A Practical Guide to (Correctly) Troubleshooting with Traceroute

Richard A Steenbergen <ras@nlayer.net> nLayer Communications, Inc.

Introduction

- Troubleshooting problems on the Internet?
  - The number one go-to tool 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 it seems like such a simple tool to use
    - Type in an IP address and it shows you a list of router hops
    - And where the traceroute stops, drops packets, or where the latency goes up a lot, that’s where the problem is, right?
    - How could this possibly go wrong?
  - Unfortunately, it almost never works out this way.

Problem Statement

- So what's wrong with traceroute?
  - Most modern commercial networks are actually well run
    - Simple issues like congestion or routing loops are becoming a smaller percentage of the total issues encountered.
    - More commonly, issues are complex enough that a naïve traceroute interpretation is utterly useless.
  - Few people are actually skilled at interpreting traceroute
    - Most ISP NOCs and even most mid-level engineering staff are not able to correctly interpret a complex traceroute.
    - Leads to a significant number of misdiagnosed issues, false reports, etc, which flood the NOCs of networks world-wide.
    - False report rate is so high that it is almost impossible to report a real traceroute-based issue through all the noise.

Traceroute Topics

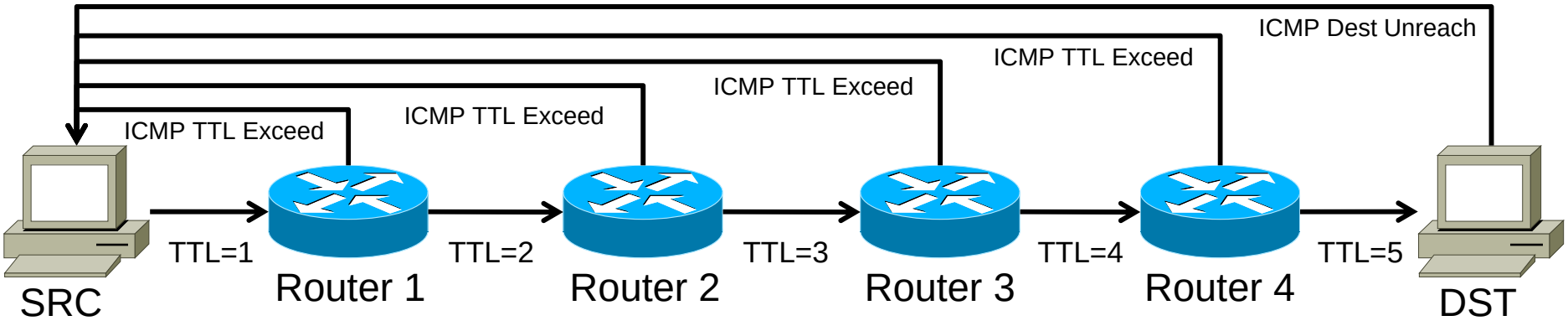
- Topics to discuss
  - How traceroute works
  - Interpreting DNS in traceroute
  - Understanding network latency
  - ICMP prioritization and rate-limiting
  - Asymmetric forwarding paths
  - Load balancing across multiple paths
  - Traceroute and MPLS

- Random Traceroute Factoid
  - The default starting destination probe port in the UNIX traceroute implementation is 33434. This comes from 32768 (2^{15}) + 666 (the mark of Satan). Coincidence?

How Traceroute Works

Traceroute – The 10,000 Ft Overview

1. Launch a probe packet towards DST, with a TTL of 1
2. Every router hop decrements the IP TTL of the packet by 1
3. When the TTL hits 0, packet is dropped, router sends ICMP TTL Exceed packet to SRC with the original probe packet as payload
4. SRC receives this ICMP message, displays a traceroute “hop”
5. Repeat from step 1, with TTL incremented by 1 each time, until…
6. DST host receives probe, returns ICMP Dest Unreachable
7. SRC stops the traceroute upon receipt of ICMP Dest Unreachable



Traceroute Implementation Details

- Traceroute can use many protocols for probe packets
  - Classic UNIX traceroute uses UDP probes
    - With a starting destination port of 33434, incrementing once per probe.
    - Cannot detect the end of the traceroute if the DST does not return an ICMP Dest Unreachable. This can happen as the result of firewalls, configuration settings, or a real application listening on the dest port.
  - Other implementations use ICMP Echo Request probes
    - Windows tracert.exe and MTR are the two biggest examples.
    - These also cannot detect the end of the traceroute if the DST does not return an ICMP Echo Response. This may or may not be more frequently firewalled than the UDP->ICMP Dest Unreachable response.
  - Many modern traceroute implementations can do all
    - Configurable UDP, TCP, or ICMP probe packets via CLI flags.
    - TCP is a poor choice for general use (frequently filtered), typically only seen as a method to work around specific firewalls.

