A Practical Guide to (Correctly) Troubleshooting with Traceroute

By: Richard A Steenbergen <ras@petabitscale.com>

Last Updated: October 12, 2020

Introduction

Troubleshooting problems on the Internet?

- The number one go-to tool is “traceroute”
  - Every OS comes with a traceroute tool of some form.
  - It’s available from thousands of websites too.
  - And there are many “visual traceroute” tools available.
- It seems like such a simple tool to use…
  - Type in an IP address, and it shows you every router hop along the way, along with a latency measurement.
  - Where the traceroute stops, or where the latency jumps up a lot, that’s where the problem is, right?
  - How could that possibly go wrong?

The Trouble with Traceroute

- Modern networks are actually pretty well run.
  - Simple issues like congestion or routing loops are rare.
- Few people are skilled at interpreting traceroute.
  - Traceroute ***looks*** relatively simple to read and use.
  - This tends to give most people the impression that they are qualified to interpret it and locate the fault.
  - In reality, most NOCs and even mid-level IP engineers may not be qualified to interpret a complex traceroute.
  - The vast number of bogus traceroute complaints out there makes it very difficult for knowledgeable people to get real issues looked at and resolved.

Example Traceroute Output

Implementations vary, but traceroute output looks a little something like this:

Hop #	Router (DNS)	Router (IP)	Latency Measurements

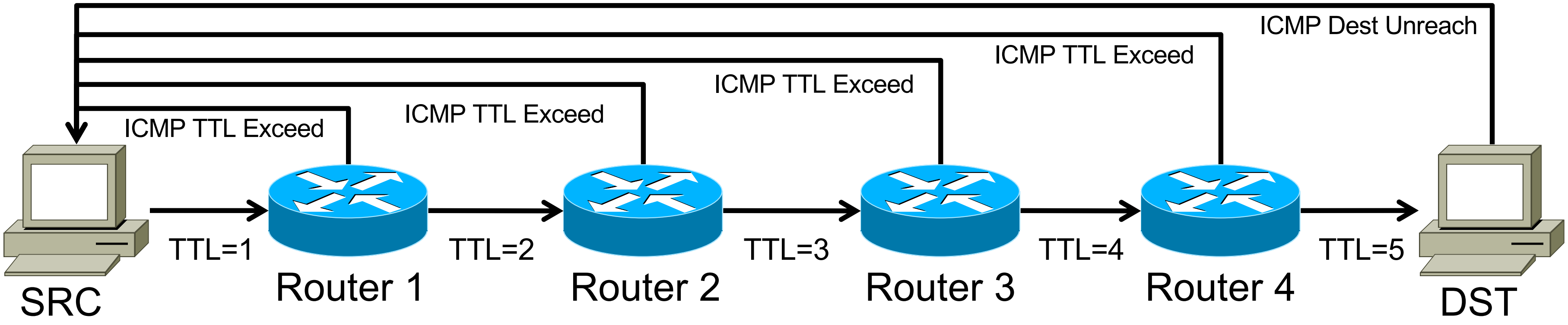
In this example, each router hop has three separate latency probes:

```
traceroute to 4.2.2.2 (4.2.2.2), 64 hops max, 52 byte packets
 1  ge0-34.aggrFZ155-2.ord6.us.scnet.net (204.93.176.73)  1.673 ms  1.187 ms  2.847 ms
 2  ge-9-2-9.ar2.ord6.us.scnet.net (75.102.0.69)  0.317 ms  0.163 ms  0.155 ms
 3  72.ae3.cr2.ord6.us.scnet.net (204.93.204.158)  0.203 ms
 4  as3549.xe-0-2-1.cr2.ord6.us.scnet.net (204.93.144.30)  0.929 ms  0.898 ms  0.893 ms
 5  ae9.503.edge3.Chicago.Level3.net (4.68.62.253)  1.005 ms  1.028 ms  1.023 ms
 6  vlan52.ebr2.Chicago2.Level3.net (4.69.138.190)  1.194 ms
 7  4.69.158.237 (4.69.158.237)  1.172 ms
 8  b.resolvers.Level3.net (4.2.2.2)  1.169 ms  1.182 ms  1.178 ms
```

Traceroute at the Packet Level

The traceroute operation is comprised of the following steps:

1. Launch a probe packet towards the DST, with an initial TTL of 1.
2. Each router that forwards the packet decrements the TTL value by 1.
3. When the TTL hits 0, the router returns an ICMP TTL Exceed to SRC.
4. SRC receives the ICMP, calculates a time difference, displays a “hop”.
5. Go back to step 1, but increment the TTL of the probe packet by 1.
6. Rinse and repeat, until DST receives probe and returns ICMP Unreach.
7. When SRC receives a ICMP Unreachable, the traceroute ends.



