Internet Engineering Task Force
INTERNET-DRAFT
draft-ietf-msdp-traceroute-06.ps

W. Fenner
AT&T Labs - Research
D. Meyer
Sprint E|Solutions
R. Roberts
Stanford University
July 2001

MSDP Traceroute

Status of this Document

This document is a product of the MSDP Working Group. Comments should be addressed to the authors, or the mailing list at msdp@network-services.uoregon.edu.

Copyright Notice

Abstract

In order to diagnose problems with the Peer-RPF-Flooding mechanisms in the Multicast Source Discovery Protocol [2], we introduce a method of tracing the path that an SA message should have taken, along with collecting statistics along that path. This occurs in a way similar to multicast traceroute [3], with each router appending information about this hop and forwarding the message towards the desired RP.

Table of Contents

1. Introduction

One of the key problems in the deployment of MSDP [2] infrastructures has been the inability to trace the path taken by MSDP control packets. The protocol specified in the document addresses this problem by providing a traceroute facility for MSDP Source Active (SA) Messages. It is important to note that, with the exception of data encapsulated in SA messages, MSDP traceroute traces a path through the MSDP control plane. It is important to note that the MSDP control path need not be the same as the path followed by data packets.

As is the case for multicast traceroute, the goals of MSDP traceroute include

- o To be able to trace the path that a SA message would take from some source router (RP) to some destination router.
- o To be able to isolate SA Message loss problems
- o To be able to isolate configuration problems
- o To minimize packets sent (e.g. no flooding, no implosion).

1.1. Definitions

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 [1].

2. Originating an MSDP Traceroute Query

As is the case for other kinds of traceroute, both routers and hosts will want to originate MSDP traceroute queries.

2.1. Routers Originating an MSDP Traceroute Query

When a router originates an MSDP traceroute Query, it formulates the Query by filling its first block and unicasts the Query packet to its neighbor toward the specified RP. Note that this is unlike multicast traceroute, in which the Query is sent to the last-hop multicast router for the destination, converted into a Request, and forwarded back towards the source and group.

2.2. Hosts Originating an MSDP Traceroute Query

While it is believed that host originated queries would enhance the usefulness of this protocol, this functionality is not addressed in this version of the specification.

3. Packet Formats

Since the messages are treated significantly differently on the outbound and return paths, MSDP traceroute messages are encapsulated in 2 different MSDP TLVs; in-progress traceroute messages are type 6, while traceroute answers are type 7.

Following a reserved byte for alignment purposes, there is a fixed request header containing the source, group, RP of the request (note that the source and group are only used for statistics gathering, and the RP is the only thing used to route the request), a query identifier, query-type bits, and the router pointer (for returning packet hop-by-hop). Following are two tables, one for router addresses and the other for fixed-length response info filled in at each hop. The remainder of the packet is available to collect more detailed variable length response info as specified by the query-type bit vector.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   MSDP Type   |          MSDP Length          |   Reserved    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Source Address                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Group Address                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        RP Address                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Query ID                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Query-type bits          |   reserved    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  skip hops    |   max hops    |   reserved    |  rtr pointer  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| ... maxhops*4  octets of router addresses ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| ... maxhops*4  octets of fixed-length response info ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| ... optional detailed response blocks ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

3.1. Type

In-progress request: 6
Answer: 7

3.2. Source Address

Source in (S,G,RP). Specifying the source and group permits returning information about cached SA messages along the path.

3.3. Group Address

Group in (S,G,RP)

3.4. RP Address

RP in (S,G,RP). The request is routed hop by hop towards the specified RP, using the Peer-RPF forwarding rules.

3.5. Query ID

This field is used as a unique identifier for this traceroute request so that duplicate or delayed responses may be detected.

3.6. Query-type bits

Bit vector specifying collection of more detailed response data to be collected for the hops between skip hops and max hops or until the packet size is exceeded. The rightmost octet is reserved and must be zero.

3.7. skip hops

The number of hops to be skipped in this request before beginning to collect detailed response data. This allows for the data collection specified by the query-type bit vector to begin at this point in the path. Specified by the requester.

