TraceNET: An Internet Topology Data Collector

M. Engin Tozal
Department of Computer Science
The University of Texas at Dallas
Richardson, TX 75080 U.S.A.
engintozal@utdallas.edu

Kamil Sarac
Department of Computer Science
The University of Texas at Dallas
Richardson, TX 75080 U.S.A.
ksarac@utdallas.edu

ABSTRACT

This paper presents a network layer Internet topology collection tool called `tracenet`. Compared to `traceroute`, `tracenet` can collect a more complete topology information on an end-to-end path. That is, while `traceroute` returns a list of IP addresses each representing a router on a path, `tracenet` attempts to return all the IP addresses assigned to the interfaces on each visited subnetwork on the path. Consequently, the collected information (1) includes more IP addresses belonging to the traced path; (2) represents "being on the same LAN" relationship among the collected IP addresses; and (3) annotates the discovered subnets with their observed subnet masks. Our experiments on Internet2, GEANT, and four major ISP networks demonstrate promising results on the utility of `tracenet` for future topology measurement studies.

Categories and Subject Descriptors

C.2 [**COMPUTER-COMMUNICATION NETWORKS**]: Network Architecture and Design

General Terms

Measurement

Keywords

Internet, Network, Subnet, Topology, Traceroute

1. INTRODUCTION

Many successful research projects and efforts have been introduced attempting to derive an accurate and large scale topology map of the Internet [17, 18, 22, 16]. These efforts focus on different but correlated topology maps: *IP level* maps show IP addresses that are in use on the Internet; *router level* maps group the interfaces hosted by the same router into a single unit (via alias resolution); *subnet level* maps enrich the router level maps with subnet level connectivity info; and *AS level* maps demonstrate the adjacency relationship between ASes.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
IMC'10, November 1–3, 2010, Melbourne, Australia.
Copyright 2010 ACM 978-1-4503-0057-5/10/11 ...$10.00.

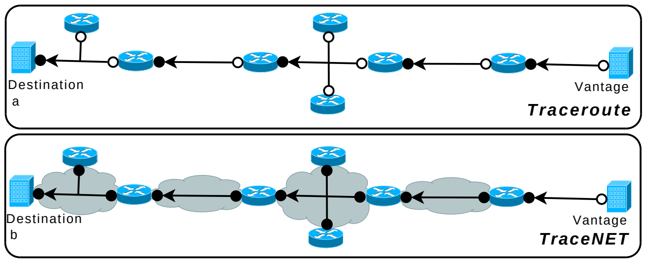

Figure 1: `Traceroute` vs `tracenet` on a path trace. Traceroute collects a single IP address vs tracenet collects a subnet at each hop.

Adverse to the benefits of having a network topology map, the main tools used to collect router or IP address level topology data are a few and operationally limited. `Traceroute` [12] and `ping` are the main data collection tools. `Traceroute` collects a list of IP addresses one for each router on the path between two hosts and `ping` is mainly used to check whether an IP address is in use or not. Almost all topology mapping projects use data collected by `traceroute` from multiple vantage points [19].

In this study, we propose a new end-to-end topology collection tool called `tracenet`. An accurate and complete Internet topology map at the router level requires identifying all routers and subnets among them. `Traceroute` attempts to collect an IP address at each router on a path between two hosts whereas `tracenet` attempts to collect a subnet at each router on the same path. In the worst case, `tracenet` returns the exact path that would be returned by `traceroute`, and, in the best case, it collects the complete topology of each subnet visited on the path. Consequently, a single session of `tracenet` (1) discovers new IP addresses that are missed by `traceroute`, (2) marks multi-access and point-to-point links, (3) reveals subnet relationship among IP addresses on the path, and (4) annotates the subnets with their observed subnet masks. `Traceroute`'s ability to collect a similar data is often limited in practice due to the difficulties of obtaining a reverse path trace and due to the dynamics of the underlying routing behavior between the two systems. As an example, consider the use of `traceroute` and `tracenet` to collect router level topology info between a vantage point

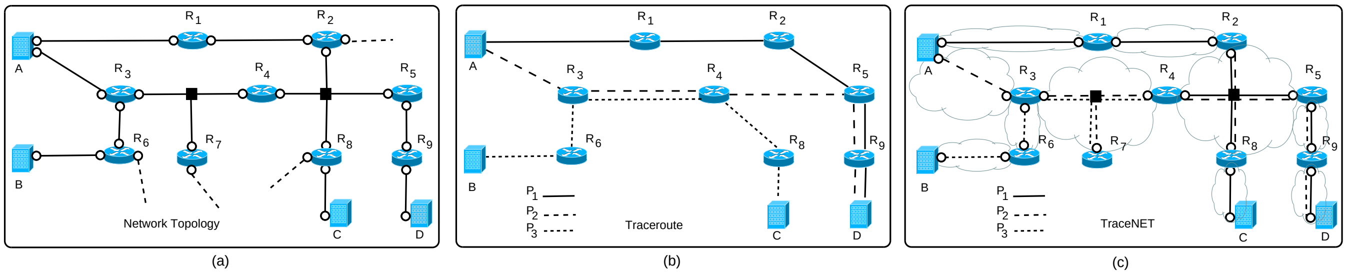

Figure 2: A network topology section among hosts A, B, C, and D with unweighed links. $P_1 = \{A, R_1, R_2, R_5, R_9, D\}$ and $P_2 = \{A, R_3, R_4, R_5, R_9, D\}$ are two paths from A to D. $P_3 = \{B, R_6, R_3, R_4, R_8, C\}$ is a path from B to C. Figures show the original network topology, `traceroute` view of the paths and `tracenet` view of the paths respectively.

and a destination as shown in Figure 1. Figures 1.a and 1.b show the data acquired by `traceroute` and `tracenet` respectively over a network segment. In the figure, small circles attached to the routers show the interfaces on the routers. An interface whose IP address is revealed during a trace is shown in black and otherwise is shown in white. The lines represent point-to-point or multi-access LANs and the arrows on the links show the routing direction of the trace between the vantage point and the destination. Note that in order for `traceroute` to return a similar topology information as `tracenet`, one needs to run `traceroute` in the reverse direction (from the destination to the vantage point) as well. However, a key limitation for `traceroute` is that one may not have access to the destination node to run a `traceroute` query in the reverse direction. Even if one has the required access, the paths between the two nodes may not be symmetric and therefore the reverse trace returned by `traceroute` might not capture the missing information from the first path trace. In fact, such a trace may return another incomplete network topology data for the reverse path. `Tracenet`, on the other hand, attempts to collect the complete topology information on the path in a single trace from the vantage point to the destination.

This valuable information comes with extra probing overhead. However, taking into account that acquiring similar information with `traceroute` requires extensive tracing conducted from many vantage points and a careful post processing [10, 8, 7], `tracenet` can be regarded as a cost effective solution in terms of bandwidth and computation.

Two important objectives in router level Internet topology mapping studies are *completeness* and *accuracy*. Completeness objective requires discovering each and every alive IP address on a given network and accuracy objective requires grouping together IP addresses that are on the same router and establishing both multi-access and point-to-point links between the routers.

A common goal in most topology discovery studies is to increase the coverage of the underlying network as much as possible. This is typically implemented by increasing the number of vantage points and destination addresses in topology discovery. However, the utility of this commonly followed approach was shown to be limited [6]. One of our primary goals in this work is to maximize the utility of our data collection process by focusing on discovering the complete topology of the visited subnets on the path. Naturally our approach allows us to collect more information at each path trace issued at a vantage point.

One critical observation about the existing topology discovery studies is that accuracy is always considered as a posterior process after data collection. Accuracy objective is achieved by addressing several functional steps in converting raw topology data to the corresponding topology maps and it involves IP alias resolution, anonymous router resolution, and subnet inference steps. Most of these tasks are shown to be computationally expensive due to the large volume of data [10, 8, 7]. `Tracenet` combines some of these functional steps, e.g., subnet inference, into topology collection phase significantly reducing the computational complexity in converting the raw data into corresponding topology maps.

Note that both completeness and accuracy conditions affect practical utility of the resulting topology maps. As an example, consider the use of a collected network topology map in designing resilient overlay network systems where the goal is to use the topology map to identify node and link disjoint overlay paths between two neighboring overlay nodes as shown in Figure 2. Figure 2.a shows the physical topology of a network that includes several point-to-point and multi-access links. Assume that our goal is to identify node and link disjoint paths between A and D as well as between B and C in this network. Figure 2.b shows the network topology collected by `traceroute` where P_1 and P_2 are two paths between A and D and P_3 is a single path between B and C. Based on this topology map one would infer that the use of P_1 for A to D path along with the use of P_3 for B and C path would satisfy the node and link disjointness requirement. However, this would be an inaccurate conclusion as routers R_2, R_4, R_5 and R_8 are sharing a multi-access link and P_1 and P_3 are not really link disjoint. On the other hand, a `tracenet` collected topology info as shown in Figure 2.c would include the subnet information hence, would help to avoid the incorrect conclusion above.