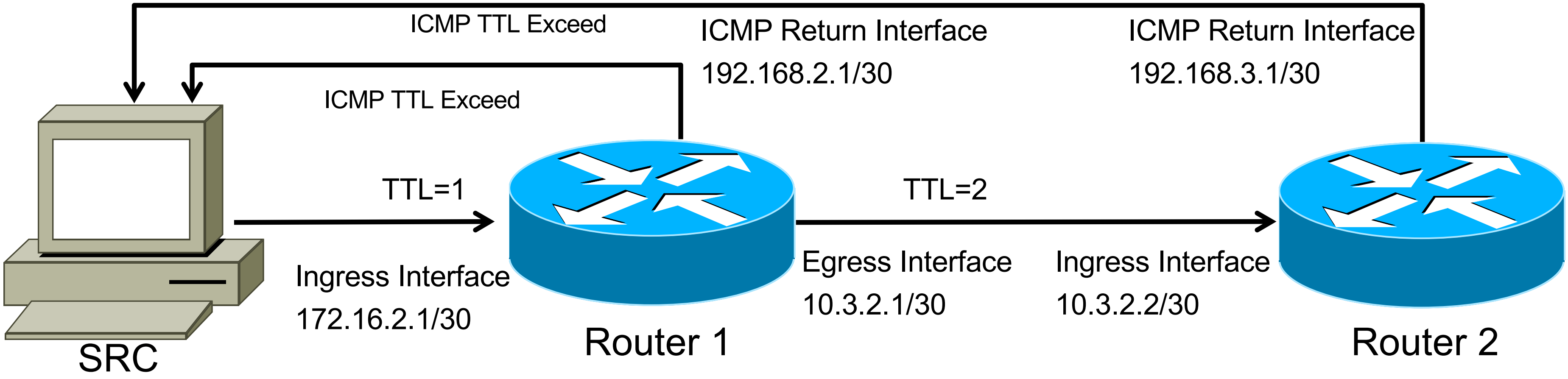
Traceroute and Multiple Probes

- Most implementations send multiple probes per hop.
  - The default for the vast majority of traceroute implementations is 3.
  - That is, 3 probe packets are launched before each TTL increment.
  - This results in 3 latency measurements per “hop”.
- Each probe packet uses a unique key to distinguish itself.
  - Most implementations use UDP packets with incrementing dest ports.
  - But ICMP or TCP can be used (e.g. Windows tracert.exe uses ICMP).
- Each probe packet is a ***completely*** independent trial.
  - Each packet MAY be forwarded down a completely different path.
  - This MAY be visible to the end-user, as multiple IP’s for each hop.
  - But it can also be completely invisible too.

Traceroute and Latency Calculation

- The traceroute “latency” calculation is very simple:
  - Timestamp when the probe packet is launched.
  - Timestamp when the return ICMP packet is received.
  - Subtract the difference to determine a round-trip time.
- Routers along the path do NOT do any time processing.
  - But they do return some of the original probe payload in the ICMP.
  - Some implementations will encode the original launch timestamp (relative to their own clocks) in the probe packet to reduce state.
- Thus the reported latency is actually the sum of:
  - The time taken to forward the packet to the displayed router hop.
  - The time taken for the router to generate an ICMP response packet.
  - The time taken for that ICMP packet to reach the sender.

Traceroute – What Hops Are You Seeing?



- Probe packet enters Router 1, TTL hits 0, packet is dropped
  - Router 1 generates an ICMP TTL Exceed message towards SRC.
  - The source address of this ICMP determines the traceroute hop IP.
- By convention, the ICMP is sourced from the ingress interface.
  - The above traceroute will read: 172.16.2.1 10.3.2.2
- Random factoid: This behavior is actually non-standard.
  - RFC1812 says the ICMP source MUST be from the egress interface.
  - If obeyed, this would prevent traceroute from working properly.

How to Interpret DNS in a Traceroute

Interpreting DNS in a Traceroute

- One does not read a Traceroute by IP alone.
  - Most operators are kind enough to include DNS info.
  - Correctly interpreting this DNS is one of the most important aspects of successfully reading a traceroute.

- Important information you can discover includes:
  - Geographic Locations
  - Interface Types and Capacities
  - Router Type and Roles
  - Network Boundaries and Relationships

Traceroute DNS Location Data

- Knowing the geographical location of the routers is an important first step to understanding an issue.
  - To identify incorrect/suboptimal routing.
  - To help you understand network interconnections.
  - And even to know when there isn't a problem at all, i.e. knowing when high latency is justified and when it isn't.
- The most commonly used location identifiers are:
  - IATA Airport Codes
  - CLLI Codes
  - Abbreviations of a city name.
  - Sometimes, you just have to guess!

Location Identifiers – IATA Airport Codes

- IATA Airport Codes
  - Provides great global coverage for large metro regions.
  - Most common in networks with fewer/larger POPs.
  - Examples:
      - Santo Domingo = SDQ
      - San Jose California = SJC
  - Sometimes represented by “Metropolitan Area” codes
    - Typically used where multiple airports service a major region.
    - Can be helpful when the individual airport code is non-intuitive.
      - New York, NY is served by JFK, LGA, and EWR airports.
        - But is covered by the code “NYC” to represent all 3.
      - Northern VA is served by IAD, Washington DC by DCA.
        - But is covered by the code “WAS” to represent both (plus BWI)

Location Identifiers – CLLI Codes

- Common Language Location Identifier
  - Full codes maintained (and sold) by Telecordia.
  - Most commonly used by North American Telcos.
    - Example: **HSTNTXMOCG0**
  - In a non-Telco role, may only use the city/state identifiers
    - Examples:
      - HSTNTX = Houston Texas
      - ASBNVA = Ashburn Virginia
  - Well defined standard covering all North American cities
    - Commonly seen in networks with a large number of POPs.
    - Not an actual standard outside of North America
      - Some providers fudge these, e.g. AMSTNL = Amsterdam NL

Location Identifiers – Arbitrary Values

- But sometimes, people just make stuff up.
  - Toronto, ON
    - IATA Airport Codes: YYZ or YTC
    - IATA Metro Code: YTO
    - CLLI Code: TOROON
    - Example Arbitrary Code: TOR
- Frequently based on the good intentions of making things readable in plain English, even though these may not follow any known standards.

