APPLE: Alias Pruning by Path Length Estimation

Alexander Marder

UC San Diego / CAIDA amarder@caida.org

Abstract. Uncovering the Internet's router graph is vital to accurate measurement and analysis. In this paper, we present a new technique for resolving router IP aliases that complements existing techniques. Our approach, Alias Pruning by Path Length Estimation (APPLE), avoids relying on router manufacturer and operating system specific implementations of IP. Instead, it filters potential router aliases seen in traceroute by comparing the reply path length from each address to a distributed set of vantage points.

We evaluated our approach on Internet-wide collections of IPv4 and IPv6 traceroutes. We compared APPLE's router alias inferences against router configurations from two R&E networks, finding no false positives. Moreover, APPLE's coverage of the potential alias pairs in the ground truth networks rivals the current state-of-the-art in IPv4, and far exceeds existing techniques in IPv6. We also show that APPLE complements existing alias resolution techniques, increasing the total number of inferred alias pairs by 109.6% in IPv4, and by 1071.5% in IPv6.

1 Introduction

Uncovering the Internet's router graph is vital to accurately analyzing and measuring the Internet. The current tool for uncovering the Internet's topology, traceroute [12], only exposes the IP addresses of router interfaces. Collapsing that to a router-level topology requires first resolving the IP address aliases for each router, a process known as *alias resolution*.

The current state-of-the-art alias resolution techniques rely on exploiting implementations of the IP on routers, such as how a router responds to Destination Unreachable packets [8,14] and populates the IP-ID field [7,15,16,25]. However, implementations can differ between router manufacturers and operating systems, limiting their ability to resolve aliases. Moreover, current RFC recommendations advise against setting the IP-ID field in IPv4 [26], and IPv6 only includes the IP-ID field for fragmented packets.

We present an alternative approach to alias resolution that avoids relying on IP implementations specific to router manufacturers and operating systems, and that resolves aliases in IPv4 and IPv6. Our approach, called Alias Pruning by Path Length Estimation (APPLE), relies only on the fact that routers in the Internet generally use destination-based forwarding. After inferring potential router aliases in traceroute graphs, we corroborate them with pings from

geographically and topologically distributed vantage points (VPs). Our hypothesis, which we validate against ground truth from two networks (§5), is that path lengths between a router and a VP remain mostly the same regardless of the source address, allowing us to distinguish between valid and invalid router aliases using reply path lengths.

In this paper, we make the following contributions,

- we present APPLE, a novel technique for inferring router aliases using reply path length;
- we compare APPLE's alias resolution inferences against a combined 71 router configurations from two large R&E networks, with no false positives; and
- we show that APPLE complements existing alias resolution techniques, increasing the total number of inferred router alias pairs of addresses by 109.6% in IPv4 and by 1071.5% in IPv6.

2 Previous Work

The earliest reliable alias resolution techniques, Mercator [8] and `iffinder`, try to induce ICMP Destination Unreachable responses. Some routers report the transmitting interface address when originating Destination Unreachable packets, indicating that the probed and transmitting interface addresses alias the same router. UAv6 [22] extends this idea, sending probes to unused addresses in /30 and /126 subnets. These techniques exploit implementations of ICMP packet generation, but many routers either report the probed address or do not respond to the probes, limiting their effectiveness.

Other approaches draw inferences from the IPv4 IP-ID field, used to aid reassembly of fragmented packets, that some routers populate using a shared counter for all of their interfaces. The Rocketfuel [25] component Ally compares pairs of addresses to see if the IP-IDs increase at similar rates. RadarGun [7] removes the need to compare each pair of addresses separately, sampling and comparing the IP-IDs for all addresses at once. MIDAR [15] also collects and analyzes IP-IDs, but ensures that the IP-IDs of inferred aliases form a monotonically increasing sequence. To address the absence of the IP-ID in normal IPv6 packets, Speedtrap [16] attempts to induce fragmented ICMP Echo Replies with IP-IDs, but some routers do not fragment packets in IPv6. In general, the future of IP-ID-based alias resolution is uncertain, as current IETF recommendations advise against setting the IP-ID in IPv4 packets outside of packet fragmentation [26].

Like APPLE, some techniques derive router aliases from the interface graph generated by traceroute. Spring *et al.* [24] assumed that most routers report the inbound interface address in response to traceroute probes, inferring aliases when addresses share a common successor. As we describe in §3, this technique tends to incorrectly infer aliases in the presence of off-path addresses, L3VPN outbound responses, hidden MPLS tunnels, and multipoint-to-point links. APAR [10] and `kapar` [13] try to discover router aliases by aligning traceroutes from multiple vantage points. When multiple ends of the same link appear in different traceroutes, they infer that addresses seen adjacent to the link are aliases of the link

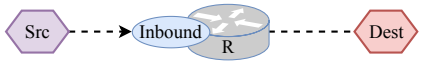


Fig. 1: Routers typically report the address of the interface that received the traceroute probe (inbound address).

(a) Traceroutes (b) Router graph.

Fig. 2: The traceroutes in (a) suggest the possible router graph in (b).

routers. Current graph analysis techniques suffer from false router alias inferences.

