Replication: Characterizing MPLS Tunnels over Internet Paths

Jarrett Huddleston
Johns Hopkins University
Baltimore, MD, USA
jhuddle6@jh.edu

Matthew Luckie
CAIDA
UC San Diego
La Jolla, CA, USA
mjl@caida.org

Alexander Marder
Johns Hopkins University
Baltimore, MD, USA
amarder@jhu.edu

Abstract

Traceroute is a critical tool in the Internet measurement toolbox, but its output can be misleading. One problem for traceroute analysis is that certain types of Multiprotocol Label Switching (MPLS) tunnels hide routers from traceroute output. Worse still, there is no simple way to detect or reveal missing routers. Any analysis that expects comprehensive topology discovery—including identifying performance bottlenecks, analyzing traffic engineering approaches, and evaluating traffic sovereignty—needs to account for MPLS.

In this paper, we replicate previous work by Vanaubel et al. [18, 21] to characterize and provide a snapshot of the current deployment of MPLS tunnels. We also release a sustainable and easily deployed tool for MPLS detection, called PyTNT. Using PyTNT, we find that the problematic types of MPLS tunnels remain prevalent, though we inferred a general decrease in MPLS usage across the Internet. We also inferred that public clouds accounted for 3 of the top 10 networks with the most routers observed to be in MPLS tunnels. Finally, we observed more MPLS routers in Europe than any in other continent, and more MPLS routers in the U.S. than any other country.

CCS Concepts

• **Networks** → **Network measurement**.

Keywords

Multiprotocol Label Switching (MPLS)

ACM Reference Format:
Jarrett Huddleston, Matthew Luckie, and Alexander Marder. 2025. Replication: Characterizing MPLS Tunnels over Internet Paths. In *Proceedings of the 2025 ACM Internet Measurement Conference (IMC '25), October 28–31, 2025, Madison, WI, USA.* ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3730567.3764457

1 Introduction

Multiprotocol Label Switching (MPLS) allows for complex traffic engineering across Internet networks, but certain MPLS configurations can confound topology analysis. In IMC 2017, Vanaubel et al. [21] demonstrated that certain types of MPLS configurations hide routers within MPLS tunnels from traceroute. They showed that some MPLS tunnels, referred to as *invisible* tunnels, turn off time-to-live (TTL) propagation, preventing traceroute from inducing responses from routers inside the tunnel. As a result, invisible MPLS tunnels cause a mismatch between the revealed traceroute path and the actual path, since the revealed path creates the false appearance that two routers are directly connected. The implications for the measurement community are clear: any analysis that expects comprehensive topology discovery—including identifying performance bottlenecks, analyzing traffic engineering approaches, and evaluating traffic sovereignty—requires revealing the routers inside MPLS tunnels.

In both the IMC 2017 paper [21] and a subsequent effort from 2019 [18], Vanaubel et al. discovered properties of router implementations that facilitate detection of invisible MPLS tunnels, and proposed a ping- and traceroute-based methodology to reveal MPLS tunnel routers, called TNT. In their 2019 paper, Vanaubel et al. implemented TNT in a hard fork of the scamper tool [14], and ran TNT on 28 vantage points (VPs) to scan ≈2.8 million IPv4 addresses. They found 200K reported MPLS tunnels, mostly deployed by tier-1 or tier-2 ISPs, and 16.5% of the tunnels were invisible; i.e., undetectable without intervention. With these results, Vanaubel et al. concluded that invisible MPLS tunnels hide a substantial number of routers from traceroute.

Replication Contributions. In this paper, we replicate the original TNT experiments to help inform topology analysis, and the Internet measurement community at large, by revealing how MPLS deployments have changed between 2019 and 2025, and how that change impacts conventional active measurement. Our analysis creates a 2025 snapshot of MPLS deployments from VPs similar to the original papers to enable comparison with previous findings. We also analyze the MPLS deployments to understand how much of the topology is unobservable to traceroute due to MPLS, and characterize the expected impact on topology analysis.

Our replication makes three contributions: First, we find that while the number of MPLS tunnel deployments has decreased between 2019 and 2025, the fraction of *invisible* MPLS tunnels remained consistent. In our data, invisible MPLS tunnels hide an average of 5.7 routers per tunnel. Second, we discover that, unlike 2019, the network class with the most observed MPLS tunnel routers in 2025 is a public cloud. In our data, three out of the top ten networks with the most MPLS tunnel routers are public clouds. Third, we confirm the finding from Vanaubel's 2017 paper [21] that high degree nodes—i.e., routers that appear to be interconnected with potentially thousands of other routers—are often explained by the presence of invisible MPLS tunnels.

Additional Contributions. In addition to a strict replication that uses the same methodology and measurement scale as the original studies, we also extend the original TNT studies. Of particular value

Acronym	Meaning	§
MPLS	Multiprotocol Label Switching	2.1
LSE	Label Stack Entry	2.1
LER	Label Edge Router	2.1
LSP	Label Switching Path	2.1
LSR	Label Switching Router	2.1
PHP	Penultimate Hop Popping	2.1
UHP	Ultimate Hop Popping	2.1
FRPLA	Forward/Return Path Length Analysis	2.3
RTLA	Return Tunnel Length Analysis	2.3
DPR	Direct Path Revelation	2.4
BRPR	Backward Recursive Path Revelation	2.4
TNT	Trace the Naughty Tunnels	2.5

Table 1: A glossary of terms used in the discussion of MPLS tunnels and their detection.

to the community, we reimplement the TNT methodology in a way that makes the tool sustainable and easily deployed. Our implementation overcomes the limitations of the original implementation that prevent it from integrating with recent versions of scamper. We also leverage recently discovered state-of-the-art vendor fingerprinting tools to better understand current MPLS deployments. We make the following additional contributions:

- We reimplement the TNT methodology using Python bindings provided by scamper to create PyTNT, a sustainable TNT analogue immediately deployable on any measurement infrastructure using scamper without the need to push new software to the distributed vantage points.
- Using PyTNT, we expand MPLS tunnel detection and revelation beyond 2.8 M destinations to all 12 M routed IPv4 /24s and every vantage point in CAIDA's Ark measurement platform [1].
- We use Hoiho [15] and IPInfo [7] to geolocate routers within MPLS tunnels, inferring more MPLS routers in Europe than any other continent.
- We infer router vendors using SNMPv3 and fingerprinting [8, 9], finding that while Cisco and Juniper dominate the observed tunnel router vendors, networks use routers from a variety of vendors in their MPLS tunnels.
- We operate a continuous two-week run of PyTNT and incorporate the data into CAIDA's August 2025 ITDK and report on the observed MPLS trends.

2 Background

Before delving into our replication efforts, we provide background on MPLS, how different MPLS configurations manifest in traceroute data, detecting MPLS tunnels, and revealing hidden router hops. Table 1 contains a set of MPLS and TNT specific acronyms relevant to the background in order of relevance.

2.1 Multiprotocol Label Switching (MPLS)

Because our work is a replication of the efforts of Vanaubel et al. and their identification of MPLS tunnels [18, 21], our descriptions of MPLS follow their characterization. MPLS routers forward packets

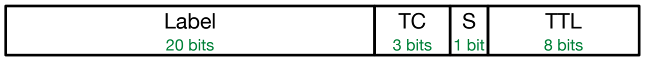


Figure 1: Each Label Stack Entry (LSE) has four components: a 20 bit label; three traffic class (TC) bits for experimental use such as quality of service, priority, or congestion notification; one bit to mark the bottom of the stack (S); and an eight bit time to live (LSE-TTL) field, functioning like the IP-TTL field.

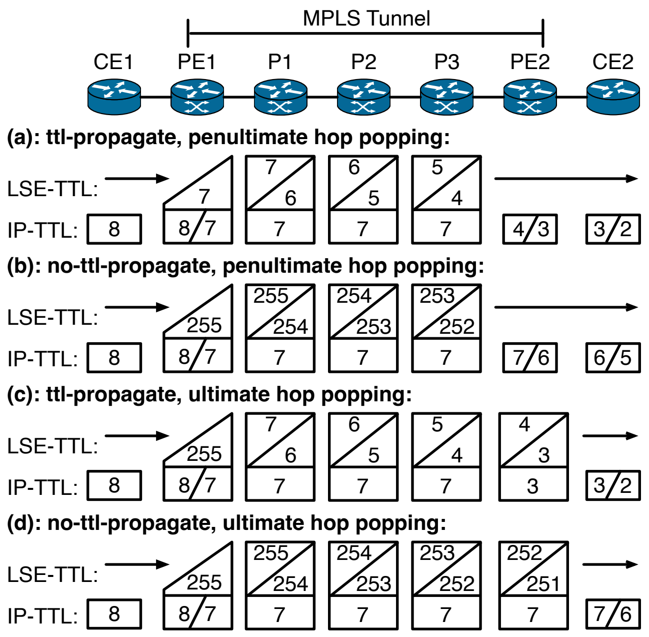

MPLS Tunnel
CE1
PE1
P1
P2
P3
PE2
CE2
(a): ttl-propagate, penultimate hop popping:
LSE-TTL:
IP-TTL:
(b): no-ttl-propagate, penultimate hop popping:
(c): ttl-propagate, ultimate hop popping:
(d): no-ttl-propagate, ultimate hop popping:

Figure 2: Examples of how the IP-TTL is either propagated or not propagated through an MPLS tunnel.

using an exact match on a label, instead of forwarding using the destination IP address. An ingress MPLS Label Edge Router (LER) prepends one or more Label Stack Entry (LSE) (Figure 1) to each IP packet when the packet enters the tunnel.