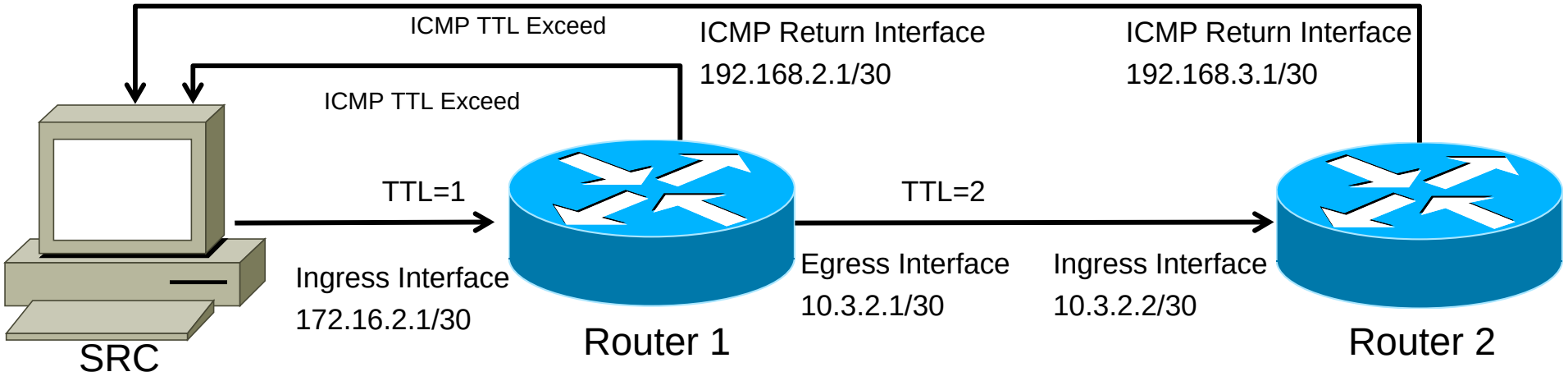
Traceroute Implementation Details

- Most implementations send multiple probes per router hop.
  - The default for classic traceroute is 3 probes per hop.
    - Giving the 3 latency results, or 3 *'s if there is no response.
  - One specific implementation (MTR) sends an endless loop of probes.
- Each probe has a unique code embedded in it
  - So the original traceroute implementation can map the responses.
  - UDP/TCP use incrementing layer 4 ports, ICMP uses the seq #.
- Layer 4 hashing can send each probe down a different path
  - This may or may not be visible to traceroute
    - Yes in the case of ECMP (Layer 3 Equal-Cost Multi-Path) load-balancing.
    - No in the case of LAG (Layer 2 802.3ad/Port-channel) load-balancing.
  - But the result is the same, each probe can behave in different ways, leading to different results for the same TTL "hop".

Traceroute – Latency Calculation

- How is traceroute latency calculated?
  - Timestamp when the probe packet is launched.
  - Timestamp when the ICMP response is received.
  - Calculate the difference to determine round-trip time.
  - 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.
  - **Most Importantly:** only the ROUND TRIP is measured
    - Traceroute is showing you the hops on the forward path.
    - But showing you latency based on the forward PLUS reverse path. Any delays on the reverse path will affect your results!

Traceroute – What Hops Are You Seeing?



- Traceroute packet with TTL of 1 enters router via the ingress interface.
- Router decrements TTL to 0, drops packet, generates ICMP TTL Exceed
  - ICMP packet dst address is set to the original traceroute probe source (SRC)
  - ICMP packet src address is set to the IP of the **ingress router interface**.
  - Traceroute shows a result based on the src address of the ICMP packet.
  - The above traceroute will read: 172.16.2.1 10.3.2.2
  - You have **NO** visibility into the return path or the egress interface used.
- Random factoid: This behavior is actually non-standard. RFC1812 says the ICMP source MUST be from the egress iface. If obeyed, this would completely change traceroute results.

Interpreting DNS in Traceroute

Interpreting DNS in a Traceroute

- Interpreting DNS is one of the most important aspects of correctly using traceroute.
- Information you can uncover includes:
  - Physical Router Locations
  - Interface Types and Capacities
  - Router Type and Roles
  - Network Boundaries and Relationships
- Deductions made from this information can be absolutely essential to troubleshooting.

Interpreting Traceroute - Location

- Why do you need to know geographical locations?
  - To identify incorrect/suboptimal routing.
    - Going from Atlanta to Miami via New York? Probably not good.
  - To know when high latency is justified and when it isn't.
    - 100ms across an ocean is normal, 100ms across town isn't.
  - To help you understand network interconnection points.
- The most commonly used location identifiers are:
  - IATA Airport Codes
  - CLLI Codes
  - Non-standard abbreviations based on a city name.
  - Of course sometimes you just have to take a guess.