Common Locations – US Major Cities

Location Name	Airport Codes	CLLI Code	Some Other Codes
Ashburn VA	IAD/DCA (WAS)	ASBNVA	WDC
Atlanta GA	ATL	ATLNGA	
Chicago IL	ORD/MDW (CHI)	CHCGIL	
Dallas TX	DFW, DAL	DLLSTX	DLS
Houston TX	IAH, HOU	HSTNTX	
Los Angeles CA	LAX	LSANCA	LA
Miami FL	MIA	MIAMFL	
Newark NJ	EWR (NYC)	NWRKNJ	NEW, NWK
New York NY	JFK, LGA (NYC)	NYCMNY	NYM
San Jose CA	SJC	SNJSCA	SJO, SV, SF
Palo Alto CA	PAO	PLALCA	PAIX, PA
Seattle CA	SEA	STTLWA	

Common Locations – Global Major Cities

Location Name	Airport Codes	CLLI Code (*)	Some Other Codes
Amsterdam NL	AMS	AMSTNL	
Frankfurt GE	FRA	FRNKGE	
Hong Kong HK	HKG	NEWTHK	HK
London UK	LHR/LGW (LON)	LONDEN	
Madrid SP	MAD	MDRDSP	
Montreal CA	YUL/YMY (YMQ)	MTRLPQ	MTL
Paris FR	CDG/ORY (PAR)	PARSFR	
Singapore SG	SIN	SNGPSI	SNG
Seoul KR	ICN/GMP (SEL)	SEOLKO	
Sydney AU	SYD	SYDNAU	
Tokyo JP	NRT/HND (TYO)	TOKYJP	TKO
Toronto CA	YYZ/YTC (YTO)	TOROON	TOR

(*) Not a real CLLI Standard

Interpreting DNS – Interface Types

- Many networks will try to put interface info in DNS
  - This may not always be up to date though.
    - Well-run networks use automatically generated DNS.
    - Others are shockingly bad at keeping their DNS updated.
  - Can potentially help you identify the type of interface
    - As well as capacity, maybe even the make/model of router.
- Example:
  - xe-11-3-0.edge1.NewYork1.Level3.net
    - XE-#/#/# is Juniper 10GE port.
    - The device has at least 12 card slots, at least 4 “PIC” slots.
    - Highly likely to be a Juniper MX960.

Common Interface Naming Conventions

Interface Type	Cisco IOS	Cisco IOS XR	Juniper
Fast Ethernet	Fa#/#		fe-#/#/#
Gigabit Ethernet	Gi#/#	Gi#/#/#/#	ge-#/#/#
10 Gigabit Ethernet	Te#/#	Te#/#/#/#	xe-#/#/#
40 Gigabit Ethernet	Fo#/#	Fo#/#/#/#	et-#/#/#
100 Gigabit Ethernet	Hu#/#	Hu#/#/#/#	et-#/#/#
SONET	Pos#/#	POS#/#/#/#	so-#/#/#
T1	Se#/#		t1-#/#/#
Ethernet Bundle	Po# / Port-channel#	BE####	ae#
SONET Bundle	PosCh#	BS####	as#
Tunnel	Tu#	TT# or TI#	ip-#/#/# or gr-#/#/#
ATM	ATM#/#	AT#/#/#/#	at-#/#/#
Vlan	Vl###	Gi#/#/#/#.###	ge-#-#-#.###

Interpreting DNS – Router Types/Roles

- Knowing the role of a router can be very useful
  - But every network is different.
  - They all have different naming convention.
  - And just to be extra confusing, they don't always follow their own rules.
- But you can usually guess the context to get a basic understanding of the roles.
  - Core routers – CR, Core, BB, CCR, BBR
  - Peering routers – BR, Border, Edge, IR, IGR, Peer
  - Customer routers – AR, Aggr, Cust, CAR, HSA, GW

Network Boundaries and Relationships

- Identifying Network Boundaries is important.
  - Tends to be where routing policy changes occur.
    - E.g. different return paths based on Local Preference.
  - These also tend to be areas where capacity and routing are the most difficult, and are likely to be problem spots.
- Identifying the relationship can be helpful too.
  - Typically: a) Transit Provider, b) Peer, or c) Customer.
  - Many networks will try to indicate demarcs in their DNS
    - Clear names like networkname.customer.alter.net
    - Or always landing customers on routers named “gw”

Network Boundaries and Relationships

- It’s easy to spot where the DNS changes
  - 4 te1-2-10g.ar3.DCA3.gblx.net (67.17.108.146)
  - 5 sl-st21-ash-8-0-0.sprintlink.net (144.232.18.65)
- Or, look for “remote party” name in the DNS
  - 4 po2-20G.ar5.DCA3.gblx.net (67.16.133.90)
  - 5 cogent-1.ar5.DCA3.gblx.net (64.212.107.90)
  - Common where one side controls the /30 DNS, and the other side doesn’t provide interface information.
- For more info, look at the other side of the /30
  - > nslookup 64.212.107.89
  - Result: te2-3-10GE.ar5.DCA3.gblx.net

Understanding Network Latency

Understanding Network Latency

There are 3 primary causes of network induced latency:

1. Serialization Delay
   - Caused by the packetization of data across the wire.
2. Queuing Delay
   - Caused when the router/switch must buffer the packet while waiting for an opportunity to transmit it.
3. Propagation Delay
   - Caused by electromagnetic propagation speeds across large distances.