Figure 2 shows four different MPLS configurations and the effect they have on the MPLS labels. In Figure 2a, the `ttl-propagate` setting is active, so the router PE1 copies the IP-TTL value (7) to the LSE-TTL field. Conversely, when `no-ttl-propagate` is set (Figure 2b), PE1 puts a default value in the MPLS TTL field, rather than the IP-TTL. These settings are configurable in routers, and we return to the impact on traceroute output in §2.2. Once the ingress LER adds the label stack, the packet is forwarded along the Label Switching Path (LSP), which is comprised of individual Label Switching Routers (LSRs). Each LSR will determine how to forward a packet using the first label on the stack. When the packet exits the tunnel, the router copies the lower of the LSE and IP TTL values into the IP-TTL field. If the IP-TTL was propagated, then this will always be the LSE-TTL. Otherwise, the IP-TTL is typically lower than the LSE-TTL and will not be replaced with the LSE-TTL, leaving the IP-TTL set to exactly the same value as when the packet entered the MPLS tunnel. MPLS routers decrement the LSE-TTL

Tunnel Type	Propagates IP-TTL	Adds MPLS extension to ICMP	Impact on Traceroute
Explicit	Yes	Yes	MPLS routers are visible and clearly labeled
Invisible	No	N/A	MPLS routers are not visible
Implicit	Yes	No	MPLS routers are visible but not labeled
Opaque	No	Yes	Only the last router is visible and is labeled

Table 2: An overview of the types of MPLS tunnels, with whether each propagates the IP-TTL, follows RFC 4950, and how it affects traceroutes.

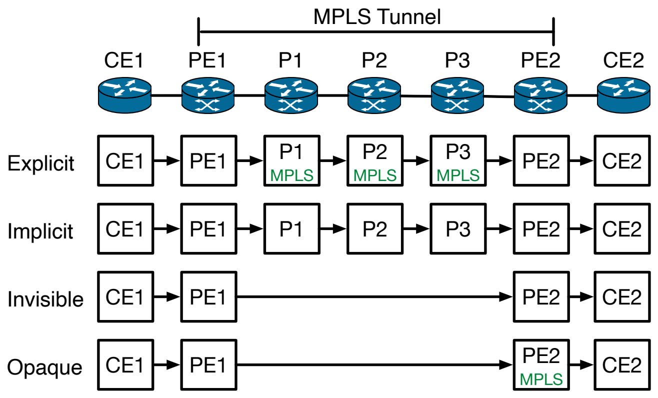


Figure 3: A sample MPLS tunnel layout with ingress LER PE1, LSRs P1 – P3, and Egress LER PE2. Each row of boxes shows which hops will appear in a traceroute between CE1 and CE2. Boxes with "MPLS" include an ICMP extension.

as they forward packets. If the LSE-TTL would be decremented to zero at an LSR, that LSR sends an ICMP time-exceeded message to the origin of the now expired packet.

There are two possible behaviors for when a packet arrives at the end of the LSP. The stack will either be removed at the hop immediately before the egress LER (P3), called Penultimate Hop Popping (PHP), or at the egress LER (PE2), called Ultimate Hop Popping (UHP). PHP reduces the load on the egress LER, as it only needs to do the longest prefix lookup to forward the packet out of the network. UHP puts more burden on the egress LER because it has to manage both the removal of the label stack and the next-hop forwarding. Figures 2c and 2d illustrate the effect that UHP has on TTL values.

2.2 MPLS Tunnel Taxonomy

Donnet et al. introduced a taxonomy of MPLS tunnel types [13, 18, 21] with two key differentiators between the types. These tunnel types cover the potential ways that MPLS tunnels manifest in traceroute output. Table 2 provides an overview of these tunnel types and their characteristics. Notably, the taxonomy outlined by Vanaubel in 2019 [18] builds on work by Vanaubel and Donnet in 2017 and 2012 respectively [13, 21].

One important characteristic is whether the ingress LER propagates the IP-TTL from a traceroute probe packet into the TTL field in the MPLS header. In Figure 2a, PE1 copies the current IP-TTL into the MPLS label stack as the value for the LSE-TTL, which allows the packet to expire inside the tunnel. When the IP-TTL is not copied into the LSE (Figure 2b) the probe packet TTL should never expire inside the tunnel, since the default value used for the LSE-TTL should always be large enough to traverse the tunnel. Consequently, the routers in a `no-ttl-propagate` tunnel should never respond to a traceroute probe traversing the tunnel.

The second important characteristic is whether MPLS routers include MPLS extensions in ICMP packets they generate in response to a packet's TTL expiring in a tunnel. Routers that include MPLS extensions in ICMP packets follow the guidelines set in RFC4950 [3]. Along with explicitly confirming that the packet inducing a response was traversing an MPLS tunnel, the extension also includes the LSEs from the MPLS header.

Figure 3 shows each category in the taxonomy and how they manifest in traceroute.

Explicit Tunnels (Exp). When operators configure MPLS tunnels with the `ttl-propagate` option and use MPLS extensions in ICMP responses, they create what Vanaubel et al. refer to as an explicit MPLS tunnel. All routers will appear in traceroute output and the responses will explicitly state that they belong to an MPLS tunnel.

Implicit Tunnels (Imp). Implicit tunnels also enable the `ttl-propagate` option, but do not include MPLS extensions in the ICMP responses. All of the routers are visible to traceroute, but the ICMP responses from the LSRs will not indicate that the probe packet was traversing an MPLS tunnel.

Invisible Tunnels (Inv). Invisible tunnels hide the routers along the LSP from conventional traceroute using the `no-ttl-propagate` option. Since the IP-TTL is not propagated into the MPLS label stack, traceroute probe packet TTLs cannot expire while traversing the MPLS tunnel. The probe packet TTLs can only expire after the MPLS header, with its separate TTL, is removed from the packet. As a result, traceroute will indicate that the ingress and egress LERs are adjacent despite the presence of the LSRs in between.

Opaque Tunnels (Opq). Opaque tunnels use MPLS ICMP extensions but set the `no-ttl-propagate` option. Opaque tunnels are also the result of when the MPLS network does not guarantee that a valid tunnel extends up to the egress LER. As a result, the probe packet will not arrive at the intended egress LER and the label stack will be removed abruptly, causing PE2 to report an MPLS label when it otherwise should not as in the case of the opaque tunnel in Figure 3. According to Vanaubel et al. [18], opaque tunnels are unique to specific models of Cisco routers. This is a revision from Donnet's earlier definition of opaque tunnels [13], which were MPLS tunnels that include MPLS headers in ICMP responses but do not propagate the IP-TTL, a definition now encompassed by invisible tunnels.

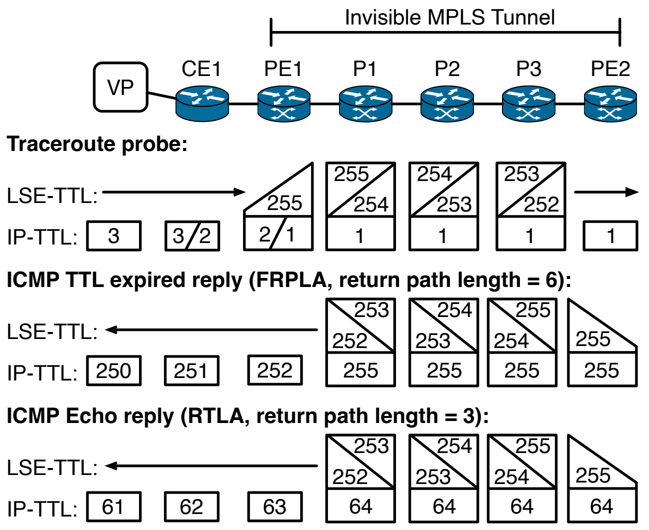


Figure 4: Examples of FRPLA and RTLA for packets through an invisible MPLS tunnel. Because the IP-TTL is not propagated to the LSE-TTL in an invisible tunnel, PE1 initializes the LSE-TTL to 255. The IP-TTL does not expire until PE2, at which point PE2 will send an ICMP time-exceeded packet, and the forward path appears to be three hops (CE1, PE1, PE2). PE2 initializes both IP-TTL and LSE-TTL of the TTL expired reply to 255, but only the LSE-TTL will be decremented through the tunnel. When the packet exits the tunnel, P1 will copy the LSE-TTL to the IP-TTL and the return path will appear to be six hops long (255-249), forming the basis of FRPLA. JunOS routers initialize the IP-TTL of an echo-reply packet to 64, so the return path appears to only be three hops long (64-61), forming the basis of RTLA.

2.3 MPLS Tunnel Detection

Explicit MPLS tunnels are immediately identifiable due to MPLS extensions in ICMP responses, and tools such as scamper [14] record them in their traceroute output. The remaining types of MPLS tunnels cannot be easily recognized in traceroute output, and each require different techniques for identifying that a traceroute path contains an MPLS tunnel.

2.3.1 *Invisible Tunnel Detection.*

There are three methods to detect invisible tunnels, two for PHP (Forward/Return Path Length Analysis and Return Tunnel Length Analysis) and one for UHP (Duplicate IP Addresses).