Our experimental evaluations of `tracenet` on Internet2 and GEANT topology, with data collected *from a single vantage point* at UT Dallas, resulted in a topology map with 86% similarity to the original Internet2 topology and 90% similarity to the original GEANT topology. (see Section 4.1). In addition, we ran `tracenet` over four major commercial ISPs from three vantage points and cross validated the obtained results (see Section 4.2). Figures show that around 60% of subnets observed by all three vantage points and roughly 80% of subnets observed by a particular vantage point is also observed from at least one other vantage point. Experimental results promote the use of `tracenet` in case studies involving network topology mapping and network analysis/debugging.

The remainder of the paper is organized as follows. Next section introduces the related work. Section 3 details the internals of `tracenet`. Section 4 presents our experimental evaluation results. Finally, Section 5 concludes the paper.

2. RELATED WORK

`Traceroute` [12] has been the main tool to collect router level topology data in the Internet. It sends TTL scoped packets in order to make the routers located at each hop between a vantage point and a destination to return ICMP TTL-Exceeded messages. Each router reports one of its IP addresses by setting it as the source address of the returned ICMP packet. All in all, `traceroute` returns a list of IP addresses each belonging to a router at each hop on the path. Certain load balancing practices may affect the accuracy of `traceroute` returned paths. A version of `traceroute`, called Paris `traceroute` [4], attempts to minimize the negative impact of such load balancing activities.

`Traceroute` can use ICMP, UDP, or TCP based probes. It has been reported that TCP `traceroute` is good at penetrating through firewalls whereas ICMP `traceroute` is more immune to path fluctuations [15, 9]. Discarte project [20] sets record-route option of probe packets to force the compliant routers to stamp the packets with outgoing IP address. As a result, it obtains two IP addresses per hop.

Various techniques have been used to discover more IP addresses in topology mapping studies. Skitter/ARK [17] traces the same set of destinations from multiple vantage points. Rocketfuel [21] aims to reveal the map of a single AS by carefully selecting the sources and the destinations to include the target AS network on the path traces. AROMA [13] advocates the selection of trace destinations from within the targeted network for better exploration of the network.

The well-known `ping` tool uses direct probing to check if a given IP address is in use or not. Census study [11] uses `ping` to determine alive IP addresses on the Internet.

Finally, our previous work [7] attempts to identify subnet relation among the IP addresses collected in a `traceroute` based topology collection study as a *post processing step* on the data set.

`Tracenet`, presented in this paper, is based on several observations related to IP address assignment practices and routing dynamics. Similar to `traceroute`, it obtains a single IP address at each hop, however, before moving to the next hop, it attempts to collect the IP addresses sharing the same LAN with the obtained IP address. *At the end of a `tracenet` run, we get a sequence of subnets between the source and destination hosts.* Unlike the approach presented in [7], `tracenet` discovers subnet topologies as part of the online data collection process.

3. TRACENET

In this section we develop necessary definitions, present several observations regarding operational and topological characteristics of the Internet, and describe internals of `tracenet`.

A router level Internet graph consists of routers and subnets. Each router and subnet hosts at least two interfaces. A router R is identified by the set of interfaces that it hosts: $R = \{l_1, l_2, \ldots, l_m\}$. Similarly, a subnet, S, is identified by a set of interfaces that are directly connected to it: $S = \{l_1, l_2, \ldots, l_n\}$. An interface l has an associated IP address defined as l^{ip} and a hop distance with respect to a certain vantage point v, shown as l^h_v. Whenever the vantage point v is obvious in a context, we drop v and use l^h. Finally, in our discussions we use $R.e$, $R.n$, $R.w$, and $R.s$ to refer to the interface located on (e)ast, (n)orth, (w)est, and (s)outh of a router R.

3.1 Definitions

In this section we introduce several definitions that we use in the rest of the paper.

(i) Direct Probing is the process of sending a probe packet with large enough TTL value destined to some IP address. It is used to test if the IP address is alive or not. In general a probe packet is an ICMP ECHO_REQUEST; a UDP packet destined to a likely unused port number; or the second packet of TCP handshake protocol. These probes force a responsive router to send back an ICMP ECHO_REPLY; an ICMP PORT_UNREACHABLE; or a TCP RESET packet, respectively.

(ii) Indirect Probing is the process of sending a probe packet with a small TTL value destined to some IP address in order to reveal an IP address of another router presumably located at TTL hops away on the path. The probe packet could be of type ICMP, UDP, or TCP. Whenever the TTL reaches zero, a responsive router would notify the originator of the probe message with an ICMP_TTL_EXCEEDED packet. The source address of this packet would be one of the IP addresses of the router based on its response configuration as explained next.

(iii) Router Response Configuration implies that a router is configured to remain reticent or reveal a certain interface's IP address in its response to a direct or indirect probe (query) packet. To the best of our knowledge, routers on the Internet are configured with five types of response policies: *nil interface routers* are configured not to respond to any probe packet; *probed interface routers* respond with the address of the probed interface; *incoming interface routers* respond with the address of the interface through which the probe packet has entered into the router; *shortest-path interface routers* respond with the address of the interface that has the shortest path from the router back to the probe originator; and *default interface routers* respond with a pre-designated default IP address regardless of the interface being probed. Usually, responsive routers are configured to behave as *probed interface routers* for direct probes and any other configuration for indirect probes. Observe that a router cannot be configured as *probed interface router* for indirect queries. Additionally, routers may be configured with multiple response configurations with respect to the protocol type of a probe packet i.e. ICMP, UDP, or TCP.

3.2 Observations

This section presents a set of operational and topological observations on the Internet.

(i) Hierarchical Addressing details the common IP address assignment practices and refers to the Classless Inter-Domain Routing (CIDR) on the Internet (RFC 4632). Given any subnetwork S on the Internet, the IP addresses assigned to the interfaces on S should share a common p bits prefix. Such a subnet S is said to have a $/p$ prefix (subnet mask) and is shown in this paper as S^p.

Any two IP addresses that have 31 or 30 bits common prefix are called *mate-31* or *mate-30* of each other.

(ii) Fixed Ingress Router implies that as long as there is no path fluctuations caused by routing updates, load balancing, or equal cost multi-path routing, there is a single path from a vantage point to any interface on a given subnet S. As an important result; two immediately successive probe packets released from the same vantage point and destined to two different interfaces on a subnet S are expected to reach the subnet through the same router. This fixed router is called *ingress router*. In Figure 3, R_2 is the ingress router of the subnet S with respect to the vantage (see Section 3.7 for more information on path fluctuations).

A fixed ingress router is resistant to intermediate path fluctuations as long as the fluctuated routes converge at or before the ingress router.

(iii) Unit Subnet Diameter implies that two interfaces on the same subnet are located at most one hop distance apart with respect to a vantage point. That is, for interfaces $i, j \in S \Rightarrow |i_v^h - j_v^h| \leq 1$. In Figure 3, interfaces $\{R_2.w, R_3.s, R_4.e, R_6.n\} \in S$ are at most one hop apart from each other with respect to the vantage.

(iv) Mate-31 Adjacency implies that given that i^{ip} and j^{ip} are alive and mate-31 of each other, then $i \in S \Rightarrow j \in S$.

3.3 Network Exploration (Growing)

In this section we detail how `tracenet` builds the subnet accommodating an IP address obtained at a particular hop and introduce the related algorithm. Similar to `traceroute`, `tracenet` gradually extends a trace path by obtaining an IP address l^{ip} (or anonymous) via indirect probing at each hop on the way from a vantage point to a destination. However, after obtaining IP address l^{ip} at a particular hop, `tracenet` collects other IP addresses that are hosted on the same subnet which accommodates interface l before moving to the next hop. We refer to the first step of acquiring an IP address at a certain hop via indirect probing as *trace collection* and refer to the second step of collecting other IP addresses sharing the same subnet as *subnet exploration*.

Given an acquired IP address l^{ip} at trace collection step, the main idea of subnet exploration is forming a subnet of /31 which covers l^{ip} and growing it as long as it does not conflict with one of the heuristics (Section 3.5) that we use for verifying the authenticity of the subnet.

Figure 3 freezes a `tracenet` session $P = \{\ldots, R_2.e, R_4.e\}$ at hop d. Interface $R_2.e$ was acquired at previous hop and interface $R_4.e$ is obtained at current hop via indirect probing in trace collection mode. Those two interfaces are called *ingress interface* and *pivot interface*, respectively. The router hosting the ingress interface (R_2 in Figure 3) is called the *ingress router*. In the figure, the interface $R_2.w$, which is located on the ingress router R_2 and accommodated on the same subnet with the pivot interface $R_4.e$, is called *contra-pivot interface*. Observe that the interface $R_4.e$ belongs to subnet S which hosts a number of other interfaces that we want to discover.

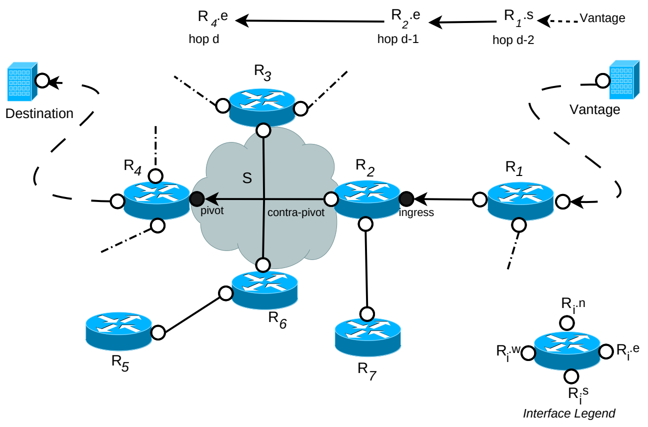

Figure 3: Subnet Exploration. Observed subnet is grown around the pivot interface discovered in trace collection mode by forming /31, /30, ... subnets.