Location Identifiers – IATA Airport Codes

- Good International coverage of most large cities.
  - Typically seen in networks with a small number of large POPs, or heavy focus in “well developed” areas.
- Examples:
  - Dallas Texas = DFW
  - San Jose California = SJC
  - Sometimes represented by pseudo-airport codes
    - Especially where multiple airports serve a region
    - Or where the airport code is non-intuitive for the city name
      - New York, NY is served by JFK, LGA, and EWR airports.
        - But may be represented by simply “NYC”.
      - Northern VA is served by IAD, Washington DC by DCA.
        - But both may be written as WDC.

Location Identifiers – CLLI Codes

- Common Language Location Identifier
  - Full codes maintained (and sold) by Telecordia.
  - Most commonly used by Telephone Companies
    - Example: **HSTNTXMOCG0**
  - For non-telco uses, only city/state codes are important
    - Examples:
      - HSTNTX = Houston Texas
      - ASBNVA = Ashburn Virginia
  - Well defined standard covering almost all NA cities
    - Commonly seen in networks with a larger number of POPs.
    - Not an actual standard outside of North America
      - Some providers fudge these, e.g. AMSTNL = Amsterdam NL

Location Identifiers – Arbitrary Values

- And then sometimes people just make stuff up
  - Chicago IL
    - Airport Code: ORD (O'Hare) or MDW (Midway)
    - CLLI Code: CHCGIL
    - Example Arbitrary Code: CHI
  - Toronto ON
    - Airport Code: YYZ (Pearson) or YTZ (City Center)
    - CLLI Code: TOROON
    - Example Arbitrary Code: TOR
  - Frequently based on the good intentions of making thing readable in plain English, even though these may not follow any standards.

Common Locations – US Major IP Cities

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

Common Locations – Non-US Major Cities

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

Interpreting DNS – Interface Types

- Most networks will try to put interface info into DNS
  - Often to help them troubleshoot their own networks.
  - Though this many not always be up to date.
    - Many large networks use automatically generated DNS.
    - Others can be surprisingly sloppy.
  - Can potentially help you identify the type of interface
    - As well as capacity, and maybe even the make/model of router.
- Examples:
  - xe-11-1-0.edge1.NewYork1.Level3.net
    - XE-#/#/# is Juniper 10GE port. The device has at least 12 slots.
    - It's at least a 40G/slot router since it has a 10GE PIC in slot 1.
    - It must be Juniper MX960, no other device could fit this profile.

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-#/#/# (*)
SONET	Pos#/#	POS#/#/#/#	so-#/#/#
T1	Se#/#		t1-#/#/#
T3			t3-#/#/#
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-#-#-#.###

(*) Some early Juniper 10GE interfaces on some platforms are named GE

Interpreting DNS – Router Types/Roles

- Knowing the role of a router can be useful
  - But every network is different, and uses different naming conventions.
  - And just to be extra confusion, they don't always follow their own naming rules.
- Generally speaking, you can guess the context and get a basic understanding of the roles.
  - Core routers – CR, Core, GBR, BB, CCR, EBR
  - Peering routers – BR, Border, Edge, IR, IGR, Peer
  - Customer routers – AR, Aggr, Cust, CAR, HSA, GW

Network Boundaries and Relationships

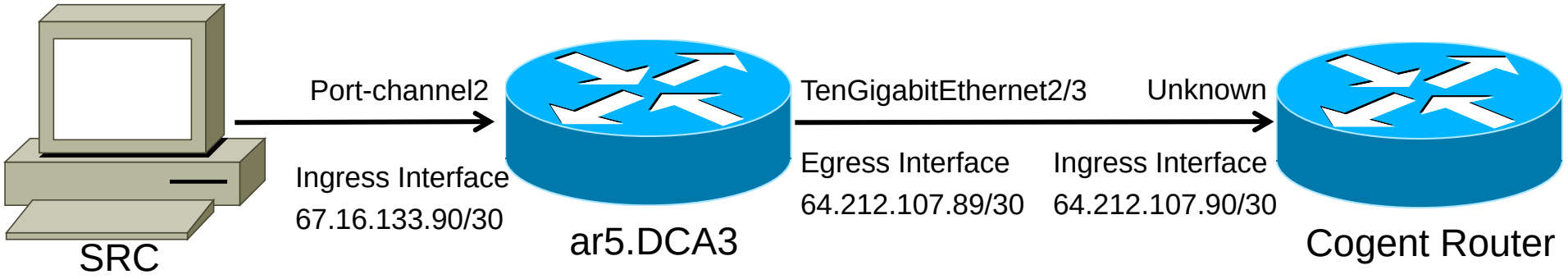
- Identifying Network Boundaries is important
  - These tend to be where routing policy changes occur
    - E.g. different return paths based on Local Preference.
  - These tend to be areas where capacity and routing are the most difficult, and thus likely to be problems.
  - It also helps to know who to blame. ☺
- 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
    - Examples:
      - Clear names like *networkname.customer.alter.net*
      - Or always landing customers on routers named “gw”