Forward/Return Path Length Analysis (FRPLA). Vanaubel et al. discovered uncommon IP header values in traceroute responses for invisible tunnels that provides a strong indicator that the path contains MPLS. For invisible tunnels, where the IP-TTL is not copied into the MPLS header, the MPLS tunnel routers will not decrement the IP-TTL and the ingress and egress LERs will appear directly connected in traceroute output. This behavior is seen in Figure 4, where the IP-TTL is not decremented between PE1 and PE2, so P1 – P3 will not appear in a traceroute through the tunnel.

However, when conducting a traceroute through an invisible tunnel, the ICMP reply packet from the egress LER will indicate the presence of additional hops prior to the ingress LER through its reply TTL. The reason is that when the reply packet exits the MPLS tunnel on its way back to the traceroute source, the LER will populate the packet's IP-TTL with whichever value is lower between the IP-TTL and the LSE-TTL. This will always be the LSE-TTL, because both TTL values were instantiated at the egress LER, but only the LSE-TTL was decremented through the tunnel.

The key insight is that when the reply packet exits the MPLS tunnel, the IP-TTL will be set to the default TTL value used on the originating router, minus the number of router hops in the tunnel. Figure 4 shows this phenomenon, with what appears to be a forward path length of three, but a return path length of six. Vanaubel et al. also showed that it is possible to infer the default value—as well as the number of router hops on the reply path—since nearly all routers use either 64, 128, or 255 as the initial IP-TTL value. However, because return paths do not necessarily have to match the forward path, there is some natural variance in their respective lengths. As a result, in the context of single traceroute FRPLA cannot precisely identify presence of a tunnel or provide the exact number of hops in the tunnel. FRPLA instead can be used in a statistical analysis on a larger scale to suggest a likely MPLS tunnel, or in the individual case to identify traceroutes for further investigation.

Return Tunnel Length Analysis (RTLA). RTLA functions similar to FRPLA, except it is able to infer the exact length of an invisible tunnel because of a behavior specific to Juniper's JunOS routers. Vanaubel et al. report that JunOS routers use different initial TTL values for ICMP Echo Reply packets (ping responses) than they do for Time Exceeded packets (traceroute responses) and MPLS header TTLs [22]. While JunOS routers use an IP-TTL of 255 for Time Exceeded packets and for the initial TTL value for LSEs that the router prepends to the reply packet, they use an IP-TTL of 64 for Echo Replies. Knowing this phenomenon, a ping to the router can reveal the exact length of the invisible MPLS tunnel.

When a Time-exceeded packet traverses the tunnel on a return path, as with FRPLA, the LSE-TTL will be copied into the IP-TTL field. However, because the echo-reply TTL is 64 for a JunOS router, it will always be lower than the value of the LSE-TTL exiting the tunnel, which was initialized to 255. Therefore, a ping to a router suspected to be the egress LER for an invisible tunnel will result in a response with a TTL indicating a shorter path than a time-exceeded response from that same router, which also used an initial TTL value of 255. The difference in the inferred number of TTL decrements for ICMP echo replies and ICMP time exceeded messages allows TNT to infer the number of hops inside the invisible tunnel. For example, in Figure 4, the echo-reply packet TTL implies three hops taken while the time-exceeded packet implies six, indicating a tunnel length of three hops.

Checking for Duplicate IP Addresses. Unlike an invisible tunnel using PHP, a traceroute through an invisible tunnel using UHP will not show the egress LER of the tunnel; e.g., PE2 in Figure 4. The reason for this behavior is that the LSE-TTL is decremented before the label stack is removed, so the earliest the IP-TTL could expire in UHP tunnels is after the tunnel (e.g. CE2 in fig. 2) As a result, RTLA and FRPLA will not work as they both rely on a traceroute probe timing out at the egress LER.

However, Vanaubel et al. observed a specific behavior in some versions of the Cisco router OS that allows these tunnels to be inferred. When receiving a packet with an IP-TTL of 1, the egress LER will not decrement the TTL and will instead forward it to the next hop. The egress LER will then not appear in the traceroute but the subsequent hop will appear twice, as it will receive the packet meant for the egress LER and its own intended probe. Identifying when an IP address is duplicated in consecutive traceroute hops can detect a potential invisible UHP tunnel.

2.3.2 *Implicit Tunnel Detection.*

Implicit tunnels, where the LSRs appear but are not annotated with MPLS extensions, can be identified using either quoted TTLs (qTTLs) in the ICMP responses to traceroute probes or by examining return path lengths for different ICMP packets. If the LSE-TTL of a packet in an MPLS tunnel expires, the LSR will respond with an ICMP time-exceeded packet that quotes the beginning of the IP packet, including the IP-TTL. This IP-TTL, referred to as the quoted TTL, should be greater than one as it would not be decremented while the packet was in the MPLS tunnel, since only the LSE-TTL is decremented. Therefore, each time-exceeded packet returned by LSRs in the tunnel will have an increasing qTTL as traceroute probes bound for subsequent hops would need higher IP-TTLs. Therefore, an implicit MPLS tunnel can be inferred by identifying where the qTTLs are both greater than one and increasing in subsequent hop responses.

An alternate method to infer implicit MPLS tunnels relies on using the difference between return path lengths for ICMP time-exceeded and echo-reply packets. For certain router manufacturers, when a packet times out at an LSR, a time-exceeded packet will be sent first back to the ingress LER of the tunnel, which will forward it to the appropriate destination. Other types of traffic, however, will be sent directly to the source of the probe packet. As a result, the return path for an echo-reply packet will be expected to be shorter than the return path for a time-exceeded packet as it does not need to traverse back through the tunnel. This difference in path lengths for time-exceeded and echo-reply packets can suggest the presence of an implicit tunnel.

2.3.3 *Opaque Tunnel Detection.*

Opaque tunnels will appear in a traceroute as a single hop with a quoted TTL (qTTL) not being equal to one. For example, in the opaque tunnel in Figure 3, the only hop from the tunnel shown is for PE2, which includes the MPLS extension and includes the quoted TTL in the response for the LSE-TTL. We can detect opaque tunnels by identifying these isolated hops in the traceroute and checking the qTTL in the response. For a regular traceroute probe destined for PE2, such as with an explicit tunnel, the qTTL would be expected to be one. However, because the IP-TTL is not propagated in an opaque tunnel, the qTTL will be the value of the LSE-TTL at the final hop in the tunnel which will always be between 1 and 255.

2.4 MPLS Router Revelation

For explicit and implicit tunnels, all hops in the tunnel appear in the traceroute and thus do not need to be revealed. Vanaubel also found no way to reveal the hops found in opaque tunnels due to how the label stack is removed. However, for invisible tunnels using PHP, there are two ways to reveal the individual routers hidden by the tunnel: Direct Path Revelation and Backward Recursive Path Revelation.

2.4.1 *Direct Path Revelation.*

Direct Path Revelation (DPR) relies on the common operational reality that operators do not have to use MPLS tunnels to reach internal IGP prefixes. With this configuration, which is the default on Juniper devices and configurable on other vendors, a traceroute to an internal address, such as the egress LER of a tunnel, will reveal all intermediary hops that would otherwise be hidden from traceroute.

2.4.2 *Backward Recursive Path Revelation.*

Backward Recursive Path Revelation (BRPR) works even when a network deploys MPLS to reach internal prefixes, provided the routers use PHP. With PHP, the last hop of the tunnel is visible to traceroute because the label stack is removed at the prior hop. For example, in the invisible tunnel in Figure 3, PE2 is visible in the traceroute to CE2 because the label stack is removed at P3. When the destination of a packet is the egress LER, in this case PE2, then the labels will be distributed such that the tunnel ends at the prior hop, P3. As a result with P3 as the final hop in the tunnel, the label stack will be removed at the new penultimate hop P2, so P3 and PE2 will both appear in the traceroute. As the name implies, this method can be applied recursively working from the end of the tunnel back toward the traceroute source until it reveals all routers in the MPLS tunnel.

2.5 Trace the Naughty Tunnels

In TMA 2019, Vanaubel et al. published an analysis of MPLS tunnel deployments across the Internet. They conducted their measurements with a tool they developed named TNT (Trace the Naughty Tunnels), which was implemented based on a hard fork of scamper [14]. TNT implements the techniques discussed in §2.3 to identify and the techniques in §2.4 to reveal MPLS tunnels.

3 Creating a Modern and Sustainable Implementation of TNT (PyTNT)

Before replicating any MPLS analysis, we first needed to reimplement TNT. The original implementation of TNT forked the May 23, 2018 release of scamper's source code, and does not work with later versions of scamper. This is because TNT relies on a modified data format that is incompatible with subsequent scamper releases. This implementation limitation in part led CAIDA to gradually phase out TNT measurements to the point that no Ark vantage points (VPs) currently run TNT measurements.

Our implementation, PyTNT, leverages newly available Python bindings to recreate TNT's methodology exactly [18]. Unlike a hard fork of the scamper source code, the Python bindings are officially supported by scamper, and will continue to be maintained in future scamper versions. PyTNT can also run in any environment where scamper can be deployed, since it can control a local or remote scamper process using sockets or a scamper mux, a multiplexed interface to a collection of remote scamper instances. This includes the ability to run on measurement platforms like Ark, where we are working to restart TNT measurements from an even wider set of VPs.