Latency - Serialization Delay

- A packet must move through the network as an atomic unit.
  - i.e. You can't transmit "half a packet" and finish the rest later.
  - Nor can you start TXing a packet until you've finished RXing it.
    - Some devices can (called "cut-through switching"), but this is rare.
- "Serialization" is the process of encoding data onto the wire.
  - The "faster" an interface is, the quicker this process will occur.
    - The math is simple, delay = packet size / link speed.
    - For example, to send a 1500 byte packet (*) over a 1 Mbps link:
      - 1 Mb/s = 1000 Kb/s = 125 KB/s = 125000 bytes/sec
      - 1500 bytes / 125000 bytes/sec = 0.012 seconds = ***12 ms of delay***.
      - Remember, networking uses true SI definitions ("Kilobyte" means 1000 bits, not 1024 "KiBibytes").
- **Every time a packet-aware device (e.g. a router or switch) touches the packet, a new serialization delay occurs.**

Serialization Delay with Layer 2 Overhead

But don’t forget about Layer 2 Overhead!

- Preamble and Start of Frame Delimiter (8 bytes)
- Ethernet header (14 bytes)
- Payload (our IP packet)
- Optional Padding (Payload min 46 bytes)
- Frame Checksum (4 bytes)
- Inter-Frame Gap (12 bytes)

A 1500 byte IP packet sends 1538 bytes across the wire.

- Plus VLAN tags (4 bytes), MPLS shims (4 bytes), etc.
- Actual Serialization Delay over a 1 Mbps link:
  - 1538 bytes / 125000 bytes/sec = 0.0123 sec = 12.3ms

Serialization Delay Examples

Fortunately, this is less of an issue on high-speed networks.

- Interface speeds have increased by many orders of magnitude over the years, while packet sizes have stayed essentially fixed.
- But this can still be a problem on lower-speed interfaces.
- This also makes an interesting counter-argument to the widespread deployment of Ethernet “jumbo frames”.

Interface Speed	Packet Size	Serialization Delay
1 Mbps	1500 bytes (1538 bytes)	12.3ms
10 Mbps	1500 bytes (1538 bytes)	1.23ms
100 Mbps	1500 bytes (1538 bytes)	0.123ms
1 Gbps	1500 bytes (1538 bytes)	0.0123ms
10 Gbps	1500 bytes (1538 bytes)	0.00123ms
100 Gbps	1500 bytes (1538 bytes)	0.000123ms

Latency – Queuing Delay

- What is a “queuing delay”?
  - “Queuing” is when a device holds a packet in memory while waiting for an opportunity to transmit it out a desired interface.
  - Every moment that the router spends holding onto the packet without transmitting it, the overall latency of the packet is increasing.
- So why do we need queuing in the first place?
  - First, a quick word about the concept of “utilization”.
    - A 10GbE port doing 5 Gbps is commonly said to be “50% utilized”.
    - But this is incorrect. At any given instant, the interface can only ever be transmitting (100% util) or not transmitting (0% util).
    - What you **actually** mean is the “avg util over X period is 50%”
  - ***SOME*** queuing is necessary for a router to function correctly.
    - Just ask anyone who has bought an under-buffered switch.

