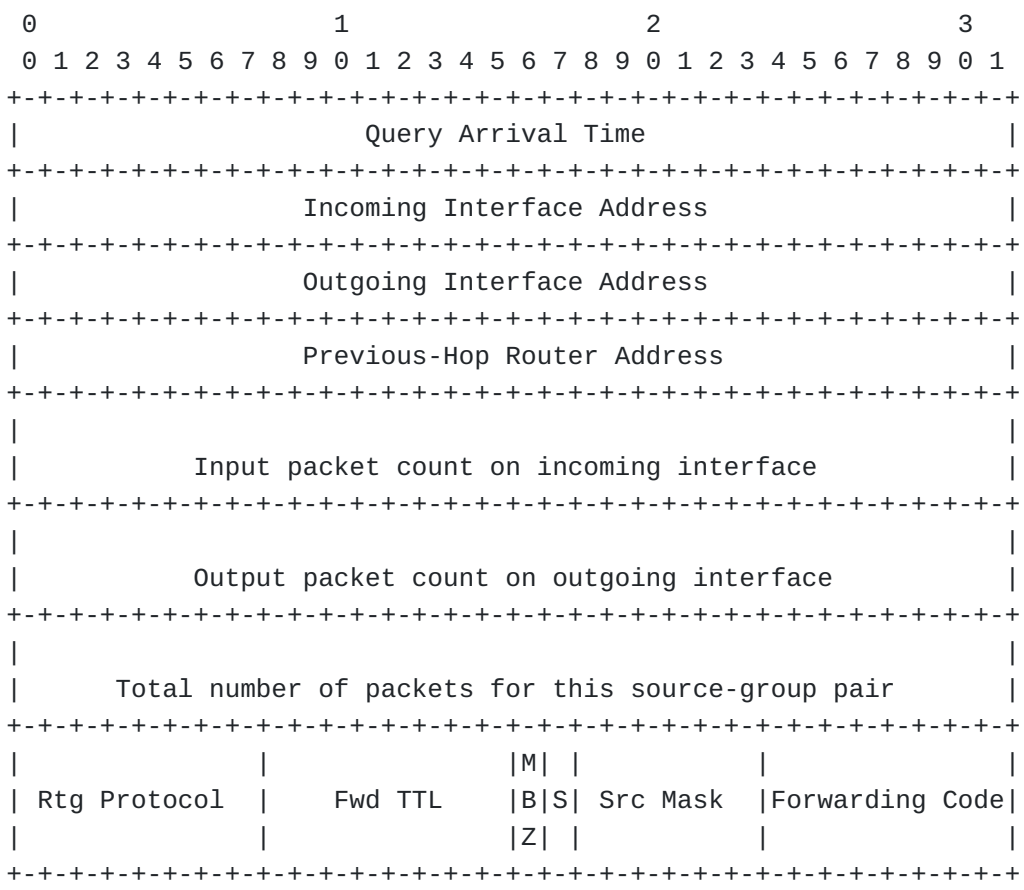
This field is used as a unique identifier for this traceroute request so that duplicate or delayed responses may be detected and to minimize collisions when a multicast response address is used.

---

5. IPv4 Multicast Traceroute Response Data

TOC

Each intermediate router in a trace path appends "response data" to the forwarded trace packet. The response data looks as follows.



5.1. Query Arrival Time: 32 bits

TOC

The Query Arrival Time is a 32-bit NTP timestamp specifying the arrival time of the traceroute 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 UNIX timeval to a 32-bit NTP timestamp:

```
query_arrival_time
= (tv.tv_sec + 32384) << 16 + ((tv.tv_usec << 10) / 15625)
```

The constant 32384 is the number of seconds from Jan 1, 1900 to Jan 1, 1970 truncated to 16 bits. ((tv.tv_usec << 10) / 15625) is a reduction of ((tv.tv_usec / 100000000) << 16).

5.2. Incoming Interface Address

TOC

This field specifies the address of the interface on which packets from this source and group are expected to arrive, or 0 if unknown.

5.3. Outgoing Interface Address

TOC

This field specifies the address of the interface on which packets from this source and group flow to the specified destination, or 0 if unknown.

5.4. Previous-Hop Router Address

TOC

This field specifies the router from which this router expects packets from this source. This may be a multicast group (e.g. ALL-[protocol]-ROUTERS.MCAST.NET) if the previous hop is not known because of the workings of the multicast routing protocol. However, it should be 0 if the incoming interface address is unknown.

5.5. Packet counts

TOC

Note that these packet counts SHOULD be as up to date as possible. If packet counts are not being maintained on the processor that handles the traceroute request in a multi-processor router architecture, the packet SHOULD be delayed while the counters are gathered from the remote processor(s). If this occurs, the Query Arrival Time should be updated to reflect the time at which the packet counts were learned.