Listing 1: PyTNT main function

```
1  pytnt(targets, initial_traces):
2    pings = []
3    blocked = []
4    traces = []
5    if len(initial_traces) > 0:
6      # read in seed traces
7      for trace in initial_traces:
8        traces.append(trace)
9        find_pings(trace, queue)
10   else:
11     for target in targets:
12       # issue initial traces
13       trace = do_trace(target)
14       traces.append(trace)
15      find_pings(trace, queue)
16   ping_results = do_pings(queue)
17
18   extra_traces = []
19   for ping in ping_results:
20     for trace in blocked:
21       process_ping(trace)
22
23   extras = issue_traces(extra_traces)
24   for extra in extras:
25     trace = find_trace(extra)
26     trace.process_trace(extra)
27
28   for trace in traces:
29     print(trace)
```

PyTNT begins by taking in either an initial set of already-run traceroutes such as from team probing [4] or a set of destinations to probe itself with traceroute. From that set of initial traceroutes, PyTNT finds every unprobed router IP address in the traceroutes (Listing 1, lines 9 and 15) and issues pings to every IP address in that set (line 16) to derive the initial TTL for both ICMP echo-reply and ICMP time-exceeded packets. If no initial traces are given, PyTNT distributes any additional probes between available VPs. If there are initial measurements, each additional probe is issued from the VP of the corresponding traceroute. PyTNT uses the probes to each router to determine both the presence of potential tunnels according to the techniques in §2.3 and any subsequent probes needed to attempt to reveal tunnel hops in accordance with the techniques in section §2.4. Finally, as each additional measurement is received, PyTNT will process it to add revealed hops to the original trace and issue any more measurements needed, typically in the case of BRPR. Once all measurements are done, PyTNT will output the list of traces annotated with any inferred tunnels and revealed hops.

Enhancements. In addition to having the same MPLS tunnel detection capabilities as TNT, we added an additional feature to PyTNT. PyTNT allows for the use of existing traceroutes produced by scamper to initialize the probing such that PyTNT integrates more easily into CAIDA's existing measurement campaigns, such as providing MPLS tunnel information as part of future ITDK releases.

Test	Total	Explicit	Invisible	Opaque	Implicit
PyTNT 1	30,360	23,383	1,585	709	4,683
PyTNT 2	30,125	23,475	1,586	678	4,386
PyTNT 3	30,330	23,312	1,582	710	4,726
Average	30,271.7	23,390.0	1,584.3	699.0	4,598.3
TNT 1	33,862	27,044	1,544	703	4,571
TNT 2	32,668	24,241	1,737	716	5,974
TNT 3	30,475	23,894	1,651	725	4,205
Average	32,335.0	25,059.7	1,644.0	714.7	4,916.7

Table 3: Tunnels identified by PyTNT and TNT in the cross-validation testing.

Comparing PyTNT to TNT. We compared PyTNT with the original TNT implementation. From a server at our institution, we conducted measurements from TNT and PyTNT to the same set of 660,058 destination IP addresses. We derived this list from the RouteViews prefix to AS mapping [6] collected on March 10th, 2025, selecting one IP address per observed /24 prefix. We probed the full set of destinations three times each with PyTNT and TNT. Table 3 summarizes the tunnels identified by the respective tools. While PyTNT detected slightly fewer tunnels than TNT, the differences can be explained at least in part by normal changes in routing topology or by certain hops being unresponsive to traceroute in one test versus another.

4 Replicating MPLS Tunnel Detection

Our goal with this replication is to provide a snapshot MPLS deployment 2025, examine how MPLS deployments have changed from 2017 [21] and 2019 [18] to 2025, and provide a tool that the Internet measurement community can use to study MPLS going forward. Aside from studying MPLS, this work is critically important for Internet topology analysis, since invisible MPLS tunnels hide routers from traceroute and create the appearance of a topology fundamentally different from reality.

Our replication effort focuses on two papers. The first, *Through the Wormhole: Tracking Invisible MPLS Tunnels* [21], outlines techniques able to identify the presence of MPLS tunnels in a traceroute path that would otherwise be undetected. The second paper, *TNT, Watch me Explode: A Light in the Dark for Revealing MPLS Tunnels*, expands the framework provided in the first paper to account for a wider array of tunnels, and provides a tool that Vanaubel et al. use to conduct an analysis of MPLS tunnel deployments.

4.1 Measurement campaign

We first replicated the experiment of running TNT on CAIDA's Ark measurement platform [1]. In May 2025, Ark used 262 VPs to continually conduct traceroutes to the entire routed IPv4 address space in cycles, where each cycle conducts a single traceroute toward a random IP address in each IPv4 /24. In a given cycle, each traceroute

Year	2019		2025		2025		2025	
Tunnel Type	TNT 28 VP		PyTNT 62 VP		PyTNT 262 VP		PyTNT ITDK	
Invisible (PHP)	28,063	14.4%	22,980	14.8%	78,860	18.5%	1,000,295	15.3%
Invisible (UHP)	4,122	2.1%	943	0.6%	2,716	0.6%	152,323	2.3%
Explicit	150,036	76.7%	128,345	82.6%	334,928	78.4%	5,160,941	79.1%
Implicit	9,905	5.1%	1,698	1.1%	7,090	1.7%	174,839	2.7%
Opaque	3,346	1.7%	1,464	0.9%	3,527	0.8%	34,139	0.5%
Total	195,525		155,421		427,121		6,522,537	

Table 4: The distribution of the types of tunnels identified in our measurement campaign compared to the values found in the 2019 TMA paper.

Continent	TNT 2019		2025 62 VP		2025 262 VP	
Europe	9	32.1%	19	30.6%	76	29.0%
North America	11	39.3%	23	37.1%	123	46.9%
South America	1	3.6%	4	6.5%	16	6.1%
Asia	4	14.3%	9	14.5%	30	11.5%
Australia	3	10.7%	7	11.3%	11	4.2%
Africa	0	0.0%	0	0.0%	6	2.3%
Total	28		62		262	

Table 5: Continental distribution of the VPs used for the original experiment (TNT 2019), our replication effort (2025 62 VP), and all Ark VPs included in CAIDA's traceroute probing campaign (2025 262 VP). We used the full set of VPs for our extended experiment.

destination is randomly assigned to one of the 262 Ark VPs, and Ark distributes traceroutes towards the destinations across the VPs.

In the original experiment, Vanaubel et al. ran TNT on 28 Ark VPs to a combined ≈2.8 M destinations [18]. Communicating with the TNT authors confirmed that the ≈2.8 M IP addresses used for the experiment exactly corresponded to the ≈2.8 M destinations that were assigned to the 28 VPs in a single cycle at the time of their experiment. We show their results in Table 4 (TNT 2019). Notably, these implicit, opaque, and invisible tunnels would have otherwise been undetected by traceroute. They also were able to reveal 32,267 LSRs previously hidden by invisible MPLS tunnels.

For our replication, we expanded the 28 VP experiment for the original 2019 iteration of TNT to run PyTNT on all 262 available Ark VPs. We bootstrapped PyTNT with traceroutes from a cycle already conducted by CAIDA to all 11.9 M routed /24s, similar to how we envision PyTNT being deployed on Ark long-term. PyTNT conducted the necessary pings and subsequent traceroute probing needed to identify and reveal any tunnels. The full probing took ≈8 hours to complete, although most VPs finished faster.

To recreate the experiment from the Vanaubel's 2019 TNT paper [18], and facilitate meaningful comparison, we downsampled our collected data to 2.8 M destinations. Ark uses more VPs in 2025 than were available in 2019, so each VP is responsible for fewer destinations in each cycle. In order to run PyTNT in a similar scope to the original TNT experiment, we used a distribution of traceroutes conducted from 62 VPs towards 2.8 M destinations rather than the original experiment's 28 VPs. We ensured that the VPs we selected maintained the same balance across continents as the original experiment, allowing us to meaningfully compare our results. As seen in Table 5, the VPs we included for our replication (2025 62 VP) closely follows the VP continent distribution from the original 2019 TNT experiment.

The 2025 62 VP column in Table 4 compares our results with the results of the original experiment (TNT 28 VP). Our probing revealed 20.5% fewer MPLS tunnels compared to 2019, indicating a potential trend away from MPLS deployments. Most importantly, explicit tunnels appear to be replacing invisible UHP, implicit, and opaque tunnels, which are primarily artifacts of unusual router configurations [18]. At the same time, the fraction of invisible PHP tunnels—the most problematic type of MPLS tunnel for traceroute analysis—remains consistent over the last 5 years.

After comparing to the historical 2019 MPLS results, we extended our analysis to the full set of traceroutes to every /24 (Table 4 Full 2025), providing a snapshot of MPLS deployments observable today. The proportion of tunnel types remains similar to the 2025 62 VP, and retains the same trends, lending confidence to the reduction in MPLS deployments we observed compared to 2019.

In addition to our experiments replicating the TNT deployment, we deployed PyTNT to run continuously for two weeks in order to collect data for CAIDA's August 2025 ITDK. Table 4 shows the distribution of unique tunnels identified in that ITDK. While many more tunnels were found than in the previous experiments, the proportions of reported tunnels is similar to what was previously reported. Explicit tunnels make up the significant majority of observed tunnels, followed by invisible, implicit, and then opaque.

