

# Use and Limits of Traceroute

*our favorite diagnostic tool that often isn't*

Caren Litvanyi  
lead network engineer  
GRNOC

22 July 2009  
Joint Techs – Indianapolis, IN

Credits: A presentation was given at NANOG45:

**“A Practical Guide to (Correctly)  
Troubleshooting with Traceroute”**

by Richard A Steenbergen <ras@nlayer.net>  
CTO/co-founder, nLayer Communications, Inc.



Much of this material is from his presentation.

[http://www.nanog.org/meetings/nanog45/presentations/Sunday/RAS\\_traceroute\\_N45.pdf](http://www.nanog.org/meetings/nanog45/presentations/Sunday/RAS_traceroute_N45.pdf)

## **For suspected network problems, the number one go-to tool for the connected public is “traceroute”.**

- Every OS comes with a traceroute tool of some kind.
- There are thousands of websites which can run a traceroute.
- There are dozens of “visual traceroute” tools available, both commercially and free, and traceroute/ping hybrids (MTR, etc.).
- Google almost anything about network troubleshooting, and traceroute is advocated as the tool for the job.
- And it seems like such a simple tool to use:

**Type in the target IP address and it shows some routers.  
Where the traceroute stops, or where the latency goes up a lot,  
that’s where the problem is, right?**

*Well, that’s what tons and tons of references on the web would have you believe!*

# So what's wrong with traceroute?

**Most core networks are actually pretty well run on decent gear.**

- Simple issues like long-term congestion or routing loops are becoming a smaller percentage of the total network issues encountered.
- Almost all routers use hardware-based forwarding of through traffic.
- Some “problems” users believe traceroutes show are not problems, but engineering.
  - changing paths (load balancing)
  - firewalls/filters/rate-limiting (security)
  - MPLS paths (traffic engineering)
- The issues involved are often complex enough to render a traceroute almost useless.

# So what's wrong with traceroute?

**Surprisingly few people are skilled at using/interpreting traceroute.**

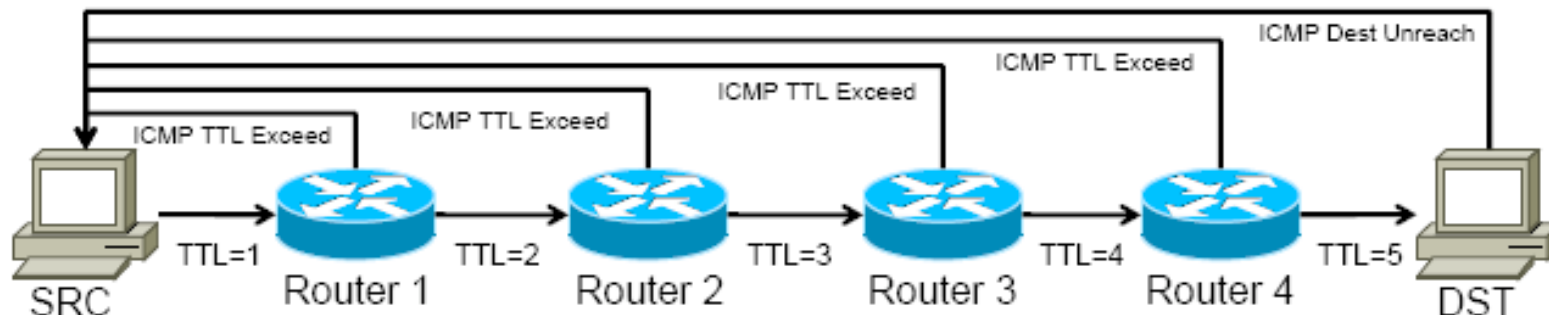
- Most local network administrators are adamant that what they believe the traceroute shows them is “the truth”. (See PingPlotter conspiracy theorists...)
- Many ISP NOCs and even many mid-level engineering staff are either not able to correctly interpret complex traceroutes, or are willing to grasp at it if other issues are not readily apparent.
- This leads many misdiagnosed issues and “false reports”, which flood the NOCs of networks worldwide, including ours.
- Because the problem of false reports is so common, it is all but impossible to submit a traceroute related ticket about a real issue and have it taken seriously at some commercial NOCs.

