



"Where the !?#! are the packets going?"

INEX Member Meeting, Dec 12th 2024

Dublin, Ireland

Luca Sani

Senior R&D Software Engineer @ Catchpoint

Traceroute

Traceroute is one of the most famous and long-lasting diagnostic tools in networking environment

First implementation by Van Jacobson in late 80s to answer the question:

"where the !?*! are the packets going" ?

```
Posted-Date: Tue, 20 Dec 88 05:13:28 PST
Received-Date: Tue, 20 Dec 88 05:14:46 PST
Received: from helios.ee.lbl.gov by venera.isi.edu (5.54/5.51)
        id AA25560; Tue, 20 Dec 88 05:14:46 PST
Received: by helios.ee.lbl.gov (5.59/s2.2)
        id AA03127; Tue, 20 Dec 88 05:13:30 PST
Message-Id: <8812201313.AA03127@helios.ee.lbl.gov>
To: ietf@venera.isi.edu, end2end-interest@venera.isi.edu
Subject: 4BSD routing diagnostic tool available for ftp
Date: Tue, 20 Dec 88 05:13:28 PST
From: Van Jacobson <van@helios.ee.lbl.gov>
Content-Length: 2373
X-Lines: 46
Status: R0
```

```
After a frustrating week of trying to figure out "where the !?*!
are the packets going?", I cobbled up a program to trace out
the route to a host. It works by sending a udp packet with a
ttl of one & listening for an icmp "time exceeded" message. If
it gets one, it prints the source address from the icmp message,
then bumps the ttl by one & etc. (As usual, I didn't come up
with this clever idea -- I heard Steve Deering mention it at an
end-to-end task force meeting.)
```

Traceroute implementations

- Many traceroute implementations have been created on different OSes
- Over the years it became one of the most used tools in the Internet measurement and topology discovery fields (multipath, de-aliasing, NAT traversal, ...)
 - Paris, Dublin, Pamplona traceroute...

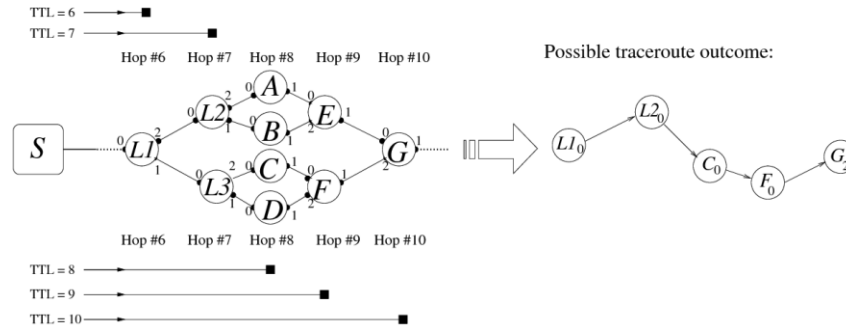


Fig. 1. Traceroute under load balancing

Linux traceroute

- We leverage Dmitry Butskoy's "[Linux traceroute](#)"
 - Very fast
 - Open source
 - Easily extendible

[Project Page](#)

[Download](#)

[Mail List](#)

This is a new modern implementation of traceroute(8) utility for Linux systems.

It has replaced the old one in the majority of distributions now, including [Fedora](#), RHEL, [Debian](#), Mandriva, [Gentoo](#), [Ubuntu](#).

- During the years we enhanced this traceroute to include new monitor capabilities
- We hope these enhancements can be useful to the community

Pietrasanta Traceroute

"A noble town since 1841 and a city of art"
(and where our Italian office is located!)