5.6. Input packet count on incoming interface

TOC

This field contains the number of multicast packets received for all groups and sources on the incoming interface, or 0xffffffffffffffff if no count can be reported. This counter should have the same value as ifInMulticastPkts from the IF-MIB (McCloghrie, K. and F. Kastenholz, "The Interfaces Group MIB," June 2000.) [refs.IFMIB] for this interface.

5.7. Output packet count on incoming interface

TOC

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 0xffffffffffffffff if no count can be reported. This counter should have the same value as ifOutMulticastPkts from the IF-MIB for this interface.

5.8. Total number of packets for this source-group pair

TOC

This field counts the number of packets from the specified source forwarded by this router to the specified group, or 0xffffffffffffffff if no count can be reported. If the S bit is set, the count is for the source network, as specified by the Src Mask field. If the S bit is set and the Src Mask field is 63, 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 IPMROUTE-STD-MIB (McWalter, D., Thaler, D., and A. Kessler, "IP Multicast MIB," March 2007.) [refs.MROUTEMIB] for this forwarding entry.

5.9. Rtg Protocol: 8 bits

TOC

This field describes the routing protocol in use between this router and the previous-hop router. Specified values include:

1	DVMRP
2	MOSPF
3	PIM
4	CBT
5	PIM using special routing table
6	PIM using a static route
7	DVMRP using a static route
8	PIM using MBGP route
9	CBT using special routing table
10	CBT using a static route
11	PIM using state created by Assert processing
12	Bi-directional PIM

Note that some of the routing protocols or functions are not supported or not used in either of IPv4 multicast nor IPv6 multicast.

5.10. Fwd TTL: 8 bits

TOC

This field contains the TTL that a packet is required to have before it will be forwarded over the outgoing interface.

5.11. MBZ: 1 bit

TOC

Must be zeroed on transmission and ignored on reception.

5.12. S: 1 bit

TOC

This S bit 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.

5.13. Src Mask: 6 bits

TOC

This field contains the number of 1's in the netmask this 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 63 (0x3f).

5.14. Forwarding Code: 8 bits

TOC

This field contains a forwarding information/error code. Defined values are as follows;

Value	Name	Description
0x00	NO_ERROR	No error
0x01	WRONG_IF	Traceroute request arrived on an interface to which this router would not forward for this source,group,destination.
0x02	PRUNE_SENT	This router has sent a prune upstream which applies to the source and group in the traceroute request.
0x03	PRUNE_RCVD	This router has stopped forwarding for this source and group in response to a request from the next hop router.
0x04	SCOPED	The group is subject to administrative scoping at this hop.
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 last-hop router.
0x07	NOT_FORWARDING	This router is not forwarding this source, group out the outgoing interface for an unspecified reason.
0x08	REACHED_RP	Reached Rendez-vous Point or Core
0x09	RPF_IF	Traceroute request arrived on the expected RPF interface for this source, group.
0x0A	NO_MULTICAST	Traceroute request arrived on an interface which is not enabled for multicast.
0x0B	INFO_HIDDEN	One or more hops have been hidden from this trace.
0x81	NO_SPACE	There was not enough room to insert another response data block in the packet.
0x82	OLD_ROUTER	The previous-hop router does not understand traceroute requests.

0x83	ADMIN_PROHIB	Traceroute is administratively prohibited.

Note that if a router discovers there is not enough room in a packet to insert its response, it puts the 0x81 error code in the previous router's Forwarding Code field, overwriting any error the previous router placed there. A multicast traceroute client, upon receiving this error, MAY restart the trace at the last hop listed in the packet. The 0x80 bit of the Forwarding Code is used to indicate a fatal error. A fatal error is one where the router may know the previous hop but cannot forward the message to it.

---

6. IPv6 Multicast Traceroute Header

TOC

IPv6 mtrace2 includes the common packet header as follows. Because of the specification of the IPv6 address, all IPv6 addresses used in each field consume 128 bits length.


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
hops
Checksum
Reserved
Multicast Address
Source Address
Destination Address
Response Address
Resp Hop Limit
Query ID

TOC

6.1. Type: 8 bits

The UDP type field is defined to be "0x1" for traceroute queries and requests. The UDP type field is changed to "0x2" when the packet is completed and sent as a response from the first hop router to the querier. Two codes are required so that multicast routers won't attempt to process a completed response in those cases where the initial query was issued from a router or the response is sent via multicast.

6.2. # hops: 8 bits

TOC

Same definition described in Section 4.2 (# hops: 8 bits)

6.3. Checksum: 16 bits

TOC

As defined in (Deering, S. and R. Hinden, “Internet Protocol, Version 6 (IPv6) Specification,” December 1998.) [refs.IPv6], the checksum is the 16-bit one's complement of the one's complement sum of the entire UDP message, starting with the UDP message type field, and prepended with a "pseudo-header" of IPv6 header fields.

