vrfinder: Finding Outbound Addresses in Traceroute

ALEXANDER MARDER, CAIDA / UC San Diego, USA
MATTHEW LUCKIE, University of Waikato, New Zealand
BRADLEY HUFFAKER, CAIDA / UC San Diego, USA
KC CLAFFY, CAIDA / UC San Diego, USA

Current methods to analyze the Internet's router-level topology with paths collected using traceroute assume that the source address for each router in the path is either an inbound or off-path address on each router. In this work, we show that outbound addresses are common in our Internet-wide traceroute dataset collected by CAIDA's Ark vantage points in January 2020, accounting for 1.7% – 5.8% of the addresses seen at some point before the end of a traceroute. This phenomenon can lead to mistakes in Internet topology analysis, such as inferring router ownership and identifying interdomain links. We hypothesize that the primary contributor to outbound addresses is Layer 3 Virtual Private Networks (L3VPNs), and propose vrfinder, a technique for identifying L3VPN outbound addresses in traceroute collections. We validate vrfinder against ground truth from two large research and education networks, demonstrating high precision (100.0%) and recall (82.1% – 95.3%). We also show the benefit of accounting for L3VPNs in traceroute analysis through extensions to bdrmapIT, increasing the accuracy of its router ownership inferences for L3VPN outbound addresses from 61.5% – 79.4% to 88.9% – 95.5%.

CCS Concepts: • **Networks** → **Network measurement**; *Public Internet*; *Packet-switching networks*; Routing protocols.

ACM Reference Format:
Alexander Marder, Matthew Luckie, Bradley Huffaker, and kc claffy. 2020. vrfinder: Finding Outbound Addresses in Traceroute. *Proc. ACM Meas. Anal. Comput. Syst.* 4, 2, Article 40 (June 2020), 28 pages. https://doi.org/10.1145/3392158

1 INTRODUCTION

Traceroute is the primary method available for uncovering the Internet's router-level topology, but it presents insidious epistemological challenges. Traceroute is a essentially a hack that exploits common router behavior. When a router discards a packet due to the expiration of the Time to Live (TTL) field in the Internet Protocol (IP) packet header, it sends an Internet Control Message Protocol (ICMP) Time Exceeded response packet to the originator of the dropped packet. When routers respond to traceroute probes, the ICMP response contains a router interface IP address in the source field, exposing an address used on the router. The traceroute tool exploits this behavior by sending probes with incrementing TTL values toward a single destination, attempting to induce responses from each router on the path from the source to the destination.

Authors' addresses: Alexander Marder, amarder@caida.org, CAIDA / UC San Diego, 9500 Gilman Dr. La Jolla, California, 92093-0505, USA; Matthew Luckie, mjl@wand.net.nz, University of Waikato, Gate 1, Knighton Road, Hamilton, New Zealand; Bradley Huffaker, bradley@caida.org, CAIDA / UC San Diego, 9500 Gilman Dr. La Jolla, California, 92093-0505, USA; kc claffy, kc@caida.org, CAIDA / UC San Diego, 9500 Gilman Dr. La Jolla, California, 92093-0505, USA.

2476-1249/2020/6-ART40 $15.00
https://doi.org/10.1145/3392158

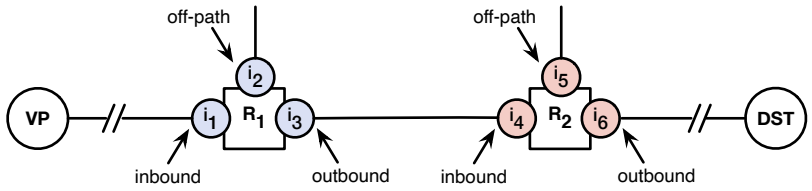


Fig. 1. Traceroute observes a sequence of router interfaces between a Vantage Point (VP) and a Destination (DST). Logically, the inbound interfaces face the VP, while the outbound interfaces face the the destination.

Most routers report the address of the interface that received the traceroute probe, known as the *inbound address*, in accordance with common network operator expectations [64], even if the router sends the response using a different interface. Other routers adhere to the specification in Request for Comments (RFC) 1812 [22], which mandates that routers report the address of the interface that transmits the ICMP response. If the route back to the source is not via the inbound interface, the router will report an *off-path address* [33], as illustrated in Fig. 1. We do not expect, however, that a source address in a traceroute response represents an interface that would have been used to forward the probe toward its destination, known as the *outbound interface*. Internet routers rarely forward a packet through the same logical interface that received the packet, as doing so could lead to a forwarding loop. The conventional interpretation, that routers report inbound or off-path interface addresses in response to TTL-expiring probes, functions as an axiom of traceroute analysis. Reliance on this axiom, either explicit or implicit, pervades the literature [19–21, 30, 42, 45, 49, 50, 54, 61, 62].

In this paper, we show that when a probe would have traversed an L3VPN had the TTL not expired, the router can report the address of the interface in the L3VPN it would have used to forward the probe toward the destination. This behavior is not described in RFC documents or in router manuals. Importantly, L3VPNs can mislead techniques [42, 49, 50] that infer which Autonomous System (AS) operates a router. In 2018, a router AS ownership inference technique [42] was used by Dhamdhere *et al.* to identify interdomain links, which they then measured for evidence of persistent congestion suggesting a long-term mismatch between installed capacity and actual traffic [25]. In 2019, Yeganeh *et al.* used a router ownership inference technique to infer and geolocate Amazon's peering connections [68]. Accurate router ownership inferences could also improve discovery and classification of critical or bottleneck routers and links [18, 34, 40, 65] whose failure could threaten networks or geographic regions, and provide additional constraints for geolocation efforts.

We report the results of our effort to develop and validate a method to infer the presence of outbound interfaces in traceroute paths primarily caused by L3VPNs. The key challenge that we address is to distinguish outbound interfaces caused by L3VPNs from apparent outbound interfaces caused by forwarding loops. While this might seem like a narrow contribution, outbound addresses can potentially confound any traceroute analysis that relies on conventional traceroute assumptions. This paper makes the following four contributions.

(1) We provide a detailed explanation of how network operators may use L3VPNs. In §4 we describe L3VPN implementation and use in Internet networks, the phenomenon that causes L3VPN routers to report the outbound interface address in response to TTL-expiring probes, and how outbound addresses can mislead traceroute analysis.

(2) We present vrfinder, a novel technique for identifying L3VPN outbound addresses in traceroutes. We use vrfinder to infer that 5.8% (IPv4) and 1.7% (IPv6) of the addresses seen in the middle of traceroutes by CAIDA's Archipelego measurement infrastructure in January 2020 were L3VPN outbound interfaces. We validate our inferences against ground truth from Internet2

and REANNZ, two large research and education networks, with 100.0% precision and 82.1% – 95.3% true positive rate in IPv4 and IPv6 traceroute collections.

(3) We extend bdrmapIT, the state-of-the-art for router ownership inferences, to account for L3VPN outbound interfaces. We show that outbound addresses can confound bdrmapIT by violating its traceroute semantic assumptions. Our improvements increase the accuracy of its router ownership inferences for the two ground truth L3VPN networks, in IPv4 and IPv6, from 61.5% – 79.4% to 88.9% – 95.5%.

(4) We publicly release our code. To allow other researchers to find and account for outbound addresses in their traceroute analyses, and support reproducibility, we publicly release our source code [48].

2 RELATED WORK

Researchers have put considerable effort into inferring router-level properties of networks to understand aspects of Internet routing behavior, performance, and resilience. The Internet was not designed with router-level measurement in mind, so researchers have used three different methods to obtain the raw data for constructing router-level Internet topologies: (1) traceroute [54, 63], (2) the Record Route IP option [28, 35, 61], and (3) Internet Group Management Protocol (IGMP) messages [51, 53]. Researchers and operators mostly use traceroute, since Record Route and IGMP only apply in IPv4, and some network operators filter packets with IP options [28], or filter IGMP packets [51]. However, using traceroute data to understand router-level paths is challenging.

Traceroute infers an IP path using a sequence of TTL-limited IP packets that each solicit an ICMP Time Exceeded message, whose source address identifies an interface of a router in the path. If the route between a source and a destination changes while traceroute is sending probes, traceroute might report adjacencies between unconnected routers. Network operators can prevent their routers from sending ICMP Time Exceeded messages, or filter classes of traceroute probes at their network borders, resulting in incomplete traceroute paths.

An operator can configure Multiprotocol Label Switching (MPLS) to hide router-level topology from traceroute, confounding traceroute inferences because edge routers appear directly connected [26]. These invisible MPLS tunnels can also create the false appearance of high degree nodes [66]. While MPLS can hide the middle of the network, the unexpected appearance of outbound addresses in traceroute can obscure network interconnections. In this work, we observe outbound addresses that are caused by a specific network configuration model (§4) and we introduce a method to detect outbound addresses in traceroute (§5).

A router, by definition, has multiple interfaces, and it can embed any of its interface IP addresses in the source address field of the ICMP response to a traceroute probe; an interface address might be inbound, off-path, or outbound (Fig. 1). There has been a particular focus on off-path addresses in the literature. In 2003, Hyun *et al.* reported that asymmetric routing might cause a router to respond with an off-path address [33] from the address space of an off-path AS, but reported that this was rare, often observed close to the traceroute destination, and caused by multi-homing. In 2013, Marchetta *et al.* [46] presented a method to infer off-path addresses using the IP Timestamp option, reporting that most of the classified IP addresses in their data were off-path. In 2014, Luckie *et al.* [41] found that most inbound interfaces were incorrectly classified by the technique from Marchetta *et al.* [46] to be off-path. Off-path addresses fundamentally differ from outbound addresses, as the former are the responding interface address, while an outbound address indicates the interface used to continue forwarding the packet to the traceroute destination.

Conventional interpretations of traceroute paths assumed that routers do not report the outbound address when responding to traceroute probes [19–21, 30, 42, 45, 49, 50, 54, 61, 62]. Some prior work used the Record Route IP option to expose *unseen* outbound addresses for routers encountered in

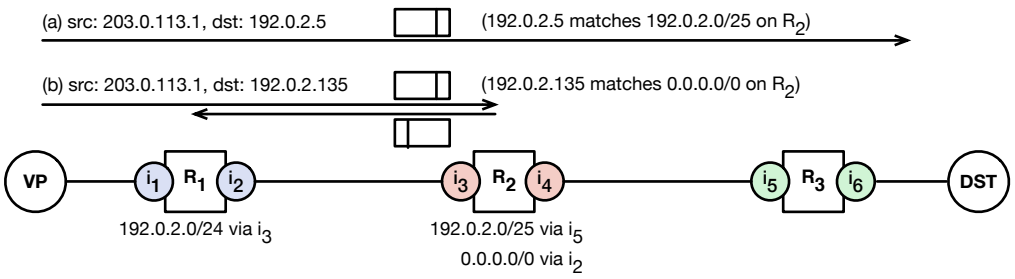


Fig. 2. A class of forwarding loop. R_2 announces `192.0.2.0/24` to R_1, but only has an internal route for `192.0.2.0/25` and a default route to R_1. Packets matching addresses in `192.0.2.0/25` will be forwarded towards the destination, but packets matching addresses in `192.0.2.128/25` will match the default route on R_2 and loop between R_1 and R_2.

traceroute paths. In 2008, Sherwood *et al.* [61] attempted to remove errors common in traceroute analysis by aligning traceroute and Record Route paths. In 2010, Katz-Bassett *et al.* [35] used Record Route to discover the reverse path between an arbitrary traceroute source and destination. In 2017, Goodchild *et al.* [28] suggested that Record Route provides useful information when used in TTL-expiring probes, opening up potential new uses for Record Route. We show (§4.2) that some routers *expose* the outbound address in response to traceroute probes due to L3VPNs.

Because a router can embed an addresses from any of its interfaces, different traceroutes can observe different IP address aliases of the same router. Researchers have developed many alias resolution techniques to infer which IP addresses belong to the same router [24, 29, 37, 43, 60, 63]. Because alias resolution techniques rely on heuristics and implementation artifacts, not all IP addresses can be resolved for aliases.

Attributing which AS operates a given router is challenging, particularly for routers that connect to other ASes, as a neighbor AS could provide the router address that traceroute observes. In 2010, Zhang *et al.* [69] found that the canonical IP2AS mapping can result in false interdomain link inferences from traceroute data. Since 2016, three techniques emerged to infer network boundaries and interdomain links from traceroute. MAP-IT [50] used iterative constraint satisfaction to infer router ownership and interdomain links from a traceroute dataset. Concurrently, bdrmap [42] used heuristics to map the border of a network hosting a traceroute vantage point. Most recently, Marder *et al.* synthesized bdrmap and MAP-IT to develop bdrmapIT [49], the current state-of-the-art in mapping routers observed in traceroute to their AS operators. In this work, we show that L3VPN outbound addresses can cause bdrmapIT to draw mistaken AS operator inferences, and we develop a method to detect and mitigate their effects on bdrmapIT's inferences.