To fully discover a subnet under investigation (S in Figure 3), `tracenet` forms a temporary subnet starting from /31 which covers the pivot interface and grows the subnet with decreasing prefix lengths (increasing subnet sizes). For each formed subnet, it directly probes the possible IP addresses within the range of the subnet to ensure that the probed IP address is assigned to an interface. Then, it applies the heuristics defined in Section 3.5 to decide whether or not the probed candidate IP address is on the subnet being explored. An investigated IP address breaking one of the heuristics implies that it is not on the subnet being built. As a result, subnet growing process immediately stops and the subnet gets shrunk to its last known valid state (i.e., the previous subnet prefix) by removing all interfaces that do not belong to the last valid state. Additionally, subnet growing stops if the number of successfully probed IP addresses for a subnet prefix is equal or less than half of the total number of IP addresses that such a subnet could accommodate.

Algorithm 1 shows the pseudocode for subnet exploration. Let $\hookrightarrow$ operator (read as "results in") be a syntactical element. $\langle i^{ip}, ttl\rangle \hookrightarrow \langle j^{ip}, RESPONSE\_MSG\_TYPE\rangle$ implies that probing of an IP address i^{ip} with TTL value of tll results in a response message of type $RESPONSE\_MSG\_TYPE$ with j^{ip} as the source IP address field of the message. If probing does not yield any response message, then j^{ip} and $RESPONSE\_MSG\_TYPE$ are set to nil. In cases where we are not interested in the value of j^{ip} but only the response message type, we use "..." in place of j^{ip}.

In the algorithm, the outer loop forms temporary growing of subnets S' starting from a /31 prefix. Inner loop traverses each candidate IP address in S' by first testing the aliveness of the IP address at distance d (line 6) and then applies the heuristics explained in Section 3.5 (line 7). As long as the heuristics hold for the current IP address, it is assumed to be part of subnet S being gradually built (line 12). Whenever any heuristic fails, the subnet gets shrunk to previous prefix (line 9). Obtaining an ICMP TTL-Exceeded message as a result of the test at line 6 implies that the subnet is over-grown and needs to be shrunk (lines 14-17). Lines 19-21 ensure that more than half of the potential IP addresses of S are utilized before growing S one more level further.

Algorithm 1 Subnet Exploration

```
Input: $l_{ingress}$, $l_{pivot}$ /*ingress and pivot interfaces respectively*/
Output: $S$ /*observed subnet*/
 1 $S \leftarrow \emptyset$
 2 $S \leftarrow S \cup \{l_{pivot}\}$
 3 for $m \leftarrow 31$ to 0 do
 4     Form temporary $S'$ covering $l_{pivot}$ with prefix $m$
 5     for each $l^{ip} \in S'$ do
 6         if $\langle l^{ip}, j^h \rangle \hookrightarrow \langle \ldots, ECHO\_REPLY \rangle$ then
 7             APPLY HEURISTICS TO $l$
 8             if $l$ breaks any heuristics then
 9                 $S^p \leftarrow m + 1$ /*Shrink $S$ and STOP*/
10                 return
11             else
12                 $S \leftarrow S \cup \{l\}$
13             end if
14         else if $\langle l^{ip}, j^h \rangle \hookrightarrow \langle \ldots, TTL\_EXCD \rangle$ then
15             $S^p \leftarrow m + 1$ /*Shrink $S$ and STOP*/
16             return
17         end if
18     end for
19     if $m \le 29$ and $|S| \le (2^{32-m})/2$ then
20         return
21     end if
22 end for
```

3.4 Subnet Positioning

As discussed in Section 3.1, routers usually report the queried IP addresses in their reply messages to direct probes. However, for indirect probes, a router may report the IP address of the incoming interface, a default interface or the interface on the shortest path from itself back to the vantage point. *Tracenet builds the subnet which accommodates the interface obtained with indirect probing at trace collection mode.* Therefore, it needs to correctly designate a pivot and an ingress interface, determine the distance to the pivot interface from the vantage point, and decide whether or not the subnet is on the trace path before growing the subnet (Section 3.3). Subnet positioning, discussed in this section, is the process of designating pivot and ingress interfaces, determining the distance of the pivot interface from the vantage point and deciding if the subnet is on the trace path.

In most cases, the perceived *direct* distance of a vantage point to an interface l that is obtained *indirectly* at hop d is also d. In some other cases, however, it might differ by one or a few hops. Consequently, direct distance to other interfaces on the subnet accommodating l changes. To determine the hop distance from the vantage point to an interface l obtained with indirect probing at hop d in trace collection mode, `tracenet` sends probe packets with increasing (forward) and decreasing (backward) TTL values starting from d until it locates the exact location of l.

Figure 4 shows portion of a `tracenet` session frozen at hop d passing through the routers R_1 and R_3. Different from Figure 3, it does not show the reported interfaces (filled black) by the routers in order to discuss various scenarios. Let u and v be the two successive interfaces obtained from R_1 and R_3 in trace collection mode at hops $d-1$ and d, respectively.

A subnet S to be explored in subnet exploration mode is said to be "on-the-trace-path" if the indirect probe packet in trace collection mode passes through it. For instance, in Figure 4, if router R_3 returns $R_3.e$ and the path to $R_3.e$ is P

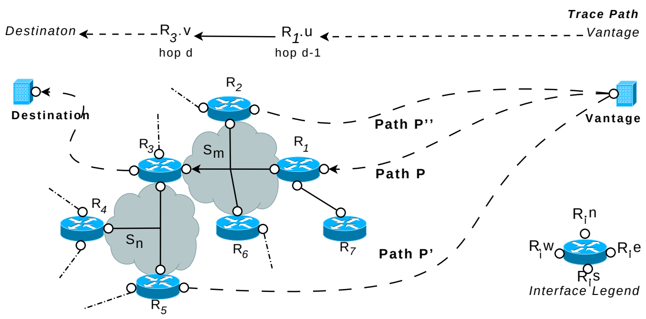

Figure 4: Subnet Positioning. Assuming path to S_m and S_n is P or P'', if $R_3.v = R_3.e$ then $l_{pivot} = R_3.e$, on the other hand, if $R_3.v = R_3.s$ then $l_{pivot} = mate31(R_3.e)$ or $l_{pivot} = mate30(R_3.e)$

or P'' then the subnet to be explored is S_m and it is on-the-trace-path. On the other hand, if it returns $R_3.s$ then the subnet to be explored is S_n and it is "off-the-trace-path". In Figure 4, `tracenet` assumes that a subnet to be explored at hop d is off-the-trace-path if the perceived direct distance to v is different from d. Instead, if the perceived direct distance to v is d and probing v^{ip} with TTL value $d-1$ results in an ICMP_TTL_EXCEEDED message with source IP address u^{ip}, i.e., $\langle v^{ip}, d-1 \rangle \hookrightarrow \langle u^{ip}, TTL\_EXCD \rangle$ then the subnet to be explored is on-the-trace-path. On the other hand, if perceived direct distance to v is d and probing v^{ip} with TTL value $d-1$ results in an ICMP_TTL_EXCEEDED message with source IP address other than u^{ip} i.e., $\langle v^{ip}, d-1 \rangle \hookrightarrow \langle l^{ip}, TTL\_EXCD \rangle$ where $l \neq u$, `tracenet` probabilistically concludes that the subnet to be explored is off-the-trace-path.

Under a stable path to a subnet S, the pivot interface is always selected among the farthest interfaces of a subnet to be explored with respect to the vantage point, i.e., $\exists l_{contra-pivot} \in S : \forall (l \neq l_{contra-pivot}), l^h = l^h_{contra-pivot} + 1$, $l \in S$, and $l_{pivot} \neq \{l_{contra-pivot}\}$ in S. To illustrate in Figure 4, in case R_3 returns $R_3.e$ and the path to subnet S_m is P or P'' then $R_3.e$ serves as pivot interface. Conversely, if R_3 returns $R_3.s$ and the path to subnet S_n is P or P'' then some other alive interface $l \in S_n$ (usually /31 mate or /30 mate of $R_3.s$) serves as pivot interface. Let $mate31(l^{ip})$ and $mate30(l^{ip})$ be functions returning /31 and /30 mate of a given IP address l^{ip}. `Tracenet` exploits mate-31 adjacency defined in Section 3.2 to determine the pivot interface. In Figure 4, if sending a probe packet to /31 mate of the IP address $v^{ip} \in R_3$ with TTL value of $v^h = d$ results in an ICMP response of $TTL\_EXCEEDED$, i.e. $\langle mate31(v^{ip}), v^h \rangle \hookrightarrow \langle \ldots, TTL\_EXCD \rangle$, then pivot interface is mate /31 or mate /30 of v^{ip}, otherwise pivot interface is v itself. Similar argument applies to /30 mate in case probing /31 does not yield any response.