Network Boundaries and Relationships

- Sometimes 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)
- Alternatively, look for the "other 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)
- Sometimes there will be no useful DNS info at all:
  2 po2-20G.ar4.DCA3.gblx.net (67.16.133.82)
  3 192.205.34.109 (192.205.34.109)
  4 cr2.wswdc.ip.att.net (12.122.84.46)
  - Is hop 3 the GBLX/AT&T border here, or is hop 4?
  - Whois says 192.205.34.109 is owned by AT&T.

Network Boundaries and Relationships

- For more info, look at the other side of the /30
  - 4 po2-20G.ar5.DCA3.gblx.net (67.16.133.90)
  - 5 cogent-1.ar5.DCA3.gblx.net (64.212.107.90)
  - > nslookup 64.212.107.89 = te2-3-10GE.ar5.DCA3.gblx.net
  - The multiple ar5.DCA3 hops are a clear indicator that hop 5 is **NOT** a gblx router, even without the “Cogent” hint in DNS.
  - Common with private peering. One side will provide the /30 but not collect/maintain DNS info from the other side, so the data gets filled in with info from their side rather than left blank.



Understanding Network Latency

Understanding Network Latency

- Three main types of network induced latency
  - Serialization Delay
    - The delay caused by having to transmit data through routers/switches in packet sized chunks.
  - 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 delays.

Latency – Serialization Delay

- The delay caused by packet-based forwarding
  - A packet moves through a network as a discrete unit.
  - Can’t transmit the next packet until last one is finished.
- Not much of an issue in high-speed networks
  - Speeds have increased by orders of magnitude over the years, while packet sizes have stayed the same.
  - 1500 bytes over a 56k link (56Kbps) = 214.2ms delay
  - 1500 bytes over a T1 (1.536Mbps) = 7.8ms delay
  - 1500 bytes over a FastE (100Mbps) = 0.12ms delay
  - 1500 bytes over a GigE (1Gbps) = 0.012ms delay

Latency – Queuing Delay

- First we must understand “Utilization”
  - A 1GE port doing 500Mbps is said to be “50% utilized”.
    - But in reality, an interface can only be transmitting (100% utilized) or not transmitting (0% utilized) at any given instant.
    - The above is actually “used 50% of the time, over a period of 1 second”.
- Some queuing is a natural function of networking
  - When an interface is in use, the next packet must be queued.
    - The odds that an interface will be in use (transmitting) at any given instant depends on how much traffic is being sent across it.
      - 90% utilization = 90% chance that the packet will have to be queued.
    - Transitions between interface speeds also require queuing.
    - As an interface reaches saturation, the time spent in queue rises rapidly.
    - When an interface is extremely full, a packet may be queued for many hundreds or thousands of milliseconds (depending on the router).
    - Thus queuing delays are often associated with congestion (full interfaces).

Latency – Propagation Delay

- Delay caused by signal propagation over distance.
  - Light travels through a vacuum at around 300,000 km/sec
    - But fiber is made of glass, not a vacuum, so it travels slower.
    - Fiber cores have a refractive index of 1.48, 1/1.48 = ~0.67c
    - Light travels through fiber at around 200,000 km/sec.
  - 200,000 km/sec = 200km (or 125 miles) per millisecond.
  - Or, 100 km (or 62.5 miles) per 1 ms of round-trip delay.
- Example:
  - A round-trip around the world at the equator, via a perfectly straight fiber route, would take ~400ms due solely to speed-of-light 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 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? 4200 miles? Normal.
  - 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 te1-4.mpd01.iad01.atlas.cogentco.com (154.54.3.222) [AS 174] 228 msec 228 msec 228 msec

    Washington DC to Washington DC in 220ms? Not normal.

Prioritization and Rate Limiting

Cosmetic Delays Affecting Traceroute

- The latency value measured by traceroute is based on:
  1. The time taken for the probe packet to reach a specific router, plus
  2. The time taken for the router to generate the ICMP TTL Exceed, plus
  3. The time taken for the ICMP TTL Exceed to return to the SRC.