Analysis by Xia *et al.* [67] found that roughly half of the forwarding loops in their 2005 traceroute dataset resulted from networks failing to configure null routes for unused parts of their address space. Fig. 2 shows a class of persistent forwarding loop commonly found at the edge of the Internet where border routers may have a default route to their provider, and internal routes that cover only a portion of the address space they announce to their provider. When a router receives a packet to an unused portion of internal address space, the destination address may match a default route and return via the router interface that received the packet. In 2017, Lone *et al.* [39] used detection of these persistent forwarding loops to infer networks that did not deploy source address validation (SAV) to filter packets with spoofed source IP addresses; in Fig. 2 R_1 should discard packets with source address `203.0.113.1` because the address is not valid for that attachment point, i.e., does not match a route via i_3. In 2018, Ruth *et al.* [58] confirmed that persistent forwarding

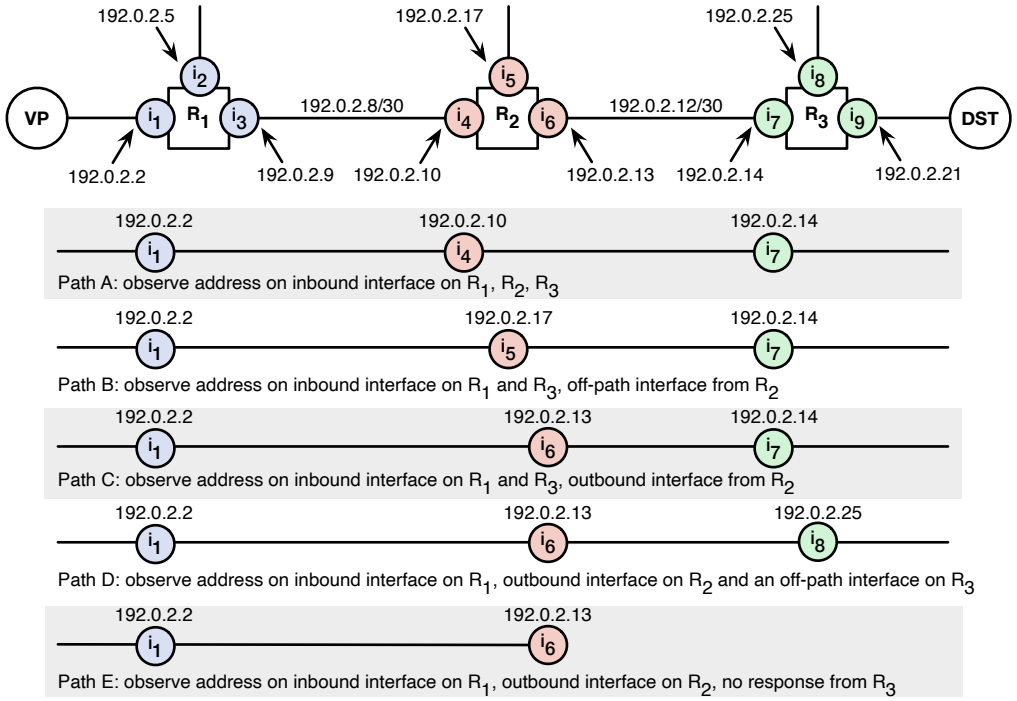


Fig. 3. Intuition and challenges in inferring outbound addresses in traceroute paths. We infer a candidate outbound address on R_2 when we observe an inbound address in the same subnet on adjacent router R_3 (Path C). If R_3 replies with an off-path or outbound address (Path D), or if R_3 does not reply at all (Path E), we cannot infer that R_2 used an outbound address because we rely on observing an inbound address in the same subnet on an adjacent router.

loops appeared regularly in Internet-wide ZMap scans. In this work, we show that forwarding loops present challenges for outbound IP address inference in traceroute paths.

3 INTUITION AND CHALLENGES

The Internet is assembled by connecting routers that speak IP. These connections are either point-to-point between two routers, or point-to-multipoint between many routers that exchange traffic using a shared fabric, such as at an Internet Exchange Point (IXP). Because IPv4 addresses are scarce resources, point-to-point connections in IPv4 are typically numbered out of a /30 or /31. In IPv6, operators most commonly address point-to-point links from /64, /126, and /127 subnets [31, 38, 59], although they may use any subnet size between 64 and 127. Host addresses, i.e., addresses usable as interface addresses, are consecutive addresses for /30, /31, /126, and /127 subnets. While there is no such guarantee for /64 subnets, current best practice is to assign consecutive addresses to the point-to-point link interfaces [70]. Fig. 3 illustrates these concepts using three routers interconnected with addresses assigned from IPv4 /30 subnets.

As discussed in §1 and §2, current methodologies that use traceroute paths typically assume that traceroute observes mostly inbound addresses, with some off-path addresses. Fig. 3 illustrates two expected traceroute observations in these cases (Paths A and B). Because traceroute paths mostly observe inbound addresses, our intuition is that we may infer the presence of an outbound address when it is followed by an inbound address, as these addresses will be part of the same subnet. In

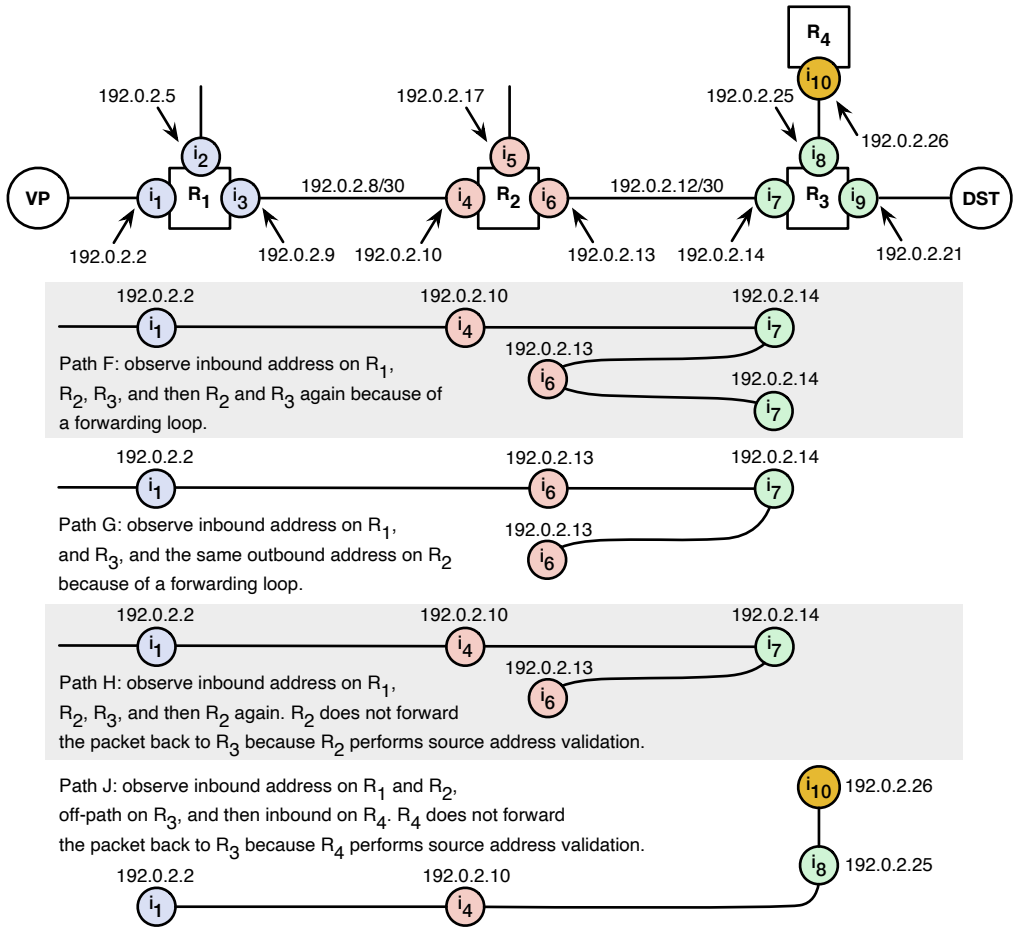


Fig. 4. Challenges in inferring outbound addresses in traceroute paths in the presence of forwarding loops. The loops can manifest in traceroute in different ways, depending on the configuration of the routers.

Fig. 3 Path C, R_2 replies with outbound address `192.0.2.13` and R_3 replies with inbound address `192.0.2.14` – consecutive addresses in `192.0.2.12/30`. In this section we introduce the primary challenges to our method of identifying outbound addresses in traceroute paths: adjacent routers that do not reply with an inbound address (§3.1) consecutive subnets (§3.3), forwarding loops (§3.2), and IXP public peering (§3.4).

3.1 Adjacent Router does not reply with an Inbound Address

Our approach requires that the router following an outbound address replies with its inbound address, but it might reply with an off-path address instead. If R_3 replies with the off-path address `192.0.2.25` (Fig. 3 Path D), then we will not infer that R_2 used an outbound address when it replied with `192.0.2.13` because the addresses are not consecutive (a false negative). If R_3 does not reply at all, as illustrated by Path E in Fig. 3, then we will also not infer R_2 used an outbound address.

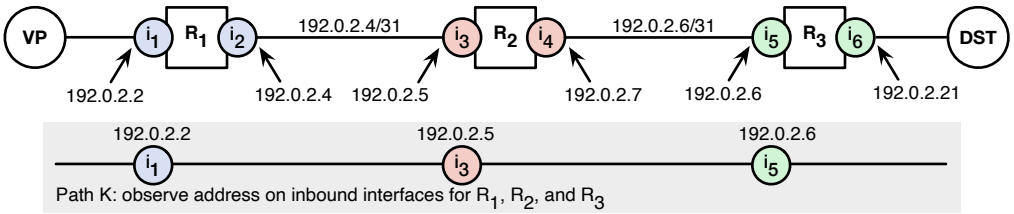


Fig. 5. Challenges in inferring outbound addresses in traceroute paths in the presence of consecutive addresses from consecutive /31 subnets. `192.0.2.5` could be assigned to R_2 and incorrectly inferred as an outbound address within `192.0.2.4/30` or correctly inferred as an inbound address within `192.0.2.4/31`.

3.2 Forwarding Loops

Forwarding loops between adjacent routers can also complicate inferences of which addresses in the traceroute are inbound, off-path, or outbound. When a traceroute path contains an address cycle, where an IP address repeats with at least one other address between the two occurrences, current practice in the research community is to truncate the traceroute path at the first occurrence of the repeated address to avoid traceroute anomalies due to forwarding loops. Paths F, G, and H in Fig. 4 show traceroute paths that expose consecutive IP addresses contained in the same subnet (`192.0.2.12/30`) – a necessary condition for our method to infer the presence of an outbound address. In Path F, R_2 uses the IP address of the inbound interface as the source address in ICMP responses, whereas in Path G, R_2 responds with the IP address of the outbound interface toward the destination. Because the loops in Paths F and G manifest similarly in traceroute output as a repeated IP address in the path separated by an address in the same /30 subnet, they are difficult to distinguish. Truncating path G at the first occurrence of a repeated address (`192.0.2.13`) would mean we could not infer that R_2 is using an outbound address because we no longer have a consecutive address (`192.0.2.14`) at the adjacent hop.

Similar to Path F, in Fig. 4 Path H R_2 uses the IP address of the inbound interface as the source address in ICMP responses. Unlike Path F, R_2 in Path H performs SAV [23, 27], so it discards subsequent traceroute packets because the source addresses of these probes is not valid for that attachment point (§2). Despite the forwarding loop, the SAV process on R_2 prevents the traceroute path from containing a repeated address, and although the adjacent addresses `192.0.2.14` and `192.0.2.13` belong to the `192.0.2.12/30` subnet, `192.0.2.14` is not an outbound address. We can detect the presence of a forwarding loop in Path H if we can infer that R_2 appeared twice using alias resolution techniques, i.e., `192.0.2.10` and `192.0.2.13` belong to the same router. However, alias resolution techniques can produce false negatives because they rely on router implementation artifacts that are not present on all routers, and some routers are unresponsive to alias resolution probe packets.

