Touring the Internet in a TCP Sidecar [*]

Rob Sherwood
Department of Computer Science
University of Maryland, College Park
capveg@cs.umd.edu

Neil Spring
Department of Computer Science
University of Maryland, College Park
nspring@cs.umd.edu

ABSTRACT

An accurate router-level topology of the Internet would benefit many research areas, including network diagnosis, inter-domain traffic engineering, and overlay construction. We present TCP Sidecar and Passenger, two elements of a system for router-level Internet topology discovery. Sidecar transparently injects measurement probes into non-measurement TCP streams, while Passenger combines TTL-limited probes with the often-ignored IP record route option. The combined approach mitigates problems associated with traceroute-based topology discovery, including abuse reports, spurious edge inference from multi-path routing, unresolved IP aliases, long network timeouts, and link discovery behind NATs and firewalls. We believe that we are the first mapping project to measure MPLS use with ICMP extensions and record route behavior when the TTL is not decremented. We are able to discover NATs when monitoring TCP connections that tunnel through them.

In this paper, we present preliminary results for TCP Sidecar and Passenger on PlanetLab. Our experiments inject measurement probes into traffic generated both from the CoDeeN Web proxy project and from a custom web crawler to 166,745 web sites.

Categories and Subject Descriptors

C.2.1 [**Communication Networks**]: Network Architecture and Design — Network Topology

General Terms

Measurement

Keywords

Network Topology Discovery, Sidecar, Passenger, Record Route

1. INTRODUCTION

Complete and accurate maps of the Internet backbone topology can help researchers and network administrators understand and improve its design. Good maps are unavailable, however, and while some of the reason for this may be social—publishing accurate information is not obviously in a network operator's interest—various research projects [7, 11, 23, 27] have shown that information helpful for research [33, 3] can be collected through traceroute-like probing.

[*]This work was supported by grants ANI 0092806 and CNS-0435065 from the National Science Foundation.

Unfortunately, the increasing use of MPLS, the size of the network, filtering of traffic directed toward router addresses, and fine-grained multi-path routing add intolerable error to traceroute-based studies. Further, large-scale traceroute-based studies typically yield abuse reports from destination hosts behind intrusion detection systems that interpret an incoming traceroute as a port-scan or intrusion attempt [28].

We present a topology discovery tool, Passenger, that revisits IP's record route option to yield more accurate (corroborated) path information, and Sidecar, a system for embedding probes within TCP connections to reduce the intrusiveness of network probing.

With Passenger, we find that the record route option has been prematurely dismissed as a useful tool for network topology discovery: specifically, that its noted limitations are not severe. Its first limitation is that only nine hops of a trace are recorded. PlanetLab [24] allows us to deploy our tools within nine hops of 87–98% of observed addresses (Section 5); distant portions of the network are poorly sampled by traceroute anyway [19]. Second, routers may forward packets with options at lower priority, but we are interested in topology, not performance. Third, firewalls may block packets with record route, yet firewalls also often block traceroute, so little is lost. Finally, intrusion detection systems are likely to report IP options as exceptional events; we find that TTL-limited record route packets can keep destination hosts from seeing and objecting to IP options.

Our challenge is *not* in these obvious limitations: it is matching traceroute-observed addresses with record-route-observed addresses to infer a path that is more correct and complete than either method can collect alone. This is challenging first because traceroute and record route observe distinct addresses with no overlap. We have also observed routers that (a) insert record route entries without decrementing TTL, (b) insert a record route entry when expiring a packet (most do not), (c) insert record route entries only sometimes, perhaps not when under load, and (d) do not insert record route entries at all. This diversity of implementation and configuration makes aligning traceroute and record route paths a daunting task.

Correct alignment of the addresses returned by both schemes is an instance of alias resolution [23, 26, 12, 17]: determining which IP addresses belong to the same router. This means that we can verify the alignment of paths using Rocketfuel's ally tool [27]: when the IP addresses discovered respond to direct probing and when they do not respond, we can discover new aliases not found by other tools. This ability to find aliases for unresponsive routers is a significant step in improving the correctness of measured network topologies.