Furthermore, our technique is not the first to use the TTL in the reply packet (reply TTL) to guide alias resolution. Vanaubel *et al.* [29] used the reply TTL to fingerprint router manufacturers, and Grailet *et al.* [9] used those fingerprints to restrict the possible alias pairs inferred via other techniques. Unlike APPLE, they used the reply TTL to restrict the search space, not to identify alias pairs.

Most recently, `Hoiho` [18] automatically learned regular expressions for extracting router name information from DNS hostnames, with the potential to provide valuable router alias constraints. As future work, we hope to use `Hoiho` to improve APPLE's router alias inferences.

Our technique avoids many of the pitfalls inherent to prior techniques for three reasons. First, many routers that respond to the traditional pings that we send do not respond to probes specifying unused ports or invalid host addresses, or always report the probe destination address. Second, we do not rely on features of the IP header specific to IPv4 or IPv6, ensuring it generalizes to both IP versions. Third, we make no assumptions about IP link prefixes and do not accept potential aliases indicated by traceroute graphs without additional evidence.

3 Common Successor Alias Resolution

Before describing our technique, we briefly discuss traceroute interpretation and the problems with relying solely on common successors for alias resolution. Conventional traceroute interpretation assumes that when a router responds to a TTL-expiring probe, it reports the address of the interface that received the probe, known as the *inbound* address (Fig. 1). Since an interface often connects its router to exactly one other router, if two addresses both precede a third address in different traceroutes, then the two addresses might belong to the same physical router. This occurs in Fig. 2a, where the addresses a and b have the common successor c. Assuming that c is the inbound interface on its router,

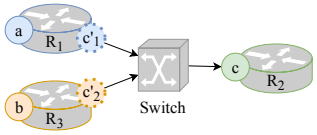


Fig. 3: Multipoint-to-point IP links connect more that two routers.

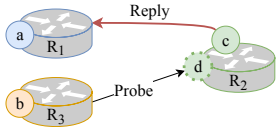


Fig. 4: Off-path address can appear subsequent to multiple routers.

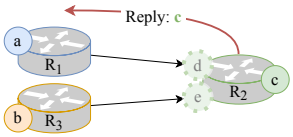


Fig. 5: The L3VPN address creates a common successor for a and b.

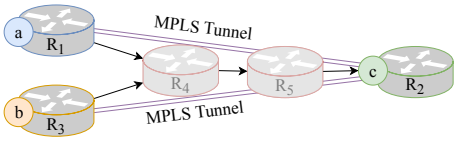


Fig. 6: Invisible MPLS tunnels make R_1 and R_3 appear interconnected with R_2.

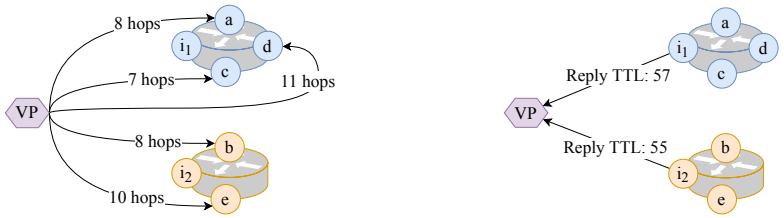
and that c connects to exactly one other router, we could infer that a and b belong to the same router (Fig. 2b). Unfortunately, many potential problems, most prominently multipoint-to-point links, off-path addresses, Layer 3 Virtual Private Networks (L3VPN), and invisible Multiprotocol Label Switching (MPLS) tunnels, confound common successor alias resolution.

Multipoint-to-Point Links Common successor alias resolution assumes that router interconnections occur over point-to-point links, but IP links can connect more than two routers. Multipoint-to-point links typically connect routers using layer 2 switches, allowing more than two routers to interconnect using the same IP subnet. In Fig. 3 the switch connects R_1, R_2, and R_3, so a and b belong to different routers but precede the inbound address c. Internet exchange points (IXPs) often use multipoint-to-point links to connect their participant's routers [3, 5].

Off-Path Addresses Even when routers connect over a point-to-point link, off-path addresses can violate the inbound address assumption. While some routers always respond with the inbound address [4], others adhere to RFC 1812 [6] and report the address of the interface used to respond to the traceroute probe. When such a router uses different interfaces to receive and reply to a traceroute probe, the router reports the off-path address instead of the inbound address [11].

In Fig. 4, R_2 received the traceroute probe through interface d but sends the reply through c, and puts c in the source address field in the reply packet. As a result, c appears immediately after b in the traceroute. If in another traceroute R_2 receives and replies to a probe through c, a might also appear prior to c.

Layer 3 Virtual Private Networks Like off-path addresses, L3VPNs violate the inbound address assumption. When L3VPN exit routers respond to traceroute probes, they report the address of the outbound interface that would have continued forwarding the packet toward the destination [17, 20], rather than the inbound interface address or a traditional off-path address. Consequently, ad-



(a) Paths from VP to router addresses. (b) Paths from routers to VP.

Fig. 7: Paths from a VP to different addresses on a router might differ (a), but the router to VP paths are often the same regardless of source address (b).