6.4. Reserved: 32 bits

TOC

Initialized to zero by the sender; ignored by receivers.

6.5. Multicast Address

TOC

Same definition described in Section 4.4 (Multicast Address)

6.6. Source Address

TOC

This field specifies the IPv6 address of the multicast source for the path being traced, or is filled with the unspecified address (::) if no source-specific information is desired. Note that non-source-specific traceroutes may not be possible with certain multicast routing protocols.

6.7. Destination Address

TOC

Same definition described in Section 4.6 (Destination Address)

6.8. Response Address

TOC

Same definition described in Section 4.7 (Response Address)

6.9. Resp Hop Limit: 8 bits

TOC

This field specifies the hop limit at which to multicast the response, if the response address is a multicast address.

6.10. Query ID: 24 bits

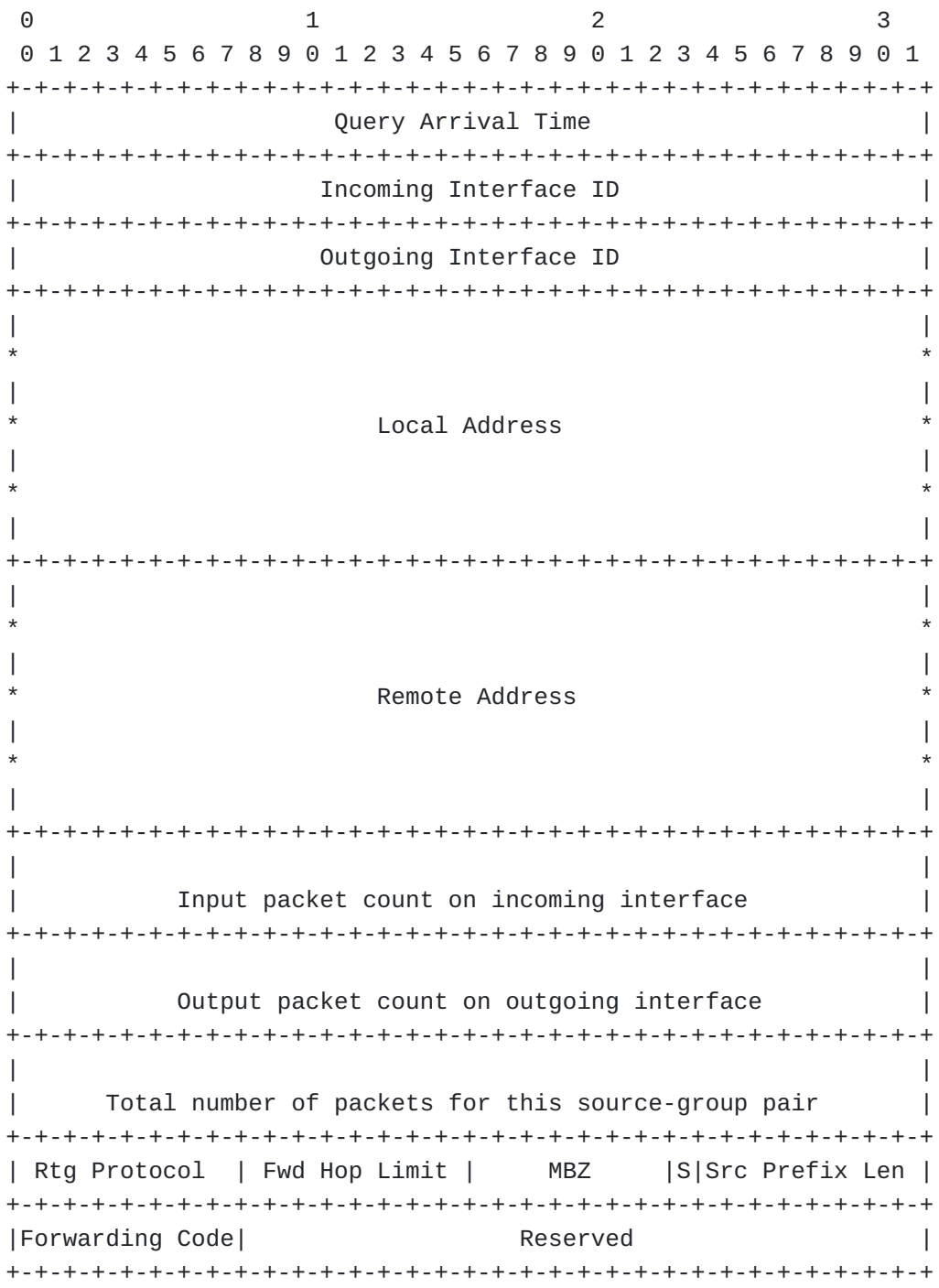
TOC

Same definition described in Section 4.9 (Query ID: 24 bits)

7. IPv6 Multicast Traceroute Response Data

TOC

Each intermediate router in a trace path appends "response data" to the forwarded trace packet. The response data looks as follows.



