A ‘traceroute’ facility for IP multicast

by Van Jacobson
(with help from Steve Casner,
Steve Deering, Dino Farinacci
and Deb Agrawal)

30th IETF
Toronto, Canada
25 July 1994

Goals

- Want to be able to trace path taken from some source to some destination.
- Want to isolate packet loss problems (e.g., congestion).
- Want to isolate configuration problems (e.g., TTL threshold).
- Want to minimize packets sent.

The way multicast works, tracing path from source to destination is hard (have to flood query over entire distribution tree). Since multicast uses reverse path forwarding, tracing from destination back to source is easy (each router must know previous hop for any given source) and traffic goes to only routers on path from source to destination.

Basic Idea

- Interested party sends traceroute request packet to last hop multicast router (the leaf router for the given destination address).

- Last hop router builds reply packet, fills in its report, then unicast sends packet to router it believes is previous hop for packets that originate at the given source.

- Each hop adds its report to end of reply packet then unicast forwards to its previous hop.

- First hop (router that believes packets from source would originate on its directly connected net) sends completed response to response destination address.

IGMP Header for Request and Response

<table>
<tr><td>version</td><td>type</td><td># reports wanted</td><td>checksum</td></tr>
<tr><td colspan="4">Multicast Group Address</td></tr>
<tr><td colspan="4">Source Address</td></tr>
<tr><td colspan="4">Destination Address</td></tr>
<tr><td colspan="4">Request ID</td></tr>
<tr><td colspan="4">Response Destination Address</td></tr>
</table>

Per-report data in Response

<table>
<tr><td colspan="5">query arrival time</td></tr>
<tr><td colspan="5">Incoming Interface Address</td></tr>
<tr><td colspan="5">Outgoing Interface Address</td></tr>
<tr><td colspan="5">Total incoming packets for group on iif</td></tr>
<tr><td colspan="5">Total outgoing packets for group on oif</td></tr>
<tr><td>routing protocol</td><td>ttl needed to forward</td><td></td><td>source mask</td><td>forwarding err code</td></tr>
</table>

0 – no err
1 – wrong if
2 – pruned
3 – scope boundary

Problems

1. May not know last hop router.

2. May not be able to unicast to last hop router.

3. May not be able to receive unicast from first hop router.

4. Intermediate router may be broken.

To solve (1) and (2), allow request to be multicast (but only correct last hop router for given group, source and dest can initiate reply).

To solve (3), allow reply to be multicast by first hop router.

To solve (4), let requestor set # reports wanted field to solicit reply from intermediate router.

Problem Diagnosis

- Can find routing inconsistencies from forwarding err code.
- Can find admin. boundary problems from forwarding err code.
- Can find threshold problems by summing TTL-to-forward from source to dest and comparing to TTL source should be using.

Problem Diagnosis (cont.)

- Can find congested links by doing two traces then look at incoming packets difference compared to previous hop's outgoing packet difference. (But note that 'incoming interface' does not necessarily correspond to 'from previous hop' if incoming is a multiaccess link).

- Can estimate per-hop link utilizations by doing two traces then looking at ratio of incoming or outgoing packets difference to timestamp difference.

- If routers are NTP synced, can estimate prop and queuing time from timestamp differences.

Support needed in router

- Need a *Groups* $\times$ *Interfaces* 2d array of counters to record outgoing packets per group.

- Need a *Groups* $\times$ *Interfaces* 2d array of counters to record incoming packets per group.