# What we will cover about Traceroute

- Refresher on how traceroute works
- Understanding network latency
- Prioritization and rate-limiting
- Asymmetric paths
- Multiple paths
- MPLS and traceroute
- Review/recommendations

## Traceroute – the basics

1. Launch a probe packet towards DST, with a TTL of 1.
2. Each router hop decrements the TTL of the packet by 1.
3. When TTL hits 0, router returns ICMP TTL Exceeded.
4. SRC host receives this ICMP, displays a traceroute “hop”.
5. Repeat from step 1, with TTL incremented by 1, until...
6. DST host receives probe, returns ICMP Destination Unreachable. Traceroute is completed!



## Traceroute – additional details

Most traceroute implementations send multiple probes.

- The default is 3 probes per TTL increment (“hop”).
- Hence the normal 3 latency results, or 3 \*’s if no response.

Each probe uses a different DST Port to distinguish itself.

- **So any layer 4 hashing can send each probe on different paths.**
- This may be visible to traceroute in the case of ECMP hashing.
- Or invisible, in the case of 802.3ad style Layer 2 aggregation.
- But the result is the same, some probes may behave differently.

Not all traceroute implementations use UDP.

- Windows uses ICMP, other tools may even use TCP.

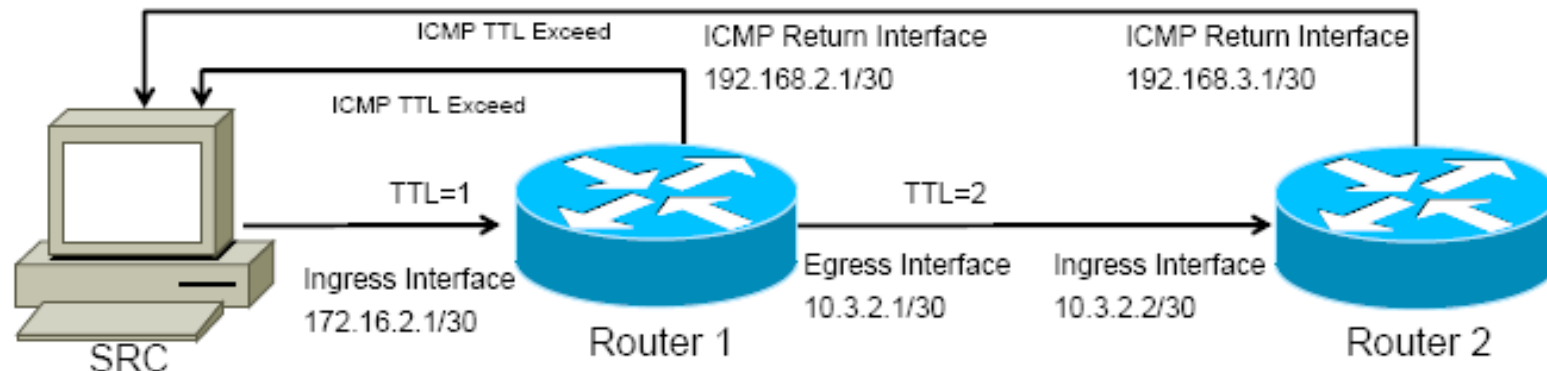
Traceroute is currently defined in RFC 1812.



# Traceroute – latency calculation

1. Timestamp when the probe packet is launched.
  2. Timestamp when the reply ICMP is received.
  3. Subtract the difference to determine round-trip value.
  4. Display this as the probe latency.
- Routers along the path do not do any time “processing”
    - They simply reflect the original packet’s data back to the SRC.
    - Many implementations encode the original launch timestamp into the probe packet, to increase accuracy and reduce state.
  - **But remember, only the ROUND-TRIP time is measured/ reported!**
  - Traceroute’s display shows you the hops on the *forward* path.
  - But showing you latency based on the **forward PLUS reverse** paths. Any delays on the reverse path *will* affect your results!

# Traceroute – what hops are you seeing, exactly?



- Packet with TTL 1 enters router via ingress interface.
- ICMP TTL Exceeded is generated as the TTL hits 0.
  - ICMP source address is that of the ingress router interface.
  - This is how traceroute sees the address of a “hop”, **the ingress IP**.
  - The above traceroute will read:  
172.16.2.1  
10.3.2.2

# Understanding Network Latency

Three primary types of network induced latency:

- Queuing Delay
  - The time spent in a router's queues waiting for transmission. This is mostly related to line contention (full interfaces), since without congestion, there is very little need for a measurable queue.
- Propagation Delay
  - The time spent "in flight", in which the signal is traveling over the transmission medium. This is primarily a limitation based on the speed of light, or other electromagnetic propagation.
- Serialization Delay
  - The delay caused by having to transmit data through routers/switches in packet sized chunks of 0's and 1's.