7.1. Query Arrival Time: 32 bits

TOC

Same definition described in Section 5.1 (Query Arrival Time: 32 bits)

7.2. Incoming Interface ID: 32 bits

TOC

This field specifies the interface ID on which packets from this source and group are expected to arrive, or 0 if unknown. This ID should be the value taken from InterfaceIndex of the IF-MIB for this interface. This field is carried in network byte order.

7.3. Outgoing Interface ID: 32 bits

TOC

This field specifies the interface ID on which packets from this source and group flow to the specified destination, or 0 if unknown. This ID should be the value taken from InterfaceIndex of the IF-MIB for this interface. This field is carried in network byte order.

7.4. Local Address

TOC

This field specifies a global IPv6 address that uniquely identifies the router. A unique local unicast address (Draves, R. and D. Thaler, “Default Router Preferences and More-Specific Routes,” November 2005.) [refs.DefRtr] SHOULD NOT be used unless the node is only assigned link-local and unique local addresses. [TBD: What if the node is only assigned link-local addresses? It should be very unlikely case, but is possible even for a properly working router.]
Note that since interface indices used in the Incoming and Outgoing Interface ID fields are node-local information, a global identifier is needed to specify the router.

7.5. Remote Address

TOC

This field specifies the address of the previous-hop router, which, in most cases, is a link-local unicast address for the queried source and destination addresses.
Although a link-local address does not have enough information to identify a node, it is possible to detect the previous-hop router with the assistance of Incoming Interface ID and the current router address (i.e., Local Address).
This may be a multicast group (e.g., ALL-[protocol]-ROUTERS.MCAST.NET) if the previous hop is not known because of the workings of the multicast routing protocol. However, it should be the unspecified address (::) if the incoming interface address is unknown.

7.6. Input packet count on incoming interface

TOC

Same definition described in Section 5.6 (Input packet count on incoming interface)

7.7. Output packet count on incoming interface

TOC

Same definition described in Section 5.7 (Output packet count on incoming interface)

7.8. Total number of packets for this source-group pair

TOC

This field counts the number of packets from the specified source forwarded by this router to the specified group, or 0xffffffffffffffff if no count can be reported. If the S bit is set, 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 IPMROUTE-STD-MIB for this forwarding entry.

7.9. Rtg Protocol: 8 bits

TOC

Same definition described in Section 5.9 (Rtg Protocol: 8 bits)
Note that some of the routing protocols or functions are not supported or not used in IPv6 multicast.

7.10. Fwd Hop Limit: 8 bits

TOC

This field contains the hop limit that a packet is required to have before it will be forwarded over the outgoing interface.

TOC

7.11. MBZ: 7 bits

Must be zeroed on transmission and ignored on reception.

7.12. S: 1 bit

TOC

This S bit 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 Prefix Len field.

7.13. Src Prefix Len: 8 bits

TOC

This field contains the decimal number of 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)

7.14. Forwarding Code: 8 bits

TOC

Same definition described in Section 5.14 (Forwarding Code: 8 bits)

7.15. Reserved: 24 bit

TOC

Initialized to zero by the sender; ignored by receivers.

---

8. Router Behavior

TOC

All of these actions are performed in addition to (NOT instead of) forwarding the packet, if applicable. E.g. a multicast packet that has TTL or the hop limit remaining MUST be forwarded normally, as MUST a unicast packet that has TTL or the hop limit remaining and is not addressed to this router.

TOC

8.1. Traceroute Query

A traceroute Query message is a traceroute message with no response blocks filled in, and uses UDP type 0x1 for IPv4 and IPv6 mtrace2.

8.1.1. Packet Verification

TOC

Upon receiving a traceroute Query message, a router must examine the Query to see if it is the proper last-hop router for the destination address in the packet. It is the proper last-hop router if it has a multicast-capable interface on the same subnet as the Destination Address and is the router that would forward traffic from the given source onto that subnet.
If the router determines that it is not the proper last-hop router, or it cannot make that determination, it does one of two things depending if the Query was received via multicast or unicast. If the Query was received via multicast, then it MUST be silently dropped. If it was received via unicast, a forwarding code of WRONG_LAST_HOP is noted and processing continues as in Section 8.2 (Traceroute Request)
Duplicate Query messages as identified by the tuple (IP Source, Query ID) SHOULD be ignored. This MAY be implemented using a simple 1-back cache (i.e. remembering the IP source and Query ID of the previous Query message that was processed, and ignoring future messages with the same IP Source and Query ID). Duplicate Request messages MUST NOT be ignored in this manner.