Pietrasanta Traceroute

- QUIC traceroute
- TCP InSession
- Work in Azure environment
- ECN-awareness (IP and transport level)
- Path MTU performance improvements
- Report ToS/DSCP hop by hop
- Report MSS when running in TCP mode
- Handle print in a separate thread (speed up)
- Overall timeout
- Avoid UDP standard filtering

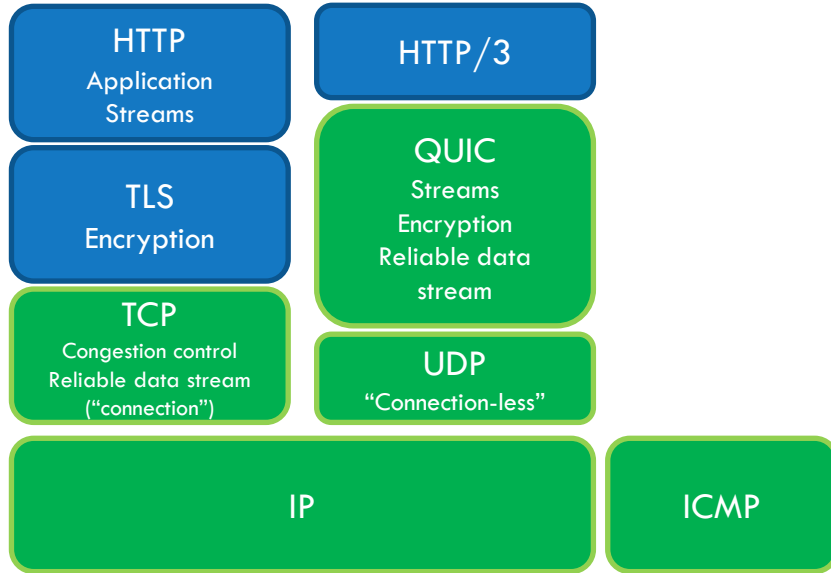


<https://github.com/catchpoint/Pietrasanta-traceroute>

QUIC traceroute

QUIC

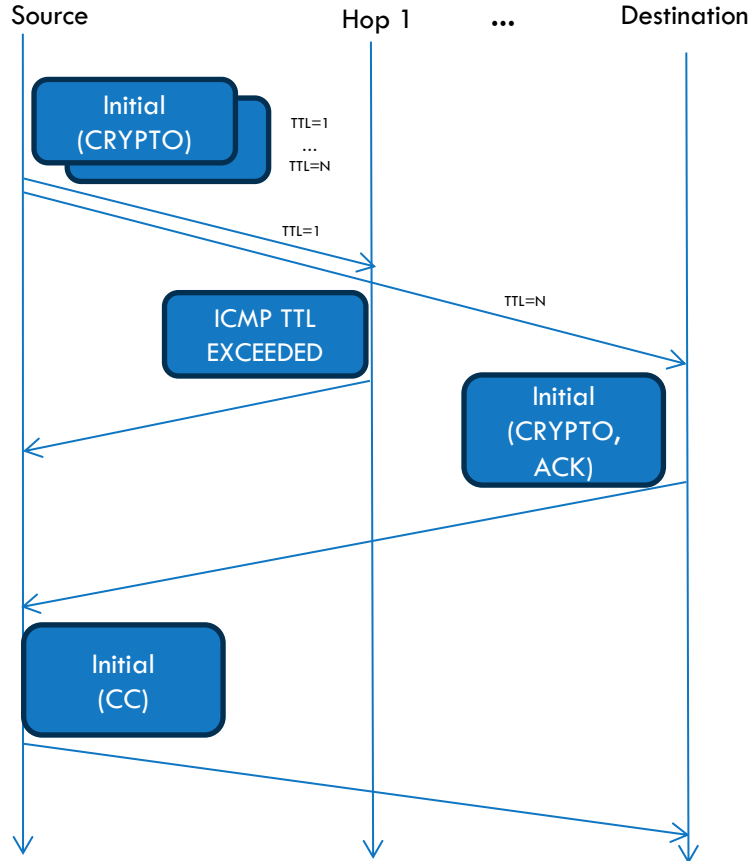
- QUIC is considered a transport layer protocol
 - More than just “UDP”
 - e.g., it is the transport layer of HTTP/3



QUIC assumes responsibility for the confidentiality and integrity protection of packets. For this it uses keys derived from a TLS handshake, but instead of carrying TLS records over QUIC (as with TCP), TLS handshake and alert messages are carried directly over the QUIC transport, which takes over the responsibilities of the TLS record layer.