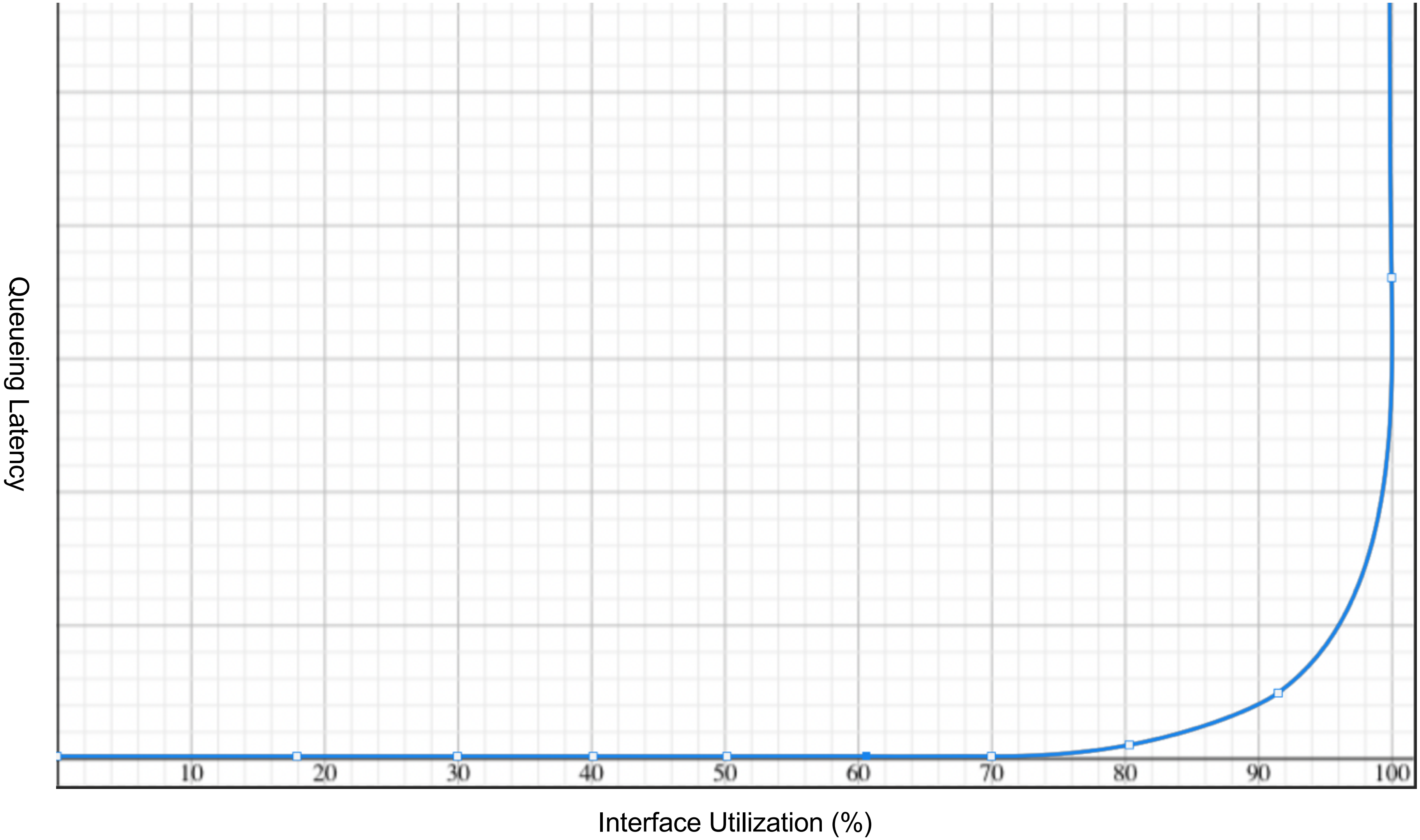
When is Queuing a Good Thing

- Queuing is especially important when:
  - You have mismatched interface speeds.
    - Packets will arrive “faster” on the higher speed interfaces, then need to be buffered while serializing to lower speed interfaces.
  - You have a lot of talkers sending traffic to a few receivers.
    - E.g. 40x GigE ports sending traffic to 4x 10GigE Uplink ports.
- Technically speaking, queuing ***always*** increases thruput.
  - The longer you hold on to the packet, the more opportunities you will have to transmit it rather than drop it.
  - The question becomes, how long should you hold it?
    - An extra 2ms to get from 80% to 90% utilization? Might be worth it.
    - If your application is extremely latency-sensitive, it might not be.

When is Queueing a Bad Thing?

- As an interface becomes more and more full, the amount of time necessary to find an open slot increases exponentially.
  - You may be able to successfully delivery every packet, but you may have to buffer them for thousands of milliseconds to do it.
  - This is probably not what your application wants, especially latency or jitter sensitive applications like voice and video.
- Beyond a certain point, queuing becomes counterproductive.
  - Getting from 98% to 99% utilization may be pointless, if it increases latency on almost every packet by 5000ms to do it.
  - Typically where most users start to notice serious problems with “internet mix” traffic is around 95% utilization (over 1 sec).
- Most routers have “bad” defaults for interface queues too.
  - Search for the term “buffer bloat” for more information.

Example Utilization vs Queueing Latency



Latency – Propagation Delay

- Propagation delay is the “time spent on the wire”.
  - The speed of light in a vacuum is ~ 300,000 km/sec.
  - Over long distances, this can cause significant latency.
- Some example math for propagation over fiber:
  - Fiber is made of glass, which has a refractive index of ~1.48
  - 1/1.48 = ~0.67c, so light travel through fiber at ~200,000 km/sec.
  - 200,000 km/sec = 200km (or 125 miles) per millisecond.
  - Divide by 2 to account for round-trip time (RTT) measurements.
    - Approx 1ms RTT propagation latency per 100 km (62.5 miles)
- Example:
  - A round-trip around the world at the equator across a perfectly straight fiber route would take ~400ms due to propagation delays.

Identifying the Latency Affecting You

- So, how do you determine if latency is normal?
  - Use location identifiers to determine geographical data.
  - See if the latency fits with the expected propagation delay.

- 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?
  - The Great Circle distance is approx. 3500 miles
  - 3500 miles / 62.5 miles = 56ms for a perfect path.
  - Sounds normal.

Identifying the Latency Affecting You

- Another example:

```
5 cr2.wswdc.ip.att.net (12.122.3.38) [MPLS: Label 17221 Exp 0] 8 msec
6 tbr2.wswdc.ip.att.net (12.122.16.102) [MPLS: Label 32760 Exp 0] 8 msec
7 ggr3.wswdc.ip.att.net (12.122.80.69) 8 msec
8 192.205.34.106 [AS 7018] 228 msec
9 te1-4.mpd01.iad01.atlas.cogentco.com (154.54.3.222) [AS 174] 228 msec
```

Washington DC to Washington DC in 220ms? Nope!