8.1.2. Normal Processing

TOC

When a router receives a traceroute Query and it determines that it is the proper last-hop router, it treats it like a traceroute Request and performs the steps listed in Section 8.2 (Traceroute Request)

8.2. Traceroute Request

TOC

A traceroute Request is a traceroute message with some number of response blocks filled in, and uses UDP type 0x1 for IPv4 and IPv6 mtrace2. Routers can tell the difference between Queries and Requests by checking the length of the packet.

TOC

8.2.1. Packet Verification

If the traceroute 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. 224/24 for IPv4, FFx2::/16 (Hinden, R. and S. Deering, “IP Version 6 Addressing Architecture,” July 1998.) [refs.IPv6Addr] for IPv6), it MUST be silently ignored.

8.2.2. Normal Processing

TOC

When a router receives a traceroute Request, 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".

1. If there is room in the current buffer (or the router can efficiently allocate more space to use), insert a new response block into the packet and fill in the Query Arrival Time, Outgoing Interface Address (for IPv4 mtrace2) or Outgoing Interface ID (for IPv6 mtrace2), Output Packet Count, and Fwd TTL or Fwd Hop Limit. If there was no room, fill in the response code "NO_SPACE" in the **previous** hop's response block, and forward the packet to the requester as described in "Forwarding Traceroute Requests".

2. Attempt to determine the forwarding information for the source and group specified, using the same mechanisms as would be used when a packet is received from the source destined for the group. State need not be instantiated, it can be "phantom" state created only for the purpose of the trace.

   If using a shared-tree protocol and there is no source-specific state, or if the source is specified as 0xFFFFFFFF, group state should be used. If there is no group state or the group is specified as 0, potential source state (i.e. the path that would be followed for a source-specific Join) should be used. If this router is the Core or RP and no source-specific information is available, note an error code of REACHED_RP.

3. If no forwarding information can be determined, the router notes an error code of NO_ROUTE, sets the remaining fields that have not yet been filled in to zero, and then forwards the packet to the requester as described in "Forwarding Traceroute Requests".

4. Fill in the Incoming Interface Address, Previous-Hop Router Address, Input Packet Count, Total Number of Packets, Routing Protocol, S, and Src Mask from the forwarding information that was determined.

5. If traceroute is administratively prohibited or the previous hop router does not understand traceroute requests, note the appropriate forwarding code (ADMIN_PROHIB or OLD_ROUTER). If traceroute is administratively prohibited and any of the fields as filled in step 4 are considered private information, zero out the applicable fields. Then the packet is forwarded to the requester as described in "Forwarding Traceroute Requests".

6. If the reception interface is not enabled for multicast, note forwarding code NO_MULTICAST. If the reception interface is the interface from which the router would expect data to arrive from the source, note forwarding code RPF_IF. Otherwise, if the reception interface is not one to which the router would forward data from the source to the group, a forwarding code of WRONG_IF is noted.

7. If the group is subject to administrative scoping on either the Outgoing or Incoming interfaces, a forwarding code of SCOPED is noted.

8. If this router is the Rendez-vous Point or Core for the group, a forwarding code of REACHED_RP is noted.

9. If this router has sent a prune upstream which applies to the source and group in the traceroute Request, it notes forwarding code PRUNE_SENT. If the router has stopped forwarding downstream in response to a prune sent by the next hop router, it notes forwarding code PRUNE_RCVD. If the router should normally forward traffic for this source and group downstream but is not, it notes forwarding code NOT_FORWARDING.

10. The packet is then sent on to the previous hop or the requester as described in Section 8.4 (Forwarding Traceroute Requests).

8.3. Traceroute Response

TOC

A router must forward all traceroute response packets normally, with no special processing. If a router has initiated a traceroute with a Query or Request message, it may listen for Responses to that traceroute but MUST still forward them as well.

8.4. Forwarding Traceroute Requests

TOC

If the Previous-hop router is known for this request and the number of response blocks is less than the number requested, the packet is sent to that router. If the Incoming Interface is known but the Previous-hop router is not known, the packet is sent to an appropriate multicast address on the Incoming Interface. The appropriate multicast address may depend on the routing protocol in use, MUST be a link-scoped group (i.e. 224/24 for IPv4, FF02::/16 for IPv6), MUST NOT be ALL-SYSTEMS.MCAST.NET (224.0.0.1) for IPv4 and All Nodes Address (FF02::1) for IPv6, and MAY be ALL-ROUTERS.MCAST.NET (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 group. Otherwise, it is sent to the Response Address in the header, as described in Section 8.5 (Sending Traceroute Responses).
Note that it is not an error for the number of response blocks to be greater than the number requested; such a packet should simply be forwarded to the requester as described in Section 8.5 (Sending Traceroute Responses).

8.5. Sending Traceroute Responses

TOC

8.5.1. Destination Address

TOC

A traceroute response must be sent to the Response Address in the traceroute header.