After determining the pivot interface l_{pivot} at a distance d for the subnet to be explored, the ingress interface $l_{ingress}$ is obtained by sending a probe packet to the pivot interface with a TTL value of $d-1$ from the vantage point. Formally, $\langle l^{ip}_{pivot}, l^h_{pivot} - 1 \rangle \hookrightarrow \langle l^{ip}_{ingress}, TTL\_EXCD \rangle$.

Algorithm 2 details the process of subnet positioning. In the algorithm $dst(l^{ip})$ is a function that determines the direct distance to the interface l from the vantage point. Lines

2-10 determine if the subnet to be explored is on-the-trace-path. Lines 11-21 designate the pivot interface and line 22 designates the ingress interface of the subnet to be explored.

Algorithm 2 Subnet Positioning

```
Input: $u$, $v$, $d$ /*last two interfaces obtained in trace collection mode
    at hop $d-1$ and $d$*/
Output: $l_{pivot}, l_{ingress}$ /*pivot and ingress interfaces*/
1  $v^h \leftarrow dst(v^{ip})$
2  if $v^h \neq d$ then
3    Subnet to be explored is off-the-trace-path
4  else
5    if $\langle v^{ip}, v^h - 1\rangle \hookrightarrow \langle u^{ip}, TTL\_EXCD\rangle$ then
6      Subnet to be explored is on-the-trace-path
7    else if $\langle v^{ip}, v^h - 1\rangle \hookrightarrow \langle i^{ip}, TTL\_EXCD\rangle$ where $i^{ip} \neq u^{ip}$
     then
8      Subnet to be explored is off-the-trace-path
9    end if
10 end if
11 if $\langle mate31(v^{ip}), v^h\rangle \hookrightarrow \langle \ldots, TTL\_EXCD\rangle$ then
12   if $mate31(v^{ip})$ is in use then
13     $l^{ip}_{pivot} \leftarrow mate31(v^{ip})$
14   else if $mate30(v^{ip})$ is in use then
15     $l^{ip}_{pivot} \leftarrow mate30(v^{ip})$
16   end if
17   $l^h_{pivot} \leftarrow v^h + 1$
18 else
19   $l^{ip}_{pivot} \leftarrow v^{ip}$
20   $l^h_{pivot} \leftarrow v^h$
21 end if
22 $l_{ingress} \leftarrow i$ where $\langle l^{ip}_{pivot}, l^h_{pivot} - 1\rangle \hookrightarrow \langle i^{ip}, TTL\_EXCD\rangle$
```

3.5 Heuristics

Heuristics given in this section are based on the common IP address assignment practices and routing behavior on the Internet. They are devised to capture a subnet regardless of its location at the core or edge of the Internet and regardless of its being a multi access LAN or a point to point link.

Let u and v be two consecutive interfaces obtained in trace collection mode and i and j be the ingress and pivot interfaces respectively determined by the subnet positioning algorithm. Note that $u = i$ and $v = j$ may or may not hold. Let l^{ip} be the IP address suspected to be sharing the same subnet with j^{ip} and S be the subnet we are trying to infer such that $j, v \in S$.

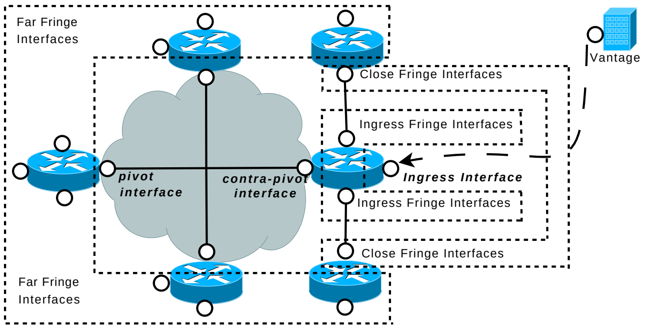

Figure 5: Fringe Interface Categorization

Figure 5 categorizes the false positive interfaces that are prone to be misinferred as members of the investigated subnet because they are located at the same distance from the vantage point as the interfaces of the subnet to be explored. None of the fringe interfaces are accommodated by the subnet being investigated and they must be caught and excluded from the observed subnet which is gradually being built. Additionally, trapping a fringe interface is a signal that the subnet exploration phase is growing the observed subnet beyond its bona-fide boundaries. As a result, `tracenet` stops growing the subnet and to shrinks it to its last known valid state.

Ingress fringe interfaces are the ones that are hosted by the ingress router. Far fringe interfaces are hosted by routers that are one hop distant from the ingress router but are not accommodated by any subnet that the ingress router has direct access. Similarly, close fringe interfaces are hosted by routers that are one hop distant from the ingress router but are accommodated by a subnet that the ingress router has direct access. Heuristics H3, H7, and H8 below catch the ingress, far fringe, and close fringe interfaces, respectively.

For each rule below, we give a description, a formal statement of a related observation on the Internet, and a code snippet demonstrating how `tracenet` utilizes the observation. Each code snippet is a test applied to address l^{ip}; keyword *apply-next-heuristic* implies l has passed the test and continue to apply the rest of the tests; keyword *stop-and-shrink* implies l has failed the test, hence, growing S should be stopped and S must get shrunk to its last known valid state; and keyword *continue-with-next-address* implies continue to grow S by getting the next candidate IP address to examine.

Although, the code snippets given for each heuristics lack details and have repetitions, our `tracenet` implementation is optimized to collect the subnets with the least number of probes and some of the rules are merged together.

Remember that in the following rules j^{ip} is the pivot IP address and l^{ip} is the candidate IP address that needs to be tested.

H1 prefix reduction: In the context of subnet exploration, whenever an IP address breaks the rules given below and causes the *stop-and-shrink* instruction to be executed, then subnet S is shrunk to last known intact status. That is S^p gets reduced to $S^p + 1$ and all false positive interfaces (i.e., interfaces conforming S^p but not $S^p + 1$), if ever passed the heuristics, gets omitted from subnet S. Obviously, as the subnet grows larger and larger the possibility of encountering a false positive causing *stop-and-shrink* instruction to be executed increases. This rule serves as a stopping condition of subnet exploration algorithm and increases our trust on the accuracy of large subnets.

H2 upper-bound subnet contiguity: This rule ensures that the examined IP address l^{ip} is in use and is not located farther from the investigated subnet with respect to the vantage point. More formally, $j, l \in S \Rightarrow \langle l^{ip}, j^h\rangle \hookrightarrow \langle \ldots, ECHO\_RPLY\rangle$.

Code snippet:

```
if $\langle l^{ip}, j^h\rangle \hookrightarrow \langle \ldots, ECHO\_RPLY\rangle$ then
  apply-next-heuristic
else if $\langle l^{ip}, j^h\rangle \hookrightarrow \langle \ldots, TTL\_EXCD\rangle$ then
  stop-and-shrink /*$l$ located farther from $S$*/
else
  continue-with-next-address
end if
```

H3 single contra-pivot interface: This rule ensures that the examined IP address l^{ip} is not one of the ingress

fringe interfaces. More formally, under a stable path to subnet S, $\exists l_{contra-pivot} \in S$ where $l^h_{contra-pivot} = j^h - 1$ and $\forall(l \neq l_{contra-pivot}), l^h = j^h$ where $l \in S$.
Code snippet:

```
if $\langle l^{ip}, j^h - 1\rangle \hookrightarrow \langle \ldots, ECHO\_RPLY\rangle$ and $l_{contra-pivot} = NIL$ then
    $i_{contra-pivot} \leftarrow l$
    apply-next-heuristic
else if $\langle l^{ip}, j^h - 1\rangle \hookrightarrow \langle \ldots, ECHO\_RPLY\rangle$ and $l_{contra-pivot} \neq NIL$ then
    stop-and-shrink /*Second contra-pivot detected*/
else
    continue-with-next-address
end if
```

Note that, regarding Figure 3, R_2 is the router hosting the contra-pivot interface $R_2.w$ and any other interface of R_2 seems to potentially pass the test given above. However, subnet exploration process demonstrated in Algorithm 1 starts to grow the subnet from prefix /31 and $R_2.w$ will be examined before any other interface hosted by R_2.