# Latency – Queuing Delay

- First you must understand “Utilization”:
  - A 1GE doing 500Mbps is said to be “50% utilized”.
  - But in reality, an interface is either transmitting (100% utilized) or not transmitting (0% utilized) at any instant.
  - The above is really “used 50% of the time, over, say, 1 second”.
- Queuing is a natural function of routers:
  - When a packet is ready to send but the interface is in use, it must be queued until the interface is free.
  - As an interface reaches saturation, the probability of a packet being queued rises exponentially.
  - When an interface is extremely full, a packet may be queued for many tens or hundreds of milliseconds.
  - Note in modern hardware-based forwarding routers, queuing delay is only because the interface is in use, not because it is “busy” with other tasks.

# Latency – Propagation Delay

- Delay caused by signal propagation over distance.
  - Light travels through a vacuum at  $\sim 300,000\text{km/sec}$
  - Fiber cores have a refractive index of  $\sim 1.48$
  - $1/1.48 = \sim 0.67c$ , light through fiber =  $\sim 200,000\text{km/sec}$
  - $200,000\text{km/sec} = 200\text{km (or 125 miles) per millisecond}$ .
  - Divide/multiply by 2 for round-trip time (RTT) measurements.

- Example:

A round-trip around the world at the equator, via a perfectly straight fiber route, would take  $\sim 400\text{ms}$  due solely to speed-of-light propagation delays.

# Latency – Serialization Delay

- Delay caused by packet-based forwarding
  - Packets move through the network as a single unit.
  - Can't transmit the next packet until last one is finished.
- Not much of an issue in modern networks

Speeds have increased by orders of magnitude over the years, while packet sizes have stayed the same (small).

- 1500 bytes over a 56k link (56Kbps) = 214.2ms delay
- 1500 bytes over a T1 (1.536Mbps) = 7.8ms delay
- 1500 bytes over a GigE (1Gbps) = 0.012ms delay
- 64 bytes over a GigE (1Gbps) = 0.0005ms delay
- 1500 bytes over a 10GigE (10Gbps) = 0.0012ms delay
- 9000 bytes over a 10GigE (10Gbps) = 0.0072ms delay

# Reasonable Latency

- Try to determine geographical information.
- See if the latency just about matches propagation delay.
- (Assume some percentage for fiber routing).

- For example:

```
3 xe-3-0-0.cr1.nyc3.us.nlayer.net (69.22.142.74) 6.570ms
4 xe-0-0-0.cr1.lhr1.uk.nlayer.net (69.22.142.10) 74.144ms
```

**New York NY to London UK in 67.6ms? 4200 miles? Yup!**

- Another example:

```
5 cr2.wswdc.ip.att.net (12.122.3.38) [MPLS: Label 17221 Exp 0] 8 msec 8 msec 8 msec
6 tbr2.wswdc.ip.att.net (12.122.16.102) [MPLS: Label 32760 Exp 0] 8 msec 8 msec 8 msec
7 ggr3.wswdc.ip.att.net (12.122.80.69) 8 msec 8 msec 8 msec
8 192.205.34.106 [AS 7018] 228 msec 228 msec 228 msec
9 tel-4.mpd01.iad01.atlas.cogentco.com (154.54.3.222) [AS 174] 228 msec 228 msec 228 msec
```

**Washington DC to Washington DC in 220ms? Nope!**

# Latency: Prioritization and Rate-limiting

Architecture of a modern router (“through it” vs. “to it”):

- Packets forwarded through the router (data plane):
  - **Fast Path:** hardware based forwarding of ordinary packets  
**Example:** Almost every packet in normal Internet traffic.
  - **Slow Path:** software based handling of “exception” packets.  
**Example:** IP Options, ICMP generation (including TTL Exceeded).
- Packets being forwarded TO the router (control plane):  
Example: BGP, IGP, SNMP, CLI access (telnet/ssh), ping, or any packets sent directly to a local IP address on the router. Routing protocols are a priority.
- These CPUs tend to be relatively underpowered.
  - A 320-640+ Gbps router may only have a 600MHz CPU.
  - **ICMP Generation is \*NOT\* a priority for any router.**



# Latency: Prioritization and Rate-limiting

On many platforms the slow-path data plane and the control-plane share the same resources.