[RFC9001](#) - Using TLS to Secure QUIC

QUIC traceroute



- Packets sent are QUIC compliant, so the header is protected and the payload (frames) are encrypted
 - We leverage openssl3
- Nice “side effects”
 - Check whether the path filters QUIC
 - Determine if the destination supports QUIC
 - Check whether ECN is supported
 - Set IP-ECN in probes

QUIC traceroute

- Like "TCP half open"
- Do a QUIC handshake then closes the session (if opened)
 - Send QUIC "Initial" packet
 - Include a CRYPTO frame with TLS "Server Hello"
 - Intermediate hops will return ICMP TTL Exceeded
 - Destination may return
 - QUIC packet
 - ICMP port unreachable (still good, dest reached)
 - Nothing (timedout)
 - Close the session if it is the case
 - Send QUIC Initial packet including a CONNECTION_CLOSE frame

TCP InSession

TCP InSession

- Classic TCP traceroute sends a different SYN for each hop
 - Different SYNs can take different paths
 - No consistency within a single traceroute
 - Many SYNs are sent per traceroute
 - Trigger firewall rules (SYN flood?)
- TCP InSession firstly opens a TCP session with the destination
- Then tracerouting is performed by sending 1-byte data packets within the session (with incremental TTL)
 - Inspired by [TCP Sidecar](#)



TCP InSession

- CONS

- Requires SACK mechanism
- Works only if the endpoint is listening in TCP on the target port
- Opens a TCP session with the target host

- PROS

- Sends 1 SYN per traceroute
 - Avoid to cause SYN flood
- Traceroute probes are seen as part of a data flow
 - Bypass firewalls
 - Data packets are "more likely" to follow the same path

Check out our blog:

<https://www.catchpoint.com/blog/traceroute-in-session-catchpoints-effort-towards-a-more-reliable-network-diagnostic-tool>

TCP InSession

```
tracert to www.bing.com(204.79.197.200), 30 hops max, 60 byte packet
 1  192.168.0.1  0.669 ms  0.621 ms  0.604 ms
 2  192.168.1.1  0.827 ms  0.571 ms  0.527 ms
 3  * * *
 4  172.18.33.212  5.071 ms  5.548 ms  172.18.33.196  5.530 ms
 5  172.18.33.226  6.503 ms  6.080 ms  172.18.33.228  6.060 ms
 6  172.19.184.88  12.701 ms  172.19.184.92  10.165 ms  172.19.184.90  10.119 ms
 7  172.19.177.66  10.087 ms  172.19.177.24  10.058 ms  172.19.177.20  12.082 ms
 8  195.22.192.144  12.046 ms  195.22.205.98  10.502 ms  10.047 ms
 9  195.22.208.79  10.003 ms  195.22.196.69  12.290 ms  195.22.208.79  12.248 ms
10  195.22.196.129  22.943 ms  195.22.196.81  13.345 ms  195.22.196.129  11.678 ms
11  13.104.182.192  11.451 ms  13.104.182.193  11.406 ms *
12  * * *
13  204.79.197.200 <MSS:1440> 13.649 ms * *
    Timeout: false
    Duration: 163.111 ms
    DestinationReached: true
```