Path J in Fig. 4 shows another case where we might mistakenly infer an outbound address. In that traceroute, R_3 responds using the off-path address of the interface facing R_4, and forwards subsequent packets to R_4, resulting in consecutive addresses in `192.0.2.24/30`. R_4 does not forward subsequent probe packets back to R_3 because R_4 discards them using SAV. Like Path H, we might incorrectly infer that the penultimate hop i_8 is an outbound address.

3.3 Consecutive Subnets

Some consecutive addresses that appear in traceroute could either belong to consecutive two-address subnets (i.e., IPv4 /31s or IPv6 /127s) or a single four-address subnet (i.e, an IPv4 /30 or an IPv6 /126). If an operator assigns IP addresses from consecutive /31 subnets to routers that appear

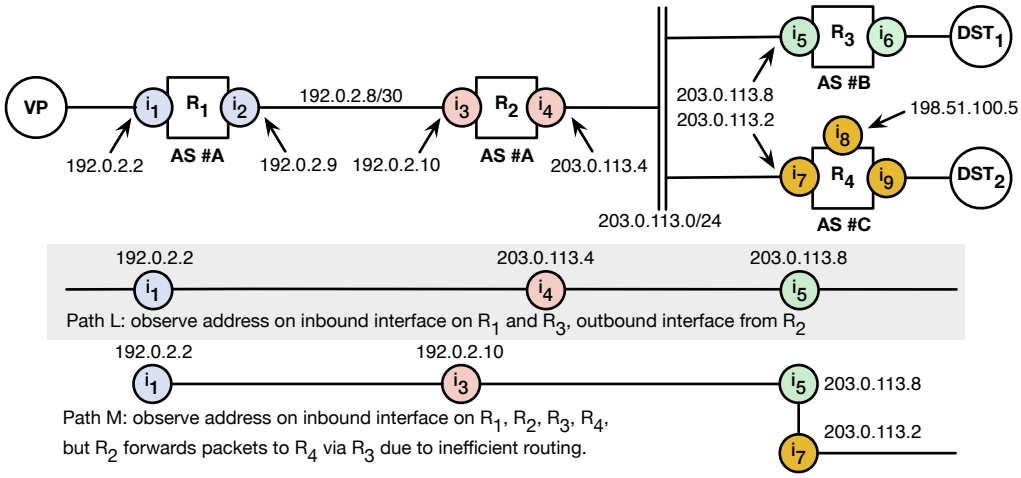


Fig. 6. Challenges in inferring outbound addresses in traceroute paths when crossing an IXP, where traceroute no longer observe consecutive addresses even when a router uses an address on an outbound interface (Path L). Some consecutive addresses from the same IXP subnet do not indicate outbound addresses (Path M).

in sequence in a traceroute path, we may observe consecutive addresses that appear to be part of a /30 subnet. In Fig. 5, Path K reveals the inbound addresses `192.0.2.5` and `192.0.2.6` on routers R_2 and R_3. These addresses could fall within `192.0.2.4/30`, suggesting that R_2 might have used an outbound address when sending an ICMP response, but in reality the addresses belong to separate /31 subnets.

3.4 IXPs

Finally, this section has so far discussed point-to-point links, which connect exactly two routers. IXPs connect many routers using a point-to-multipoint peering fabric, where the IXP operator provides a subnet containing many addresses that allows many routers to interconnect. Path L in Fig. 6 shows that the usual heuristic of detecting an outbound address by observing consecutive addresses no longer holds – R_2 responds using outbound address `203.0.113.4` and R_3 responds using inbound address `203.0.113.8` – because the routers use addresses assigned from the `203.0.113.0/24` subnet with 254 usable addresses. To capture outbound addresses at IXP peering routers, we rely on CAIDA's external dataset of IXP prefixes [16] that contains self-reported IXP prefixes collated by PeeringDB [9], and prefixes learned by routers participating in public peering collated by Packet Clearing House (PCH) [8] and Hurricane Electric (HE) [6]. Instead of looking for addresses from the same point-to-point subnet, we look for adjacent addresses from a subnet used for public peering at the same IXP. However, these datasets are incomplete because they either require the IXP or a participant to register the subnet with PeeringDB or have a route collector from PCH or HE present at the IXP. Furthermore, the issues of an adjacent router not responding with an inbound address in the same subnet, that we discussed in a point-to-point subnet context (Fig. 3), also apply to IXP subnets.

Path M in Fig. 6 shows a more challenging issue: a series of routers might all respond with their inbound addresses, but we observe addresses from the same IXP subnet consecutively in traceroute, which would usually indicate that the first router used an address from an outbound interface to reply. This scenario can occur when the AS operating R_2 (AS #A) is a peer of the AS operating R_3

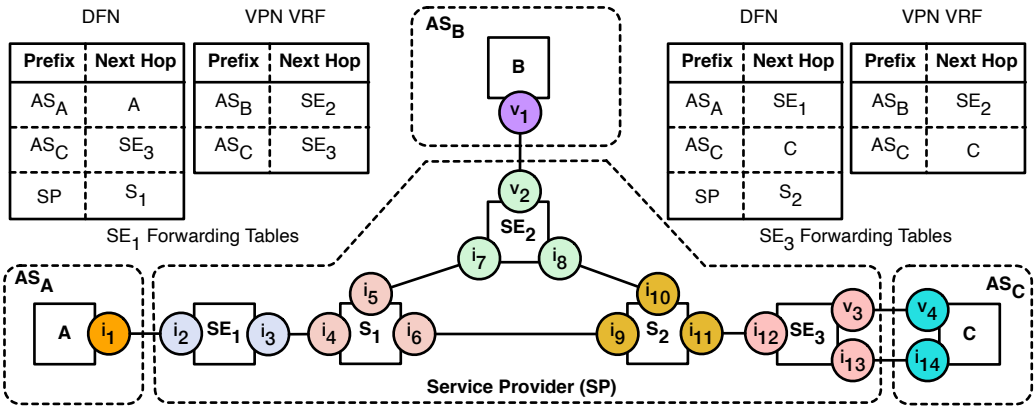


Fig. 7. A service provider with an L3VPN. *S* backbone routers form the backbone. *SE* edge routers connect the backbone to edge routers in other networks. Interfaces labeled *i* connect to the DFN, while *v* interfaces are part of the L3VPN. *SE* routers have two forwarding tables, one for the DFN and one for the L3VPN.

(AS #B), the AS operating R_4 (AS #C) is a customer of AS #B, and #A and #C do not peer. In this scenario, the path between ASes #A and #C crosses the IXP peering fabric twice, so we observe multiple inbound addresses from the IXP subnet.

4 BGP/MPLS LAYER 3 VPNS

Before explaining the traceroute phenomenon caused by L3VPNs, it is important to understand why networks implement L3VPNs and how L3VPNs work. Primarily, L3VPNs allow networks to maintain separate virtual networks using a single physical infrastructure. In service provider networks, L3VPNs typically share a single MPLS backbone. Packets are *tunneled* across the backbone by encapsulating them with an MPLS header containing a label that signals to backbone routers the path a packet will take. Packets in different L3VPNs might travel in the same MPLS tunnels. Crucially, each virtual network maintains its own Virtual Routing and Forwarding (VRF) tables, keeping traffic in one L3VPN logically separate from other L3VPNs, even if they include the same physical router. Common uses for L3VPNs in service provider networks include enforcing complex routing policies, enabling reliable and high bandwidth connections to third-party services, connecting geographically distributed customer sites, and providing carrier-of-carrier services [15].

Other types of networks, such as enterprise networks, might also employ L3VPNs. Enterprise L3VPNs often do not use a shared MPLS backbone, instead running on regular IP links. Cisco refers to these L3VPNs as *VRF Lite* deployments, and Juniper calls them *Virtual-Router Routing Instances*. In our experiments on a Juniper switch and a Cisco switch, neither switch reported the outbound address in response to traceroute probes when configured with these lightweight versions of L3VPNs. In this paper, the L3VPN and VRF terms refer exclusively to MPLS/BGP L3VPNs and the VRFs that support them.

4.1 L3VPN Implementation Over MPLS

The most common implementation of L3VPNs in service provider networks relies on an MPLS backbone to connect the service provider's edge routers, and uses BGP to propagate routes as described primarily in RFCs 2547 [56] and 4364 [57]. Both the Cisco and Juniper router implementations adhere to the RFC specifications [4, 14], representing most deployments in service provider networks.

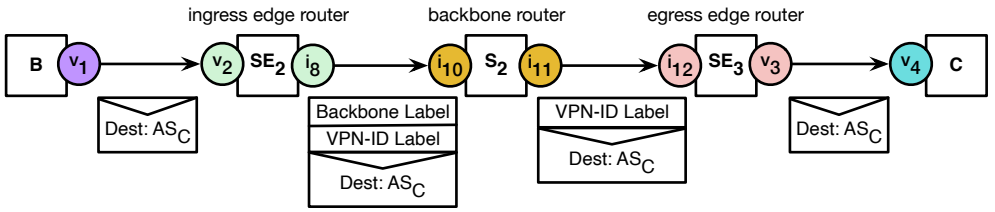


Fig. 8. When forwarding a packet from Router B to Router C, SE_2 pushes two MPLS labels: a VPN-ID label, and a backbone label. S_2 decides how to forward the packet (to SE_3) using only the backbone label, and pops the backbone label because S_2 is the penultimate hop in the tunnel. SE_3 determines the forwarding table for the packet using the VPN-ID label, pops the VPN-ID label, and forwards the packet to C.

Fig. 7 shows an L3VPN in a service provider network, where the service provider uses an MPLS backbone comprised of routers S_1 and S_2 to connect its edge routers SE_1, SE_2, and SE_3. Each of these service provider edge routers connects to a router operated by a different network, in addition to connecting to the MPLS backbone. Here, the service provider added an L3VPN to enforce policies that dictate which connected networks are allowed to exchange traffic using the service provider's routing infrastructure. To accomplish this, the service provider logically partitions the network into the Default Forwarding Network (DFN), and the L3VPN.

In this example, the service provider's L3VPN uses the same physical routers as the DFN, but they maintain separate forwarding tables and routing policies using two building blocks. First, each *SE* router maintains a VRF table to separate the L3VPN routes from the default forwarding table (§4.1.1). Second, packet labels allow traffic from both the DFN and L3VPN to traverse the shared backbone, while providing sufficient information for edge routers to implement network policy (§4.1.2).

4.1.1 Virtual Routing and Forwarding. The first building block is the ability to create VRF tables, which function like default routing and forwarding tables. Each edge router in an L3VPN maintains a VRF table specific to that VPN. To preserve routing separation, VRF tables on a router do not share routes with other VRF tables or with the default forwarding table, unless explicitly configured to leak certain routes to other routing tables. Fig. 7 shows this separation, where edge router SE_3 has two forwarding tables. The first table is the DFN table, which exists regardless of whether the router participates in an L3VPN. DFN tables contain routes to support the MPLS tunnels, such as the *SP* route, and might also include routes to other networks, like the AS #A and AS #C routes.

Unlike the DFN table, SE_3's VRF table contains only routes to edge routers in the same L3VPN (SE_2) and networks connected to the L3VPN (AS #B and AS #C). If an L3VPN packet arrives at SE_3, and the L3VPN VRF at SE_3 does not contain a route to the packet's destination, SE_3 will not look up the destination in the DFN table. This behavior preserves routing separation between the L3VPN and the DFN.

4.1.2 Packet Forwarding. The second building block lets edge routers distinguish between L3VPN and DFN packets when both traverse a shared MPLS backbone. Essentially, when a packet enters the service provider over an L3VPN link, the edge router pushes a VPN identifier (VPN-ID) onto the packet. If the next hop requires tunneling the packet across the MPLS backbone, the edge router also pushes a backbone label on top of the VPN-ID.