H4 lower-bound subnet contiguity: This rule increases our confidence level on contra-pivot interface before adding l to the subnet S if the examined IP address l^{ip} found to be a contra-pivot interface by H3. More formally, $\{l$ is found to be contra-pivot interface$\} \Rightarrow \langle l^{ip}, j^h - 2\rangle \hookrightarrow \neg\langle \ldots, ECHO_RPLY\rangle$.
Code snippet:

```
if $l_{contra-pivot} = l$ and $\langle l^{ip}, j^h - 2\rangle \hookrightarrow \langle \ldots, ECHO\_RPLY\rangle$ then
    stop-and-shrink
else
    $S \leftarrow S \cup l$
    continue-with-next-address
end if
```

H5 mate-31 subnet contiguity: This rule is a shortcut to add the examined IP address l^{ip} to subnet S if it is /31 mate of the pivot. The same rule is valid $l^{ip} = mate30(j^{ip})$ only if $mate31(j^{ip})$ is found not to be in use. More formally, $l^{ip} = mate31(j^{ip}) \Rightarrow l \in S$.
Code snippet:

```
if $l^{ip} = mate31(j^{ip})$ then
    $S \leftarrow S \cup l$
    continue-with-next-address
else
    apply-next-heuristic
end if
```

H6 fixed entry points: This rule is used to test whether the examined IP address l^{ip} resides on a different subnet located at the same distance with S or not. More formally, $\{l$ is found not to be the contra-pivot interface$\} \Rightarrow (\langle l^{ip}, j^h - 1\rangle \hookrightarrow \langle i, TTL_EXCD\rangle$ or $\langle l^{ip}, j^h - 1\rangle \hookrightarrow \langle u, TTL\_EXCD\rangle$.
Code snippet:

```
if $\langle l^{ip}, j^h - 1\rangle \hookrightarrow \langle i, TTL\_EXCD\rangle$ then
    apply-next-heuristic
else if $S$ is on-the-trace-path and $\langle l^{ip}, j^h - 1\rangle \hookrightarrow \langle u, TTL\_EXCD\rangle$ then
    apply-next-heuristic
else if $\langle l^{ip}, j^h - 1\rangle \hookrightarrow \langle k, TTL\_EXCD\rangle$ where $k \neq i$ and $k \neq u$ then
    stop-and-shrink
else if $\langle l^{ip}, j^h - 1\rangle \hookrightarrow \langle i, ECHO\_RPLY\rangle$ then
    stop-and-shrink
end if
```

Remember that u is obtained in trace collection mode and i is obtained by subnet positioning algorithm. To minimize effects of path fluctuations both i and u are considered to be valid entry points to subnet S. Note that the rule is valid in case i and/or u are anonymous.

H7 upper-bound router contiguity: This rule is used to detect whether the examined IP address l^{ip} is one of the far fringe interfaces or not. More formally, $l \in S \Rightarrow \langle mate31(l^{ip}), j^h\rangle \hookrightarrow \neg\langle \ldots, TTL\_EXCD\rangle$.
Code snippet:

```
if $\langle mate31(l^{ip}), j^h\rangle \hookrightarrow \langle \ldots, TTL\_EXCD\rangle$ then
    stop-and-shrink /*Far Fringe interface detected*/
else
    apply-next-heuristic
end if
```

A router hosting an interface that resides on S at distance j^h also hosts some other interfaces that are not on S. Those interfaces may pass all the rules given above and they must be eliminated.

To illustrate, in Figure 3, $R_6.w$ or $R_4.s$ behaves as if they are on the subnet. However, a probe directed to their /31 mates with $TTL = d$ result in a TTL-Exceeded message with a very high probability. Conversely, /31 mate of an interface that is really on subnet S, for example /31 mate of $R_6.n$ must result in an ICMP Echo-Reply or anonymous.

Note that if l is a far fringe interface, then the probe packet destined to its /31 expires one hop earlier. In case probing /31 mate of l^{ip} does not yield any response or yields an ICMP Host-Unreachable the same heuristic is performed with /30 mate.

H8 lower-bound router contiguity: This rule is used to detect whether the examined IP address l^{ip} is one of the close fridge interfaces or not. More formally, $l \in S \Rightarrow \langle mate31(l^{ip}), j^h - 1\rangle \hookrightarrow \neg\langle \ldots, ECHO\_RPLY\rangle$ such that
$mate31(l^{ip}) \neq l^{ip}_{contra-pivot}$.
Code snippet:

```
if $mate31(l^{ip}) \neq l^{ip}_{contra-pivot}$ then
    if $\langle mate31(l^{ip}), j^h - 1\rangle \hookrightarrow \langle \ldots, ECHO\_RPLY\rangle$ then
        stop-and-shrink /*Close Fringe interface detected*/
    else
        apply-next-heuristic
    end if
end if
```

To illustrate, in Figure 3, interface $R_7.n$ behaves as if it is on the subnet being investigated. However, a probe directed to its /31 mate with $TTL = d$ results in an ICMP Echo-Reply if the /31 mate is on the ingress router of the subnet being investigated. If probing /31 mate of l^{ip} does not yield any response or yields an ICMP Host-Unreachable, the same heuristic is performed with /30 mate. In case the subnet between R_2 and R_7 accommodates more than two interfaces, all false positive interfaces will be added to the investigated subnet until `tracenet` encounters an interface whose /31 or /30 mate is located on the ingress router. Remember that whenever `tracenet` traps a false positive interface, it shrinks S by removing all false positive interfaces from S.

H9 boundary address reduction: A final subnet that is populated with interfaces passing all the tests above cannot contain any of the boundary addresses, i.e., network address and broadcast address, unless $S^p = 31$. As a post processing task after collecting the subnet, as long as the subnet contains a boundary address, `tracenet` divides the subnet S into S_1 and S_2 where $S^p_1 = S^p_2 = S^p - 1$ drops S_i if $j \notin S_i$.

Remember that these nine heuristics are not designed for any specific type of network but are generic enough to work over various subnet configurations appearing on the Internet.

3.6 Probing Overhead

The exact number of probes for a `tracenet` session depends on the length of the trace, utilization/configuration of the IP addresses of the subnet being explored as well as its fringe subnets. In this section we give a model for lower and upper bounds on the probing complexity as a function of the size $|S|$ for a subnet S.

For each subnet S_i, *initial cost* is retrieving an IP address at the trace collection mode and determining pivot and ingress interfaces at the subnet positioning phase. The former requires a single probe and the latter requires one or three probes depending on the position of the subnet.

For each interface l on S_i; *intermediate cost* is incurred by heuristics and it depends on the type of l: no probing overhead for pivot interface; if l is /31 (or /30) mate of the pivot interface, then only H2 and H5 are applied; if l is contra pivot interface then, H2, H3, and H4 are applied; and otherwise, H2 to H4 and H6 to H8 are applied. Among those heuristics, H5 does not require probing, H7 and H8 may take one or two probes depending on the existence of $mate31(l^{ip})$, and the rest takes a single probe each where both H3 and H6 requires the same single probe.

For each subnet S_i; *final cost* is incurred by encountering a stopping condition and H9. Heuristic H9 has no probing overhead. On the other hand, stopping condition can be encountered because of either S_i being under-utilized (lines 19-21 of Algorithm 1) or trapping into a stop-and-shrink statement. The former has no probing cost, the latter depends on the number of heuristics applied until executing the stop-and-shrink statement.

The lower bound on probing overhead is when the discovered subnet S is a point-to-point link in which case only heuristics H2 and H5 are applied. Assuming, as the most typical case, the stopping condition is an immediate stop-and-shrink statement execution and the subnet is on-the-trace-path, the total cost is only four probes for discovering the point-to-point subnet.

The upper bound of probing overhead is when the discovered subnet S is off-the-trace-path and is a multi access LAN accommodating $|S| > 2$ interfaces. Each subnet has a single contra-pivot interface which requires three probes. Let $l^{ip} \neq mate31(i_{pivot})$, $\forall(l \in S)$, hence, $\forall(l \in S)$ requires all heuristics H2-H8 where heuristics H7 and H8 require two probes each. Also let the execution of the stopping condition is delayed until the last heuristic, i.e., H8. Under this scenario, exploring S has the *initial cost* of three probes, *final cost* of eight probes, and *intermediate cost* of three probes for contra-pivot interface and seven probes for other interfaces except pivot interface. As a result the upper bound probing cost would be $7|S| + 7$. Note that upper bound probing complexity is a worst case scenario with probability $p < 1/\binom{2^{\lceil \lg(2|S|-1)\rceil}}{|S|}$ of encountering and is only valid for multi access LANs. This scenario takes place when an administrator utilizes half of an IP address range where only the odd or even numbered IP addresses are assigned to interfaces.

3.7 Path Fluctuations

Path fluctuations [5] such as load balancing and equal cost multi-path routing on the Internet depend on local network traffic and router configuration that the packets pass through at a certain time. Although a single load balancing enabled router may potentially damage the authenticity of all trace paths passing on it, the effect is not the same for `tracenet` because *`tracenet` is based on the stable ingress router concept rather than stable path trace concept.* That is, as long as two packets destined to two different IP addresses on the same subnet enter into the subnet through the same ingress router we are not interested if they took the same path or not.

Heuristics defined above are meaningful at local sites identified by two neighboring routers and immune to path fluctuations taking place before and after the sites as long as the hop distance is preserved. On the other hand, path fluctuations with varying hop distances while exploring a subnet potentially causes the subnet being observed smaller than its actual size.

In case a path fluctuation occurs at a local site, only H6 among the rules given above gets affected. H6 expects that the packets destined to an interface on the subnet should enter into the subnet through certain router(s). The result of a path fluctuation in the context of H6 would be stopping to grow the subnet prematurely before it reaches its actual size. In order to minimize this negative effect we resort to the following methods: (1) our implementation of `tracenet` is completely based on ICMP probes which are shown to be the least affected by load balancing [15] and (2) `tracenet` always attempts to obtain at most two ingress routers to the subnet being investigated (one is in trace collection mode and the other is in subnet positioning phase) and applies the test H6 against both routers. As a result, any packet entering to the subnet through either of the routers does not cause early halt of subnet exploration phase.