- And often don't have the best schedulers for the CPU.
- As a result, control-plane activity such as BGP churn, SNMP walks, CLI use, and periodic software processes can consume CPU and slow the generation of ICMP TTL Exceeds.
- This results in random “spikes” in traceroute latency, which is often misinterpreted as a network issue.
- The most infamous process which causes these spikes is called “BGP Scanner”, and runs every 60 seconds on all Cisco IOS devices.
- The most common for Internet2 seems to be SNMP querying.

## Rate-limited ICMP Generation

Most routers also rate limit their ICMP generation

- Often with arbitrary hard-coded limits.
- Which may be insufficient under heavy ICMP load.
- This comes into play more on fast-pinging, but both at once can also mess up a traceroute.

Juniper

- Hard limit of 50pps per interface, 250pps on FPC3s
- Hard limit of 500pps per PFE as of JUNOS 8.3+

Foundry/Brocade

- Hard limit of 400pps per interface

Force10

- Hard limit of 200pps or 600pps per interface

**Additionally, networks may configure to limit ICMPs even further.**

# Rate-limited ICMP Generation

```
litvanyi@CHIC-re1> show pfe statistics ip icmp
```

## ICMP Statistics:

```
271315023 requests
  5293837 network unreachables
160358076 ttl expired
    0 ttl captured
    0 redirects
    80 mtu exceeded
    0 icmp/option handoffs
```

## ICMP Errors:

```
    0 unknown unreachables
    0 unsupported ICMP type
    0 unprocessed redirects
    0 invalid ICMP type
    0 invalid protocol
    0 bad input interface
989946 throttled icmps
    0 runts
```

# Traceroute - Spotting The Fake Latency

The most important rule of all – understanding this alone will put you ahead of 99% of the people out there capable of running a traceroute:

- If there is an actual issue somewhere, the latency will continue or increase for all future hops.
- Latency spikes in the middle of a traceroute mean *absolutely nothing* if they do not continue forward.
- Really, I mean it! Latency spikes in the middle of a traceroute mean *absolutely nothing* if they do not continue forward.
  - At worst it could be the result of an asymmetric path.
  - But it is probably an artificial rate-limit or CPU prioritization issue.
  - *By definition, if regularly forwarded packets are being affected you should see the issue persist on all future hops.*

# Traceroute - Spotting The Fake Latency

## Quiz time!

This is from my home. My network seems to be “slow”, particularly when I try to work. Should I open a ticket with my cable provider complaining about hop 4 in Greater Victoria? How about the second traceroute? Is there a problem in Vancouver?

```
2  24.69.80.1 (24.69.80.1)  19.889 ms  11.057 ms  12.911 ms
3  rd1cv-ge4-2-2.gv.shawcable.net (64.59.166.146)  11.434 ms  9.872 ms  9.520 ms
4  rd2cv-ge1-0-0.gv.shawcable.net (66.163.72.6)  191.892 ms  167.192 ms  129.870 ms
5  rc1wt-pos3-0-0.wa.shawcable.net (66.163.77.182)  13.962 ms  24.222 ms  11.963 ms
6  internet2-six.net.internet2.edu (198.32.180.67)  13.915 ms  25.740 ms  23.969 ms
```

traceroute to www.uva.nl (145.18.10.70), 64 hops max, 40 byte packets

```
2  24.69.80.1 (24.69.80.1)  13.229 ms  13.442 ms  10.133 ms
3  rd1cv-ge4-2-2.gv.shawcable.net (64.59.166.146)  13.821 ms  11.224 ms  12.577 ms
4  rc1bb-pos4-0-0.vc.shawcable.net (66.163.77.185)  14.116 ms  11.717 ms  57.263 ms
5  rc2wh-tge0-7-1-0.vc.shawcable.net (66.163.69.73)  82.143 ms  *  121.419 ms
6  rc1so-pos10-0-0.cg.shawcable.net (66.163.76.185)  34.387 ms  28.685 ms  29.224 ms
7  66.163.77.30 (66.163.77.30)  45.291 ms  43.585 ms  44.292 ms
8  66.163.77.133 (66.163.77.133)  64.998 ms  64.660 ms  74.522 ms
9  XSR03.Asd001A.surf.net (195.69.144.50)  176.402 ms  174.174 ms  175.356 ms
10 ae2.500.jnr01.asd001a.surf.net (145.145.80.78)  193.424 ms  175.480 ms  175.484 ms
11 UvA-CPE1.amsterdam1.surf.net (145.145.19.130)  175.155 ms  172.488 ms  172.715 ms
```