Prioritization and Rate Limiting

Prioritization and Rate-Limiting

- Remember, traceroute latency is the sum of:
  1. The time required for the probe packet to reach the router hop.
  2. The time required for the router to drop the probe packet and generate an ICMP TTL Exceed heading back to the original SRC.
  3. The time required for that ICMP TTL Exceed to reach the SRC.

- No. 1 and No. 3 come from real network characteristics.
  - But No. 2 has nothing to do with forwarding conditions.
  - This affects only traceroute packets, not real network traffic.

- A wide variety of conditions can cause routers to:
  - Not send the ICMP TTL Exceed packet (causing artificial loss).
  - Be slow in the generation of the ICMP (causing artificial latency).

Understanding “To It” vs. “Through It”

- Modern routers have distinct processing paths:
  - Control Plane - Packets being forwarded ***to*** the router
    - Example: BGP, ISIS/OSPF, SNMP, CLI access (telnet/ssh), ping, or any packets sent directly to a local IP address.
  - Date Plane - Packets forwarded ***through*** the router
    - 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***, logging, etc.
- Router CPUs tend to be relatively underpowered
  - A 320-640+ Gbps router may have a 600MHz MIPS CPU
- ICMP Generation is ***NOT*** a priority for the router.

The Infamous BGP Scanner

- On some platforms, the slow-path data plane and the control-plane share the same resources.
  - And historically have not had the best process schedulers.
  - Control-plane activity such as BGP churn or CLI use can consume CPU, and slow generation of ICMP TTL Exceeds.
  - This results in random “spikes” in traceroute latency, which is often misinterpreted as a network issue.

- One infamous example of this is the “BGP Scanner” process, which runs every 60 seconds on many classic Cisco IOS devices.

Rate Limited ICMP Generation

- ICMP generation is a slow-path data plane operation
  - There are ***many*** other reasons for ICMP generation besides TTL Exceed generation, and the process can be very complex.
  - No commercial router has “HW assisted traceroute” to date.
  - So, the general-purpose CPU is used to generate the ICMP pkt.
- These CPUs also handle other slow-path functions.
  - If left unchecked, a routing loop or TTL=0 Denial of Service attack could bring down the other functions of the router.
  - As a result, most routers rate-limit ICMP generation.
  - These rate-limits tend to vary wildly by vendor, platform, and software revision, and are often not configurable or logged!
    - A few users running MTR against a network can easily hit these!

Spotting The Cosmetic Loss/Latency

- If there is an ***actual*** forwarding issues, the loss/latency will persist across ***ALL*** future hops as well.
- Example (not a real issue in hop 2):

```
1 ae3.cr2.iad1.us.nlayer.net  0.275 ms  0.264 ms  0.137 ms
2 xe-1-2-0.cr1.ord1.us.nlayer.net  18.271 ms  18.257 ms  68.001 ms
3 tge2-1.ar1.slc1.us.nlayer.net 53.373 ms  53.213 ms  53.227
```

- 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 more often than not, this is an indication of an artificial rate-limit or prioritization issue.
    - Try a non-TTL expiring method like “ping” to confirm the behavior.

Asymmetric Paths

Asymmetric Paths

- Traceroute ***shows*** you the forward path ***only***
  - But remember, the latency values also include the time it takes for the TTL Exceed reply packet to come back.
  - The reverse path is a significant component of the latency values, but is ***completely invisible*** to the user.
- Not only is the reverse path hidden, but it can be completely different at every hop in the forward path.
  - The ***only*** way to confidently analyze a traceroute is to have traceroutes in **BOTH DIRECTIONS!**
  - And even then, it can’t catch ALL of the potential asymmetric paths in the middle.

Asymmetric Paths and Network Boundaries

- Asymmetric paths often start at network boundaries
  - Why? Because that is where administrative policies change.

```
te1-1.ar2.DCA3.gblx.net (69.31.31.209)  0.719 ms  0.560 ms  0.428 ms
te1-2-10g.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
```