3.8. max hops

The max. # of hops requested in this traceroute request.

3.9. reserved

This octet is reserved for future use.

3.10. rtr pointer

The rtr pointer is a pointer into both the router addresses table and the fixed-length response info table. When the query is being filled in (outbound) the rtr pointer points to the next available entry. On return to the query originator, it points to the next hop back (toward the originator).

3.11. router addresses

Table of the outgoing interfaces that this request has traversed. Inserted by requester, max hops * 4 octets.

3.12. Fixed-length response info

Table of fixed response words containing limited status info for each router along the path. Inserted by requester, max hops * 4 octets.

Fixed response word:

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Peer Uptime   | Cache Entry   |       |D|R|N|C|  Return Code  |
|               |    Uptime     |       | |P|C| |               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Peer uptime: RPF peer uptime in minutes, capped at 255.

Cache Entry Uptime: cache entry uptime in minutes, capped at 255.

Bits:

D-bit: set if have this (S,G) in cache but with a different RP.

RP-bit: set if this router is an RP

NC-bit: set if this router is not caching SA's

C-bit: set if this (S,G,RP) tuple is in the cache.

Return Code:

0	No-error
1	Hit-src-RP
2	Filled-packet
3	No-neighbor
4	Reached-max-hops
5	SA-found
6	Non-RPF-neighbor-used

3.13. Detailed response block

Each detailed response block begins with a bit vector describing what types of responses are present (equal to or a sub-set of the query-type bits) and the current rtr pointer in the rightmost octet:

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               Response Type Bits              |  rtr pointer  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Types of info:

Bit 0: Next Hop info

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Next-Hop Router Address                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Bit 1: SA info

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        Count of SA messages received for this (S,G,RP)        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Count of encapsulated data packets received for this (S,G,RP) |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    SA cache entry uptime                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 SA cache entry expiry time                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Bit 2: Peering info

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Peering Uptime                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         # of Peering Resets                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Bit 3: terminate if SA found

This bit selects "SA-found" as a terminating condition. Note that selecting this option defeats loop detection.

4. Protocol Processing

4.1. Processing an in-progress traceroute packet

The following steps are taken when receiving an in-progress traceroute:

1. Ensure that the traceroute request arrived from the router whose address appears in the router address table at index (rtr pointer - 1). If not, drop the request, optionally logging a warning message.
2. If rtr pointer is greater than max hops, something bad has happened; immediately process as a partial response.
3. If this router is the source RP, set the Hit-src-RP return code and the RP-bit in the fixed-length response info word and store in the fixed-length response info table as offset by rtr pointer. Proceed to Return packet as a response.
4. If this router is an RP, set the RP-bit in the fixed-length response info word. If this router is not doing SA caching, set the NC-bit in the fixed-length response info word and proceed to the RPF peer check. Otherwise look up the SA cache information for the (S,G,RP). If the SA cache entry exists, set the C-bit in the fixed-length response info word and insert the Cache Entry Uptime field in the fixed-length response info word. If no SA cache entry exists for the (S,G,RP), check if there is a cache entry for (S,G) with a different RP. If so, then set the D-bit in the fixed-length response info word.
5. Look up the MSDP RPF neighbor for the RP, using the MSDP peer-RPF-forwarding rules in the MSDP spec [2]. If the peer-RPF-forwarding rules do not provide an RPF neighbor for this RP, but there is a cached SA for this (S,G,RP), we choose the peer which advertised this cached SA to us and set the Non-RPF-neighbor-used return code.
6. If this lookup returns a neighbor with which we have a peering (i.e. the RPF peer or the advertising peer), insert your IP address on this peering into the router address table as offset by the current rtr pointer, otherwise insert your IP address on the peering on which the traceroute packet was received. Insert the Peer Uptime field in the fixed-length response info word and move it to the fixed-length response info table as offset by the current rtr pointer. If all the query-type bits are zero, proceed to Forward packet to RPF neighbor. Otherwise proceed to Append detailed response data.
7. If no neighboring peer was selected in step 5, set the No-neighbor return code in the fixed-length response info word and move it to the fixed-length response info table as

offset by the current rtr pointer and proceed to Return packet as a response.