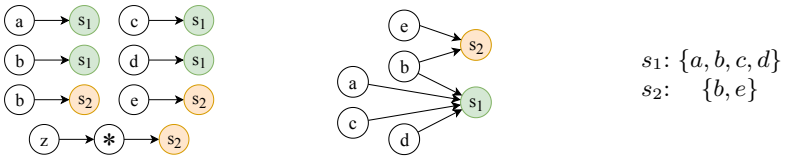
dresses on any prior router could precede the outbound address in a traceroute. In Fig. 5, R_2 reports the outbound address c, so a and b appear prior to c.

MPLS Tunnels The fourth prominent reason is that MPLS tunnels might violate the assumption that adjacent hops in traceroute indicate directly connected routers. Invisible MPLS tunnels can cause addresses from unconnected routers to appear adjacent in a traceroute path [27, 28]. When a probe packet enters an MPLS tunnel, the entry router encapsulates the probe inside an MPLS packet. Network operators can either configure the router to propagate the TTL from the encapsulated packet, or use a default value. The tunnel routers only decrement the TTL in the MPLS packet header, and not the probe's TTL, so if the entry router does not propagate the TTL, the exit router's response appears immediately after the entry router's response. This occurs in Fig. 6, where R_1 and R_3 do not propagate the probe packet TTL to the MPLS header, so R_2's response appears immediately subsequent to the responses from R_1 and R_3.

4 Methodology

Clearly, the fact that two addresses share a common successor does not always mean that the two addresses belong to the same router. However, common successors can help constrain the process of alias resolution by providing an initial set of possible router aliases. Our goal is to find pairs of addresses that belong to the same router (*alias pairs*) among addresses that share a common successor.

We infer that a pair of addresses belong to the same router by comparing the reply paths from each address to several vantage points (VPs). While the path from a VP to different addresses on the same router might differ significantly (Fig. 7a), especially when the addresses have different longest matching prefixes, all responses from the router to a given VP use the same destination address, and we hypothesize that they often share the same path (Fig. 7b). This follows from the fact that routers primarily forward packets according to their destination addresses, so the path from a router to the same destination should mostly remain the same regardless of source address. Thus, we discard potential alias



(a) Traceroute segments. (b) Interface graph. (c) Common successor sets.

Fig. 8: Based on the traceroutes (a), we create the interface graph (b). We exclude (z, s_2) due to the unresponsive hop between them. Using the incoming edges for s_1 and s_2 we create the potential router alias sets (c).

pairs when we infer that a sufficient number of the reply paths differ.We discuss how we make this decision in §4.2.

When a router originates an IP packet to a VP, that packet does not record the actual path that it traversed, but the packet does include a TTL value that routers decrement as they forward it to the VP. Routers typically initialize that TTL to either 32, 64, 128, or 255 [27,29], and the same router will always initialize the TTLs with the same value. Thus, when a VP receives a reply packet from a router, the TTL value in the packet header (*reply TTL*) indicates the path length from the router to the VP.

Our approach, APPLE, relies on the path length indications given by reply TTLs to evaluate the similarity of reply paths between addresses in potential alias pairs. APPLE performs this alias resolution in two steps. First, APPLE uses traceroutes to group the addresses according to common successors (§4.1). Second, APPLE evaluates potential alias pairs in each group, filtering unlikely pairs based on their reply TTLs (§4.2).

4.1 Group Addresses by Common Successor

In order to create the potential alias pairs for evaluation, we group addresses according to common successors. To do so, we represent the traceroutes in our collection with a directed interface-graph. First, we truncate each traceroute at the first occurrence of a repeated address separated by at least one other address. These address cycles [30] indicate forwarding loops, violating our assumption that a traceroute continually moves away from the initiating VP. We also strip the last traceroute hop if it responded with an ICMP Echo Reply, since routers always report the probed address in Echo Replies, violating the inbound interface address assumption [17,19,21]. Then we create an edge from each address to the next hop, provided no unresponsive hops separate the addresses in traceroute.

In Fig. 8, we use the traceroutes in Fig. 8a to create the graph in Fig. 8b. We construct an edge from each address to its successors, except from z to s_2, since an unresponsive hop separates them. Then, we create the sets of possible router aliases in Fig. 8c using the incoming edges for each node. We do not perform the transitive closure on these sets for the reasons in §3, so both sets contain b.

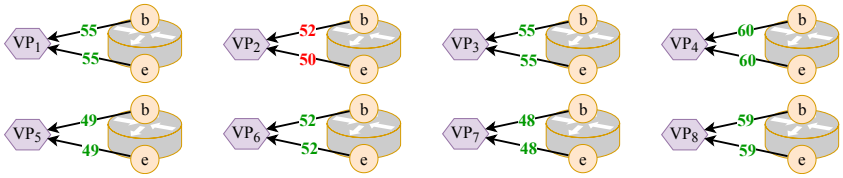


Fig. 9: Reply TTLs to 8 VPs from each address in the possible alias pair (b, e).