Moreover, as long as the intermediate path fluctuations, if ever occur, converge to an ingress router determined by `tracenet`, H6 does not get affected from it and correctly grows the subnet. To increase the confidence level regarding path fluctuations on a collected subnet, the same subnet could be re-collected at a different time or from a different vantage point.

3.8 Limitations

`Tracenet` gradually enlarges a subnet around the IP address l_{pivot} after obtaining it via subnet positioning. Forming a subnet from a single IP address in a bottom-up fashion allows us to efficiently build large sized subnets compared to the top-down approach which assumes a very large subnet and shrinks it by testing if each IP address is within the boundaries of the assumed subnet. Since `tracenet` stops growing a subnet one level upwards in case it cannot fill at least half of the current level, sparsely utilized subnets might potentially get underestimated. A similar behavior would be observed when a consecutive portion of the IP addresses space of a subnet remain silent to `tracenet` probes. In our implementation we re-probe an IP address if we do not get a response for the first probe. Moreover, if the path length to the ingress router changes in the middle of subnet exploration mode then the resulting subnet would also be an underestimated subnet. Nevertheless, we believe that path length changes in the middle of subnet exploration mode is not very likely.

Trace collection mode of `tracenet` is similar to traditional `traceroute`. We plan to include the approach used in `Paris traceroute` in order to keep the end-to-end paths more stable. Finally, `tracenet` is designed mainly for IPv4 and its extension to IPv6 remains as a future work.

4. EVALUATIONS

In this section we measure accuracy and consistency of `tracenet` using two different types of experiments. In the first experiment set, we ran `tracenet` over Internet2 and GEANT and compare the collected subnets with the subnet list that we derived using the information provided by these research networks. Our comparisons are based on the rate of exact prefix length matches, similarity of subnet prefix lengths, and similarity of subnet sizes. For the second set of experiments, we ran `tracenet` from three different vantage points with a common target IP address set belonging to four commercial ISPs (Sprintlink, AboveNet, Level3, NTT America) and cross validate the collected subnets of each vantage point against the others. Finally, we ran `tracenet` from a single vantage point with ICMP, UDP, and TCP probe packets to see its behavior with different protocols. Note that the main focus in our experiments is limited to demonstrating the accuracy and consistency of `tracenet` rather than deriving complete [13, 21] or representative [14, 3] sample topologies of commercial ISPs.

Before continuing with the experiments and their evaluations we would like to point out a few issues regarding the authenticity of the subnets collected by `tracenet`. First of all, `tracenet` is based on active probing and completely unresponsive subnets, i.e., subnets located behind a firewall which blocks probe packets or their responses, cannot be captured. In this study we focus on sketching subnets at the core of the Internet which are mostly open for discovery. Secondly, a subnet sketched by `tracenet` is only an observable subnet. That is, if there is a partially unresponsive subnet, i.e., a subnet which consists of a mixture of responsive and unresponsive routers, `tracenet` can capture only the responsive portion of it. Related to observable subnet concept, in case there is a subnet utilizing only a portion of its whole IP address range, `tracenet` could only reveal what it observes. To illustrate, if a network administrator utilizes only a /30 portion of a subnet which is assigned a /29 subnet mask, `tracenet` collects it as a /30 subnet. Finally, Internet accommodates virtual subnets whose hosts connect to each other through tunnels. Even though such a tunnel is realized over a sequence of routers, `tracenet` captures it as a single subnet.

4.1 TraceNET Accuracy over Internet2 and GEANT

In this set of experiments, we run `tracenet` over Internet2 and GEANT and collect their subnet topologies. We measure the similarity of the collected topology against the derived subnet topologies of Internet2 and GEANT using four different metrics. In the first approach we directly compare the rate of exact matches i.e., rate of subnets that are collected exactly as announced in the original topologies. Results show that exact match rate of `tracenet` for Internet2 is 73.7% and for GEANT is 53.5%. In the second approach, we exclude those unresponsive subnets, i.e., the ones that do not reply back to our probes, and evaluate the exact match rate again. With this approach, our exact match rate raises to 94.9% and 97.3% for Internet2 and GEANT, respectively. In the third approach, we use each of the subnet prefix lengths in the original topology as a point in an n-dimensional euclidean space and measure the percentage of similarity of the collected topology to the original topology. This approach allows us to asses the amount of deviation from the original topology rather than looking upon it as a binary match/not-match value. Results show that the subnet prefix length similarity including unresponsive subnets for Internet2 and GEANT are 0.83 and 0.900 respectively. Our fourth approach is also based on the similarity in euclidean space, however, we use subnet size instead of prefix length as the coordinate space. We obtain similarity rates of 0.86 and 0.907 for Internet2 and GEANT respectively again including unresponsive subnets.

We first derive original Internet2[2] and GEANT[1] subnet topologies as our ground truth. `Tracenet` collects a subnet on-the-fly while tracing towards a destination if it appears on the end-to-end path. However, we cannot control the path taken by a packet and make it to pass through a certain subnet or router. Thus, we build destination IP address sets for Internet2 and GEANT by selecting a random IP address from each of their original subnets. Below we present the details of the experiments in the context of the aforementioned approaches.

4.1.1 *Exact Match Rate Evaluation*

Tables 1 and 2 show the original and collected subnet distributions of Internet2 and GEANT respectively. In both tables the first row (*orgl*) show the original subnet prefix distribution. The second row (*exmt*) shows the distribution of the subnets collected exactly as in the original topology. The third (*miss*) and fifth (*undes*) rows respectively show missing subnets that could not be discovered at all and underestimated subnets that inferred to be smaller than the originals. We further divide missing and underestimated distributions into two sets shown in fourth ($miss\backslash unrs$) and sixth ($undes\backslash unrs$) rows in order to observe whether the missing or underestimated subnets result from unresponsive hosts or incompetence of our heuristics. We believe that this distinction is important to show that a misinferred subnet which is behind a firewall that is filtering out our probe packets, e.g., *totally unresponsive subnet*, or utilizing only a sub-portion of its capacity, e.g., *partially unresponsive subnet* cannot be attributed as drawback of `tracenet`. The seventh (*ovres*), eighth (*splt*), and ninth (*merg*) rows show the overestimated subnets that are inferred to be larger than the original subnet, split, and merged subnets, respectively.

Given two subnets S_a and S_b that have their IP addresses from two consecutive address ranges, if the two subnets are incorrectly inferred to be a single subnet S_{ab} with the corresponding IP addresses, then they are considered to be merged. On the other hand, if one of them is collected accurately and the other one is collected as S_{ab} then the first one is considered to be an exact match and the second one is considered to be an overestimation.

The exact match rate of `tracenet` over Internet2 is 73.7% including unresponsive subnets and is 94.9% excluding totally unresponsive subnets. Internet2 topology is a special topology that consists of many small sized (mostly point-to-point) subnets at the backbone and a few larger subnets

Table 1: Internet2, Original and Collected Subnet Distribution

	/24	/25	/26	/27	/28	/29	/30	/31	total
orgl	6	1	0	2	26	20	101	23	179
exmt	0	0	0	0	2	16	92	22	132
miss	1	0	0	0	2	0	0	0	3
miss\unrs	4	1	0	2	1	4	8	1	21
undes	1	0	0	0	2	0	0	0	3
undes\unrs	0	0	0	0	19	0	0	0	19
ovres	0	0	0	0	0	0	1	0	1
splt	0	0	0	0	0	0	0	0	0
merg	0	0	0	0	0	0	0	0	0

behind some firewall that filters out ICMP messages or configured not to respond to any direct probe. After collecting the subnets we further probed every IP address within the address range of the missing and underestimated subnets to identify the unresponsive subnets. The results show that 19 out of 22 of the underestimated and 21 out of 24 of the missing subnets are caused by partially and totally unresponsive subnets, respectively. When we analysed the two underestimated /28 subnetworks shown in Table 1, we found that only 2 IP addresses were observed to be utilized in the first network and only 5 are observed to be utilized in the second network. Besides, the utilized IP addresses of the IP address range of the second network have large gaps, so line 19 of algorithm 1 stops growing the subnet prematurely.

Table 2: GEANT, Original and Collected Subnet Distribution

	/24	/25	/26	/27	/28	/29	/30	/31	total
orgl	0	0	0	0	24	109	138	0	271
exmt	0	0	0	0	0	41	104	0	145
miss	0	0	0	0	0	1	0	0	1
miss\unrs	0	0	0	0	10	53	34	0	97
undes	0	0	0	0	3	0	0	0	3
undes\unrs	0	0	0	0	11	14	0	0	25
ovres	0	0	0	0	0	0	0	0	0
splt	0	0	0	0	0	0	0	0	0
merg	0	0	0	0	0	0	0	0	0

The exact match rate of `tracenet` over GEANT is 53.5% including unresponsive subnets and is 97.3% excluding totally unresponsive subnets. Published GEANT topology mostly consists of /30 and /29 subnets. Although `tracenet`'s success rate with /30 networks is high, it misses most of /29 and /28 subnets. Studying those missed or underestimated subnets further by probing each and every possible IP address in the original subnet reveals the fact that those subnets mostly do not respond to the probes. That is, either our probe packets or their responses were filtered out or those subnets are not realized despite they are published to exist.