We further extended our analysis to investigate the number of routers hidden by invisible tunnels, when a traceroute path traverses the invisible tunnel. Figure 5 shows the distribution of hops that were revealed in the invisible tunnels inferred as part of the May 2025 262 VP experiment. This probing revealed on average 5.7 routers hidden from traceroute output per tunnel, demonstrating the necessity of accounting for invisible MPLS tunnels in topology analysis. Even in the 21.4% of cases where we could detect an invisible tunnel but not reveal any hops, the detection could let researchers know that information is missing from the topology.

In addition to the number of tunnels identified, we examined how many traceroutes contained an MPLS tunnel. We observed

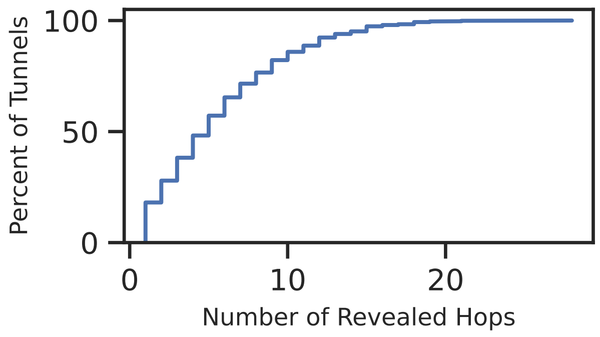


Figure 5: A CDF for the number of revealed hops in invisible MPLS tunnels. This does not include 15,752 cases where no hops could be identified.

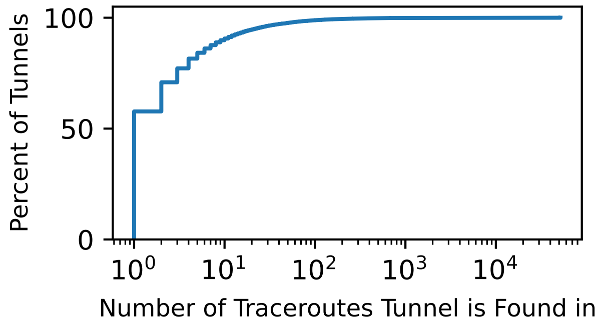


Figure 6: A CDF for the number traceroutes each reported MPLS tunnel was found on.

Vendor	Count	255,255	255,64	64,64	Other
Cisco	377785	99.8%	0.1%	0.0%	0.1%
Huawei	89640	99.9%	0.0%	0.0%	0.0%
MikroTik	60224	0.2%	0.5%	99.2%	0.1%
H3C	24439	99.8%	0.0%	0.0%	0.1%
Juniper	11228	0.1%	99.6%	0.2%	0.1%
OneAccess	11032	61.7%	0.3%	37.9%	0.1%
Nokia	5709	0.1%	0.9%	99.0%	0.0%
Ruijie	4590	0.1%	0.6%	83.5%	15.8%
Other	43679	10.6%	4.5%	82.3%	2.7%
Total	628326				

Table 6: Initial TTLs inferred for IPv4 router interfaces that sent time exceeded messages, echo replies, and provided a vendor via SNMPv3 probes. In the August 2025 ITDK, most self-identifying Juniper routers used an initial TTL of 255 for ICMP time exceeded messages, and 64 for ICMP echo replies.

that in the 11.9 M traceroutes, 61.0% (7,056,488) contained at least one tunnel. 53.4% of traceroutes (6,177,152) had explicit tunnels, 11.0% (1,269,837) had invisible tunnels, 0.9% (109,312) had implicit tunnels, and 0.5% (61,369) had opaque tunnels. Figure 6 shows how frequently the MPLS tunnels appeared in the traceroutes. Half of the tunnels appeared in a single traceroute, ≈80% appeared in ten or fewer, ≈10% in at least 100 traceroutes, and the most prolific tunnel appeared in 317,015 traceroutes. That almost 11% of traceroutes contained at least one invisible tunnel highlights the importance of being able to measure the presence of invisible tunnels and the ability to reveal the routers within the tunnel for accurate Internet topology measurement.

Takeaway: Compared to 2019, MPLS deployments appear less prevalent, but invisible PHP tunnels remain prevalent with the potential to obscure a significant number of routers from traceroute output.

4.2 Most Common Vendors in MPLS Tunnels

Vanaubel's tunnel revelation paper from 2017 [21] used the router vendor fingerprinting technique proposed in 2013 by Vanaubel et al. [22] as part of the MPLS tunnel detection. Specifically, TNT uses inferred initial TTLs for ICMP time-exceeded and echo-reply messages to determine whether to use RTLA or FRPLA, as RTLA infers the exact tunnel length but only works with Juniper routers. To begin, we investigated the degree to which Juniper routers still use IP-TTL signatures that trigger RTLA (§2.3.1) – an initial TTL of 255 for ICMP time exceeded messages, and an initial TTL of 64 for ICMP echo replies. CAIDA's August 2025 ITDK process sent SNMPv3 probes to all inferred router interfaces, which induce some routers to self-identify their vendor [8]. The ITDK process also sent ICMP echo requests to these same router interfaces, and we collected TTLs from time exceeded messages recorded in the team-probing traceroutes covered by the August 2025 ITDK. Table 6 shows that we inferred nearly all observed Juniper routers continue to use a (255, 64) TTL signature. We therefore used the same fingerprinting technique proposed in 2013 by Vanaubel et al. [22] – used in TNT – to also trigger tunnel inference in PyTNT.

Next, we used both SNMPv3 [8] and light-weight vendor fingerprinting techniques proposed by Albakour et al. [9] to analyze the router vendors used by networks for their MPLS tunnels in the May 2025 262 VP experiment. Our analysis with PyTNT yielded a set of 346,610 unique IP addresses belonging to MPLS tunnels. We were able to identify the router vendor for 95,013 of these addresses. We identified vendors for 49,390 routers by the routers disclosing their own manufacturer in the SNMPv3 probe response. The remaining 45,623 addresses had vendors inferred using light-weight vendor fingerprinting from the March 2025 ITDK process.

Table 7 identifies the nine most frequent router vendors encountered in MPLS tunnels. In our data, Cisco was the most frequently encountered router vendor in all tunnel types by a significant margin. Juniper routers, although not as common as Cisco, were still much more prevalent in all four MPLS tunnels varieties than even the third most common vendor, MikroTik. Overall, Cisco and Juniper made up 90.5% of fingerprinted routers, the remaining 9.5% of routers were manufactured by 67 other vendors.

Table 8 provides the distribution of MPLS routers observed in the August 2025 ITDK by vendor type. Nine of the ten most frequent vendors are the same between that ITDK and our May 2025 262 VP PyTNT dataset, with the only exception being the tenth most

Vendor	Explicit	Invisible	Implicit	Opaque
Cisco	43,396	6,638	1,899	2,517
Juniper	23,405	4,865	1,248	58
MikroTik	2,864	225	102	0
Huawei	2,187	180	136	7
Nokia	1,229	870	407	0
H3C	322	73	52	1
OneAccess	346	31	7	0
Juniper/Unisphere	251	49	0	0
Brocade	86	42	4	0

Table 7: The most frequent router manufacturers observed to be present in MPLS Tunnels by SNMP probing from the May 2025 262 VP experiment broken down according to the tunnel taxonomy in §2.

Vendor	Explicit	Invisible	Implicit	Opaque
Cisco	128,896	24,884	24,339	7,419
Juniper	83,373	12,972	3,580	48
MikroTik	12,116	1,647	2,824	1
Nokia	4,385	2,078	4,489	2
Huawei	4,185	2,536	1,600	8
H3C	767	655	2,569	1
TELDAT, S.A.	152	17	1,015	0
OneAccess	736	118	115	1
Juniper/Unisphere	348	118	2	0
SonicWall	265	3	168	0

Table 8: The most frequent router manufacturers from the August 2025 ITDK observed to be present in MPLS tunnels by SNMP and LFP probing according to the tunnel taxonomy in §2.

common vendor of MPLS routers in the ITDK is SonicWall instead of Brocade. In total, 332,886 router vendors were reported directly by SNMPv3 probes [8] or inferred with light-weight fingerprints [9] in the August 2025 ITDK, and these ten most frequent vendors comprised 98.9% of routers with a vendor annotation.

Takeaway: Cisco and Juniper routers are by far the most prevalent router manufacturers across all examined types of MPLS tunnels, but networks use routers from a variety of vendors in their MPLS tunnels.

4.3 Networks Where MPLS is Found

We inferred the Autonomous Systems (ASes) operating the routers for 86.2% of the 206,857 router IP addresses observed in MPLS tunnels using bdrmapIT [16]. Table 9 shows the 10 ASes that most frequently used MPLS in our May 2025 262 VP dataset. Most are large ISPs, but the three largest public clouds (Amazon, Microsoft, and Google) all appear in the list, indicating that MPLS is common in public cloud networks. Nearly all of the ASes used explicit tunnels far more than implicit tunnels, with nearly all top ASes skewing more toward explicit tunnels than the May 2025 262 VP dataset at large. For one AS, Spectrum in the U.S., we never observed an invisible MPLS tunnel. Another surprising result is the disproportionate use of implicit tunnels in Telefonica ES (23.8%), since implicit tunnels comprised less than 2% of all MPLS tunnels we observed. Interestingly, these ISPs only account for a single observed opaque routers, which were more commonly observed in smaller providers.