As seen in Fig. 8, when ingress edge router SE_2 receives a packet over the L3VPN link, SE_2 looks up the next hop in the VRF. Before forwarding the packet, SE_2 first pushes the VPN identifier onto the packet, and then pushes the MPLS backbone label on top of it. As the packet is forwarded

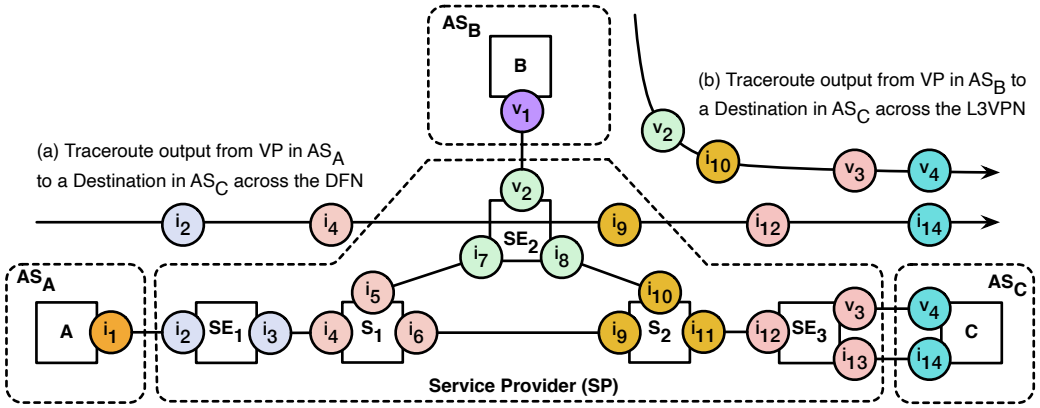


Fig. 9. A traceroute across the DFN from a VP in AS_A toward a destination in AS_C yields normal traceroute output, where routers report inbound interface addresses. A traceroute across the L3VPN from a VP in AS_B yields outbound address v_3 on router SE_3, instead of the inbound address i_{12} observed by the first traceroute.

toward SE_3, the backbone router S_2 uses only the backbone label to determine the next hop. In Fig. 8, S_2 pops the backbone label before forwarding the packet, as S_2 is the penultimate hop in the path and the subsequent router does not need that label to make its forwarding decision.

Finally, when the L3VPN packet arrives at the egress edge router SE_3, SE_3 uses the VPN-ID to distinguish L3VPN packets from DFN packets, and to differentiate among L3VPNs. SE_3 then performs the next hop lookup in the VRF associated with the VPN-ID. In Fig. 7, SE_3 looks up the packet in the L3VPN VRF, and forwards the packet through the VRF interface v_3 to reach C. It is this lookup in the egress VRF that causes problems for traceroute analysis. If the TTL expires and the egress edge router SE_3 sends an ICMP message, SE_3 reports the outbound interface address it would have used to forward the packet in the ICMP message, instead of reporting the inbound or off-path interface address that traceroute analysis typically assumes.

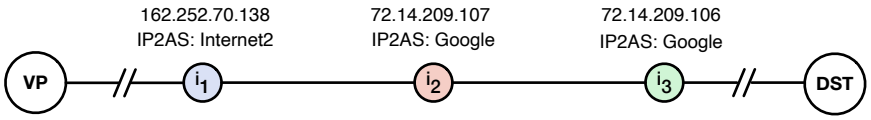
4.2 Impact on Traceroute

To show the impact of L3VPNs on traceroute output, we contrast a traceroute through a DFN with a traceroute through an L3VPN.

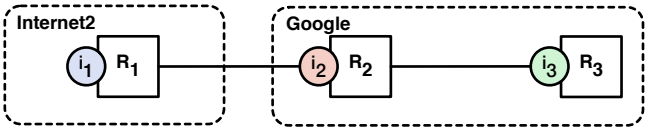
4.2.1 Traceroute Through a DFN. The first example focuses on a traceroute through the DFN. In Fig. 9a, a vantage point in AS #A sends TTL-expiring probes toward a destination in AS #C over the DFN. When the first probe enters the service provider network, SE_1 decrements the TTL, drops the probe, and responds with the inbound address i_2. If the backbone routers S_1 and S_2 respond, they also respond with their inbound addresses – i_4 and i_9, respectively. When the probes reach SE_3 without a VPN ID label, the next hop lookup uses the default forwarding table. Finally, both SE_3 and C respond with their inbound interface addresses i_{12} and i_{14}, as expected.

This example demonstrates the expected output when traceroutes traverse default forwarding networks. Each responding hop reports the address of the interface that received the traceroute probe. In reality, a router might report an off-path address if it responds through a different interface. Regardless of whether the addresses correspond to inbound or off-path interfaces, no router in the DFN responds with an outbound address from the traceroute destination's side of the router.

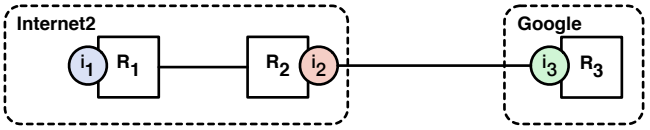
4.2.2 Traceroute Through an L3VPN. The situation is different for the L3VPN. In Fig. 9b, a VP in B sends traceroute probes to a host in AS_C across the L3VPN. The service provider ingress router SE_2



(a) Traceroute segment that traversed an L3VPN configured on an Internet2 router with interface i_2.



(b) Incorrect topology inferred by bdrmapIT, which inferred R_2 to be operated by Google.



(c) Underlying topology, where R_2 is operated by Internet2.

Fig. 10. Traceroute segment that traversed an L3VPN leading to an incorrect AS operator inference, because bdrmapIT assumed that `72.14.209.107` was an inbound address on R_2.

recognizes that all of the probes belong to the L3VPN, so SE_2 responds to the first traceroute probe with the inbound VRF address v_2. The backbone router S_2 responds as it would to traceroute probes through the DFN, reporting the inbound address i_{10}. When a traceroute probe's TTL expires at the egress edge router SE_3, rather than report the inbound address i_{12} or some other off-path address in the ICMP response, SE_3 reports the outbound interface address v_3, which is the interface that SE_3 would have used to forward the probe packet to its destination. Finally, router C responds with its inbound address v_4, representing the other side of that same IP link.

4.2.3 Real World Example. We demonstrate the potential problems caused by L3VPN outbound address responses using traceroutes between December 24, 2019 and January 7, 2020 from 159 CAIDA Ark vantage points. We inferred router operators using the bdrmapIT approach [49]. These router ownership inferences provide an example of erroneous topology inference caused by an L3VPN egress edge router reporting the address of its outbound interface. One example was caused by the traceroute segment shown in Fig. 10a. bdrmapIT first mapped the addresses to their originating AS numbers, correctly mapping the first address to Internet2 (AS11537), and the remaining addresses to the Google (AS15169).

The problem for bdrmapIT, and for its predecessors bdrmap and MAP-IT, is that they rely on the assumption that the reported addresses in traceroute either correspond to the inbound or an off-path interface. bdrmapIT concludes that `72.14.209.107` identifies an inbound interface on R_2 operated by Google, since the alternative unlikely explanation is a link between Internet2 routers R_1 and R_2 using Google IP addresses. This assumption leads bdrmapIT to incorrectly infer the topology in Fig. 10b, where `72.14.209.107` is assigned to an interface on a Google router.

In reality, `72.14.209.107` identifies an interface on an Internet2 router used for the Internet2 Peer Exchange (Fig. 10c). Internet2 implements this service as an L3VPN, separating the Peer Exchange traffic from the traffic between its members that traverses its DFN. As a result, when the L3VPN egress edge router responds to the traceroute probe, it reports the outbound address `72.14.209.107`.

This particular interface uses an address from Google's address space, since it connects a Google router to the L3VPN. We provide a solution to this mistaken inference in §6.4.

4.3 Discussion

Networks have used L3VPNs for many years, even in the public Internet, but to the best of our knowledge, this outbound address phenomenon is unknown to the research community except for a brief mention by Luckie *et al.* in 2016 [42]. The authors there mentioned the possibility of outbound addresses in traceroute, but did not cite any documentation or consider limitations that L3VPNs posed to their technique, leaving their approach just as susceptible to mistaken inferences as the bdrmapIT technique. This behavior, where L3VPN egress edge routers report the outbound address, is also not specified in any RFC or in official manuals. It does appear, however, in tutorials and official forum answers from Cisco [52] and Juniper [13].

Despite strong evidence that Cisco and Juniper routers report L3VPN outbound addresses in response to ICMP TTL-expiring probes, including confirmation from employees and observations against our ground truth, it is possible that some routers respond differently. There might also be scenarios where routers report outbound addresses when sending ICMP Time Exceeded messages outside of L3VPNs, of which we are unaware. In this work, we identify outbound addresses in traceroute responses. The more ambitious goals of discovering all instances of L3VPNs in a traceroute collection, when the L3VPN egress edge router does not respond with its outbound address, or disambiguating the causes of the outbound addresses, remains future work.

5 METHODOLOGY

In this section, we present vrfinder, a technique for identifying L3VPN outbound addresses in traceroute. Our approach in vrfinder follows from the observation that outbound addresses typically cause the appearance of addresses from the same IP subnet in a traceroute (§3) when the adjacent router responds with an inbound address. vrfinder exploits this knowledge when trying to identify outbound addresses in a traceroute collection. The first step (§5.1) builds a set of candidate outbound addresses seen in the middle of traceroutes by looking for adjacent addresses that likely represent the same point-to-point IP link. The second step (§5.2) prunes the candidates using additional probing, IXP datasets, and router aliases. Finally, the third step (§5.3) uses additional traceroute probing to identify outbound addresses among those addresses that only appear at the end of traceroutes.

5.1 Finding Candidate Outbound Addresses

The first step tries to find candidate outbound addresses (COAs) in the middle of traceroutes, using two heuristics. The first heuristic looks for adjacent consecutive addresses that may belong to the subnet used to form a point-to-point link, the type of link generally used for interdomain links, and by extension the types of L3VPN links used in service provider networks. As described in §3, IPv4 point-to-point links are usually addressed from either a /30 or /31 subnet [55]. IPv6 point-to-point links are usually addressed from /64, /126, and /127 subnets [31, 38, 59], although operators may use any subnet size between 64 and 127. We detect a candidate outbound address when we observe the other address in an assumed subnet on the adjacent traceroute hop. The second heuristic looks for adjacent addresses in traceroute that belong to the same IXP subnet; the address that appeared first in traceroute is a candidate outbound address.

5.2 Pruning Candidate Outbound Addresses

The goal of the second step is to distinguish COAs that are L3VPN outbound addresses from COAs that are caused by consecutive subnets (§5.2.1), forwarding loops (§5.2.2), and inefficient forwarding (§5.2.3).

5.2.1 Pruning Consecutive Subnets. Traceroute can expose addresses from consecutive subnets that lead to adjacent consecutive addresses in traceroute output (§3.3). The goal of this step is to identify traceroute-adjacent addresses that belong to different two-address subnets. In Path K in Fig. 5, `192.0.2.6` could be a COA due to the adjacent address `192.0.2.5`, and both addresses could be contained in the `192.0.2.4/30` four-address subnet. In reality, `192.0.2.5` and `192.0.2.6` are from two consecutive IPv4 /31 two-address subnets.

To prune these COAs, we examine the responsiveness of all addresses in a four-address subnet (IPv4 /30 or IPv6 /126) whose address sequence in traceroute could imply a four-address subnet. If we elicit a response from an unusable address, we infer the COA does not belong to a four-address subnet. Returning to the example in Fig. 5, we ping the four addresses in the potential `192.0.2.4/30` subnet; if we receive a response from an unusable address, we conclude that the COA does not come from a four-address subnet, and thus `192.0.2.6` likely belongs to a two-address /31 subnet, ruling it out as a outbound address. If we receive responses from both usable host addresses, and none of the unusable addresses respond, we retain our inference that the COA belongs to a four-address subnet. There are three other possible test results.

First, we might receive a response only from the COA, which suggests that the candidate belongs to a four-address subnet. If the COA belonged to a two-address subnet, we would usually expect the COA's router to use an address in the adjacent subnet and therefore respond to probes sent to that address as well. For example, router R_2 in Fig. 5 is assigned both `192.0.2.4` and `192.0.2.6`, so if R_2 replies to pings for `192.0.2.6`, we would expect replies for pings to `192.0.2.4` as well. We therefore classify this response pattern as a *probable* indication that the COA is an outbound address.

Second, we might receive a reply only from the other address in the assumed four-address subnet, but not from the COA itself. Here, we cannot apply the same reasoning as before, because there is no guarantee that the router with the other address in the subnet uses any other pinged address, rendering the test *inconclusive*. Third, we might not receive a reply to any ping, once again rendering the test inconclusive.

Aside from these inconclusive test results, the most important limitation of this pruning approach is that we remove any COA when our ping test demonstrates it does not belong to a four-address subnet, in keeping with our assumption that L3VPNs use point-to-point links to connect to other networks. Occasionally, we might discard an outbound address that belongs to a point-to-multipoint link with a subnet containing more than four addresses.

5.2.2 Pruning Forwarding Loops. Traceroute can expose forwarding loops that lead to adjacent addresses from the same subnet in traceroute output (§3.2). The goal of this step is to distinguish COAs caused by L3VPNs (Path C in Fig. 3) from router loops where we do not observe repeated addresses (Path H in Fig. 4). This step does not consider COAs inside an address cycle, which exist when an address appears multiple times in a single traceroute path with at least one other address separating the repeated address (Paths F and G in Fig. 4). Our approach to resolving loops consists of three heuristics, which we apply to COAs outside of an IXP.