- Items #1 and #3 are based on actual network conditions.
- But Item #2 is not.
  - It is by definition impossible for item #2 to cause impact to “real” traffic.
  - Only the traceroute probes and responses themselves are affected.
  - This results in “cosmetic” issues which are mistaken for real issues.

Routing “To It” vs. “Through It”

- Architecture of a modern router:
  - Packets forwarded **through** the router (data-plane)
    - Fast Path: hardware based forwarding of ordinary packets
      - Examples: Almost every packet in normal Internet traffic.
    - Slow Path: software based handling of “exception” packets
      - Examples: IP Options, **ICMP generation** ← Traceroute happens here
  - Packets being forwarded **to** the router (control-plane)
    - Examples: BGP, IGP, SNMP, CLI access (telnet/ssh), ping, or any other packets sent directly to a local IP address on the router.
  - Router CPUs tend to be relatively underpowered
    - A 320-640+ Gbps router may only have a single 600MHz MIPS CPU
    - Which is usually busy enough doing things other than traceroute
      - ICMP Generation is **NOT** a priority for the router.
      - And in most cases is specifically rate-limited and de-prioritized.

The Infamous BGP Scanner

- On some popular router platforms the slow-path data plane and the control-plane share the same resources.
  - And often don't have the best software schedulers either.
  - As a result, control-plane activity such as BGP churn, CLI use, and periodic software processes can consume enough CPU to slow the generation of ICMP TTL Exceed packets.
  - This results in "spikes" in traceroute reported latency.
- The most infamous process is Cisco IOS "BGP Scanner"
  - Runs every 60 seconds on all BGP speaking IOS routers.
  - Does periodic removal of routes with invalid next-hops, etc.
  - Impact significantly reduced with "Next-Hop Tracking" feature.

Rate-Limited ICMP Generation

- Most routers also rate limit their ICMP generation
  - Often with arbitrary values which can't be changed.
  - These may be insufficient under heavy traceroute load.
    - Especially with more and more users running MTR.
- Juniper M/T/MX-series
  - Distributed ICMP generation, runs on FPC CPU, doesn't touch RE.
  - Hard-coded limit of 50pps per FPC for type 1/2, 250pps on FPC3s.
  - FPC3 hard-coded limit bumped to 500pps per FPC in JUNOS 8.3+.
- Foundry MLX/XMR
  - Hard-coded limit of 400pps per interface.
- Force10 E-series
  - Hard-coded limit of 200pps or 600pps per interface.

Rate-Limited ICMP Generation

- Cisco 6500/7600 Routers (SUP720+)
  - Configurable rate-limit for TTL expiring packets
    - mls rate-limit all ttl-failure 1000 255
    - Affects all ICMP TTL Exceed generations for the entire chassis.
  - Centralized processing of ICMP generation
    - Runs on MSFC for the entire chassis, along-side control-plane operations.
- Cisco GSR
  - Hard-coded rate-limit per line-card, ICMP done on LC CPU.
- Cisco IOS-XR Platforms (CRS-1)
  - Hard-coded rate-limit prior to IOS XR 3.6
  - 3.6+ rate-limit somewhat configurable via LPTS
    - hw-module forwarding mpls ttl-expiry police rate 1000

Spotting The Fake Latency Spikes

- The most important rule for troubleshooting latency
  - If there is an **actual** issue, the latency will continue or increase for all future hops afterwards.
    - 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 68.257 ms 18.001 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 it is probably an artificial rate-limit or prioritization issue.
    - By definition, if the regularly forwarded packets are being affected you should see the issue persist on all future hops.

Asymmetric Forwarding Paths

Asymmetric Paths

- Routing on the Internet has no guarantee of symmetry
  - In fact, it is almost always going to be asymmetric.
- Traceroute shows you **only** the forward path
  - Even though the latency is based on the round-trip time.
- The reverse path is completely invisible to traceroute
  - It can be completely different at every hop along the path.
  - The only practical solution is to look at both forward and reverse path traceroutes to try and spot reverse path issues.
  - And even that won't catch asymmetric paths in the middle.