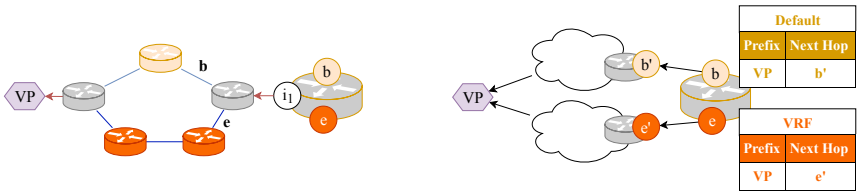
4.2 Probing and Filtering Alias Pairs

After creating the common successor groups, we evaluate each potential alias pair. First, we ping each address with a possible alias pair from every VP, recording the reply TTLs. This requires $O(numAddresses \times numVPs)$ probes, allowing it to scale to large traceroute collections. We also run the probes from each VP concurrently, reducing the run time to the time required for one VP.

We do not require that reply TTLs from each address in a potential alias pair match at every VP. Instead, we require a minimum number of matches (*minimum match threshold*) designed to limit the impact of random reply TTL collisions, which we set using a generalized solution to the birthday problem [23]. With v total VPs, r unique reply TTLs per VP, and a potential alias pairs, $p(a, r, v) \approx 1 - e^{(-a/r^v)}$ computes the probability that any pair of unrelated addresses will have the same combination of values (§A). The ping probes and common successor pairs dictate r and a respectively, so we set the minimum match threshold to the smallest v where $p(a, r, v) < 1/a$. We reject any pair with fewer than v matches.

We also reject alias pairs based on the number of comparisons required to reach the minimum match threshold. For each alias pair, we first sort the pairs of responses according to the minimum RTT to either address. This reflects our assumption that replies to nearby VPs generally encounter fewer network technologies that might confound reply TTL comparison. Next, we compare reply TTLs in sorted order until reaching the minimum match threshold, and prune the alias pair if the ratio of matches to comparisons falls below a predetermined *acceptance threshold*. In Fig. 9, we need eight comparisons to reach the required seven matches, so we discard the pair if $7/8 = 0.875$ falls below the acceptance threshold. Defining the acceptance threshold in terms of the minimum match threshold, and not as a fixed constant, allows it to scale with the required number of matches.

Finally, we create transitive alias pairs based on the transitive closure of the pairs inferred through reply TTLs. We do so by constructing an undirected graph with the common successor alias pairs as edges. Then, we infer alias pairs for every combination of addresses in each graph component, ensuring that our alias pairs cover all inferred aliases of the same router.



(a) Load-balanced paths. (b) L3VPN.

Fig. 10: Load-balanced paths of and L3VPNs can misalign reply TTLs.

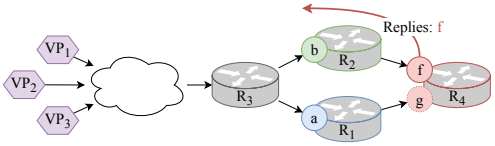


Fig. 11: The off-path address f succeeds both a and b. Both R_1 and R_2 sends replies to the VPs through R_3, so pings to a and b indicate the same reply TTLs.

4.3 Limitations

Addresses on the same router might not always have identical reply TTLs to a VP if network or router configurations cause the replies to traverse a different number of routers, such as load-balanced paths and L3VPN virtual routing and forwarding (VRF) tables (Fig. 10). When load-balanced paths use different numbers of hops, as in Fig. 10a, the reply packets traverse a different number of routers, resulting in different reply TTLs. Similarly, some routers have multiple virtual forwarding tables, known as VRFs, in addition to the default forwarding table. In Fig. 10b, the router includes e in a VRF that uses a different path to reach the VP, so the reply TTLs differ.

Conversely, we might falsely infer alias pairs when a parallel or load-balanced path exists between the VPs and a common successor for a potential alias pair. This occurs in Fig. 11, where R_4 responded with the off-path address f, creating a common successor for a and b. In this case, R_1 and R_2 are on load-balanced paths between R_3 and R_4. Since all responses to the VP first go to R_3, most VPs will receive responses from a and b with the same reply TTL, causing us to incorrectly identify (a, b) as an alias pair.

The transitive nature of alias resolution can cause cascading false inferences, so preventing false alias pairs is paramount. Currently, topologically and geographically distributing the set of VPs provides our only defense against load-balanced and parallel paths. As future work, we hope to investigate how to determine the set of VPs to include and exclude for each pair to maximize the ability of our acceptance threshold to prune incorrect alias pairs. We also plan to experiment with including other alias resolution techniques, such as `Hoiho` [18], to add additional constraints based router identifiers in DNS hostnames.

	ITDK	Pings Sent	Probed	Responses	Resp. %	Pairs
IPv4	04-2019	04-2019	366,469	292,141	79.7%	5,022,839
IPv6	01-2019	05-2019	76,098	59,778	78.6%	563,489

(a) Ping probing statistics.

	Total	ASNs	Countries	Cities
IPv4	99	71	37	83
IPv6	78	61	29	63

(b) VP statistics.

	Total		Probed		Responses	
	IPv4	IPv6	IPv4	IPv6	IPv4	IPv6
Internet2	2176	1095	719	616	646	536
R&E	1651	137	352	137	352	137

(c) Ground truth alias pairs.