Traceroute Test: First, we use a traceroute test to determine if the COA is observed in a traceroute path toward the apparent inbound address that followed the COA. In order to confirm the COA, the new traceroute path must terminate at the apparent inbound address and contain a response from the COA at the hop immediately prior. If the new traceroute exposes a different

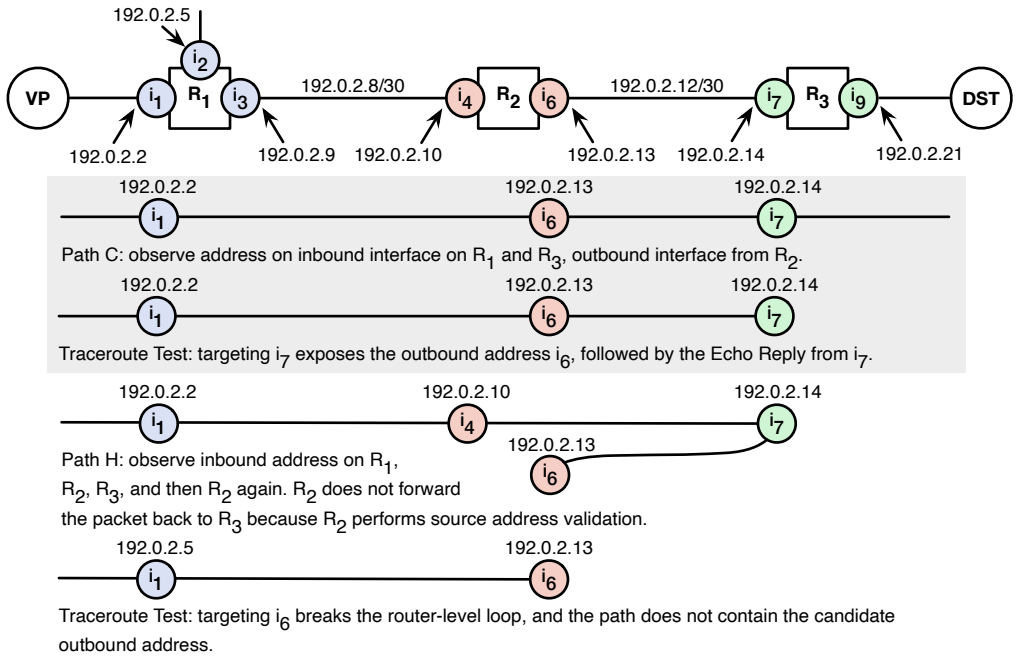


Fig. 11. The traceroute test can differentiate COAs caused by L3VPNs from those caused by incomplete forwarding loops.

address at the penultimate hop, and appears to traverse the same AS to reach the apparent inbound address, we reject the COA. We check that the new traceroute traverses the same AS as the original traceroute to ensure that it did not reach the apparent inbound address via an alternate AS path, potentially bypassing the L3VPN. Otherwise, the test is inconclusive.

Fig. 11 illustrates the approach, using two paths we previously discussed in §3. Path C in Fig. 11 shows COA `192.0.2.13` appearing before `192.0.2.14`, so we issue a new traceroute toward `192.0.2.14`. Since this traceroute still exposes `192.0.2.13`, we infer that the COA was actually an outbound address and not an artifact of a forwarding loop. When we conduct the traceroute test for Path H in Fig. 11, which contains a forwarding loop, we do not see the COA on R_3 in the path. Instead, the traceroute terminates at R_2, which responds with `192.0.2.13`. In this case the penultimate hop `192.0.2.5` belongs to AS #A's address space, as does the address that appeared before COA in the original traceroute, allowing us to reject the COA. If `192.0.2.5` belonged to a different AS address space, we could not confirm or reject the COA since the traceroute might have taken different AS paths to R_2 in the original and new traceroutes.

Alias Test: Second, because not all routers respond to probes addressed to them, we use an alias resolution test to infer if the COA is an alias of a router that appeared earlier in the path, i.e., the COA is part of a loop. For Path H in Fig. 11, if `192.0.2.10` and `192.0.2.13` were aliases, then we infer there was a loop in the path and the COA `192.0.2.14` is not an outbound address. To infer aliases, we use the MIDAR [37] and iffinder [36] probe-based alias resolution techniques, and the Hoiho [43] DNS-based alias resolution technique.

Probable Loops: Third, because not all routers respond to traceroute probes addressed to them, or support alias resolution, we use an analytic approach to infer if a COA is likely part of a loop.

When a COA primarily appears inside of address cycles, we infer that the COA likely results from a persistent forwarding loop.

5.2.3 Multiple IXP Hops. Traceroute can expose addresses from an IXP subnet that could either be caused by an outbound address or inefficient forwarding (§3.4). We try to distinguish these situations by comparing the AS that announces the addresses observed immediately prior to the COA with the AS that operates the COA according to a mapping of IXP addresses to AS operators. If the ASes are the same, or are siblings, the we infer the COA is an outbound address. Returning to Path L in Fig. 6, we check if the AS assigned to the COA `203.0.113.4` is the same AS that originates the longest matching prefix for `192.0.2.2`; in Path L, this is AS #A. For Path M in Fig. 6, AS #B operates the COA `203.0.113.8`, while AS #A operates the prior address `192.0.2.10`, so we infer this path was caused by inefficient routing. If the IXP dataset does not have a mapping for an IXP COA, the test is inconclusive.

5.3 Last Hop Probing

Some addresses in a traceroute collection never appear in the middle of a traceroute, i.e. when they appear in a traceroute, they are always the last hop. This occurs for last hop addresses in a service provider where subsequent routers in the connected networks do not respond to TTL-expiring probes. This situation is uncommon for normal traceroute responses, since service provider border routers often connect to multiple networks, one of which might respond to traceroute probes. The likelihood increases, however, for outbound addresses that connect the L3VPN to one other network. When that other network's routers do not respond to TTL-expiring probes, the L3VPN outbound address always appears at the end of traceroutes. In Path E of Fig. 3, the address `192.0.2.13` never appears in the middle of a traceroute, so we could not identify it as a COA using this path alone. We use additional probing to discover COAs when the other address in the subnet did not appear in the traceroutes.

Traceroute collections might have many last hop addresses, so we only probe the most likely candidates. Since an address can appear in the middle of traceroutes from one vantage point (VP) but not from others, we collect the last hops from each VP, recording their prefix-to-AS mapping and the destination ASes of their traceroutes. Service providers usually use their own subnets for interconnections with their customers, so we remove any last hops with a destination AS that matches its prefix-to-AS mapping. This keeps only the last hops most likely to indicate an AS interconnection.

Next, we try to expose the other side of the IP link for each potential last hop outbound address. For each potential last hop outbound address, we conduct a traceroute to the possible other addresses in their two and four address subnets from the original VP. Using the example Path E from Fig. 3 with potential last hop outbound address `192.0.2.13`, we conduct traceroutes to `192.0.2.12` and `192.0.2.14`, as these traceroutes might elicit a response from `192.0.2.12` or `192.0.2.14`, with the potential last hop outbound address `192.0.2.13` appearing in the path immediately before. These traceroute-adjacent addresses from the same subnet indicate a COA. Finally, we use the ping test to confirm the new COA if we inferred it from a four-address subnet.

6 RESULTS

We evaluated vrfinder on the traceroute collections used for the January 2020 CAIDA ITDK [17]. The IPv4 collection consists of 475.8 M traceroutes collected by 159 VPs, and the IPv6 collection has 237.2 M traceroutes collected by 69 VPs. The dataset also includes MIDAR [37] and iffinder [36] alias resolution for IPv4, and provides DNS hostnames for the IP addresses seen in the IPv4 and IPv6

traceroutes. We supplemented the alias resolution included in the ITDK using router identifiers that Hoiho [43] inferred from the IPv4 hostnames.

We collected IXP public peering prefixes from CAIDA's IXPs dataset [16]. The IXP dataset also provided many individual IP addresses assigned to ASes at IXPs, required for pruning the IXP COAs. We found AS assignments for 57.3% and 70.7% of the IPv4 and IPv6 IXP COAs respectively. If an assigned address found in the dataset was not covered by an IXP prefix, then we added a /24 covering prefix for IPv4 addresses and a /64 covering prefix for IPv6 addresses. This added 200 to the 1055 IPv4 prefixes and 100 to the 903 IPv6 prefixes.

Our IXP pruning heuristic also relies on AS address space and organizational information. We inferred AS address space ownership using contemporaneous BGP route announcements collected by RouteViews [12] and RIPE RIS [11], supplemented with additional prefixes found in RIR extended delegation files [1–3, 7, 10]. CAIDA's AS-to-organization dataset [5] provided a list of sibling ASes.

6.1 Running vrfinder

We ran vrfinder on the traceroute collections to identify L3VPN outbound addresses (Table 1). The first step was generating the initial COA sets. vrfinder inferred more COAs from four-address subnets than from two-address subnets (4.0x for IPv4 and 2.5x for IPv6) indicating four-address subnets are more prevalent in service provider networks. The last hop probing added 1406 and 392 COAs to the IPv4 and IPv6 COAs respectively.

After creating the initial set of COAs, we began pruning with the ping test. We received sufficient replies for a conclusive result in 89.9% of the IPv4 COA tests and 64.4% of IPv6 COA tests, removing 16.4% and 16.1% of those in IPv4 and IPv6 respectively. We kept the inconclusive COAs in the set of L3VPN outbound addresses when presenting our results, since the ping test invalidated few of the COAs with a conclusive result.

Next, we removed COAs mistakenly inferred due to forwarding loops. In IPv6, we removed 61.4% of the remaining service provider COAs, while in IPv4 we only removed 12.9%. We suspect that the higher number of unused address in IPv6 results in more forwarding loops that cause adjacent consecutive addresses in traceroute. In fact, 8.7% of the traceroutes in the IPv6 collection result in an address cycle, compared to 1.5% of the IPv4 traceroutes, supporting our theory that IPv6 traceroutes more often contain a forwarding loop than traceroutes in IPv4.

Finally, we pruned the IXP COAs, of which 56.4% and 71.8% in IPv4 and IPv6 respectively had an AS assignment in the IXP dataset. Since we retained few of the IXP COAs with an AS assignment, we removed any IXP COA without an AS assignment. Without knowing the AS assigned using the IXP COA, we cannot differentiate between inefficient peering and L3VPN outbound addresses, so we removed those COAs to avoid false inferences.

Of the addresses in the middle of traceroutes in the collection, the inferred L3VPN outbound addresses accounted for 5.8% of the IPv4 addresses and 1.7% of the IPv6 addresses. These results demonstrate the prevalence of L3VPN outbound addresses in the IPv4 traceroutes, and suggest that L3VPN outbound addresses appear less frequently in the IPv6 traceroutes.

6.2 Validation Against Ground Truth

We validated our inferences against ground truth from Internet2 and REANNZ, two large research and education networks (Table 2). Internet2 provided us with its Juniper router configurations, and REANNZ provided us with their L3VPN addresses. Internet2 used its DFN to connect customers to each other, and its L3VPN to connect its customers to transit providers and third party services. Both the DFN and L3VPN addresses appeared in traceroutes, and the Internet2 router configurations included both L3VPN and DFN addresses. REANNZ used L3VPNs to enforce routing policies, while

	IPv4		IPv6	
Candidate outbound addresses				
Service provider middle addresses	139,193		11,607	
Service provider last-hop addresses	1,406		392	
IXP middle addresses	1,002		529	
Total COAs	**141,601**		**12,528**	
Phase 1: Filter service provider four-address subnets				
Confirmed: only usable addresses responded	79,619	56.6%	4,483	37.4%
Rejected: unusable address responded	(16,544)	11.8%	(888)	7.4%
Probable: only COA responded	4,939	3.5%	150	1.3%
Inconclusive: insufficient responses	11,350	8.1%	3,048	25.4%
Untested: two-address subnet COAs	28,147	20.0%	3,430	28.6%
Remaining service provider COAs	**124,055**		**11,111**	
Phase 2: Filter service provider loops				
Confirmed: traceroute test	67,586	54.5%	1,562	14.1%
Rejected: traceroute test	(11,988)	9.7%	(2,560)	23.0%
Rejected: alias test established loop	(1,890)	1.5%	(53)	0.5%
Rejected: candidate mostly in loops	(2,152)	1.7%	(4,214)	37.9%
Inconclusive: no loop information	40,439	32.6%	2,722	24.5%
Remaining service provider COAs	**108,025**		**4,284**	
Phase 3: Classify IXP middle addresses				
Confirmed: Inferred outbound	109	10.9%	89	16.8%
Rejected: Inferred inefficient routing	(456)	45.5%	(291)	55.0%
Inconclusive: no participant information	(437)	43.6%	(149)	28.2%
Total IXP outbound addresses	**109**		**83**	
Outbound addresses inferred				
Service provider middle addresses	106,619		3,892	
Service provider last-hop addresses	1,406		392	
IXP middle addresses	109		89	
Total inferred outbound addresses	**108,134**		**4,373**	

Table 1. The initial set of COAs, and number of COAs kept or (pruned) at each phase, and the number of inferred outbound addresses. Phase 1 only prunes service provider COAs, Phase 2 only considers the service provider COAs retained in Phase 1, and Phase 3 only prunes IXP COAs.

the DFN only supported the MPLS tunnels. As a result, all REANNZ addresses in the traceroute collection belonged to L3VPNs.