Asymmetric Paths and Network Boundaries

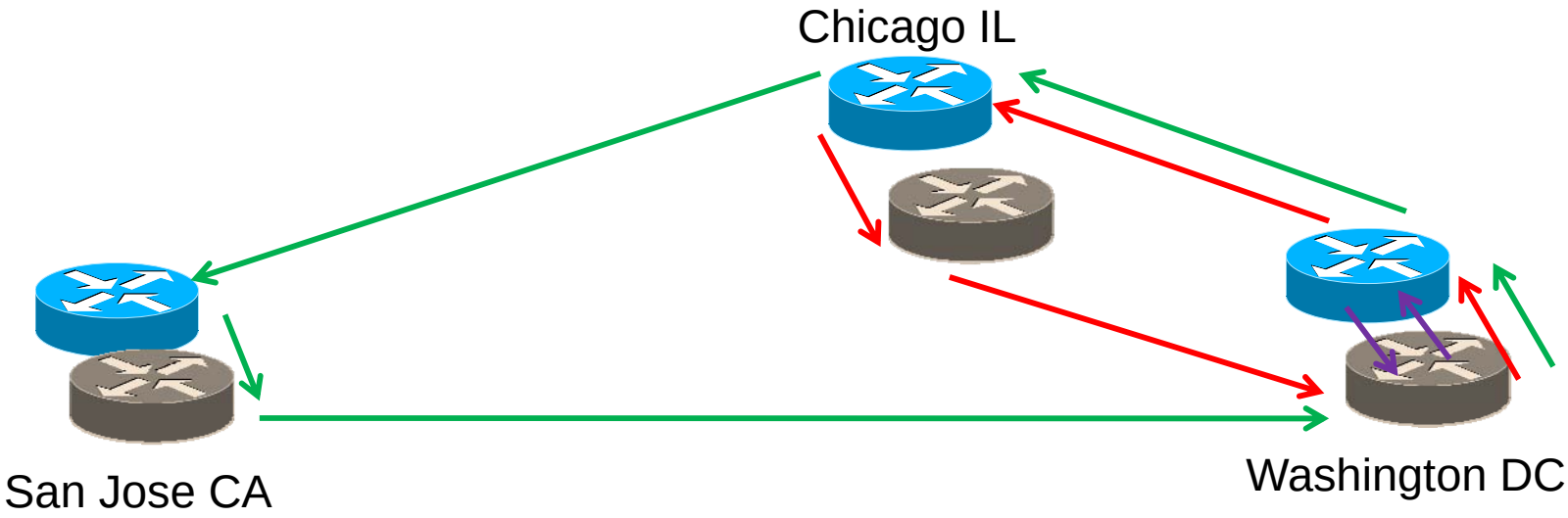
- Asymmetric paths often start at network boundaries
  - Why? Because that is where administrative policies change

```
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 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.

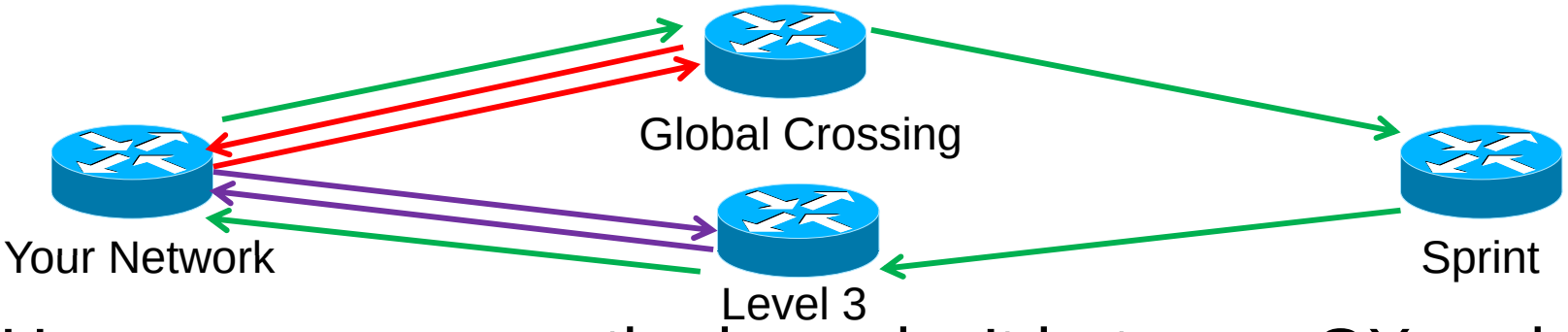
Multiple Interconnection Points

- Asymmetric paths can potentially happen at every router hop.
- Especially where networks connect in multiple locations.
  - The forward path of all hops goes via Washington DC interconnection.
  - Hop 1 (purple) returns via the Washington DC interconnection.
  - Hop 2 (red) returns via the Chicago interconnection.
  - Hop 3 (green) returns via the San Jose interconnection.
  - Congestion at the Chicago interconnection would disappear by hop 3.



Using Source Address in your Traceroute

- How can you test around asymmetric paths?
  - Consider the previous example of a problem with Sprint
    - You are multi-homed to GX and Level 3.
    - Traceroute shows You -> GX -> Sprint and latency starting at Sprint



  - How can you prove the issue isn't between GX and Sprint?
    - Run a traceroute using your side of the GX /30 as your src address.
    - This /30 comes from your provider (GX)'s larger aggregate block.
    - The reverse path will be guaranteed to go Sprint->GBLX
    - If the latency doesn't persist, you know the issue is on the reverse.

Using Source Address in your Traceroute

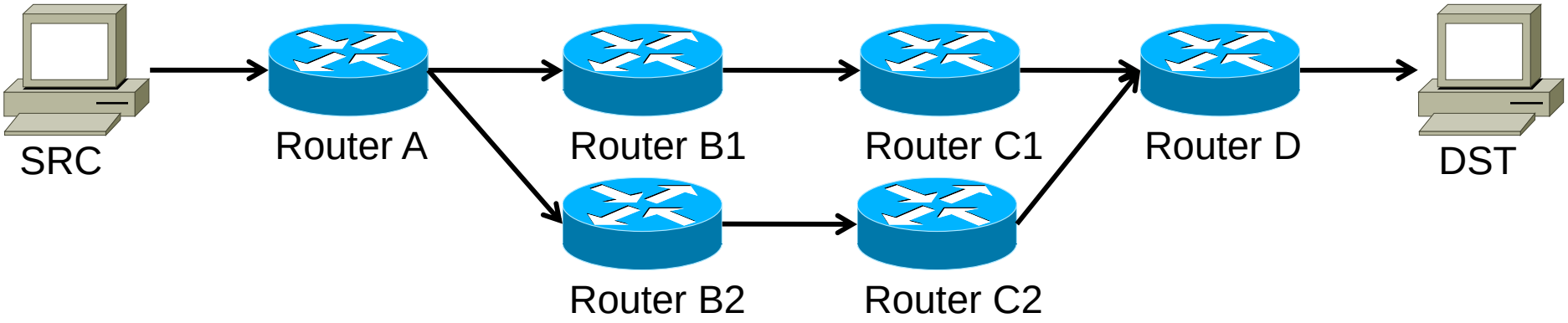
- But what if the /30 is numbered out of my space?
  - As in the case of a customer or potentially a peer.
- You can still see some benefits from setting SRCs
  - Consider trying to examine the reverse path of a peer who you have multiple interconnection points with.
    - A traceroute sourced from your 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, setting the traceroute source to that /30 would force the return path to come back via that interconnection.
    - Trying both options can give you different viewpoints.

Tracerouting From a Router

- Default Source Address
  - 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 using “system default-address-selection”.
- Clock granularity
  - Some platforms may be less accurate than others.
    - For example, Cisco IOS has a 4ms latency granularity.

Load Balancing Across Multiple Paths

Equal Cost Multi-Path Routing



- Flow hashing keeps a single TCP/UDP flow mapped to a single path.
- UDP/TCP traceroute probes with incrementing layer 4 ports look like unique flows, which may cause them to go down different parallel paths.
  - 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
```