# Traceroute - Spotting The Fake Latency

Quiz time! How about this one?

```
tel-1.ar2.DCA3.gblx.net (69.31.31.209) 0.719 ms 0.560 ms 0.428 ms
tel-2-10g.ar3.DCA3.gblx.net (67.17.108.146) 0.574 ms 0.557 ms 0.576 ms
sl-st21-ash-8-0-0.sprintlink.net (144.232.18.65) 100.280 ms 100.265 ms 100.282 ms
144.232.20.149 (144.232.20.149) 102.037 ms 101.876 ms 101.892 ms
sl-bb20-dc-15-0-0.sprintlink.net (144.232.15.0) 101.888 ms 101.876 ms 101.890 ms
```

Say GBLX is one of my transit providers. Should I open a ticket complaining that their circuit to Sprint in DC is congested, and they had better fix it? When they tell me it's not congested, should I escalate and threaten to not renew our contract with them?

Not so fast!

Yes, the latency does "continue" on.

BUT, there are other possible issues you should investigate first.

So far, we see a "possible" issue, but *we don't know what or where it is.*

# Asymmetric Paths and Network Boundaries

Asymmetric paths often start at network boundaries.

Why? Because that is where administrative policies change.

Here's the previous GBLX/Sprint traceroute:

```
tel1-1.ar2.DCA3.gblx.net (69.31.31.209) 0.719 ms 0.560 ms 0.428 ms
tel1-2-10g.ar3.DCA3.gblx.net (67.17.108.146) 0.574 ms 0.557 ms 0.576 ms
sl-st21-ash-8-0-0.sprintlink.net (144.232.18.65) 100.280 ms 100.265 ms 100.282 ms
144.232.20.149 (144.232.20.149) 102.037 ms 101.876 ms 101.892 ms
sl-bb20-dc-15-0-0.sprintlink.net (144.232.15.0) 101.888 ms 101.876 ms 101.890 ms
```

So what can we say about the possible problem in the path above?

- It COULD be congestion between GBLX and Sprint.
- But it could also be an asymmetric reverse path.
- At this GBLX/Sprint boundary, the reverse path policy changes.
- This is often seen in multi-homed networks with multiple paths.
- In the example above, Sprint's reverse route goes via a circuit that is congested, but that circuit is NOT shown in the traceroute.

# Asymmetric Paths and Network Boundaries

How can you troubleshoot around asymmetric paths?

- The most powerful option is to control your SRC address.
- In the previous example, assume that:
  - You are multi-homed to Global Crossing and Level3.
  - Global Crossing reaches you via (of course) Global Crossing.
  - Sprint reaches you via Level3.
  - There is a problem between Sprint and Level3.
- How can you prove the issue isn't between GX and Sprint?
  - Run a traceroute using your side of the GBLX /30 as your source.
  - This /30 comes from your provider (GBLX)'s larger aggregate.
  - The reverse path will be guaranteed to go Sprint->GBLX.
  - If the latency doesn't persist, you know the issue is on the reverse.
- The point is, think about and investigate the reverse path before jumping to conclusions.



# Asymmetric Paths and SRC Addresses

But what if the /30 is numbered out of my space?

- As in the case of a customer or possibly a peer.

You may still see some benefits from setting SRCs.

- Consider trying to examine the reverse path of a peer with whom you have multiple interconnection points.
  - A traceroute sourced from your IP space (such as loopback) may come back via any of multiple interconnection points.
  - But if the remote network carries the /30s of your interconnection in their IGP (i.e. they redistribute connected into their IGP)...
  - Then the traffic will come back over their backbone, and return to you via the /30 you are testing from.
  - Trying both options can give you different viewpoints.

Other than that, try to find traceroute servers in the reverse direction. Or admit you don't know.

# What SRC Are You Using, Anyway?

## When tracerouting from a router...

- Most routers default to using the source address of the egress interface that the probe leaves from.
- This may or may not be what you want to see.
- Some platforms can be configured to default to a loopback address rather than the egress interface.
  - For example, Juniper
- My recommendation is to ALWAYS SPECIFY your SRC IP address, and make sure that is included in any traceroute sent to anyone.