4.2. Append detailed response data

The detailed response data, as specified by the query-type bits, is appended to the received packet.

1. Check if rtr pointer is greater than skip hops. If not, then proceed to Forward packet to RPF neighbor.
2. Verify that space remains in the packet to add the detailed response data block(s) as specified by the query-type bits without exceeding the MSDP MTU [2] section XXX. If not, set the Filled-packet return code in the fixed-length response info word and move it to the fixed-length response info table as offset by the current rtr pointer. Proceed to Return packet as a response.
3. Copy the query-type bits to the response type bits word (the first word of the detailed response data). Insert the current value of rtr pointer in the low-order 8 bits of this word.
4. If the Next Hop info query-type bit is set, check if we determined a next hop neighbor. If so, insert this neighbor's address into the Next-Hop Router Address word. If not, turn off the Next Hop info bit in the copied response.
5. If the SA info query-type bit is set, check if we have a cached SA matching this (S,G,RP). If so, fill in the SA-info query words. If not (e.g. no matching SA entry or the router is not caching SAs), turn off the SA info bit in the copied response.
6. If the Peering info query-type bit is set, check if we determined a next hop neighbor. If so, fill in the Peering-info query words with values from this peering. If not, turn off the Peering info bit in the copied response.
7. If the terminate if SA found bit is set, finish these rules but instead of proceeding to forwarding packet to RPF neighbor, proceed to Return packet as a response.

4.3. Forward packet to RPF neighbor

1. Increment the rtr pointer field.
2. If the rtr pointer == max hops, decrement the rtr pointer field, set the Reached-max-hops return code in the fixed-length response info word and move it to the fixed-length response info table as offset by the current rtr pointer. Proceed to Return packet as a response.
3. Otherwise, unicast the current packet over the MSDP peering to the RPF neighbor (or advertiser, as selected previously).

4.4. Return packet as a response

Turn the MSDP type from traceroute request to traceroute response.

Decrement the router pointer.

If the router pointer was already 0, you're the one who requested the trace.

Otherwise, send the message on your MSDP peering with the router at the current router pointer's offset in the router address table.

4.5. Processing a traceroute answer

Decrement the router pointer.

If the router pointer was already 0, you're the one who requested the trace. Display it to the user, or otherwise handle appropriately.

Otherwise, send the message on your MSDP peering with the router at the current router pointer's offset in the router address table.

5. Security Considerations

Topology discovery is possible. Internal topology may be hidden by ensuring MSDP peerings between all border routers and ensuring that the MSDP Peer-RPF-algorithm prefers the border router peerings to internal peerings.

Packet amplification occurs (i.e. a small request elicits a large reply) but is not a concern because the reply must return to the originator, so you can effectively only attack yourself.

It is difficult to forge an MSDP traceroute packet if you have not compromised the routing system, since MSDP traceroute packets are only accepted from configured MSDP peers on existing TCP connections.

6. References

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

[2] Fenner, W. and D. Meyer, Editors, "Multicast Source Discovery Protocol (MSDP)", draft-ietf-msdp-spec-10.txt, May, 2001.

[3] Fenner, W. and S. Casner, "A "traceroute" facility for IP Multicast", draft-ietf-idmr-traceroute-ipm-06.txt, March, 2000.

7. Author's Addresses

William C. Fenner
AT&T Labs - Research
75 Willow Rd
Menlo Park, CA 94025
Phone: +1 650 330 7893
Email: fenner@research.att.com

David Meyer
Sprint E|Solutions
12502 Sunrise Valley Dr
Reston VA, 20191
dmm@sprint.net

Ronald G. Roberts
Stanford University
241 Panama St
Pine Hall Room 175
Stanford, CA 94305-4122
Phone: +1 650 723 3352
Email: rgr@stanford.edu