8.5.2. TTL and Hop Limit

TOC

If the Response Address is unicast, the router inserts its normal unicast TTL or hop limit in the IP header, and may use any of its interface addresses as the source address. If the Response Address is multicast, the router copies the Response TTL or hop limit from the traceroute header into the IP header.

TOC

8.5.3. Source Address

If the Response Address is unicast, the router may use any of its interface addresses as the source address. Since some multicast routing protocols forward based on source address, if the Response Address is multicast, the router MUST use an address that is known in the multicast routing topology if it can make that determination.

8.5.4. Sourcing multicast responses

TOC

When a router sources a multicast response, the response packet MUST be sent on a single interface, then forwarded as if it were received on that interface. It MUST NOT source the response packet individually on each interface, in order to avoid duplicate packets.

8.6. Hiding information

TOC

Information about a domain's topology and connectivity may be hidden from multicast traceroute requests. The exact mechanism is not specified here; however, the INFO_HIDDEN forwarding code may be used to note that, for example, the incoming interface address and packet count are for the entrance to the domain and the outgoing interface address and packet count are the exit from the domain. The source-group packet count may be from either router or not specified (0xffffffff).

---

9. Using multicast traceroute

TOC

9.1. Sample client

TOC

This section describes the behavior of an example multicast traceroute client.

TOC

9.1.1. Sending initial query

When the destination of the mtrace2 is the machine running the client, the mtrace2 Query packet can be sent to the ALL-ROUTERS.MCAST.NET (224.0.0.2) for IPv4 or All Routers Address (FF02::2) for IPv6. This will ensure that the packet is received by the last-hop router on the subnet. Otherwise, if the proper last-hop router is known for the mtrace2 destination, the Query could be unicasted to that router. Otherwise, the Query packet should be multicasted to the group being queried; if the destination of the mtrace2 is a member of the group, this will get the Query to the proper last-hop router. In this final case, the packet should contain the Router Alert option (Katz, D., "IP Router Alert Option," February 1997.) [refs.RA], to make sure that routers that are not members of the multicast group notice the packet. See also Section 9.2 (Last hop router) on determining the last-hop router.

9.1.2. Determining the Path

TOC

The client could send a small number of initial query messages with a large "# hops" field, 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 response, then 2, and so on. If no response is received at a certain hop, the hop count can continue past the non-responding hop, in the hopes that further hops may respond. These attempts should continue until a user-defined timeout has occurred. See also Section 9.3 (First hop router) and Section 9.4 (Broken intermediate router) on receiving the results of a trace.

9.1.3. Collecting statistics

TOC

After a client has determined that it has traced the whole path or as much as it can expect to (see Section 9.5 (Mtrace2 termination)), 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 10.3 (Packet loss) and Section 10.4 (Link Utilization).

TOC

9.2. Last hop router

The mtrace2 querier 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 group being traced (since the last hop router listens to that group). All routers except the correct last hop router should ignore any mtrace2 request received via multicast. Mtrace2 requests which are multicasted to the group being traced must include the Router Alert option (Katz, D., “IP Router Alert Option,” February 1997.) [refs.RA].
Another alternative is to unicast to the trace destination. Traceroute requests which are unicasted to the trace destination must include the Router Alert option, in order that the last-hop router is aware of the packet.
If the traceroute querier is attached to the same router as the destination of the request, the traceroute request may be multicasted to ALL-ROUTERS.MCAST.NET (224.0.0.2) for IPv4 or All Routers Address (FF02::2) for IPv6 if the last-hop router is not known.

9.3. First hop router

TOC

The mtrace2 querier may not be unicast reachable from the first hop router. In this case, the querier should set the traceroute response address to a multicast address, and should set the response TTL (or hop limit) to a value sufficient for the response from the first hop router to reach the querier. It may be appropriate to start with a small TTL and increase in subsequent attempts until a sufficient TTL is reached, up to an appropriate maximum (such as 192).
The IANA has assigned 224.0.1.32, MTRACE.MCAST.NET as the default multicast group for IPv4 mtrace2 responses, and will assign MTRACE2_IPV6RESPADDR (TBD (see Section 11 (IANA Considerations)) for IPv6 mtrace2 responses. Other groups may be used if needed, e.g. when using mtrace2 to diagnose problems with the IANA-assigned group.

9.4. Broken intermediate router

TOC

A broken intermediate router might simply not understand traceroute packets, and drop them. The querier would then get no response at all from its traceroute requests. It should then perform a hop-by-hop search by setting the number of responses field until it gets a response (both linear and binary search are options, but binary is likely to be slower because a failure requires waiting for a timeout).

9.5. Mtrace2 termination

TOC

When performing an expanding hop-by-hop trace, it is necessary to determine when to stop expanding.

9.5.1. Arriving at source

TOC

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 Previous Hop router is zero.