ISP (AS)	Explicit	Invisible	Implicit	Opaque
Amazon (16509)	9,487	32	189	0
Telefonica DE (6805)	7,014	72	97	0
Telefonica ES (3352)	4,318	2	1,347	0
Microsoft (8075)	5,132	167	2	0
Spectrum (33363)	4,148	0	4	0
Google (15169)	3,792	27	2	0
Vodafone (3209)	1,645	1,016	8	0
Viettel (7552)	1,934	36	450	0
Kaztelecom (9198)	2146	17	0	0
Claro (4230)	1,546	132	383	0

Table 9: The most frequent ISPs observed to operate MPLS tunnels broken down according to the tunnel taxonomy in §2. Most of these ISPs appear to use a mix of tunnel types, likely reflecting a variety of routers and legacy deployments, although we never observed invisible tunnels within or Spectrum.

ISP (AS)	Exp	Inv	Imp	Opq
Telefonica DE (6805)	2,414	14	226,985	0
Telefonica ES (3352)	70,115	5	90,458	0
Viettel (7552)	11,255	93	32,669	0
Telia (3301)	4,430	70	37,379	0
Amazon (16509)	33,380	134	711	0
Tele2 (1257)	3,148	6	31,034	0
V.Tal (8167)	1,231	5	30,656	0
Google Fiber(16591)	1,287	7	25,769	0
Meditelecom (36925)	130	0	25,627	0
China Unicom (4837)	17,771	6,052	22	1

Table 10: The most frequent ISPs observed to operate MPLS tunnels from the August 2025 ITDK broken down according to the taxonomy in §2. Most of these ISPs appear to use a mix of tunnel types, likely reflecting a variety of routers and legacy deployments, although we never observed invisible tunnels within Meditelecom.

Table 10 shows the ISPs that operate the most MPLS routers from the August 2025 ITDK as inferred with bdrmapIT [16]. Similar to differences in vendor patterns between the May 2025 262 VP PyTNT and August 2025 ITDK datasets (§7,8), the distribution of MPLS routers among ASes in the August 2025 ITDK did not follow the general AS distribution observed in the May 2025 VP dataset, however there were many more implicit routers inferred in the August 2025 ITDK. With the exception of Amazon, each of the most frequently identified ISPs have significantly more implicit MPLS

Continent	MPLS Routers	
Europe	77,047	37.6%
North America	72,102	35.2%
Asia	32,431	15.8%
South America	11,235	6.6%
Africa	5,221	2.5%
Australia	4,635	2.3%

Table 11: Continent locations of the router interface IP addresses used in MPLS tunnels in our May 2025 262 VP dataset.

routers than explicit, and none were found to have any opaque tunnels. A likely explanation for this behavior is that implicit tunnels are much more concentrated, only found in 5,236 ASes, while we observed explicit tunnels in 31,733 ASes.

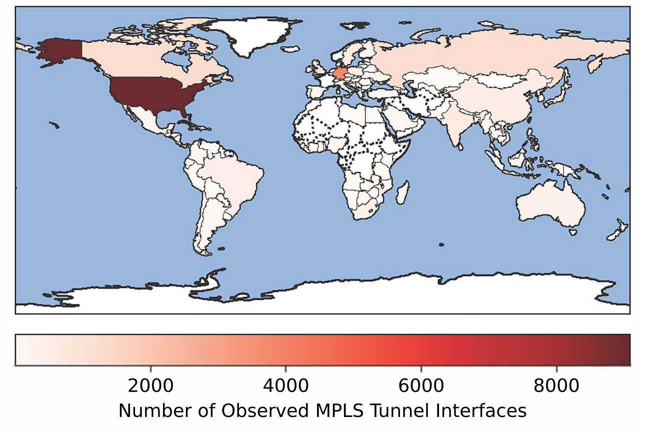
Takeaway: MPLS is common in public cloud WANs, and the types of tunnels observed in large ISPs can differ widely from the more general deployment trends we observed.

4.4 Geolocation of MPLS Tunnels

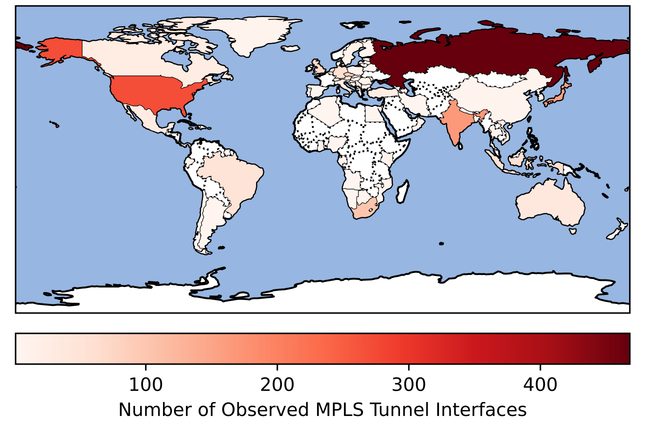
We used IP geolocation to infer where MPLS tunnels were more prevalent in our dataset. Using the MPLS tunnel router IP addresses in our 2025 262 VP dataset, we first attempted to geolocate routers using Hoiho [15], which learns to extract geolocations based on clues that network operators provide in DNS hostnames. We performed reverse DNS lookup on all tunnel IP addresses, and obtained DNS hostnames for 129,874 out of the 201,774 (64.4%) tunnel IP addresses. We used the Hoiho-generated regular expressions (regexes) included in the March 2025 ITDK, and if the regexes did not extract a location for a hostname we tried the August 2024 ITDK regexes, followed by the March 2024 ITDK regexes. In total, Hoiho inferred locations for 32,008 (15.9%) of the tunnel addresses, with 506 and 505 added by the August 2024 and March 2024 regexes, respectively. For all remaining IP addresses, we geolocated at the country-level using the free version of IPinfo [7], which prior work demonstrates is typically reliable at country-level and uses delay measurements to inform its location inference [12]. Using the same approach we geolocated 71.2% of the 2,848,307 IP addresses inferred to belong to MPLS routers in the August 2025 ITDK dataset.

Figure 7 shows the locations for routers in invisible tunnels (Figure 7a) and opaque tunnels (Figure 7b) in our May 2025 262 VP PyTNT experiment. The heatmap for invisible tunnels is representative of the heatmaps for explicit and implicit tunnels as well. Figure 8 shows the locations for routers in invisible (Figure 8a), implicit (Figure 8b), and opaque (Figure 8c) tunnels inferred from the August 2025 ITDK. The locations indicate that MPLS is more prevalent in the U.S. than any other country, although Europe contains more MPLS tunnel router IP addresses than North America (Table 11). Surprisingly, India contains far more opaque tunnels than any other country, with most (85%) belonging to Jio (AS55836). The frequent use of opaque MPLS tunnels in India likely reflects a router vendor and network operator preference within Jio.

Takeaway: The U.S. contains more MPLS tunnel router IP addresses than any other country in our dataset, and India has disproportionately more opaque tunnels than any other country.



(a) Invisible tunnel router locations.



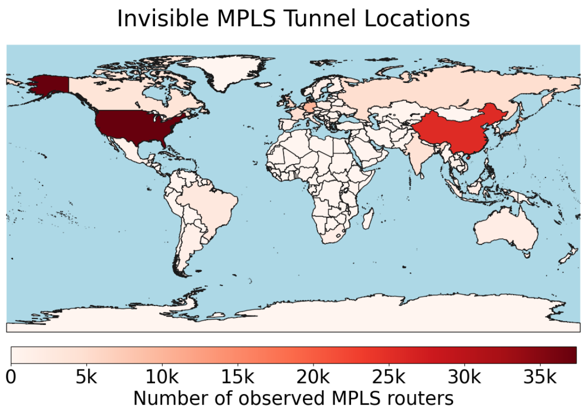
(b) Opaque tunnel router locations.

Figure 7: Locations of invisible (a) and opaque (b) tunnels for the 2025 262 VP dataset. The locations for explicit and implicit tunnels are nearly identical to invisible tunnels at a country-level granularity.

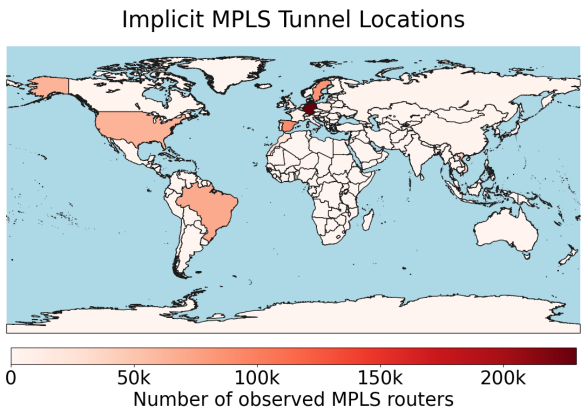
4.5 High Degree Nodes in CAIDA's ITDK

Our last replication effort focuses on CAIDA's Internet Topology Data Kit (ITDK) [2], which is released approximately twice per year. Each ITDK uses two weeks of traceroute across a distributed set of VPs, and collapses the IP-layer topology into a router-level topology using alias resolution.