Table 3 shows the results of our inferences compared to our ground truth as vrfinder added last hop COAs and progressed through its pruning stages. The positive predictive value (PPV) indicates the reliability of our positive L3VPN outbound address identifications. Here, a false positive is a DFN address that vrfinder flagged as an L3VPN outbound address, so we only present PPV results for Internet2. The true positive rate (TPR) provides the fraction of L3VPN outbound addresses in the ground truth that vrfinder correctly detected.

The initial set of COAs suffered from both false detections and missing inferences. Unresponsive routers caused all but one of the missing inferences for REANNZ, while 38.5% – 50.0% of the

Dataset	Description	IPv4	IPv6
L3VPN Outbound Address Datasets			
Internet2 VPN	Internet2's L3VPN outbound addresses.	172	130
REANNZ VPN	REANNZ's L3VPN outbound addresses.	63	26
Normal Inbound or Off-Path Address Datasets			
Internet2 DFN	Addresses in Internet2's DFN.	196	155
Internet2 Neighbor	Addresses on routers operated by Internet2 neighbors used for interconnection with Internet2.	425	268
REANNZ Neighbor	Addresses on routers operated by REANNZ neighbors used for interconnection with REANNZ.	52	20
IXP	IXP public peering addresses associated with a specific participant AS in PeeringDB [9].	16,134	9,063
Regex	Addresses with hostnames indicating the operating AS, extracted using Hoiho [43] regular expressions.	782	137
Total		17,824	9,795

Table 2. The intersection of the addresses in the seven ground truth datasets and the traceroute collections. We use the two L3VPN outbound address datasets to evaluate vrfinder (§6.2), and use all of the datasets to ensure our extensions to bdrmapIT (§6.4) improve AS ownership inferences for the L3VPN outbound addresses, and remain similar for the other datasets.

	Internet2				REANNZ	
	IPv4		IPv6		IPv4	IPv6
	PPV	TPR	PPV	TPR	TPR	TPR
Initial candidate outbound addresses	93.1%	94.8%	95.0%	90.6%	83.1%	86.4%
Add last hop service provider addresses	93.2%	95.3%	95.2%	91.5%	83.6%	–
Phase 1: Filter four-address subnets	97.6%	–	96.0%	–	–	–
Phase 2: Filter loops						
Traceroute test	–	–	98.3%	–	–	–
Alias test established loop	98.2%	–	99.2%	–	82.1%	–
Candidate mostly in loop	-	–	100.0%	–	–	–
Phase 3: Classify IXP middle addresses	100.0%	95.3%	100.0%	91.5%	82.1%	86.4%

Table 3. The true positive rate (TPR) through the different vrfinder stages for Internet2 and REANNZ. We also present the positive predictive value (PPV) for Internet since the ground truth includes DFN addresses.

Internet2 missing inferences were also due to unresponsive routers. For the remaining missing inferences, the router following the outbound address responded with addresses belonging to other subnets, suggesting they were off-path.

Next, we probed to find outbound last hop addresses, which added new outbound inferences for all but REANNZ IPv6. For the unidentified outbound last hop addresses, we either never probed them, or never received a response from the subsequent router. The last hop probing did not add any false L3VPN outbound inferences.

To address false detections among service provider COAs, vrfinder first used the ping test to identify COAs incorrectly inferred to belong to four-address subnets, correctly identifying all eight

invalid IPv4 COAs in Internet2's DFN. Next, we tried to identify transient and persistent forwarding loops. The traceroute test was inconclusive for most of the remaining false COAs, but correctly rejected three incorrect COAs in Internet2 IPv6. Alias resolution identified additional router loop COAs in all but REANNZ IPv6, and removing COAs primarily seen in address cycles correctly removed a router loop COA in Internet2 IPv6. It also discarded a valid REANNZ IPv6 COA, since it appeared in an address cycle nine times, but only once outside of a cycle. In the last pruning step, vrfinder removed 3 false IXP address COAs in Internet2 IPv4.

Overall, vrfinder achieved 100.0% PPV and 82.1% – 95.3% TPR for our two ground truth datasets. The missing inferences, especially for REANNZ, highlight the fact that vrfinder cannot identify an outbound address when the subsequent router does not respond to traceroute or ping probing. Finally, these results only include addresses that responded with either ICMP Time Exceeded or Destination Unreachable error types, since traceroute interpretation typically expects routers to report the inbound interface address. Including the ICMP Echo Replies in the evaluation reduced the recall for Internet2 and REANNZ in IPv4 to 94.8% and 78.6% respectively, and to 88.8% for Internet2 in IPv6.

6.3 Discussions with Network Operators

After developing all but the traceroute test of the vrfinder methodology, we spoke with network operators at five large commercial ISPs and a European R&E network. We asked them about specific vrfinder outbound address inferences from their IPv4 address space, looking for additional insight. One operator at a global ISP tested our L3VPN outbound address assumption on the ISP's network, confirming that the L3VPN egress router always reported the outbound address. Another operator at a transit provider in Oceania informed us that the ISP virtualizes nearly all of its topology, and confirmed that the outbound inferences we sent belonged to VRFs. The R&E operator also confirmed all but one of our outbound inferences for the R&E network.

However, operators at the other three ISPs told us that none of their L3VPNs should appear in our traceroute collection. After sending them a sample of the original traceroutes that led to the outbound inferences, they informed us that many of our mistaken inferences resulted from traceroutes doubling-back toward the VP. Their routers, upon receiving the packet back from their customers, responded to the first traceroute probe, but SAV discarded subsequent probes. In one of the networks, transient route changes also appeared to account for some of the mistakes. These conversations led us to create the traceroute test (§5.2.2), and suggest that vrfinder's outbound inferences might not be as accurate as the evaluation against ground truth indicates in §6.2.

6.4 Extending bdrmapIT to Account for L3VPN Outbound Addresses

To show the potential benefits of accounting for L3VPN outbound addresses, we extended bdrmapIT to account for them when inferring AS ownership for the IP addresses in traceroute. This required first using vrfinder to identify outbound addresses in the traceroute collection, changing how bdrmapIT represents traceroute adjacencies involving outbound addresses, and modifying the way bdrmapIT infers AS ownership for outbound addresses. Appendix A contains a complete explanation of our approach and the extensions to bdrmapIT.

We validated the L3VPN extensions to bdrmapIT against the L3VPN ground truth described in §6.2, using the same IPv4 and IPv6 traceroute collections as before, and included five additional ground truth datasets to ensure that our modifications did not adversely affect ownership inferences for inbound or off-path addresses (Table 2). The Internet2 and REANNZ datasets included addresses used on the routers in neighboring networks to connect to the L3VPNs, and Internet2 included the DFN addresses used on its routers as well as the addresses used on routers in other networks to connect to the DFN. We also included the IXP participant address assignments in PeeringDB [9],

Dataset	IPv4			IPv6		
	bdrmapIT	L3VPN Ext.	Change	bdrmapIT	L3VPN Ext.	Change
L3VPN Outbound Address Datasets						
Internet2 VPN	63.4%	92.4%	+29.1%	61.5%	89.2%	+27.7%
REANNZ VPN	79.4%	88.9%	+9.5%	72.7%	95.5%	+22.7%
Normal Inbound or Off-Path Address Datasets						
Internet2 DFN	96.9%	96.9%	–	93.5%	93.5%	–
Internet2 Neighbor	95.5%	95.8%	+0.2%	92.2%	93.3%	+1.1%
REANNZ Neighbor	92.3%	90.4%	-1.9%	100.0%	100.0%	–
IXP	92.3%	92.4%	+0.1%	91.1%	91.2%	+0.1%
Regex	92.7%	92.8%	+0.1%	83.9%	83.9%	–

Table 4. Our extensions increased router AS ownership accuracy for Internet and REANNZ L3VPN outbound addresses, without significantly affecting the other addresses.

and adapted Hoiho [43] to generate regular expressions that extract AS ownership information from DNS hostnames (Regex).

Table 4 shows the accuracy of our AS ownership inferences with and without the L3VPN extensions for bdrmapIT. Our extensions improved the AS ownership accuracy for the L3VPN outbound addresses by 9.5% – 29.1%, and produced nearly identical results for the other ground truth datasets. Crucially, the extensions reduced the ownership accuracy only for REANNZ Neighbor in IPv4, with one additional incorrect inference. For the other datasets, the new heuristics either had no effect or slightly improved the accuracy of bdrmapIT's inferences. Overall, these results show that detecting and accounting for L3VPN outbound addresses can improve the accuracy of AS ownership inferences for the outbound addresses.

7 CONCLUSION

Traceroute analyses, and the Internet measurements that build on them, rely on axioms, such as which router interfaces are represented in traceroute. In this paper, we described how L3VPN exit routers violate these axioms by reporting outbound addresses in response to traceroute probes. We also showed that these L3VPN outbound addresses cause bdrmapIT, the state-of-the-art in router ownership inference, to incorrectly infer router operators by violating its foundational assumptions. These kinds of mistakes can impact a variety of Internet infrastructure research, such as causing interdomain congestion measurement tools to incorrectly measure internal links, so we modified bdrmapIT to account for L3VPN outbound addresses.

In 2006, Augustine *et al.* [20] showed that load balancing affected measured traceroute paths, potentially confounding Internet measurements. They showed that changing Internet technologies force us to revisit assumptions to ensure the validity of our measurements, and invented Paris Traceroute as a way to avoid these problems. Similarly, the widespread deployment of BGP/MPLS L3VPNs must alter the way we analyze traceroute. To that end, we presented vrfinder to detect L3VPN outbound addresses in traceroute, so that researchers can account for them in their analyses.

We validated our approach in vrfinder with ground truth from two research and education networks on large traceroute collections, demonstrating its reliability and coverage. We have publicly released our vrfinder [48] and bdrmapIT [47] code so researchers can account for L3VPN outbound addresses in their measurements and analysis. As future work we hope to use vrfinder to

study L3VPN deployment throughout the Internet, hopefully providing a better understanding of ISP networks and their evolution.

ACKNOWLEDGMENTS

We thank the network operators at REANNZ and Internet2 for providing detailed information about their network and taking the time to answer our questions. This work was supported by NSF OAC-1724853, NSF C-ACCEL OIA-1937165, and DARPA Cooperative Agreement HR00112020014.