Classic TCP traceroute

- Almost each hop has multiple IP addresses
- The destination replied only once

TCP InSession

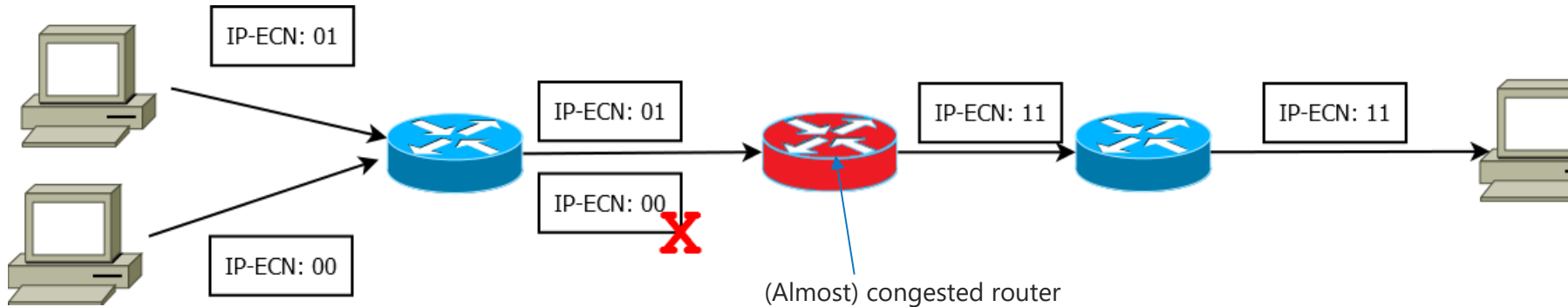
- Each hop has one IP address
- The destination replied to all probes

```
tracert to www.bing.com(95.101.20.193), 30 hops max, 53 byte packet
 1  192.168.0.1  0.490 ms  0.452 ms  0.440 ms
 2  192.168.100.1  5.520 ms  5.511 ms  5.672 ms
 3  * * *
 4  172.18.33.212  5.640 ms  5.632 ms  6.129 ms
 5  172.18.33.230  6.124 ms  6.776 ms  6.774 ms
 6  172.19.184.92  10.463 ms  10.996 ms  10.975 ms
 7  172.19.177.60  10.612 ms  9.895 ms  9.872 ms
 8  151.99.72.197  26.336 ms  26.313 ms  26.302 ms
 9  23.210.57.131  10.219 ms  10.192 ms  10.887 ms
10  95.101.20.193  9.663 ms  12.033 ms  9.457 ms
    Timeout: false
    Duration: 71.904 ms
    DestinationReached: true
    MSS: 1452
```

ECN bleaching detection

ECN mechanism

- *The Addition of Explicit Congestion Notification to IP*, [rfc3168](#), 2001
 - Two bits in the IP header (in TOS/Traffic Class)
- The source declares that a packet should be treated with ECN
 - Set IP-ECN fields either to 01 or 10
- When congestion is (almost) happening, instead of dropping the packet the router sets the IP-ECN fields to 11 (Congestion Experienced)
- The destination reports this event to the source (typically at transport level)



ECN and L4S

- Recently, ECN mechanism got renewed attention due to L4S (Low Latency, Low Loss, and Scalable Throughput – [rfc9330](#), [rfc9331](#), [rfc9332](#) - 2023)
- L4S requires an ECN feedback *more accurate* wrt the “classic” 2001 version
 - “More accurate” = Feedback contains exact counters of IP-ECN values received

L4S

Nokia Bell Labs pioneers L4S enabler for large-scale deployment time applications

WWDC23

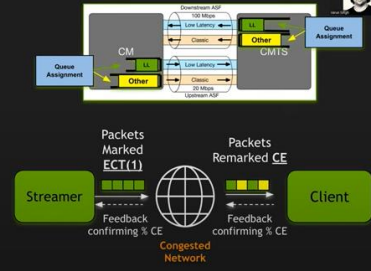
Reduce network delays with L4S

Shawn Zhang, Internet Technologies

Last-Mile: L4S for Cloud Streaming

→ Problem 2: Handling impairments in user's network (bufferbloats, packet loss ...)