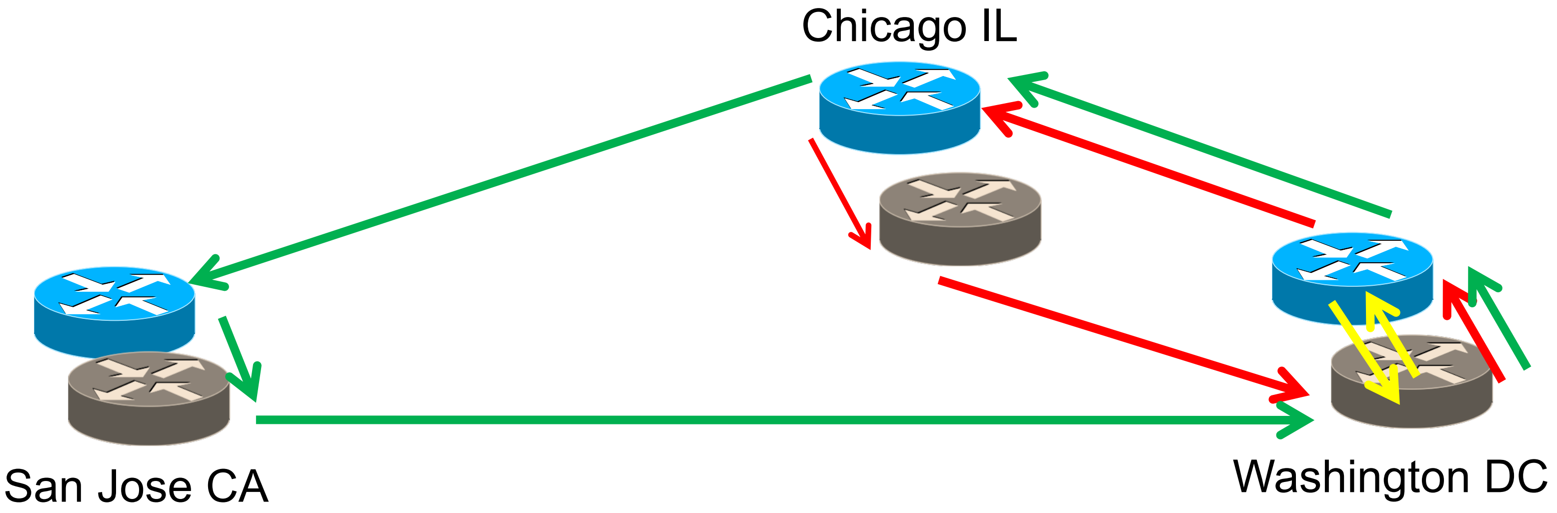
  - What’ s wrong in the path above?
    - It COULD be congestion between GBLX and Sprint.
    - But it could also be an asymmetric reverse path.
    - At this GBLX/Sprint boundary, the reverse path policy changes.
    - This is most often seen in multi-homed network with multiple paths.
    - In the example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

Using Source Address in your Traceroute

- How can you work 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 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.

Asymmetric Paths with Multiple Exits

- But remember, asymmetric paths can happen *anywhere*
- Especially where networks connect in multiple locations
  - And use closest-exit (hot potato) routing, as is typically done.
  - Hop 1 (yellow) returns via a Washington DC interconnection.
  - Hop 2 (red) returns via a Chicago interconnection.
  - Hop 3 (green) returns via a San Jose interconnection.



Using Source Address in your Traceroute

- But what if the /30 is numbered out of my space?
  - As in the case of a customer or potentially a peer.
- You can still see some benefits from setting SRCs
  - Consider trying to examine the reverse path of a peer who you have multiple interconnection points with.
    - A traceroute sourced from your IP space (such as a 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.

Default Source Addresses

- 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 “system default-address-selection”.

Multiple Paths and Load Balancing

Multiple Paths

- Remember, every probe is an independent trial.
  - UDP/TCP traceroute probes typically use a different layer 4 port every time, to identify which probe is which.
  - Equal-Cost Multi-Path (ECMP) may make multiple potential paths show up for each “hop” TTL value.
  - Example:

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

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

Multiple Paths - Examples

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.

Multiple Unequal Length Paths

- The most confusing scenario is ECMP across multiple paths that are of unequal router hop length.
  - This can make the traceroute appear to go back and forth.
  - The end result is extremely confusing and difficult to read.
- Consider the following equal-cost paths:
  - A – B – C – D – E
  - A – X – B – C – D – E
  - A Traceroute with 3 probes per hop ends up looking like:

```
1  A A A
2  B X B
3  C B C
4  D C D
5  E D E
```

How To Handle Multiple Paths

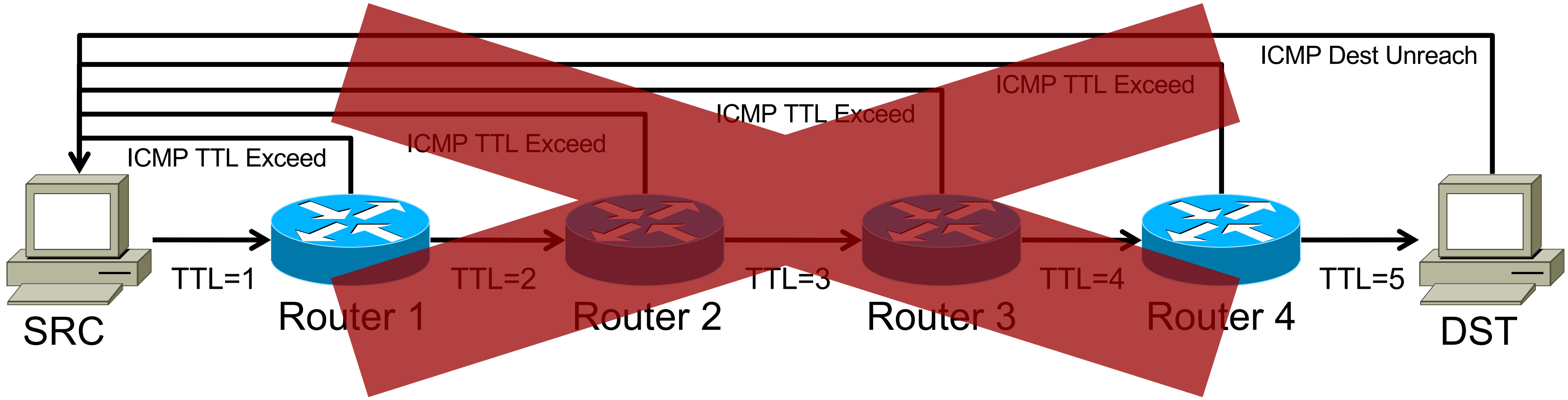
- When in doubt, try looking at just a single probe.
  - Set your traceroute client to only send 1 probe per hop.
    - For many Unix implementations, the command is “-q 1”.
    - JUNOS CLI lacks this, but you can do it in CLI from their unix shell.
  - But be aware that this may not be the path which your actual traffic forwards over.
  - And remember, EVERY PROBE is an independent trial.
    - Even when doing 1 probe per hop, you aren’t guaranteed you’re going to see a single contiguous path as taken by a single flow.
  - One way to try out different paths which may be available is to increment the dest IP by 1, or try different source IPs.
    - This can come into play when a network is doing ECMP hashing based on only Layer 3 information.