Table 1: Statistics from our IPv4 and IPv6 ping probing (a), the ping probing VPs (b), and the alias pairs in the ground truth for Internet2 and R&E visible in our traceroute collections and ping probing (c). In (a) Pairs indicates the potential common successor alias pairs among the responding addresses.

5 Evaluation

We evaluated APPLE on separate IPv4 and IPv6 traceroute collections (Table 1). For IPv4 we used the traceroutes included in CAIDA's Internet Topology Data Kit (ITDK) for April 2019 [1]. While we ran our ping probes in the same month, human error caused us to only ping 83.4% of the addresses seen prior to a Time Exceeded or Destination Unreachable reply. The ITDK also includes a combination of MIDAR and `iffinder` alias resolution, allowing us to compare APPLE against existing techniques. For our IPv6 evaluation we use traceroutes from the January 2019 ITDK [2], the most recent ITDK to include IPv6 alias resolution. We pinged 366,052 and 75,979 IPv4 and IPv6 addresses respectively from 99 VPs in 83 different cities for IPv4 and from 78 VPs in 63 cities for IPv6.

We compared APPLE's alias pair inferences against router configurations from Internet2 and another large R&E network in the United States (Table 1c). Our evaluation focuses on the alias pairs that APPLE inferred and those visible in the traceroute collections. First, we set the minimum match threshold, and explore the trade-offs between the positive predictive value (PPV) and the true positive rate (TPR) related to the acceptance threshold. Then, we evaluated APPLE's TPR by comparing it against the ground truth router configurations (§5.2), and compared the alias pairs generated by APPLE to those found by state-of-the-art alias resolution techniques (§5.3). Finally, we explore how the number of VPs affects APPLE's accuracy (§5.4).

5.1 Evaluating Input Parameters

Before evaluating our results, we set the minimum match and acceptance thresholds from §4.2. To set the minimum match threshold, we first need to determine the possible reply TTLs seen at a given VP. For each VP, we grouped responses by their reply TTL, selected the largest group, and computed the

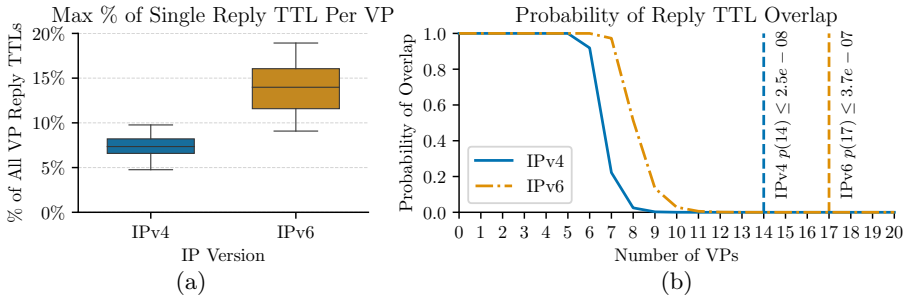


Fig. 12: Using the maximum percentage of all reply TTLs at a VP accounted for by a single value (a), we approximate the probability of reply TTL collision (b).

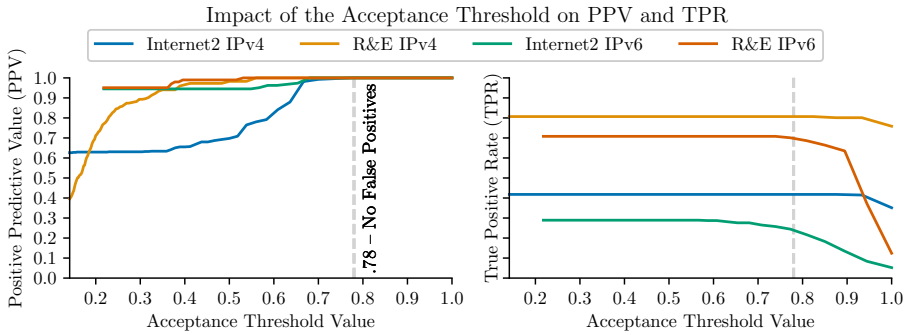


Fig. 13: The acceptance threshold impact on precision and the true positive rate. Lines starts at the first possible acceptance threshold value.

fraction of all responses to the VP included in the group, e.g., for a VP with replies $[55, 59, 55, 53]$, 55 has the most responses, accounting for 50% of the responses seen by that VP.

Fig. 12a shows the distribution of these fractions across the VPs in our experiments. No reply TTL accounted for more than 10%/20% of the responses to an individual VP in IPv4/IPv6, so we set the number of possible replies per VP to $r = \frac{1}{0.1} = 10$ in IPv4 and $r = \frac{1}{0.2} = 5$ in IPv6. Using the number of possible alias pairs (a) from Table 1a, we computed the lower bound on the probability of an anomalous match for 1 – 20 VPs (Fig. 12b). The smallest number of VPs that reduces the probability to less than $\frac{1}{a}$ is 14/17 for IPv4/IPv6, so we set the minimum match threshold to those values in the remaining experiments.