- L4S [RFC9332] addresses bufferbloats by allowing sender to react faster to queue build-up vs black-box E2E queue build-up estimation
- Use of L4S requires a compliant TX/RX and network (marking, CE feedback, on-path AQM, and new CC)
- CloudXR 4.0 SDK has initial L4S CC support
- PoC L4S support in GeForce NOW in evaluation



ECN bleaching detection

- Intermediate hops can bleach/alter the value of ECN into the IP header
- See for example: *The Benefits of Using Explicit Congestion Notification (ECN)* – [rfc8087](#), 2017:

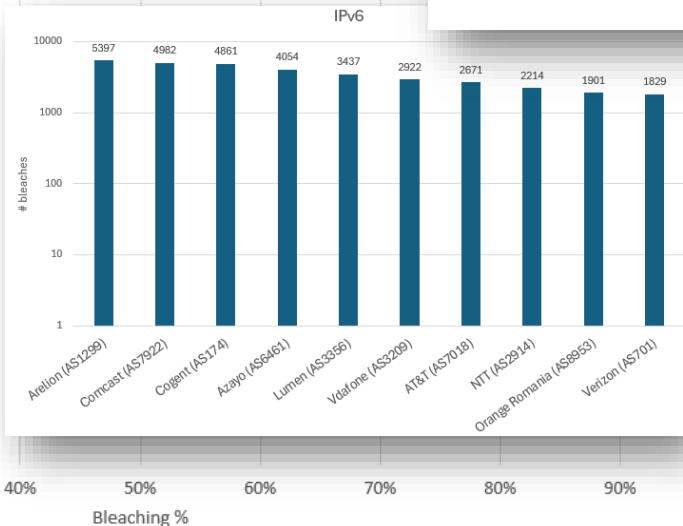
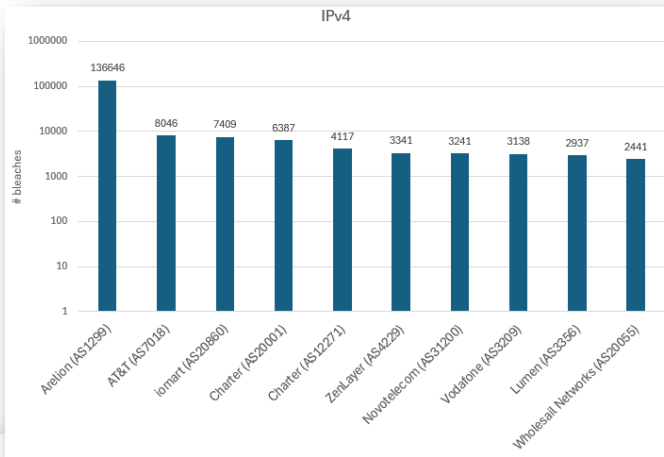
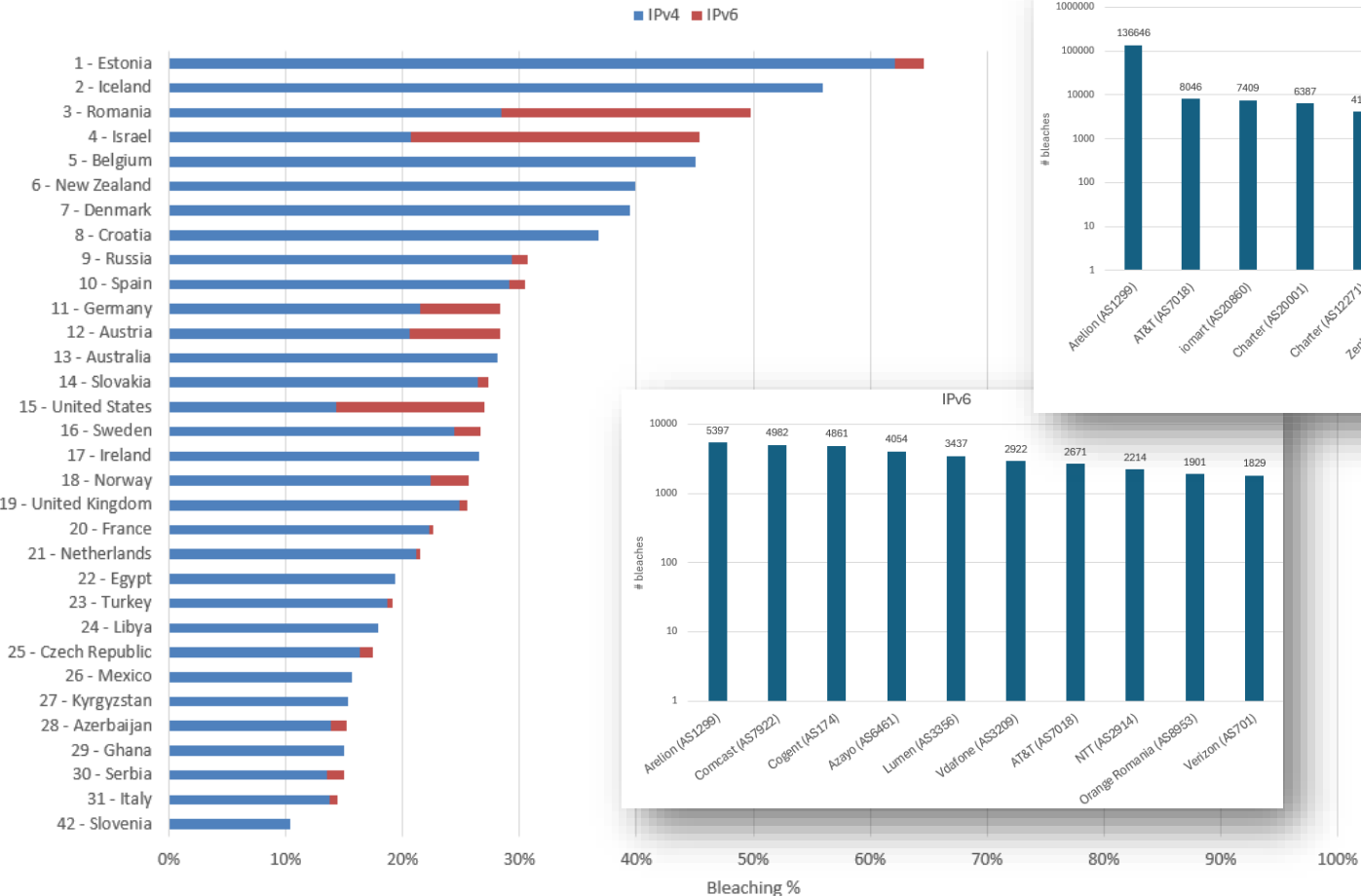
Cases have been noted where an endpoint sends a packet with a non-zero ECN mark, but the packet is received by the remote endpoint with a zero ECN codepoint [TR15]. This could be a result of a policy that erases or "bleaches" the ECN codepoint values at a network edge (resetting the codepoint to zero). Bleaching may occur for various reasons (including normalizing packets to hide which equipment supports ECN). This policy prevents use of ECN by applications.