## What SRC Are You Using, Anyway?

- Pet peeve: people that don't include the SRC address they are tracing FROM in a traceroute, so I can't trace back to it correctly, or look for routing information about it.
- This is especially annoying from routers or multi-homed hosts.

```
litvanyi@SEAT-rel> traceroute routing-instance cps 24.69.80.1
traceroute to 24.69.80.1 (24.69.80.1), 30 hops max, 40 byte packets
 1  rclwt-ge4-1.wa.shawcable.net (198.32.180.54)  1.460 ms  0.617 ms  0.804 ms
 2  rd2cv-pos2-0-0.gv.shawcable.net (66.163.77.181)  2.750 ms  2.844 ms  2.819 ms
 3  * * *
```

- If I send this to Shaw Cable complaining about something, how are they supposed to know where I am coming from?
- **YOU** do the right thing, and make sure your src address is clear if you send a traceroute to anyone.

## Other Confusions – Multiple Paths

- Because each (UDP/TCP) traceroute probe uses a different layer 4 port, equal-cost multi-path may make multiple paths show up within a single “hop”.
- This is relatively harmless, but may be confusing.
- Example:

```
6 ldn-bb2-link.telial.net (80.91.251.14) 74.139 ms 74.126 ms
  ldn-bb1-link.telial.net (80.91.249.77) 74.144 ms
7 hbg-bb1-link.telial.net (80.91.249.11) 89.773 ms
  hbg-bb2-link.telial.net (80.91.250.150) 88.459 ms 88.456 ms
8 s-bb2-link.telial.net (80.91.249.13) 105.002 ms
  s-bb2-link.telial.net (80.239.147.169) 102.647 ms 102.501 ms
```

- Of the 3 probes, 2 go over one path, 1 goes over another.

## Other Confusions – Multiple Paths

A slightly more complex example:

```
4 p16-1-0-0.r21.asbnva01.us.bb.verio.net (129.250.5.21) 0.571 ms 0.604 ms 0.594 ms
5 p16-1-2-2.r21.nycmny01.us.bb.verio.net (129.250.4.26) 7.279 ms 7.260 ms
p16-4-0-0.r00.chcgil06.us.bb.verio.net (129.250.5.102) 25.981 ms
6 p16-2-0-0.r21.sttlwa01.us.bb.verio.net (129.250.2.180) 71.027 ms
p16-1-1-3.r20.sttlwa01.us.bb.verio.net (129.250.2.6) 66.730 ms 66.535 ms
```

- ECMP between two parallel but different paths
  - Ashburn VA – New York NY – Seattle WA
  - Ashburn VA – Chicago IL – Seattle WA
  - Also harmless, but potentially confusing.
- A much worse scenario is ECMP where the load-balanced paths are of unequal hop length.
- This can make the traceroute appear to go back and forth, and is extremely confusing and difficult to read.

## Handling Multiple Paths

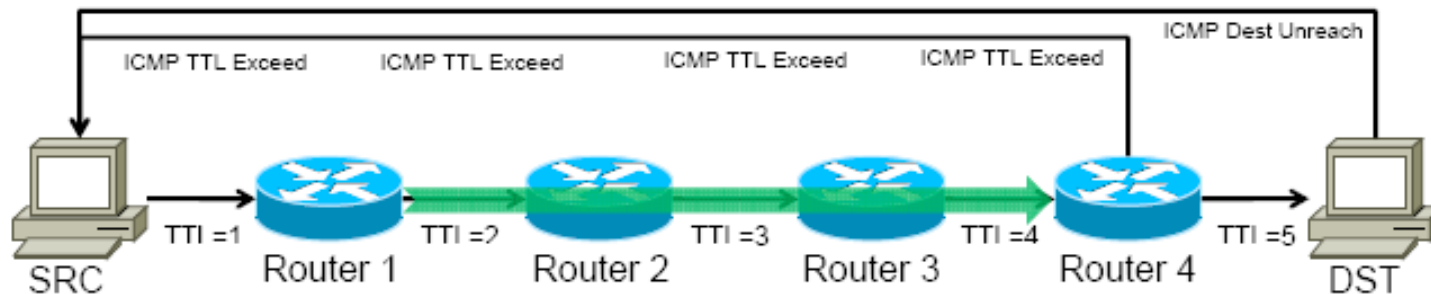
- **When in doubt, only look at a single path!**
- Set your traceroute client to only send a single probe.
- But be aware that this may not be the path which your actual traffic forwards over.
- One method is to try out different paths which may be available by incrementing the dest IP by 1 (resulting in a different hashing) or try different source IPs.

# MPLS and Traceroute

- Many large networks operate an MPLS based core.
  - This presents a problem when ICMPs need to be generated.
- How does the MPLS-only hop/router deliver an ICMP?
- One solution is called ICMP Tunneling:
  - If generating an ICMP about a packet inside an LSP...
  - Then put the generated ICMP back into the same LSP.
  - Works for delivering the message, but...
  - It can make traceroutes look really weird, and like there is a congestion issue somewhere.