Next, we investigated the trade-off between excluding false alias pairs and discarding valid pairs using the acceptance threshold. When the ratio of matching reply TTLs to comparisons falls below the acceptance threshold, we discard the pair. In this analysis, we exclude transitive pairs, and only evaluate the common successor pairs with at least the minimum number of required matches.

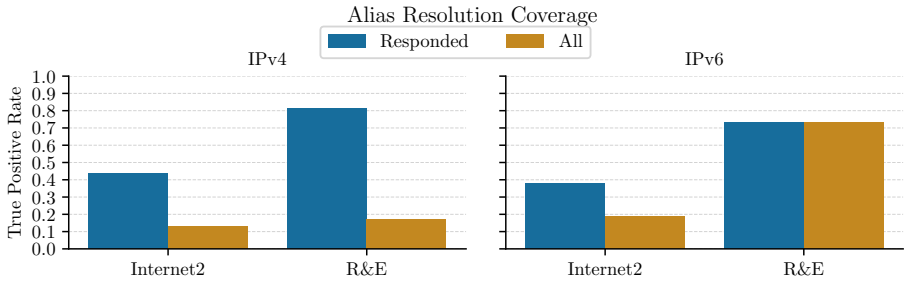


Fig. 14: The TPR for the set of alias pairs where both addresses responded to pings (Responded), and the set of all alias pairs in the traceroute collection (All).

As seen in Fig. 13, increasing the acceptance threshold removes false alias pairs but decreases coverage. Generally, we value increased PPV when inferring alias pairs, rather than increased TPR, since the transitive nature of alias resolution tends to cascade false inferences. We use an acceptance threshold of 0.78 in the remaining evaluation, preventing all false alias pairs in our ground truth.

5.2 Evaluating APPLE's Accuracy:

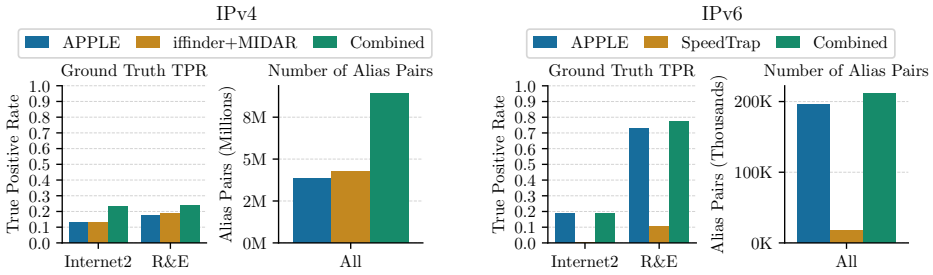
Using the parameters from §5.1, we validate APPLE's alias pair inferences against our two ground truth networks. These parameter settings eliminated all of the incorrect alias pairs, so we only present the true positive rate (TPR), which indicates the fraction of alias pairs in the ground truth that we detected. In this evaluation, we also include the transitive alias pairs in the results.

Fig. 14 shows the TPR for IPv4 and IPv6. The Responded TPR refers to the alias pairs where both addresses responded to the ping probing, indicating the practical ceiling for our performance. APPLE generally performs better for R&E than Internet2, possibly due to the extensive use of L3VPNs in Internet2. For IPv4, the TPR for R&E exceeds 80%, and for Internet2 APPLE found 43.8% of the alias pairs. APPLE achieves worse coverage for IPv6, with TPRs of 73.0% and 37.9% for R&E and Internet2 respectively. We remain unsure what caused the difference in coverage between IPv4 and IPv6, but ruled out insufficient responses to VPs in common.

Fig. 14 also provides the the coverage for all of the possible alias pairs in the traceroute collections (All TPR). Since the number of inferred alias pairs remained the same, while the number of missing pairs increased, the coverage is worse when considering all visible alias pairs. Overall, APPLE found 13.0% – 17.3% of the IPv4 alias pairs and 18.5% – 73.0% of the IPv6 alias pairs.

5.3 Comparing APPLE's Coverage to Current Techniques

Next, we show that APPLE complements current alias resolution techniques by finding additional alias pairs. Specifically, we compare APPLE to the alias res-



(a) IPv4 alias pairs.

(b) IPv6 alias pairs.

Fig. 15: Comparing APPLE to `iffinder`+MIDAR in IPv4 (a) and `Speedtrap` (b) in IPv6, for all addresses seen in the traceroute collections. Each graph shows the TPR for Internet2 and R&E, and the total number of alias pairs.

olution datasets included in the ITDKs, which analyze all intermediate hop addresses in the traceroute collection. In IPv4, the ITDK uses a combination of `iffinder` [14] and MIDAR [15], and in IPv6 it uses SpeedTrap [16]. Both MIDAR and `Speedtrap` rely on global IP-ID counters, and prioritize minimizing false alias pairs.