IP-ECN bleaching in the wild

- We run Pietrasanta traceroute from Catchpoint nodes deployed around the world to understand how many traceroutes show the effects of ECN bleaching
- Period of reference: August 2024
- Besides research curiosity, this can be useful to understand how much the network is prepared to accommodate L4S
- This is not intended to be a rigorous research work
 - The results presented are obviously biased by the node selection
 - We tried to be as fair and distributed as possible in selecting sources and destinations

Results



Analyses and campaigns

Not only traceroute...

- DNS performance: Recently we run a DNS measurement campaign in collaboration with IBM to measure the performance of the global DNS infrastructure
 - <https://www.catchpoint.com/asset/the-need-for-speed>
- BGP analyses
 - BGP outage analyses
 - We are working on a way to report several kinds of BGP routing anomalies (hijacks and leaks, stay tuned!)
- And many more! (feel free to checkout our blog!)
 - <https://www.catchpoint.com/blog>

How to contribute

- Join and participate in expanding our global observability network. Two different ways:
- **Active measurements**
 - Host a Catchpoint synthetic node to run active measurements if you can provide a VPS connected to the Internet
- **Passive measurements**
 - Share a BGP feed with us simply by setting up a multihop session against one of our collectors, more on <https://www.catchpoint.com/bgp/partner>

Thank you!



<https://github.com/catchpoint/Pietrasanta-traceroute>