MPLS and Traceroute

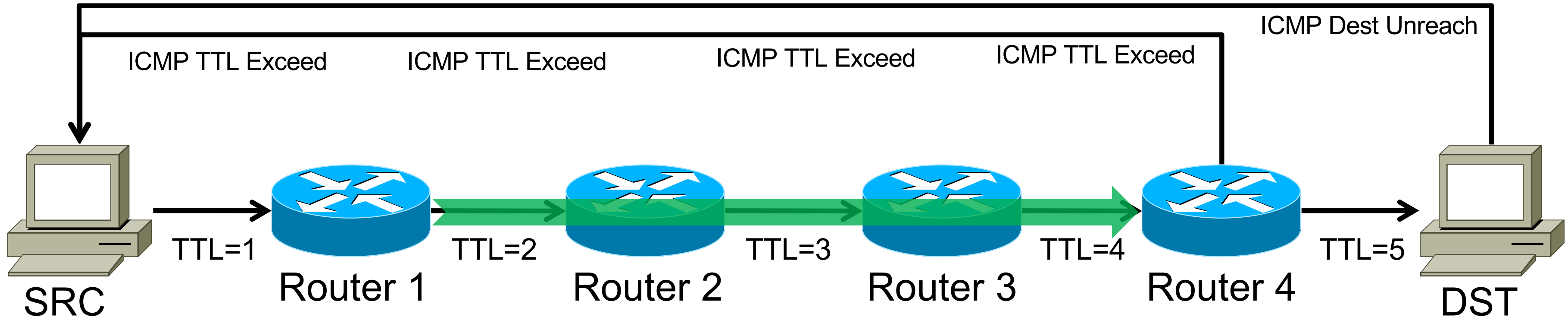
MPLS ICMP Tunneling

- Many large networks operate an MPLS based core
- Some devices don’t even carry an IP routing table
  - This is fine for switching MPLS labeled packets.
  - But presents a problem when ICMPs are generated.
  - How does the MPLS-only 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
  - This works for delivering the message, but…
  - It can make traceroutes look really WEIRD!

MPLS ICMP Tunneling Diagram



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 the same RTT as the final hop.



MPLS ICMP Tunneling Example

```
1.  te2-4.ar5.PAO2.gblx.net (69.22.153.209)  1.160 ms  1.060 ms  1.029 ms
2.  192.205.34.245 (192.205.34.245)  3.984 ms  3.810 ms  3.786 ms
3.  tbr1.sffca.ip.att.net (12.123.12.25)  74.848 ms  74.859 ms  74.936 ms
4.  cr1.sffca.ip.att.net (12.122.19.1)  74.344 ms  74.612 ms  74.072 ms
5.  cr1.cgcil.ip.att.net (12.122.4.122)  74.827 ms  75.061 ms  74.640 ms
6.  cr2.cgcil.ip.att.net (12.122.2.54)  75.279 ms  74.839 ms  75.238 ms
7.  cr1.n54ny.ip.att.net (12.122.1.1)  74.667 ms  74.501 ms  77.266 ms
8.  gbr7.n54ny.ip.att.net (12.122.4.133)  74.443 ms  74.357 ms  75.397 ms
9.  ar3.n54ny.ip.att.net (12.123.0.77)  74.648 ms  74.369 ms  74.415 ms
10.12.126.0.29 (12.126.0.29)  76.104 ms  76.283 ms  76.174 ms
11.route-server.cbbtier3.att.net (12.0.1.28)  74.360 ms  74.303 ms  74.272
    ms
```

Final Thoughts: The Traceroute Checklist

- Before beginning any serious traceroute analysis, you should **ALWAYS** ask for:
    - Traceroutes in both directions (forward ***AND*** reverse).
    - The real source and destination IPs for those traceroutes.
- Beware of:
    - Snippits of traceroutes with missing information.
    - Users running a traceroute towards an IP they saw in another traceroute, rather than a real destination host.
        - These can route VERY differently, and aren’t the same as the real destination IP for network performance purposes.

Thanks for your time.

Send comments, questions, complaints, to:

Richard A Steenbergen <ras@petabitscale.com>