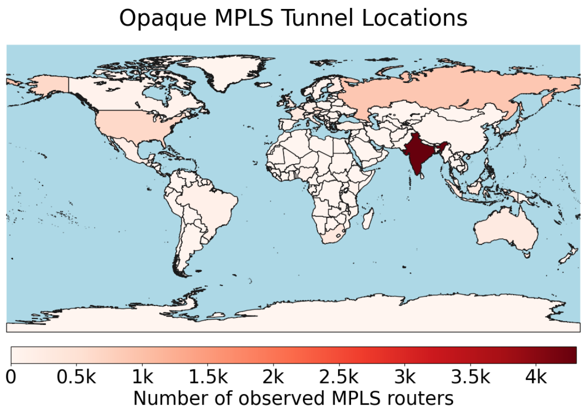
In the 2017 IMC paper [21], Vanaubel et al. identified 17,944 high-degree nodes (HDNs) in CAIDA's March 2016 ITDK. HDNs are inferred routers—interface IP addresses that alias resolution infers to belong to the same router—with observed interconnections to at least 128 other inferred routers. Vanaubel et al. justified 128 as a reasonably likely upper bound on the number in-use router interfaces in ISP networks. These HDNs were previously assumed to be multipoint-to-point links over a layer 2 switching fabric, but Vanaubel et al. showed that another potential explanation for the presence of these HDNs was that invisible MPLS tunnels may create the false appearance of directly connected routers in traceroute


Invisible MPLS Tunnel Locations
0
5k
10k
15k
20k
25k
30k
35k
Number of observed MPLS routers

(a) Invisible tunnel router locations.


Implicit MPLS Tunnel Locations
0
50k
100k
150k
200k
Number of observed MPLS routers

(b) Implicit tunnel router locations.



(c) Opaque tunnel router locations.

Figure 8: Locations of invisible (a), implicit (b), and opaque (c) tunnel for the August 2025 ITDK. The explicit tunnel distribution closes matches that of the invisible tunnels in Figure 7.

output. HDNs in the ITDK can also be caused by false positives in the alias resolution process, in which multiple addresses are mistakenly inferred to belong to the same router. We replicated this experiment as closely as possible, deriving a list of HDNs from the March 2025 ITDK with the same 128 link threshold. We could not exactly reproduce the number of HDNs previously observed in the March 2016 ITDK, but our analysis is consistent with both the original description and the intention of the HDN analysis.

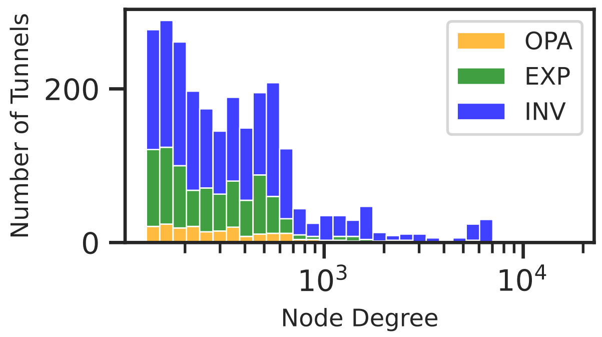


Figure 9: The degree distribution for high degree nodes inferred in the ingress LERs of an MPLS tunnel for invisible (INV), explicit (EXP), and opaque (OPA) tunnels.

Our approach is to identify an HDN as those with at least 128 distinct next-hop routers observed over the two week measurement period, since this indicates potentially false links. We extracted all immediately adjacent interface IP addresses in the two weeks of traceroutes voered by the March 2025 ITDK. An immediate adjacency is two consecutive hops in traceroute output with no unresponsive hops in between. We also ensured that both hops responded with ICMP Time Exceeded messages, to ensure that they were routers. Since we were looking for HDNs without obvious explanations, we also filtered out any adjacency where the adjacent hop belonged to an IXP public peering prefix reported in PeeringDB [5], since IXP public peering can create the appearance of HDNs due to the layer 2 switching fabric. Using the remaining IP-level adjacencies, we mapped each IP address to an inferred router using the MIDAR, iffinder, and SNMPv3 alias resolution graph included in the March 2025 ITDK, and created a directed graph of the inferred routers. We identified any inferred router with ≥ 128 outgoing edges as an HDN, yielding 9,239 HDNs.

Using PyTNT, we attempted to identify if invisible MPLS tunnels could explain the HDNs observed in the March 2025 ITDK. For each HDN, we identified the set of traceroutes traversing the HDN from the two week collection for that ITDK, and used those traceroutes as the seed for PyTNT's analysis. We identified 1,623 HDNs as ingress LERs for invisible tunnels, 724 for explicit tunnels, and 196 for opaque tunnels. Figure 9 gives the distribution of the degrees of these HDNs. Figure 10 includes the degree distribution of non-MPLS HDNs. While only 16.7% of observed HDNs are in invisible tunnels, these HDNs account for 37% of nodes with a degree over 512, and 33% of nodes with a degree over 10,000.

Takeaway: While there are still several causes for HDNs, it remains plausible that invisible MPLS tunnels do contribute to HDN presence in topology datasets.

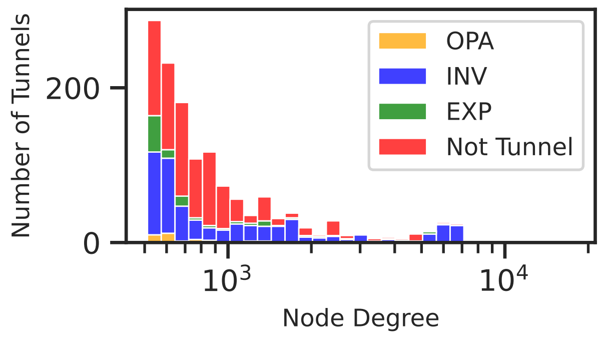


Figure 10: The degree distribution for HDNs with a degree over 512 either not in an MPLS tunnel, in an invisible (INV) tunnel, in an explicit tunnel (EXP), or in an opaque tunnel (OPA).

Vendor	Count	255,255	255,64	64,64	Other
Huawei	21977	0.9%	1.2%	97.4%	0.5%
Cisco	20471	1.3%	10.9%	87.6%	0.2%
Juniper	10033	0.2%	8.5%	91.1%	0.1%
Mikrotik	8413	0.8%	0.2%	98.9%	0.2%
Nokia	3752	0.0%	0.0%	99.9%	0.1%
H3C	1341	4.6%	0.3%	94.9%	0.1%
Ruijie	616	0.8%		99.0%	0.2%
Oneaccess	30	86.7%		13.3%	
Other	2215	1.4%	3.7%	94.8%	0.2%
Total	68848				

Table 12: Initial TTLs inferred for IPv6 router interfaces that provided a vendor through SNMPv3 probes, TTL expired messages, and ICMP echo replies. In August 2025, ≈10% of self-identifying Juniper and Cisco routers used an initial TTL of 255 for ICMP TTL expired messages, and 64 for ICMP echo replies.

4.6 MPLS in IPv6 Infrastructure

The original implementation of TNT focused on IPv4 tunnels, although MPLS tunnels are also present in IPv6 infrastructure. In general, the structure and function of an MPLS tunnel in IPv6 is similar to one in IPv4. However, there are key differences that may make detecting and revealing MPLS tunnels over IPv6 more difficult than in IPv4.

One of the main uses of MPLS in IPv6 is 6PE, a technique that uses MPLS to carry IPv6 traffic over infrastructure that only supports IPv4 [20]. 6PE works roughly by creating an MPLS tunnel in which the ingress and egress LERs both support IPv6, but the LSRs along the LSP only support IPv4. Because tunnels are set up such that the LSR may not support IPv6, if a packet from a traceroute times out at an LSR, that LSR may not be able to reply with the appropriate time-exceeded IPv6 error message. This will then show in the traceroute as a missing hop. While the hop can still be inferred as part of an MPLS tunnel, information about the router such as IP address will be unavailable.

Another crucial difference between IPv4 and IPv6 in the context of detecting invisible MPLS tunnels is the different initial TTL behaviors exhibited by routers in IPv6 responses. Invisible tunnel detection techniques (§2.3.1) currently rely on TTL-based router fingerprinting to determine the appropriate tunnel detection method. However, the IPv6 router responses to pings and traceroutes issued as part of CAIDA's August 2025 ITDK suggest that the initial TTL signatures for IPv6 routers follow a different pattern than IPv4 routers. Table 12 shows the distribution of observed signatures for IPv6 routers, with 64,64 being the most common signature across all identified vendors – substantially different to the signatures observed for IPv4 reported in Table 6. As a result, RTLA is a less viable option for detecting MPLS tunnels over IPv6. While FRPLA will still be able to suggest the presence of MPLS tunnels, regular shifts in routing behavior may create false positives that otherwise could have been avoided using RTLA's more exact method.

Both of these factors warrant a deeper investigation into MPLS behavior in IPv6.

5 Insights from a Tier-1 Operator

We spoke with a network operator at a Tier-1 network about their use of MPLS and share the insights here for the benefit of the Internet measurement community. This Tier-1 network is also on the top ten list of networks using MPLS in §4.3 and has deployed MPLS for over 25 years.