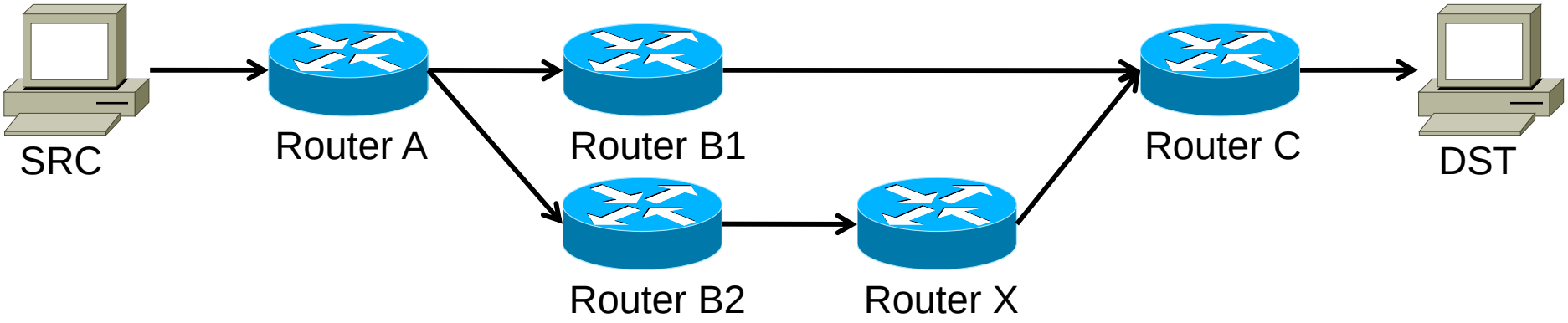
Multiple Paths - Examples

- 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 “somewhat” parallel paths
  - Ashburn VA – New York NY – Seattle WA
  - Ashburn VA – Chicago IL – Seattle WA
- Completely harmless, flow hashing protects against reordering, but the resulting traceroute is potentially confusing.