REFERENCES

[1] [n.d.]. AFRINIC Extended Allocation and Assignment Reports. ftp://ftp.afrinic.net/pub/stats/afrinic.
[2] [n.d.]. APNIC Extended Allocation and Assignment Reports. ftp://ftp.apnic.net/pub/stats/apnic.
[3] [n.d.]. ARIN Extended Allocation and Assignment Reports. ftp.arin.net/pub/stats/arin.
[4] [n.d.]. Border Gateway Protocol (BGP) VPNs. https://www.cisco.com/c/en/us/products/ios-nx-os-software/border-gateway-protocol-bgp-vpns/index.html.
[5] [n.d.]. The CAIDA UCSD AS to Organization Mapping Dataset. http://www.caida.org/data/as_organizations.xml.
[6] [n.d.]. Internet Exchange Report. https://bgp.he.net/report/exchanges.
[7] [n.d.]. LACNIC Extended Allocation and Assignment Reports. ftp.lacnic.net/pub/stats/lacnic.
[8] [n.d.]. PCH: Packet Clearing House. https://www.pch.net/resources/Routing_Data/.
[9] [n.d.]. PeeringDB. https://peeringdb.com/.
[10] [n.d.]. RIPE Extended Allocation and Assignment Reports. ftp://ftp.ripe.net/pub/stats/ripencc.
[11] [n.d.]. Routing Information Service (RIS). https://www.ripe.net/analyse/internet-measurements/routing-information-service-ris.
[12] [n.d.]. University of Oregon Route Views Project. http://www.routeviews.org/routeviews/.
[13] 2018. Understanding Traceroute Behavior in L3VPN Setup on Junos OS. https://kb.juniper.net/InfoCenter/index?page=content&id=KB33434&cat=MX_SERIES&actp=LIST&showDraft=false.
[14] 2019. Types of VPNs. https://www.juniper.net/documentation/en_US/junos/topics/topic-map/l3-vpns-overview.html.
[15] 2019. Understanding Carrier-of-Carriers VPNs. https://www.juniper.net/documentation/en_US/junos/topics/concept/vpn-carrier-of-carriers-vpns.html.
[16] 2020. The CAIDA UCSD IXPs Dataset. https://www.caida.org/data/ixps.
[17] 2020. Macroscopic Internet Topology Data Kit (ITDK). http://www.caida.org/data/internet-topology-data-kit/.
[18] Réka Albert, Hawoong Jeong, and Albert-László Barabási. 2000. Error and Attack Tolerance of Complex Networks. *Nature* 406 (June 2000).
[19] Lisa D Amini, Anees Shaikh, and Henning G Schulzrinne. 2002. Issues with Inferring Internet Topological Attributes. In *Internet Performance and Control of Network Systems III*.
[20] Brice Augustin, Xavier Cuvellier, Benjamin Orgogozo, Fabien Viger, Timur Friedman, Matthieu Latapy, Clémence Magnien, and Renata Teixeira. 2006. Avoiding Traceroute Anomalies with Paris Traceroute. In *IMC 2006 – 6th ACM Internet Measurement Conference*.
[21] Brice Augustin, Timur Friedman, and Renata Teixeira. 2007. Measuring load-balanced paths in the Internet. In *IMC*. San Diego, CA, USA, 149–160.
[22] Fred Baker. 1995. *RFC 1812: Requirements for IP Version 4 Routers.* Technical Report. Internet Engineering Task Force.
[23] F. Baker and P. Savola. 2004. *Ingress Filtering for Multihomed Networks.* RFC 3704.
[24] Adam Bender, Rob Sherwood, and Neil Spring. 2008. Fixing Ally's Growing Pains with Velocity Modeling. In *Proceedings of the 8th ACM SIGCOMM conference on Internet measurement*. ACM, 337–342.
[25] Amogh Dhamdhere, David D. Clark, Alexander Gamero-Garrido, Matthew Luckie, Ricky K. P. Mok, Gautam Akiwate, Kabir Gogia, Vaibhav Bajpai, Alex C. Snoeren, and Kc Claffy. 2018. Inferring Persistent Interdomain Congestion. In *SIGCOMM*.
[26] Benoit Donnet, Matthew Luckie, Pascal Mérindol, and Jean-Jacques Pansiot. 2012. Revealing MPLS Tunnels Obscured from Traceroute. *ACM SIGCOMM Computer Communication Review* (2012).
[27] P. Ferguson and D. Senie. 2000. *Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing.* RFC 2827.
[28] Brian J Goodchild, Yi-Ching Chiu, Rob Hansen, Haonan Lu, Matt Calder, Matthew Luckie, Wyatt Lloyd, David Choffnes, and Ethan Katz-Bassett. 2017. The Record Route Option is an Option!. In *IMC*.
[29] Ramesh Govindan and Hongsuda Tangmunarunkit. 2000. Heuristics for Internet Map Discovery. In *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE.*

[30] Mehmet H Gunes and Kamil Sarac. 2006. Analytical IP Alias Resolution. In *Communications, 2006. ICC'06. IEEE International Conference on*, Vol. 1. IEEE, 459–464.

[31] Robert M. Hinden and Stephen E. Deering. 2006. *RFC 4291: IP Version 6 Addressing Architecture.* Technical Report. IETF.

[32] Bradley Huffaker, Amogh Dhamdhere, Marina Fomenkov, et al. 2010. Toward Topology Dualism: Improving the Accuracy of AS Annotations for Routers. In *International Conference on Passive and Active Network Measurement*. Springer, 101–110.

[33] Y. Hyun, A. Broido, and k. claffy. 2003. On Third-party Addresses in Traceroute Paths. In *PAM*.

[34] M. S. Kang, S. B. Lee, and V. D. Gligor. 2013. The Crossfire Attack. In *2013 IEEE Symposium on Security and Privacy*.

[35] Ethan Katz-Bassett, Harsha V Madhyastha, Vijay Kumar Adhikari, Colin Scott, Justine Sherry, Peter Van Wesep, Thomas E Anderson, and Arvind Krishnamurthy. 2010. Reverse Traceroute. In *NSDI*, Vol. 10. 219–234.

[36] Ken Keys. [n.d.]. iffinder. https://www.caida.org/tools/measurement/iffinder/.

[37] Ken Keys, Young Hyun, Matthew Luckie, and Kim Claffy. 2013. Internet-Scale IPv4 Alias Resolution with MIDAR. *IEEE/ACM Transactions on Networking (TON)* (2013).

[38] Miya Kohno, Becca Nitzan, Randy Bush, Yoshinobu Matsuzaki, Lorenzo Colitti, and Thomas Narten. 2011. *RFC 6164: Using 127-Bit IPv6 Prefixes on Inter-Router Links.* Technical Report. IETF.

[39] Qasim Lone, Matthew Luckie, Maciej Korczyński, and Michel van Eeten. 2017. Using Loops Observed in Traceroute to Infer the Ability to Spoof. In *International Conference on Passive and Active Network Measurement*. Springer, 229–241.

[40] Matthew Luckie and Robert Beverly. 2017. The Impact of Router Outages on the AS-level Internet. In *Proceedings of ACM SIGCOMM*.

[41] Matthew Luckie and Kc Claffy. 2014. A Second Look at Detecting Third-Party Addresses in Traceroute Traces with the IP Timestamp Option. In *Proceedings of the 15th International Conference on Passive and Active Measurement-Volume 8362*.

[42] Matthew Luckie, Amogh Dhamdhere, Bradley Huffaker, David Clark, and kc claffy. 2016. bdrmap: Inference of Borders Between IP Networks. In *IMC*.

[43] Matthew Luckie, Bradley Huffaker, and k claffy. 2019. Learning Regexes to Extract Router Names from Hostnames. In *Proceedings of the Internet Measurement Conference*. 337–350.

[44] Matthew Luckie, Bradley Huffaker, Amogh Dhamdhere, Vasileios Giotsas, and kc claffy. 2013. AS Relationships, Customer Cones, and Validation. In *Proceedings of the ACM SIGCOMM Internet Measurement Conference (IMC)*.

[45] Zhuoqing Morley Mao, Jennifer Rexford, Jia Wang, and Randy H Katz. 2003. Towards an Accurate AS-Level Traceroute Tool. In *Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications*. ACM, 365–378.

[46] Pietro Marchetta, Walter de Donato, and Antonio Pescapé. 2013. Detecting Third-Party Addresses in Traceroute Traces with IP Timestamp Option. In *Proceedings of the 14th international conference on Passive and Active Measurement*.

[47] Alexander Marder. 2020. bdrmapIT GitHub Code Respository. https://github.com/alexmarder/bdrmapit.

[48] Alexander Marder. 2020. vrfinder GitHub Code Respository. https://github.com/alexmarder/vrfinder.

[49] 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 *IMC*.

[50] Alexander Marder and Jonathan M Smith. 2016. MAP-IT: Multipass Accurate Passive Inferences from Traceroute. In *Proceedings of the 2016 Internet Measurement Conference*. ACM, 397–411.

[51] Pascal Mérindol, Benoit Donnet, Jean-Jacques Pansiot, Matthew Luckie, and Young Hyun. 2011. MERLIN: MEasure the Router Level of the INternet. In *Proceedings of the 7th Euro-NF Conference on Next Generation Internet (NGI)*. 1–8.

[52] Nagendra Kumar Nainar. 2018. MPLS VPN Traceroute. https://community.cisco.com/t5/routing/mpls-vpn-traceroute/m-p/3768060.

[53] Jean-Jacques Pansiot, Pascal Mérindol, Benoit Donnet, and Olivier Bonaventure. 2010. Extracting Intra-Domain Topology from mrinfo Probing. In *PAM*. 81–90.

[54] Vern Paxson. 1997. End-to-End Internet Packet Dynamics. In *ACM SIGCOMM Computer Communication Review*, Vol. 27. ACM, 139–152.

[55] Alvaro Retana, Russ White, Vince Fuller, and Danny McPherson. 2000. *RFC 3021: Using 31-Bit Prefixes on IPv4 Point-to-Point Links.* Technical Report. IETF.

[56] Eric Rosen and Yakov Rekhter. 1999. *RFC 2547: BGP/MPLS VPNs.* Technical Report. Internet Engineering Task Force.

[57] Eric Rosen and Yakov Rekhter. 2006. *RFC 4364: BGP/MPLS IP Virtual Private Networks (VPNs).* Technical Report. IETF.

[58] Jan Rüth, Torsten Zimmermann, and Oliver Hohlfeld. 2019. Hidden Treasures–Recycling Large-Scale Internet Measurements to Study the Internet's Control Plane. In *International Conference on Passive and Active Network Measurement*. Springer, 51–67.

[59] Pekka Savola. 2003. *RFC 3627: Use of /127 Prefix Length Between Routers Considered Harmful.* Technical Report. IETF.

[60] Justine Sherry, Ethan Katz-Bassett, Mary Pimenova, Harsha V Madhyastha, Thomas Anderson, and Arvind Krishnamurthy. 2010. Resolving IP Aliases with Prespecified Timestamps. In *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement*. ACM, 172–178.
[61] Rob Sherwood, Adam Bender, and Neil Spring. 2008. DisCarte: A Disjunctive Internet Cartographer. In *SIGCOMM*. 303–314.
[62] Neil Spring, Mira Dontcheva, Maya Rodrig, and David Wetherall. 2004. *How to Resolve IP Aliases*. Technical Report UW-CSE-TR 04–05–04. University of Washington.
[63] Neil Spring, Ratul Mahajan, and David Wetherall. 2002. Measuring ISP topologies with Rocketfuel. In *SIGCOMM*. 133–145.
[64] Richard A Steenbergen. 2009. A Practical Guide to (Correctly) Troubleshooting with Traceroute. *NANOG* (2009).
[65] Ahren Studer and Adrian Perrig. 2009. The Coremelt Attack. In *European Symposium on Research in Computer Security*.
[66] 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*. ACM, 29–42.
[67] Jianhong Xia, Lixin Gao, and Teng Fei. 2007. A Measurement Study of Persistent Forwarding Loops on the Internet. *Computer Networks* 51, 17 (2007), 4780–4796.
[68] Bahador Yeganeh, Ramakrishnan Durairajan, Reza Rejaie, and Walter Willinger. 2019. How Cloud Traffic Goes Hiding: A Study of Amazon's Peering Fabric. In *IMC*. Amsterdam, NL, 202–216.
[69] Yu Zhang, Ricardo Oliveira, Hongli Zhang, and Lixia Zhang. 2010. Quantifying the Pitfalls of Traceroute in AS Connectivity Inference. In *International Conference on Passive and Active Network Measurement*.
[70] Jan Zorz, Sander Steffan, Primoz Drazumeric, Mark Townsley, Andrew Alston, Gert Doering, Jordi Palet, Jen Linkova, Luis Balbinot, Kevin Meynell, and Lee Howard. 2017. *Best Current Operational Practice for Operators: IPv6 Prefix Assignment for End-users – Persistent vs Non-persistent, and What Size to Choose*. Technical Report. RIPE NCC.

A EXTENDING bdrmapIT

After identifying L3VPN outbound addresses, we extend bdrmapIT to account for them. As described in §4.2, outbound addresses violate the assumptions in bdrmapIT, underlying decisions made primarily when constructing the router-graph (§A.1) and annotating each router with its AS operator (§A.2), both of which we modify here.

A.1 Modifying Graph Construction

In order to properly construct a graph that includes outbound address edges, we need to identify not just the addresses that were outbound addresses at least once in the traceroute collection, but also the specific interface adjacencies in the collection that involve outbound addresses. This is difficult since some interface addresses appear in the collection as both outbound addresses and inbound addresses. In Fig. 12, `192.0.2.13` is outbound on router R_2 in Paths A and B, but inbound in Path C.