As seen in Fig. 15, APPLE adds alias pairs for both networks in IPv4 and IPv6, exceeding the ITDK's alias resolution coverage for all but R&E in IPv4, despite only comparing common successor alias pairs. In total, combining APPLE and the ITDK increased the number of inferred alias pairs for the entire traceroute collection, and not just those seen in the ground truth, by 109.6% in IPv4 and by 1071.5% in IPv6, over the ITDK alias resolution alone. The increased coverage is especially important for IPv6 (Fig. 15b), which does not include the IP-ID in the normal IP packet header. `Speedtrap` only works when it can induce fragmentation and expose a global IP-ID counter on a router. This does not work for Juniper routers [16], used for all routers in the Internet2 ground truth and three of the R&E routers. It also did not resolve any aliases for the nine Cisco routers in R&E with multiple addresses in the traceroutes. All of `Speedtrap`'s alias pair inferences in our ground truth include addresses on Brocade routers.

5.4 Reducing the Number of VPs

Our final experiment shows the impact of fewer VPs on APPLE's accuracy. We reran our experiments for IPv4 with the same parameters, but artificially limited the number of VPs. We experimented with random groups of VPs from 15 to 95 in increments of five, using the same IPv4 parameters as before.

Fig. 16 shows the precision and recall of the ten random groups created for each increment, excluding the transitive pairs. APPLE filters out all incorrect R&E alias pairs, but keeps incorrect Internet2 pairs for 50 of the 160 groups. Increasing the acceptance threshold to 0.85 removes all false alias pairs for 32 of

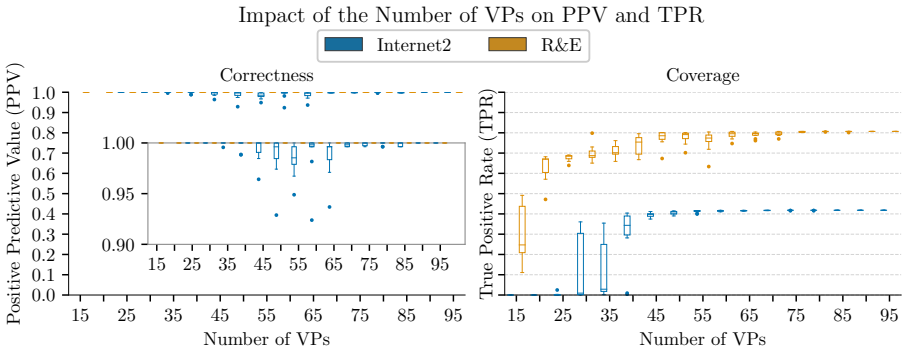


Fig. 16: We re-ran our experiments for IPv4 but limited the available VPs.

those groups with little effect on the TPRs, suggesting that we set the acceptance threshold too low. Interestingly, for the false alias pairs in this experiment, the VPs with shorter RTTs to the addresses in the false alias pairs generally see mismatched reply TTLs more frequently than those further away. As future work, we plan to investigate weighting VPs according to their relative RTT to the addresses in a potential alias pair.

6 Caveats

Although we found no incorrect alias pairs when validating against our ground truth using the full set of VPs, we have anecdotal evidence that APPLE draws incorrect inferences in some cases. The addresses (`89.149.137.33`, `141.136.108.26`) provide an example of a likely incorrect alias pair outside of our ground truth. Their DNS hostnames `xe-11-0-5.cr2-sjc1.ip4.gtt.net` and `xe-4-1-1.cr1-pao1.ip4.gtt.net` indicate that one address is on a router in Palo Alto, while the other is on a router in San Jose. As future work, we hope to improve the precision of our approach by gathering more ground truth and incorporating other constraints, like parsing DNS hostnames [18], in addition to the reply TTL.

7 Conclusion

We presented APPLE, a technique for resolving router aliases seen in traceroute using reply TTLs. We intend for APPLE to complement, rather than replace, existing alias resolution techniques; combining APPLE with existing alias resolution techniques yielded 109.6% and 1071.5% more alias pairs in IPv4 and IPv6 respectively. Despite perfect precision compared to ground truth, we expect some false positives in APPLE's inferred alias pairs. We plan to continue experimenting and improving APPLE to increase its reliability. We also plan to release our source code, allowing other researchers to use and improve on our technique.

Acknowledgments

We thank kc claffy, Matthew Luckie, and Young Hyun for their invaluable feedback. This work was supported by NSF grants OAC-1724853 and OIA-1937165.

A Generalizing the Birthday Problem to Alias Resolution

The birthday problem computes the probability that any combination of n people share the same birthday. A common approximate general solution [23] takes the form,

$$p(n, d) \approx 1 - \exp\left(\frac{-n(n-1)}{2d}\right),$$

where d is the number of days in the year. Of note, the $\frac{n(n-2)}{2}$ term corresponds to the number of possible two-person combinations. Using a to represent the number of combinations, the equation takes the form,

$$p(a, d) \approx 1 - \exp\left(\frac{-a}{d}\right).$$

Applying this equation to our problem, we first replace the number of combinations with the number of potential alias pairs. Second, we must determine the potential reply space for each address. When an address replies to a VP, the VP sees a reply TTL from the space of possible reply TTLs, r. If we assume that a reply TTL to one VP is independent of all the others, then the potential reply space for an address is r^v. Practically, we consider r^v an upper bound on the possible combinations, since we expect that the network topology and control plane create dependent probabilities. Plugging r^v in for d we get the approximate probability that a pair of addresses will have the same combination of replies to all v VPs,

$$p(a, r, v) \approx 1 - \exp\left(\frac{-a}{r^v}\right).$$

To limit collisions, while maximizing the number of true alias pairs, we use the smallest value of v such that $p(a, r, v) < 1/a$.