9.5.2. Fatal error

TOC

A trace has encountered a fatal error if the last Forwarding Error in the trace has the 0x80 bit set.

9.5.3. No previous hop

TOC

A trace can not continue if the last Previous Hop in the trace is set to 0.

9.5.4. Traceroute shorter than requested

TOC

If the trace that is returned is shorter than requested (i.e. the number of Response blocks is smaller than the "# hops" field), the trace encountered an error and could not continue.

9.6. Continuing after an error

TOC

When the NO_SPACE error occurs, the client might try to continue the trace by starting it at the last hop in the trace. It can do this by unicasting to this router's outgoing interface address, keeping all fields the same. If this results in a single hop and a "WRONG_IF"

error, the client may try setting the trace destination to the same outgoing interface address.
If a trace times out, it is likely to be because a router in the middle of the path does not support multicast traceroute. That router's address will be in the Previous Hop field of the last entry in the last reply packet received. A client may be able to determine (via mrinfo or SNMP [refs.DefRtr] (Draves, R. and D. Thaler, “Default Router Preferences and More-Specific Routes,” November 2005.)[refs.MROUTEMIB] (McWalter, D., Thaler, D., and A. Kessler, “IP Multicast MIB,” March 2007.)) a list of neighbors of the non-responding router. If desired, each of those neighbors could be probed to determine the remainder of the path. Unfortunately, this heuristic may end up with multiple paths, since there is no way of knowing what the non-responding router's algorithm for choosing a previous-hop router is. However, if all paths but one flow back towards the non-responding router, it is possible to be sure that this is the correct path.

9.7. Multicast Traceroute and shared tree routing protocols

TOC

When using shared-tree routing protocols like PIM-SM and CBT, a more advanced client may use multicast traceroute to determine paths or potential paths.

9.7.1. PIM-SM

TOC

When a multicast traceroute 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 trace Query may be unicasted to the RP.

9.7.2. Bi-directional PIM

TOC

Bi-directional PIM (Handley, M., Kouvelas, I., and T. Speakman, “Bi-directional Protocol Independent Multicast (BIDIR-PIM),” February 2007.) [refs.Bidir] 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 sources to the RPA (Rendezvous Point Address) and from

the RPA to receivers without requiring source-specific state. In contrast to PIM-SM, RP always has the state to trace.
A Designated Forwarder (DF) for a given RPA is in charge of forwarding downstream traffic onto its link, and forwarding upstream traffic from its link towards the RPL (Rendezvous Point Link) that the RPA belongs to. Hence mtrace2 reports DF addresses or RPA along the path.

9.7.3. CBT

TOC

When a multicast traceroute reaches a CBT (Ballardie, T., “Core Based Trees (CBT version 2) Multicast Routing -- Protocol Specification --,” September 1997.) [refs.CBT] Core, it must simply stop since CBT does not have source-specific state. However, a second trace can be performed, setting the trace destination to the traffic source, the trace group to the group being traced, and the trace source to the Core (or to 0, since CBT does not have source-specific state). This trace Query may be unicasted to the Core. There are two possibilities when combining the two traces:

9.7.3.1. No overlap

TOC

If there is no overlap between the two traces, the second trace can be reversed and appended to the first trace. This composite trace shows the full path from the source to the destination.

9.7.3.2. Overlapping paths

TOC

If there is a portion of the path that is common to the ends of the two traces, that portion is removed from both traces. Then, as in the no overlap case, the second trace is reversed and appended to the first trace, and the composite trace again contains the full path.
This algorithm works whether the source has joined the CBT tree or not.

9.8. Protocol-specific considerations

TOC

TOC

9.8.1. DVMRP

DVMRP's dominant router election and route exchange guarantees that DVMRP routers know whether or not they are the last-hop forwarder for the link and who the previous hop is.

9.8.2. PIM-DM

TOC

Routers running PIM Dense Mode 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, multicast traceroute can only trace existing paths, not potential paths. When there are multiple possible paths but the branch points are not on shared media, the previous hop router is known, but the last hop router 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 last-hop forwarder for the link (because they won or lost an Assert battle) and know who the previous hop is (because it won an Assert battle). Therefore, multicast traceroute is always able to follow the proper path when traffic is flowing.

---

10. Problem Diagnosis

TOC

10.1. Forwarding Inconsistencies

TOC

The forwarding error code can tell if a group is unexpectedly pruned or administratively scoped.

10.2. TTL or hop limit problems

TOC

By taking the maximum of (hops from source + forwarding TTL (or hop limit) threshold) over all hops, you can discover the TTL required for the source to reach the destination.

TOC

10.3. Packet loss

By taking two traces, you can find packet loss information by comparing the difference in input packet counts to the difference in output packet counts at the previous hop. On a point-to-point link, any difference in these numbers implies packet loss. Since the packet counts may be changing as the trace query is propagating, there may be small errors (off by 1 or 2) in these statistics. However, these errors will not accumulate if multiple traces are taken to expand the measurement period. 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 response data 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 response data 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.