A.1.1 Marking Forwarding Adjacencies. To avoid mistaking `192.0.2.13` for an outbound address in Fig. 12 Path C, we need to determine when it represents an outbound address rather than an inbound address. Complicating matters, `192.0.2.13` might not always appear prior to `192.0.2.14` when it is an outbound address, such as in Path B, where it appears prior to the off-path address `192.0.2.25`. To account for this problem, we use the *definitive instances* of `192.0.2.13` as an outbound address – when it appears immediately prior to the other side of its IP link – to determine the preceding AS address space when `192.0.2.13` represents an outbound address. In Path A, an address from AS #A's address space appears before a definitive instance of the outbound address `192.0.2.13`, indicating that `192.0.2.13` is likely outbound when it follows an address from address space belonging to AS #A.

Using this information, we mark all adjacencies leading *to* an outbound address as outbound adjacencies (OA), distinguishing them from normal adjacencies when we construct edges in the graph, and remove any adjacency going *from* an outbound address. Thus, we mark the adjacency OA(`192.0.2.2`, `192.0.2.13`) before the definitive instance of the outbound address in Fig. 12 Path

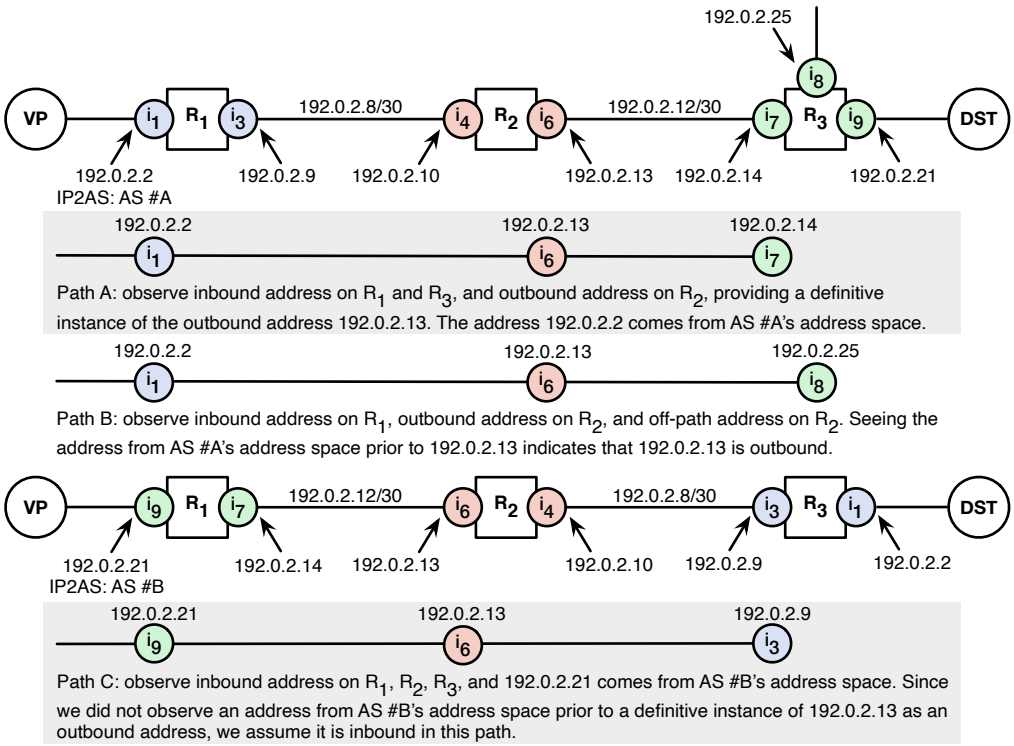


Fig. 12. We use definitive instances of an outbound address (Path A) to collect the AS address space immediately before the outbound address. We then use the prior address space to rule in outbound addresses that appear before off-path addresses (Path B), and rule out ingress addresses (Path C).

A, and use the prior AS #A to mark OA(`192.0.2.2`, `192.0.2.13`) in Fig. 12 Path B. We also discard the (`192.0.2.13`, `192.0.2.14`) and (`192.0.2.13`, `192.0.2.25`) adjacencies. In Path C, we do not mark (`192.0.2.21`, `192.0.2.13`) or remove (`192.0.2.13`, `192.0.2.9`), since AS #B never appeared before a definitive instance of `192.0.2.13` as an outbound address .

A.1.2 Adding Forwarding Adjacencies to the Graph. Next, we use the marked adjacencies to create OA edges in bdrmapIT's inferred router graph, where each router represents a group of one or more addresses on a physical router. Previously, when bdrmapIT constructed the graph, it directed all edges from a router node to a node representing an interface seen subsequently in a traceroute. This reflected the assumption that addresses in traceroute typically represent the inbound interface, and therefore provide router alias constraints. Outbound addresses violate that assumption, so we instead direct all OA edges from one router to another router, representing neighboring router operator constraints. For outbound addresses, we rely on the assumption that routers typically interconnect more frequently with routers operated by the same network than with routers operated by any other single network [32].

To reflect these neighboring router constraints, in Fig. 13a we construct the edge $R_1 \xrightarrow{OA} R_2$, since the same network likely operates R_1 and R_2. We also create the edge $R_2 \xrightarrow{OA} R_1$, instead of an edge from R_2 to R_3, for the same reason. We expect that an outbound address often indicates the

(a) OA edge to L3VPN outbound address. (b) OA edge from L3VPN outbound address.

Fig. 13. We create two edges (a) and (b) between routers for every OA adjacency.

Edge Confidence Label	Priority
Nexthop	1
Nexthop OA	2
Multihop	3
Multihop OA	4

Table 5. The updated edge confidence labels still prioritize nexthop edges over multihop edges, but also prioritize normal nexthop edges over OA edges.

border of the L3VPN service provider, in which case different networks operate R_2 and R_3 while the same network often operates R_1 and R_2.

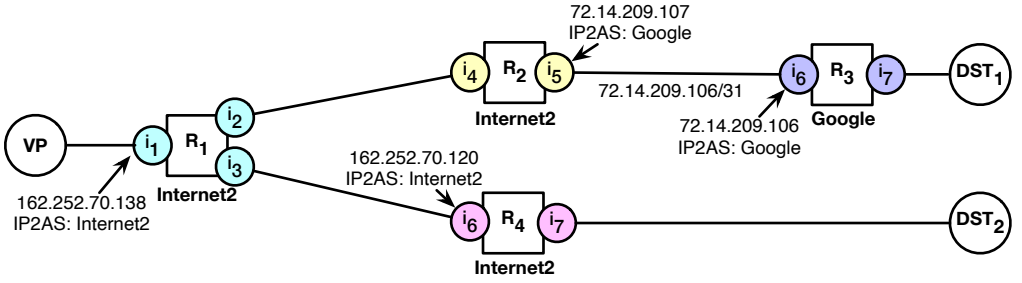
bdrmapIT previously had two edge confidence categories, prioritizing nexthop above multihop edges such that a router only uses its multihop edges when it has no nexthop edges. We modify the link confidences to include OA edges, assigning a lower edge confidence to OA edges than to normal edges, so that bdrmapIT prioritizes normal edges if available (Table 5). More precisely, bdrmapIT only uses normal nexthop edges if available, but if we converted all of a router's outgoing edges into OA edges, then the router uses those edges rather than any multihop edges. Prioritizing normal edges over outbound address edges lets bdrmapIT rely on the router alias constraints the normal edges provide when available, and also helps prevent false router operator inferences due to a false outbound address inference.

In Fig. 14 we extend the real example from §4.2.3 to include the traceroute segments in Figs. 14b and 14c. Here, vrfinder identified `72.14.209.107` as an outbound address, so we create the OA link from R_1 to the outbound address. Since R_1 still has a normal outgoing edge (Fig. 14d), bdrmapIT will only use that edge during router annotation. bdrmapIT also discards the edge from R_2 to `72.14.209.106`, redirecting it as a OA edge to R_1 instead (Fig. 14e).

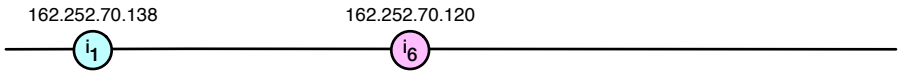
A.2 Modifying Router Annotation

When relying on OA edges, bdrmapIT determines router ownership using neighboring router constraints, but still benefits from bdrmapIT's iterative constraint satisfaction. bdrmapIT primarily annotates routers with their AS operator by selecting the most frequent AS among the constraints, and we do the same here, since we want to determine the most frequent AS operator of neighboring routers. We annotate OA routers in a separate step after annotating normal routers, allowing the neighboring router constraints to leverage normal router annotations.

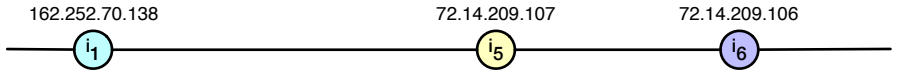
A.2.1 Annotating IRs with OA Edges. Algorithm 1 describes the technique for annotating routers relying on OA edges. Similar to bdrmapIT, we use votes to represent the constraints. Each router connected by an outgoing OA edge votes with its current AS annotation, reflecting the current router operator inference for that router. We try to rely only on votes from neighboring routers with normal outgoing edges if possible. Thus, if the router has any outgoing edges to normal routers, we use only those votes when selecting the most frequent AS. Otherwise, we add a vote for each



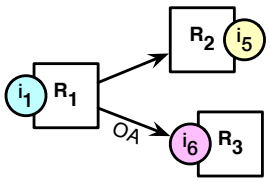
(a) Network topology based on ground truth.



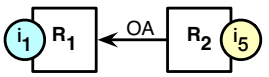
(b) Normal traceroute path.



(c) Traceroute with an L3VPN outbound address.



(d) Pruned OA edge.



(e) Redirected OA edge.

Fig. 14. Network topology based on ground truth from Internet2. From the traceroutes in (b) and (c), we create the OA links in (d) and (e).

of the router interface origin ASes, and use all of the neighboring AS votes in the election. The origin ASes for the router represent traditional router alias constraints, and provide potential AS operators for the router in addition to the neighboring router AS operators.

Next, we use the election to select the most frequent AS among the votes. First, we filter the votes to only include ASes that have a relationship with a router origin AS, and the router origin ASes themselves. This helps ensure that the selected AS directly connects to a router origin AS. When none of the neighboring router ASes have a relationship with a router origin AS, we revert to using all the votes, as this might indicate incomplete AS relationships inferred from BGP paths. We then use whatever votes remain to annotate the router with the AS receiving the most votes, breaking ties using the number of direct and indirect transit customers to select the likely customer AS, relying on the technique by Luckie *et al.* [44] to infer AS relationships from BGP AS paths. We use the same tiebreak as bdrmapIT, since we expect that the addresses for L3VPN interconnections between a service provider and a customer typically come from the provider's address space.

Returning to the real example (Fig. 15), we use the modified graph to annotate the routers with their AS operators. We annotate R_1 in the first step, using only the normal nexthop edge. Finally,

Algorithm 1 Annotating L3VPN Router, r

V: counter for AS votes
N: counter for normal router votes
for all router $j \in$ SUBSEQUENT$[r]$ **do**
    $a \leftarrow$ ANNOTATION$[j]$
    if a != NULL **then**
        INCREMENT$(V[a])$
        if j has normal edges **then** INCREMENT$(N[a])$
for all $i \in r$.INTERFACES **do** INCREMENT$(V[i.\text{AS}])$
if N is not empty **then** $V \leftarrow N$
$R \leftarrow r$.ORIGINS $\cup \{v \in V \mid \exists i \in r.\text{ORIGINS} : \text{REL}(o, v)\}$
if R != r.ORIGINS **then** $a \leftarrow \max_{v \in R} V[v]$
else $a \leftarrow \max_{v \in V} V[v]$
ANNOTATION$[r] \leftarrow a$

(a) Step 1: normal routers.

(b) Step 2: OA routers.

Fig. 15. In the first stage of router annotation (a), we annotate R_1 using its normal edge, leading to the correct annotation for R_2 in the second stage (b).

in the step that annotates OA routers, using the reversed OA edge, we rely on the inferred AS operator for R_1 to correctly determine that Internet2 operates R_2.

A.2.2 Annotation Flapping. Relying on router annotations for OA edges can lead to router annotation flapping when they point to another router that relies on OA edges. To avoid this, we annotate the routers that rely on OA edges in a fixed order, and use annotations from the same iteration of the loop. We first sort the routers in descending order by the number of outgoing edges, since we expect more reliable annotations for routers with more outgoing edges. If two routers have the same number of edges, then we break ties using the smallest number of direct and indirect transit customers among the router origin ASes. This full ordering of OA routers prevents annotation flapping and ensures that bdrmapIT always outputs the same annotations for the same input.

Received August 2019; revised September 2019; accepted March 2020