# MPLS and Traceroute

- All returned ICMP packets must travel to the end of the LSP before going back to the sender. This makes every hop in the LSP appear to have approximately the same RTT as the final hop.



```
2 loop0.lns3.adl4.internode.on.net [203.16.215.xxx] 38 ms 39 ms 39 ms
3 vl13.cor1.adl4.internode.on.net [203.16.213.172] 38 ms 38 ms 38 ms
4 gi0-0-0.bdr2.adl5.internode.on.net [203.16.215.33] 219 ms 219 ms 219 ms
5 pos3-1.bdr1.syd7.internode.on.net [203.16.212.178] 218 ms 218 ms 218 ms
6 pos1-3.bdr1.sjc2.agile.on.net [203.16.213.37] 222 ms 220 ms 223 ms
7 eqixsj-google-gige.google.com [206.223.116.21] 218 ms 219 ms 219 ms
```

Is there a problem with congestion between hops 3 and 4?

No, this is just an MPLS LSP through on.net from Australia to San Jose.



# MPLS and Traceroute

- Known Linux bug (Redhat?):

```
[litvanyi@foo ~]$ traceroute 24.69.80.1
traceroute to 24.69.80.1 (24.69.80.1), 30 hops max, 38 byte packets
 1  192.12.206.193 (192.12.206.193)  0.332 ms  0.188 ms  0.233 ms
 2  * * *
 3  ge-0-1-0.111.rtr.atla.net.internet2.edu (149.165.254.23)  16.375 ms  16.682 ms  20.724 ms
Icmp checksum is wrong
 4  so-3-2-0.0.rtr.hous.net.internet2.edu (64.57.28.43)  39.085 ms Icmp checksum is wrong
   39.071 ms Icmp checksum is wrong
   39.081 ms
Icmp checksum is wrong
 5  so-3-0-0.0.rtr.losa.net.internet2.edu (64.57.28.44)  74.305 ms Icmp checksum is wrong
   74.807 ms Icmp checksum is wrong
   71.063 ms
 6  198.32.176.46 (198.32.176.46)  81.053 ms  81.432 ms  80.944 ms
 7  rc2wh-pos0-15-2-0.vc.shawcable.net (66.163.77.109)  112.517 ms  112.661 ms  112.800 ms
 8  rc2bb-tge0-15-1-0.vc.shawcable.net (66.163.69.122)  91.933 ms  92.416 ms  91.563 ms
 9  66.163.76.230 (66.163.76.230)  91.802 ms  91.802 ms  91.688 ms
```

Is something in Internet2 CPS horribly corrupting packets? – NO.

# MPLS and Traceroute

- This can be worked around by using larger packet sizes:

```
[litvanyi@jump ~]$ traceroute 24.69.80.1 1500
traceroute to 24.69.80.1 (24.69.80.1), 30 hops max, 1500 byte packets
 1  192.12.206.193 (192.12.206.193)  0.409 ms  0.318 ms  0.361 ms
 2  * * *
 3  ge-0-1-0.111.rtr.atla.net.internet2.edu (149.165.254.23)  36.076 ms  29.671 ms  34.340 ms
 4  so-3-2-0.0.rtr.hous.net.internet2.edu (64.57.28.43)  39.440 ms  39.451 ms  39.468 ms
    MPLS Label=287712 CoS=6 TTL=1 S=0
    MPLS Label=289072 CoS=0 TTL=1 S=0
 5  so-3-0-0.0.rtr.losa.net.internet2.edu (64.57.28.44)  71.552 ms  71.683 ms  72.196 ms
    MPLS Label=289072 CoS=6 TTL=1 S=0
 6  198.32.176.46 (198.32.176.46)  80.930 ms  81.184 ms  81.068 ms
 7  rc2wh-pos0-15-2-0.vc.shawcable.net (66.163.77.109)  112.298 ms  112.911 ms  112.424 ms
 8  rc2bb-tge0-15-1-0.vc.shawcable.net (66.163.69.122)  91.936 ms  92.174 ms  91.812 ms
 9  66.163.76.230 (66.163.76.230)  91.937 ms  92.547 ms  91.812 ms
```

<http://rhn.redhat.com/errata/RHBA-2008-0883.html>

Routers using Multiprotocol Label Switching (MPLS) send certain MPLS data in ICMP packets. The traceroute utility incorrectly rendered this MPLS data, resulting in output such as "Icmp checksum is wrong". In this updated package, traceroute renders MPLS data in accordance with RFC 4950 (ICMP Extensions for Multiprotocol Label Switching), resulting in the correct output for MPLS data, as well as making traceroute RFC 4950 compliant.