Multiple Unequal-Length Paths



- A far more confusing scenario is equal-cost unequal-length paths.
- This makes the traceroute appear to jump “back and forth” between hops
- It can be **extremely** confusing to end users and very difficult to parse.
- An example traceroute would end up looking something like this:
  - 1 A A A
  - 2 B1 B2 B1
  - 3 C X C
  - 4 D C D
  - 5 E D E

Coping With Multiple Paths

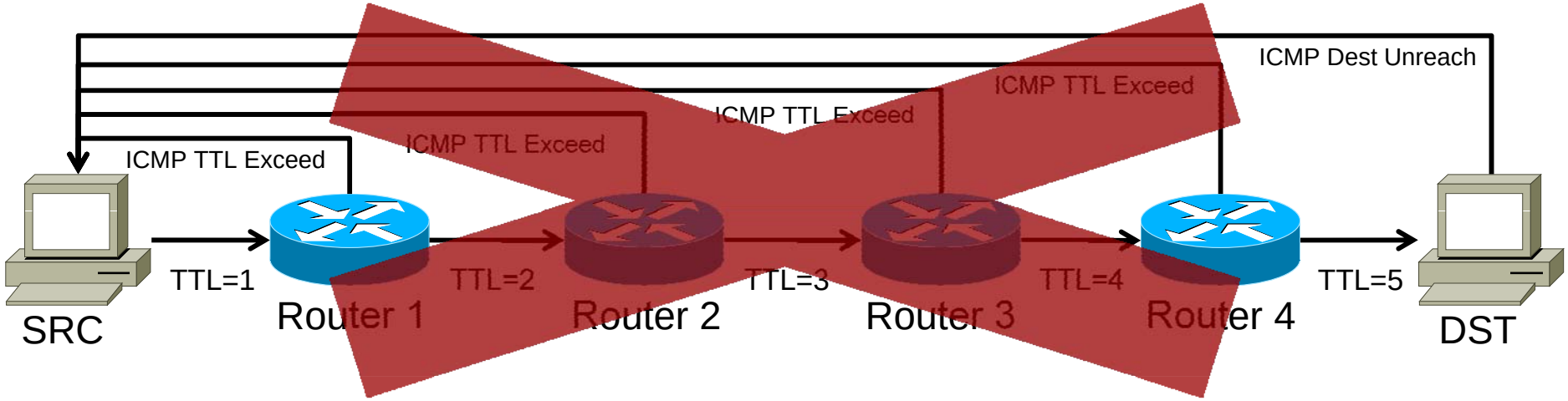
- When in doubt, only look at a single path
  - Set your traceroute client to only send a single probe.
  - But be aware that this may not be the path which your actual traffic forwards over.
  - One way to try out different paths manually is to increment the source or destination IP by 1 across multiple complete traceroutes.

MPLS and Traceroute

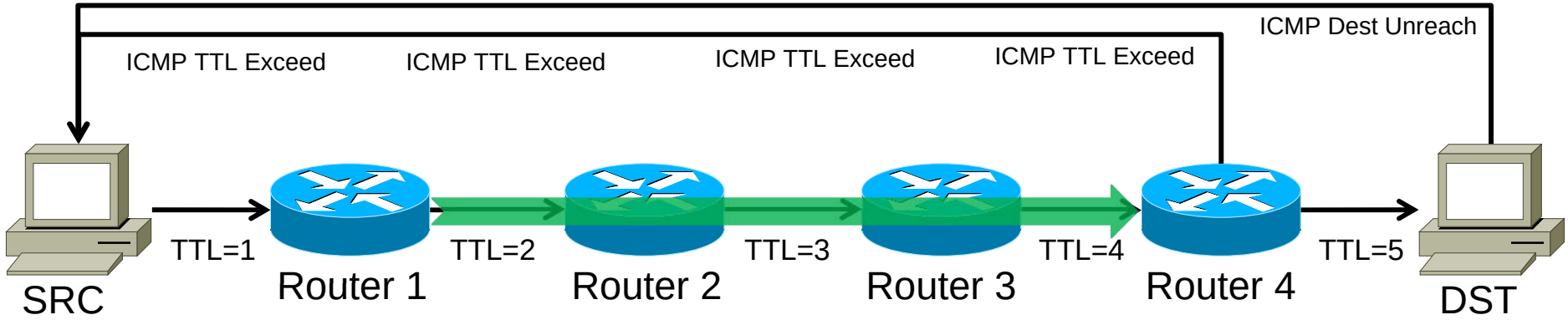
MPLS ICMP Tunneling

- 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 an ICMP is generated
  - How does the MPLS-only router deliver an ICMP msg?
- 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!

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

Send questions, complaints, to:

Richard A Steenbergen <ras@nlayer.net>