4.1.2 Similarity Rate Evaluation

To have an idea on the success rate other than exact match rate, we analyze the divergence of the collected Internet2/GEANT subnet topologies from the original ones. One can regard each subnet as a feature of the topology and subnet prefix as the value of that feature. Therefore, we can measure the similarity of the collected topology to the original topology in an n-dimensional euclidean space, where n is the number of subnets, by defining a proper distance function. We compute the distance factor of a given subnet S_i simply by taking the absolute value of the difference between its original and collected prefix lengths. Below is the *prefix distance factor* function:

$$d(S_i) = \begin{cases} 0 & S_i \in \text{exmt} \\ |s_i^o - s_i^c| & S_i \in \text{undes} \\ |s_i^o - s_i^c| & S_i \in \text{ovres} \\ \max\{|s_i^o - p_u|, |s_i^o - p_l|\} & S_i \in \text{miss} \\ |s_i^o - s_i^c| & S_i \in \text{merg} \\ |s_i^o - \max\{s_i^c\}| & S_i \in \text{splt} \end{cases} \quad (1)$$

such that s_i^o, s_i^c are the original and collected prefix values for subnet i, p_u and p_l are upper and lower prefix values found in the original or collected topology. Original Internet2 network has upper and lower boundary values of $p_u = 31$, $p_l = 24$ respectively and GEANT has $p_u = 30$, $p_l = 28$. For missing subnets we take the maximum of distances to the boundaries in favor of dissimilarity. Hereby, the dissimilarity of the topologies can be evaluated by the well known Minkowski distance of order k as

$$\left(\sum_i^n d(S_i)^k\right)^{1/k} \quad (2)$$

and normalized similarity equation with parameter $k = 1$ can be defined as

$$1 - \frac{\sum_i^n d(S_i)}{\sum_i^n \max\{(s_i^o - p_l), (p_u - s_i^o)\}}. \quad (3)$$

Based on equation (3) the similarity rate for Internet2 and GEANT are 0.83 and 0.900 respectively while zero means totally dissimilar and one means exactly the same. Compared to Internet2, GEANT, having a smaller prefix length range inherently makes its similarity to be larger.

Although equations (1), (2), and (3) present valuable insights related to similarity, they are not sensitive to the differences in the sizes of subnets (e.g., $|/29\ subnet| - |/30\ subnet| = 4$ vs $|/23\ subnet| - |/24\ subnet| = 256$ though both have a difference of one in terms of prefix length). To reflect this difference, we compute the distance factor of a given subnet S_i simply by taking the absolute value of the difference between its original and collected subnet sizes. Below, we define a new distance factor function called as *size distance factor* function:

$$\widehat{d}(S_i) = \begin{cases} 0 & S_i \in \text{exmt} \\ |2^{32-s_i^o} - 2^{32-s_i^c}| & S_i \in \text{undes} \\ |2^{32-s_i^o} - 2^{32-s_i^c}| & S_i \in \text{ovres} \\ \max\{(2^{32-p_l} - 2^{32-s_i^o}), & \\ \quad (2^{32-s_i^o} - 2^{32-p_u})\} & S_i \in \text{miss} \\ |2^{32-s_i^o} - 2^{32-s_i^c}| & S_i \in \text{merg} \\ |2^{32-s_i^o} - \max\{2^{32-s_i^c}\}| & S_i \in \text{splt} \end{cases} \quad (4)$$

The corresponding similarity equation becomes

$$1 - \frac{\sum_i^n \widehat{d}(S_i)}{\sum_i^n \max\{(2^{32-p_l} - 2^{32-s_i^o}), (2^{32-s_i^o} - 2^{32-p_u})\}}. \quad (5)$$

Again for missing subnets we take the maximum of size distances to the boundaries in favor of dissimilarity.

Based on the equation (5), the subnet size similarities between the collected and the original Internet2 and GEANT topologies are 0.86 and 0.907 respectively while zero means totally dissimilar and one means exactly the same.

4.2 TraceNET Performance over the Internet

In this section we use `tracenet` on public Internet domains to evaluate its performance. A limitation regarding the verification of topology discovery tools over commercial ISPs is that their original topologies are considered to be proprietary and are not available to the public. In our work, we resort to cross validation by running `tracenet` at three different vantage points with the same target IP address set and demonstrate the level of agreement/disagreement among the observed subnets. The results show that all three vantage points agrees on around 60% of all collected subnets by a particular vantage point. Additionally, roughly 80% of the collected subnets by a particular vantage point is also verified by at least one other vantage point. On the other hand, analyzing the collected subnets further show that some subnets are inferred to be larger when collected from another vantage point. One explanation of this could be that like any other active probing based topology collection tool, `tracenet` is affected from rate limiting routers or ISPs. That is, routers or ISPs regulate their responsiveness to probes based on the traffic load or any other rate limiting policies. However, all three vantage points consistently infers that most of the subnets at the non-edge Internet are point-to-point links of /31 or /30 subnets. Then follows /29 subnets and a sharp continuing decrease after /29 with a small increase for /24 subnets. We believe that /24 is a de-facto standard subnet mask for small or medium organizations, however, most of the organizations are also behind probe blocking firewalls.

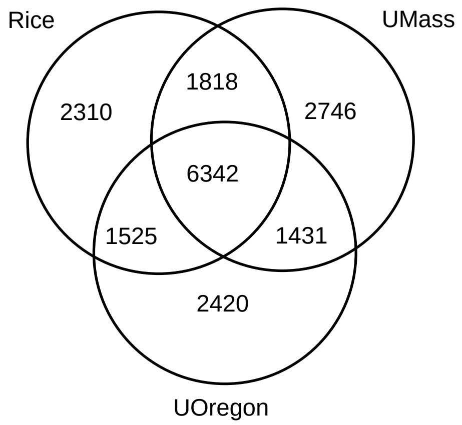

Figure 6: Distribution of exact math subnets among three different PlanetLab sites

As our experimental set-up, we designated PlanetLab sites at Rice University, University of Oregon and University of Massachusetts as our vantage points located at the south, west and east of the United States, respectively. We built a target set of 34084 IP addresses belonging to four major ISPs including Sprintlink, AboveNet, Level3, and NTT America. This target set does not contain IP addresses belonging to the customers of these major ISPs.

Figure 6 demonstrates the distribution of exactly matching subnets collected from three different PlanetLab vantage points. In the figure, almost 80% of subnets observed from a particular vantage point are also observed from one other vantage point. Besides, around 60% of subnets of a particular vantage point is observed from all three vantage points. Note that we selected those vantage points expecting that `tracenet` probes enter into each ISP from different border routers and take different paths towards a destination IP address. As a result, around 20% of subnets being observed uniquely by each vantage point is a natural outcome stemming from different border routers appearing in the paths and various paths being taken toward the destinations.

Next, for each vantage point, we assess the number of IP addresses in our target set, the number of IP addresses that are put into a subnet, and the number of IP addresses for which `tracenet` failed to grow a subnet. In Figure 7, the left aligned bars show the number of target IP addresses for our traces; central bars show the number of IP addresses that are found and placed into subnets of different sizes; and the right aligned bars show the number of IP addresses that are found but could not be placed into a subnet, i.e., these are the IP addresses for which we could not find a subnet larger than /32.

In Figure 7, SprintLink seems to be the least responsive ISP to our probes, the number of unsubnetized IP addresses are large compared to other ISPs consistently from all vantage points. Additionally, as observed from PlanetLab site at Rice, the total number of subnetized and unsubnetized IP addresses is less than the number target IP addresses. Further analysis of our data shows that not all target IP addresses has responded to our probe packets as a result they are not included in un-subnetized /32 IP address set. NTT America is the most responsive ISP. Note that, `tracenet` collects subnets along a path between a vantage point and a destination IP address, hence, path lengths as well as the number of responsive routers on the discovered subnets effect total number of subnetized IP addresses. Path length factor explains the fact that despite the number of un-subnetized IP addresses is large for SprintLink the number of subnetized IP addresses is close to that of Level3 and AboveNET on the average.

Figure 8 shows the distribution of total number of subnets per ISP at three vantage points. The numbers of collected subnets from each ISP are close to each other with respect to all vantage points. One counter intuitive observation takes place when we combine Figure 7 and Figure 8. In the former figure, NTT America has the largest number of subnetized IP addresses whereas in the latter figure it has the least number of subnets. Analyzing our data reveals the fact that NTT America accommodates large subnets of mask /20, /21, /22 whereas other ISPs mostly host small subnets of mask /30 and /31. That is, small number of large sized subnets hosts more IP addresses than large number of small sized subnets in our figures.