We asked the operator several questions about their MPLS deployment and learned five key takeaways. (1) MPLS is enabled in what the operator considers the core of the network, but some edge networks obtained through acquisition remain IP-only networks. (2) The network uses both PHP and UHP. The use of UHP resulted in some cases from legacy deployments, but it also lets them tunnel IPv6 over IPv4 without maintaining an IPv6 iBGP mesh in the network core. (3) With regard to setting the `ttl-propagate` option, they primarily use router vendor defaults, and the operator saw no compelling reason to hide the routers from traceroute with the `no-ttl-propagate` option. (4) The Tier-1 network uses Juniper, Cisco, and Nokia to implement their MPLS tunnels. (5) Finally, they use MPLS to route between ASes under their organizational control.

6 Related Work

Identification of MPLS Tunnels: Since the TNT paper was published in 2019 [18], we are not aware of other papers that have used similar techniques to identify MPLS tunnels. Prior work in MPLS detection used patterns in RTTs [17] or MPLS label distribution [19]. However, these techniques require more overhead, such as training a machine learning model to identify tunnels [17], and do not discern between different configurations of MPLS tunnels.

Other work explores how the behavior of MPLS deployments change in IPv6 networks. Vanaubel et al. [20] identify additional MPLS behaviors not present in IPv4, particularly with how labels are handled. For example, dual stack IPv4 and IPv6 deployments generally use a larger MPLS label stack despite the fact that the IPv4 and IPv6 MPLS tunnels often share the same router paths.

Augmenting studies using topology measurement: Many studies that draw conclusions from traceroute topology discovery could benefit from incorporating MPLS into the analysis. This is because the potential for hidden routers and false links in traceroute output can mislead inferences. As far back as 2006, Alderson et al. [10] identified MPLS as one of the ways that networks can hide their router-level infrastructure from outside measurement. Recent examples where MPLS tunnel revelation might benefit measurement are studies examining router diversity [9] and characterizing long-haul infrastructure [11]. We plan to incorporate PyTNT into CAIDA's ITDK, increasing the richness and fidelity of the best available router-level map of the Internet.

7 Conclusion

We replicated the work of Vanaubel et al. identifying MPLS tunnels for the purpose of improving topology analysis with traceroute. A goal of our replication was to identify changes in the MPLS landscape between 2016 and 2025. While the prevalence of MPLS tunnels in public clouds has changed and the number of MPLS deployments has decreased, the presence of invisible tunnels that obscure measured topologies remains a critical problem for traceroute analysis.

As part of our replication, we reimplemented the existing state-of-the-art to be easily maintainable and deployable. We used our tool, called PyTNT, to reveal MPLS tunnels on paths to every routed /24 from all 262 VPs that CAIDA's Ark measurement platform uses for continuous traceroute probing. We release PyTNT as open source on GitHub to facilitate higher-fidelity Internet measurements and plan to continue to incorporate PyTNT into future ITDK releases. Our results, particularly the consistent presence of certain varieties of MPLS tunnel that confound traditional measurements, demonstrate the potential usefulness of a tool able to identify MPLS tunnels in topology datasets, and motivates continued work to integrate PyTNT into active topology measurement campaigns.

Acknowledgments

This work was funded in part by NSF ITE-2326928, NSF OAC-2519417, NSF OAC-2131987, NSF CNS-2212241, and NSF OAC-2530871.

References

[1] [n. d.]. Archipelago (Ark) Measurement Infrastructure. https://www.caida.org/projects/ark/
[2] [n. d.]. The CAIDA Macroscopic Internet Topology Data Kit - 2025-03. https://www.caida.org/catalog/datasets/internet-topology-data-kit.
[3] [n. d.]. ICMP Extensions for Multiprotocol Label Switching. https://datatracker.ietf.org/doc/html/rfc4950
[4] [n. d.]. Macroscopic Topology Measurements. https://www.caida.org/projects/macroscopic/
[5] [n. d.]. PeeringDB. https://www.peeringdb.com/
[6] [n. d.]. Routeviews Prefix to AS mappings Dataset for IPv4 and IPv6. https://www.caida.org/catalog/datasets/routeviews-prefix2as/
[7] 2025. IPinfo Lite. https://ipinfo.io/lite.
[8] Taha Albakour, Oliver Gasser, Robert Beverly, and Georgios Smaragdakis. 2021. Third time's not a charm: exploiting SNMPv3 for router fingerprinting. In *Proceedings of the 21st ACM Internet Measurement Conference* (Virtual Event) *(IMC '21)*. Association for Computing Machinery, New York, NY, USA, 150–164. doi:10.1145/3487552.3487848
[9] Taha Albakour, Oliver Gasser, Robert Beverly, and Georgios Smaragdakis. 2023. Illuminating Router Vendor Diversity Within Providers and Along Network Paths. In *Proceedings of the 2023 ACM on Internet Measurement Conference* (Montreal QC, Canada) *(IMC '23)*. Association for Computing Machinery, New York, NY, USA, 89–103. doi:10.1145/3618257.3624813
[10] David Alderson, Hyunseok Chang, Matthew Roughan, Steve Uhlig, and Walter Willinger. 2006. The many facets of internet topology and traffic. (2006).
[11] Esteban Carisimo, Caleb J. Wang, Mia Weaver, Fabián E. Bustamante, and Paul Barford. 2023. A Hop Away from Everywhere: A View of the Intercontinental Long-haul Infrastructure. *Proc. ACM Meas. Anal. Comput. Syst.* 7, 3, Article 47 (Dec. 2023), 26 pages. doi:10.1145/3626778
[12] Omar Darwich, Hugo Rimlinger, Milo Dreyfus, Matthieu Gouel, and Kevin Vermeulen. 2023. Replication: Towards a publicly available internet scale ip geolocation dataset. In *Proceedings of the 2023 ACM on Internet Measurement Conference*. 1–15.
[13] Benoit Donnet, Matthew Luckie, Pascal Mérindol, and Jean-Jacques Pansiot. 2012. Revealing MPLS tunnels obscured from traceroute. *SIGCOMM Comput. Commun. Rev.* 42, 2 (March 2012), 87–93. doi:10.1145/2185376.2185388
[14] Matthew Luckie. 2010. Scamper: a Scalable and Extensible Packet Prober for Active Measurement of the Internet. In *IMC*. 239–245.
[15] Matthew Luckie, Bradley Huffaker, Alexander Marder, Zachary Bischof, Marianne Fletcher, and KC Claffy. 2021. Learning to extract geographic information from internet router hostnames. In *Proceedings of the 17th International Conference on emerging Networking EXperiments and Technologies*. 440–453.
[16] Alexander Marder, Matthew Luckie, Amogh Dhamdhere, Bradley Huffaker, kc claffy, and Jonathan M. Smith. 2018. Pushing the Boundaries with bdrmapIT: Mapping Router Ownership at Internet Scale. In *Proceedings of the Internet Measurement Conference 2018* (Boston, MA, USA) *(IMC '18)*. Association for Computing Machinery, New York, NY, USA, 56–69. doi:10.1145/3278532.3278538
[17] Joel Sommers, Paul Barford, and Brian Eriksson. 2011. On the prevalence and characteristics of MPLS deployments in the open internet. In *Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference* (Berlin, Germany) *(IMC '11)*. Association for Computing Machinery, New York, NY, USA, 445–462. doi:10.1145/2068816.2068858
[18] Yves Vanaubel, Jean-Romain Luttringer, Pascal Mérindol, Jean-Jacques Pansiot, and Benoit Donnet. 2019. TNT, Watch me Explode: A Light in the Dark for Revealing MPLS Tunnels. arXiv:1901.10156 [cs.NI] https://arxiv.org/abs/1901.10156
[19] Yves Vanaubel, Pascal Mérindol, Jean-Jacques Pansiot, and Benoit Donnet. 2015. MPLS Under the Microscope: Revealing Actual Transit Path Diversity. In *Proceedings of the 2015 Internet Measurement Conference* (Tokyo, Japan) *(IMC '15)*. Association for Computing Machinery, New York, NY, USA, 49–62. doi:10.1145/2815675.2815687
[20] Yves Vanaubel, Pascal Mérindol, Jean-Jacques Pansiot, and Benoit Donnet. 2016. A Brief History of MPLS Usage in IPv6. In *Passive and Active Measurement*, Thomas Karagiannis and Xenofontas Dimitropoulos (Eds.). Springer International Publishing, Cham, 359–370.
[21] Yves Vanaubel, Pascal Mérindol, Jean-Jacques Pansiot, and Benoit Donnet. 2017. Through the wormhole: tracking invisible MPLS tunnels. In *Proceedings of the 2017 Internet Measurement Conference* (London, United Kingdom) *(IMC '17)*. Association for Computing Machinery, New York, NY, USA, 29–42. doi:10.1145/3131365.3131378
[22] Yves Vanaubel, Jean-Jacques Pansiot, Pascal Mérindol, and Benoit Donnet. 2013. Network fingerprinting: TTL-based router signatures. In *Proceedings of the 2013 Conference on Internet Measurement Conference* (Barcelona, Spain) *(IMC '13)*. Association for Computing Machinery, New York, NY, USA, 369–376. doi:10.1145/2504730.2504761

A Ethics

Our only potential ethical concern with this work is that we engage with a number of tools to identify router topology over a significant swath of the Internet. Additionally, our measurements are conducted through the CAIDA Archipelago, which is an opt-in measurement platform. CAIDA also minimizes the bandwidth usage of the Ark nodes on their host networks, and all of our measurements use low packet-per-second rates.