10.4. Link Utilization

TOC

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.

10.5. Time delay

TOC

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.

---

11. IANA Considerations

TOC

The following new assignments can only be made via a Standards Action as specified in (Narten, T. and H. Alvestrand, “Guidelines for Writing an IANA Considerations Section in RFCs,” October 1998.) [refs.IANA].

11.1. Routing protocols

TOC

The IANA is responsible for allocating new Routing Protocol codes. The Routing Protocol code is somewhat problematic, since in the case of protocols like CBT and PIM it must encode both a unicast routing algorithm and a multicast tree-building protocol. The space was not divided into two fields because it was already small and some combinations (e.g. DVMRP) would be wasted.

11.2. Forwarding codes

TOC

New Forwarding codes must only be created by an RFC that modifies this document's Section 9 (Using multicast traceroute), fully describing the conditions under which the new forwarding code is used. The IANA may act as a central repository so that there is a single place to look up forwarding codes and the document in which they are defined.

11.3. UDP destination port and IPv6 address

TOC

The IANA should allocate UDP destination port for multicast traceroute version 2 upon publication of the first RFC. Additionally, the well-known multicast address (MTRACE2_IPV6RESPADDR) intended for default use by IPv6 multicast traceroute should be registered and defined by the first RFC published.

TOC

12. Security Considerations

12.1. Topology Discovery

TOC

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.

12.2. Traffic Rates

TOC

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.

12.3. Unicast Replies

TOC

The "Response address" field may be used to send a single packet (the traceroute Reply packet) to an arbitrary unicast address. It is possible to use this facility as a packet amplifier, as a small multicast traceroute Query may turn into a large Reply packet.

13. Acknowledgements

TOC

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 unicasting a multicast traceroute Query to the destination of the trace with Router Alert set is due to Tony Ballardie. The idea of the "S" bit to allow statistics for a source subnet is due to Tom Pusateri.

TOC

14. References

14.1. Normative References

TOC

[refs.KEYWORDS]	Bradner, S., "Key words for use in RFCs to indicate requirement levels," RFC 2119, March 1997.
[refs.IPv6]	Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification," RFC 2460, December 1998.
[refs.IPv6Addr]	Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture," RFC 2373, July 1998.
[refs.IGMPv3]	Cain, B., Deering, S., Kouvelas, I., Fenner, B., and A. Thyagarajan, "Internet Group Management Protocol, Version 3," RFC 3376, October 2002.
[refs.IANA]	Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs," RFC 2434, October 1998.

14.2. Informative References

TOC

[refs.DefRtr]	Draves, R. and D. Thaler, "Default Router Preferences and More-Specific Routes," RFC 4191, November 2005.
[refs.Cksum]	Braden, B., Borman, D., and C. Partridge, "Computing the Internet Checksum," RFC 1071, September 1988.
[refs.RA]	Katz, D., "IP Router Alert Option," RFC 2113, February 1997.
[refs.IFMIB]	McCloghrie, K. and F. Kastenholz, "The Interfaces Group MIB," RFC 2863, June 2000.
[refs.MROUTEMIB]	McWalter, D., Thaler, D., and A. Kessler, "IP Multicast MIB," draft-ietf-mboned-ip-mcast-mib-05.txt (work in progress), March 2007.
[refs.PIMSM]	Fenner, B., Handley, M., Holbrook, H., and I. Kouvelas, "Protocol Independent Multicast - Sparse Mode (PIM-SM): Protocol Specification (Revised)," RFC 4601, August 2006.
[refs.CBT]	Ballardie, T., "Core Based Trees (CBT version 2) Multicast Routing -- Protocol Specification --," RFC 2189, September 1997.
[refs.Bidir]	Handley, M., Kouvelas, I., and T. Speakman, "Bi-directional Protocol Independent Multicast (BIDIR-PIM)," draft-ietf-pim-bidir-09.txt (work in progress), February 2007.

Authors' Addresses

TOC

	Hitoshi Asaeda
	Keio University
	Graduate School of Media and Governance
	Fujisawa, Kanagawa 252-8520
	Japan
Email:	asaeda@wide.ad.jp
	Tatsuya Jinmei
	Toshiba Corporation
	Corporate Research & Development Center
	Kawasaki, Kanagawa 212-8582
	Japan
Email:	jinmei@isl.rdc.toshiba.co.jp
	William C. Fenner
	AT&T Research
	Menlo Park, CA 94025
	US
Email:	fenner@research.att.com
	Stephen L. Casner
	Packet Design, Inc.
	Palo Alto, CA 94304
	US
Email:	casner@packetdesign.com

Full Copyright Statement

TOC

Intellectual Property

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at http://www.ietf.org/ipr.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.