Figure 9 shows the distribution of subnet prefix length at three different vantage points in a logarithmic scale. The shape of the figure shows the fact that the distribution of collected subnets are coherent with each other. In the figure, there is a big decrease between /30 and /29 from 4499 to 1546 as observed at PlanetLab site at Rice and even bigger

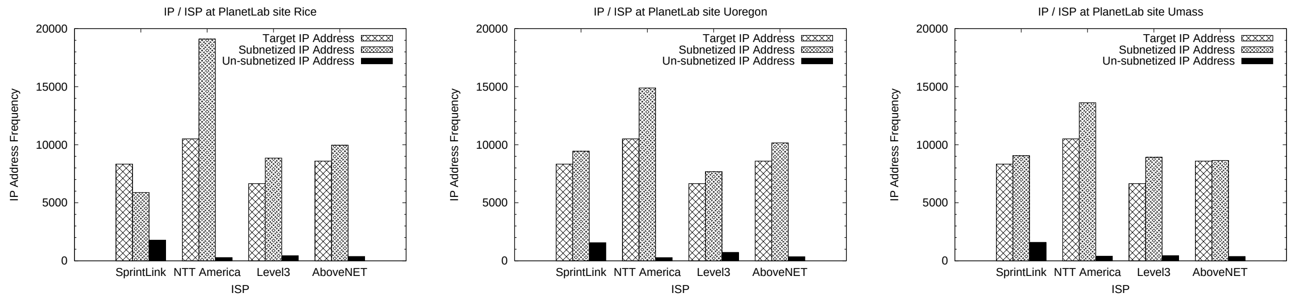

Figure 7: Target, Subnetized, Un-Subnetized IP address distribution at different Planetlab sites

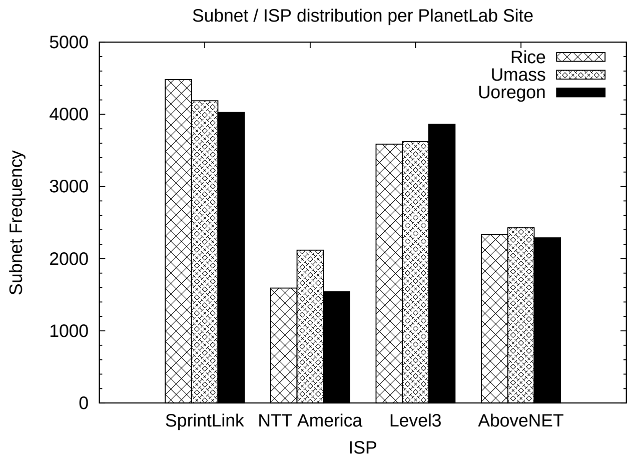

Figure 8: Subnet per ISP distribution at different PlanetLab sites

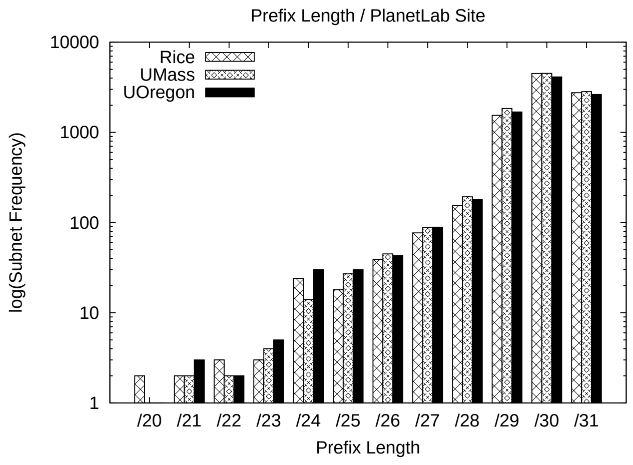

Figure 9: Subnet prefix length distribution at different PlanetLab sites

decrease between /29 and /28 from 1546 to 154. The figure shows that most of the links are point to point links (/31 and /30), then follows /29 multi access links, then the number of subnets decrease dramatically.

As our final experiment, we investigated performance of `tracenet` with different probing protocols. Table 3 shows the number of subnets collected by `tracenet` with ICMP, UDP, and TCP protocols over four ISPs. In the figure, ICMP protocol probing clearly outperforms UDP and TCP protocols in terms of the number of collected subnets. Our findings support the recent observations [9, 15] that routers respond better to ICMP probes compared to UDP and TCP probes. Additionally, the results collected with TCP probing is neglectable compared to ICMP and UDP.

5. CONCLUSIONS

Topology discovery studies make use of every bit of available data to sketch a complete and accurate map of a given network. BGP table dumps, `traceroute` paths, DNS tables, `ping` responses, registered prefix information of ASes, common IP assignment practices, and limitedly supported IP options such as loose source routing and record-route are among the sources of raw data. Among all, `traceroute` is the most prevalent tool used to reveal lists of IP addresses ordered by hop distance at a given network.

In this study, we have devised a set of heuristics to develop a router level Internet topology collector called `tracenet`. `Tracenet` collects topology information of each visited subnet on the path from a source to a destination whereas the traditional `traceroute` collects a single IP address at each visited router. With `tracenet`, we are able to (1) disclose a large number of IP addresses that could be exposed with traditional `traceroute` only if executed from many vantage points; (2) form multi-access and point-to-point links on the networks; (3) detect "being on the same LAN" association

Table 3: Tracenet under ICMP, UDP, TCP probing protocols at PlanetLab site Rice

	ICMP	UDP	TCP
SprintLink	4482	1834	13
NTT America	1593	106	4
Level3	3587	1062	11
AboveNET	2333	777	40
Total	11995	3779	68

among IP addresses; and (4) annotate subnets with their observed prefix values.

As a result, `tracenet` could be utilized as an effective data collection tool to improve completeness and correctness of current and future topology mapping studies. Finally, an implementation of `tracenet` is publicly available on our project web site at http://itom.utdallas.edu.

6. REFERENCES

[1] Geant2 looking glass. Available at http://stats.geant2.net/lg/.

[2] Internet2 observatory data collections. Available at http://www.internet2.edu/observatory/archive/data-collections.html.

[3] D. Achlioptas, A. Clauset, D. Kempe, and C. Moore. On the bias of traceroute sampling. In *ACM STOC*, Baltimore, MD, USA, May 2005.

[4] B. Augustin, X. Cuvellier, B. Orgogozo, F. Viger, T. Friedman, M. Latapy, C. Magnien, and R. Teixeira. Avoiding traceroute anomalies with Paris traceroute. In *ACM IMC*, Rio de Janeiro, Brazil, Oct 2006.

[5] B. Augustin, T. Friedman, and R. Teixeira. Measuring load-balanced paths in the internet. In *IMC Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, New York, NY, USA, 2007.

[6] P. Barford, A. Bestavros, J. Byers, and M. Crovella. On the marginal utility of network topology measurements. In *ACM Internet Measurements Workshop*, San Francisco, CA, USA, Nov 2001.

[7] M. Gunes and K. Sarac. Inferring subnets in router-level topology collection studies. In *ACM IMC*, San Diego, CA, USA, Oct 2007.

[8] M. Gunes and K. Sarac. Resolving anonymous routers in internet topology measurement studies. In *IEEE INFOCOM*, Phoenix, AZ, USA, Apr 2008.

[9] M. Gunes and K. Sarac. Analyzing Router Responsiveness to Active Measurement Probes. In *PAM*, Seoul, Korea, Apr 2009.

[10] M. Gunes and K. Sarac. Resolving IP aliases in building traceroute-based internet maps. *IEEE/ACM Transactions on Networking*, 17(6):1738–1751, Dec 2009.

[11] J. Heidemann, R. Govindan, C. Papadopoulos, G. Bartlett, and J. Bannister. Census and survey of the visible internet. In *ACM IMC*, Vouliag., Greece, Oct 2008.

[12] V. Jacobson. *Traceroute*. Lawrence Berkeley Laboratory (LBL), Feb 1989. Available from ftp://ee.lbl.gov/traceroute.tar.Z.

[13] S. Kim and K. Harfoush. Efficient estimation of more detailed Internet IP maps. In *IEEE ICC*, Glasgow, Scotland, Jun 2007.

[14] A. Lakhina, J. Byers, M. Crovella, and P. Xie. Sampling biases in IP topology measurements. In *IEEE INFOCOM*, San Francisco, CA, USA, Mar 2003.

[15] M. Luckie, Y. Hyun, and B. Huffaker. Traceroute probe method and forward IP path inference. In *ACM IMC*, Vouliag., Greece, Oct 2008.

[16] H. Madhyastha, T. Isdal, M. Piatek, C. Dixon, T. Anderson, A. Krishnamurthy, and A. Venkataramani. iPlane: An information plane for distributed services. In *OSDI*, Seattle, WA, USA, Nov 2006.

[17] D. McRobb, K. Claffy, and T. Monk. *Skitter: CAIDA's macroscopic Internet topology discovery and tracking tool*, 1999. Available from http://www.caida.org/tools/skitter/.

[18] Y. Shavitt and E. Shir. DIMES: Distributed Internet measurements and simulations. Project page http://www.netdimes.org.

[19] Y. Shavitt and U. Weinsberg. Quantifying the importance of vantage points distribution in internet topology measurements. In *IEEE INFOCOM*, Rio de Janeiro, Brazil, Apr 2009.

[20] R. Sherwood, A. Bender, and N. Spring. DisCarte: A disjunctive internet cartographer. In *ACM SIGCOMM*, Seattle, WA, USA, Aug 2008.

[21] N. Spring, R. Mahajan, D. Wetherall, and T. Anderson. Measuring ISP topologies with Rocketfuel. *IEEE/ACM Transactions On Networking*, 12(1):2–16, Feb 2004.

[22] University of Oregon. Route views project. http://www.routeviews.org.