# MPLS and Traceroute, Internet2 CPS

This is from my home. I am a Shaw Cable customer.

```
pellet:~ litvanyi$ traceroute www.grnoc.iu.edu
traceroute to globalnoc.iu.edu (134.68.107.125), 64 hops max, 40 byte packets
 1  192.168.0.1 (192.168.0.1)  1.409 ms  0.930 ms  0.273 ms
 2  24.69.80.1 (24.69.80.1)  16.674 ms  9.312 ms  7.879 ms
 3  rd1cv-ge4-2-2.gv.shawcable.net (64.59.166.146)  9.664 ms  33.145 ms  7.462 ms
 4  rd2cv-ge1-0-0.gv.shawcable.net (66.163.72.6)  44.451 ms  11.729 ms  23.752 ms
 5  rc1wt-pos3-0-0.wa.shawcable.net (66.163.77.182)  31.850 ms  63.270 ms  13.448 ms
 6  internet2-six.net.internet2.edu (198.32.180.67)  16.722 ms  15.068 ms  24.315 ms
 7  * * *
 8  * * *
 9  * * *
10  so-1-0-0.0.rtr.atla.net.internet2.edu (64.57.28.4)  111.630 ms  101.314 ms  103.335 ms
11  ge-0-3-0.111.rtr.ll.indiana.gigapop.net (149.165.254.22)  130.086 ms  113.738 ms  126.691 ms
```

But... everything but the LSP ends in Internet2 shows stars!

What's going on here?

Uses master instance table to return ttl exceeded.

# MPLS and Traceroute, Internet2 CPS

This is from a GRNOC desktop (same OS), tracing back to my home.

```
bar:~ user$ traceroute 24.69.80.1
traceroute to 24.69.80.1 (24.69.80.1), 64 hops max, 40 byte packets
 1  ge-2-3.11.cr.ictc.net.uits.iu.edu (134.68.11.254)  59.072 ms  4.516 ms  0.844 ms
 2  tge-1-1.912.br.ul.net.uits.iu.edu (149.166.5.7)  0.542 ms  0.509 ms  0.490 ms
 3  149.165.183.17 (149.165.183.17)  0.472 ms  0.493 ms  0.468 ms
 4  * * *
 5  ge-0-1-0.111.rtr.atla.net.internet2.edu (149.165.254.23)  24.056 ms  25.871 ms  34.774 ms
 6  so-3-2-0.0.rtr.hous.net.internet2.edu (64.57.28.43)  39.506 ms  39.394 ms  39.460 ms
 7  so-3-0-0.0.rtr.losa.net.internet2.edu (64.57.28.44)  84.897 ms  71.511 ms  71.412 ms
 8  198.32.176.46 (198.32.176.46)  81.183 ms  81.180 ms  81.283 ms
 9  rc2wh-pos0-7-2-0.vc.shawcable.net (66.163.76.65)  102.855 ms  103.016 ms  102.421 ms
10  rc2bb-tge0-0-1-0.vc.shawcable.net (66.163.69.66)  96.632 ms  96.993 ms  98.000 ms
11  rd1cv-pos2-0-0.gv.shawcable.net (66.163.77.186)  96.594 ms  96.713 ms  96.616 ms
```

This one shows up fine because the source is connected to both the R&E and commodity instances.

## Other Traceroute Tidbits

- So, if it's not great at debugging latency, what can we say traceroute *\*is\** good for? -- Mostly just forward path discovery, and it's not perfect at that, either.
- Traceroute is not a good tool for determining multicast paths, but if you are going to use it at all, it is only even possibly useful in the direction from Receiver → Source. Presenting a traceroute from the source towards a receiver to attempt to show the path of a multicast flow shows a fundamental misunderstanding of multicast.
- On Internet2 CPS, we need to traceroute within the routing-instance:  
`traceroute routing-instance cps source 64.57.29.247 24.69.80.1`
- Common codes:

asterisk \* - no response.  
!H - No route to host.  
!N - No route to network.  
!P - Invalid protocol.

!X - Route administratively blocked.  
!S - Source route failed.  
!F - Fragmentation needed.  
!# - ICMP error.

**Questions?**

**Thanks!**