References

1. Internet topology data kit - April 2019. `http://www.caida.org/data/internet-topology-data-kit/` (Apr 2019)
2. Internet topology data kit - January 2019. `http://www.caida.org/data/internet-topology-data-kit/` (Jan 2019)
3. Ager, B., Chatzis, N., Feldmann, A., Sarrar, N., Uhlig, S., Willinger, W.: Anatomy of a large european IXP. ACM SIGCOMM CCR (2012)
4. Amini, L.D., Shaikh, A., Schulzrinne, H.G.: Issues with inferring Internet topological attributes. In: Internet Performance and Control of Network Systems III (2002)

5. Augustin, B., Krishnamurthy, B., Willinger, W.: IXPs: Mapped? In: IMC (2009)
6. Baker, F.: RFC 1812: Requirements for IP version 4 routers. Tech. rep., Internet Engineering Task Force (1995)
7. Bender, A., Sherwood, R., Spring, N.: Fixing ally's growing pains with velocity modeling. In: IMC (2008)
8. Govindan, R., Tangmunarunkit, H.: Heuristics for Internet map discovery. In: INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE (2000)
9. Grailet, J.F., Donnet, B.: Towards a renewed alias resolution with space search reduction and IP fingerprinting. In: Network Traffic Measurement and Analysis Conference (TMA) (2017)
10. Gunes, M.H., Sarac, K.: Resolving IP aliases in building traceroute-based Internet maps. IEEE/ACM Transactions on Networking (2009)
11. Hyun, Y., Broido, A., claffy, k.: On third-party addresses in traceroute paths. In: PAM (2003)
12. Jacobson, V.: traceroute. `ftp://ftp.ee.lbl.gov/traceroute.tar.gz`
13. Keys, K.: Internet-scale IP alias resolution techniques. ACM SIGCOMM Computer Communication Review (CCR) (2010)
14. Keys, K.: iffinder. `https://www.caida.org/tools/measurement/iffinder/`
15. Keys, K., Hyun, Y., Luckie, M., Claffy, K.: Internet-scale IPv4 alias resolution with MIDAR. IEEE/ACM Transactions on Networking (2013)
16. Luckie, M., Beverly, R., Brinkmeyer, W., et al.: Speedtrap: Internet-scale IPv6 alias resolution. In: IMC (2013)
17. Luckie, M., Dhamdhere, A., Huffaker, B., Clark, D., claffy, k.: bdrmap: Inference of borders between IP networks. In: IMC (2016)
18. Luckie, M., Huffaker, B., et al.: Learning regexes to extract router names from hostnames. In: IMC (2019)
19. Marder, A., Luckie, M., Dhamdhere, A., Huffaker, B., kc claffy, Smith, J.M.: Pushing the Boundaries with bdrmapIT: Mapping Router Ownership at Internet Scale. In: IMC (2018)
20. Marder, A., Luckie, M., Huffaker, B., kc claffy: vrfinder: Finding forwarding addresses in traceroute. In: POMACS (2020)
21. Marder, A., Smith, J.M.: MAP-IT: Multipass accurate passive inferences from traceroute. In: IMC (2016)
22. Padmanabhan, R., Li, Z., Levin, D., Spring, N.: UAv6: Alias resolution in IPv6 using unused addresses. In: PAM (2015)
23. Sayrafiezadeh, M.: The birthday problem revisited. Mathematics Magazine **67**(3), 220–223 (1994)
24. Spring, N., Dontcheva, M., Rodrig, M., Wetherall, D.: How to resolve IP aliases. Tech. Rep. UW-CSE-TR 04–05–04, University of Washington (2004)
25. Spring, N., Mahajan, R., Wetherall, D.: Measuring ISP topologies with Rocketfuel. ACM SIGCOMM CCR (2002)
26. Touch, J.: RFC 6864: Updated specification of the ipv4 id field. Tech. rep., Internet Engineering Task Force (Feb 2013)
27. Vanaubel, Y., Luttringer, J., Mérindol, P., Pansiot, J., Donnet, B.: TNT, watch me explode: A light in the dark for revealing MPLS tunnels. In: Network Traffic Measurement and Analysis Conference (June 2019)
28. Vanaubel, Y., Mérindol, P., Pansiot, J.J., Donnet, B.: Through the wormhole: Tracking invisible MPLS tunnels. In: IMC (2017)
29. Vanaubel, Y., Pansiot, J.J., Mérindol, P., Donnet, B.: Network fingerprinting: TTL-based router signatures. In: IMC (2013)

30. Viger, F., Augustin, B., Cuvellier, X., Magnien, C., Latapy, M., Friedman, T., Teixeira, R.: Detection, understanding, and prevention of traceroute measurement artifacts. Computer networks **52**(5) (2008)