With Sidecar, we show how to embed Passenger's record route and TTL-limited probes within TCP connections to collect path information with limited intrusiveness. Embedding within TCP requires tracking connection state and disambiguating acknowledgments of probe TCP packets from those of the normal transfer. Although passive observation of TCP behavior and timing has been useful for measurement [18, 22, 32, 4], and traceroute can be embedded with paratrace [16], we believe this is the first demonstration of the feasibility of running traceroute-like probing within TCP connections to pass through firewalls and avoid false accusation by intrusion detection systems.

This paper is organized as follows. In Section 2, we describe the problem of aligning record route with traceroute. We present Sidecar in Section 3. In Section 4, we describe Passenger and our data collection methodology with the results in Section 5. We then conclude and describe our plans for future work in Section 6.

2. MAPPING WITH RR

2.1 Conventional Wisdom

In this section, we describe why the record route (RR) IP option has been unnecessarily discounted as a topology discovery technique and describe why its limitations are not flaws.

As an IP packet with the record route option traverses a router, the router enters its address into an array at a given offset in the IP header and updates the offset. Because space in the IP header is limited, the record route array can hold at most nine addresses. Paths through the Internet are often longer than nine hops, so much of the network would be undiscovered by record route. Fortunately, we can send record route packets from PlanetLab [24]. In Section 5, we find that at least 87% of addresses in our survey are reachable in nine hops from at least one PlanetLab node.

IP options increase the chance of delay, discard, or alarm at intrusion detection systems (IDSs). Delay matters little for topology discovery. Discard is common at firewalls for traceroute packets and record route is not much different. Fonseca et al. [10] found that 46% the paths between PlanetLab hosts drop packets with RR set but that only 8% of those paths are blocked in transit networks. We believe that firewalls and IDSs are typically close to the end-hosts that they protect. To reduce the likelihood of intrusion alarm without sacrificing data from the core of the network, we prevent RR probes from reaching hosts by limiting the TTL. We set probe TTLs to the minimum of the hop count to the end host minus three or eleven, because more information in record route is very unlikely after eleven hops.

2.2 Simple Topology Discovery

We first describe the process of discovering network topology using record route when the network is simple: all routers always decrement TTL and append to the record route array when not expiring the packet. The diversity of router implementation and configuration means that this model is too simple to be directly applied, but it remains useful as an introduction. In the next subsection, we dive into this complexity.

The addresses discovered by traceroute and by record route do not overlap. RR records the address of the *outgoing* interface onto which the packet is sent or the router's designated "loopback" address. By contrast, the "time-exceeded" messages solicited by the TTL-limited probes of traceroute [15], by convention, come from the *incoming* interface where the packet was received.

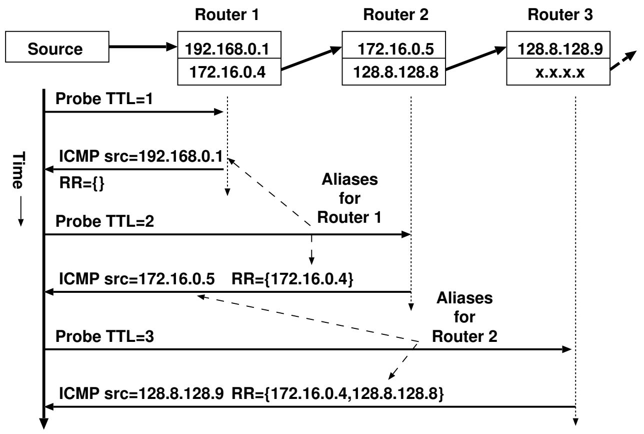

Figure 1: Alias resolution with TTL-limited record route.

In Figure 1, we discover the incoming and outgoing interfaces of each router by sending probes with the RR option and different TTLs. We retrieve the RR array from the header of the packet encapsulated in the ICMP time exceeded message. (The IP header of the response does not include record route.) The ith address of the RR array is an alias for the router that sends the ICMP time exceeded message for TTL=i.

Load-balancing can cause incorrect topology inference when only traceroute is used. When packets from the same traceroute traverse multiple paths, especially of different lengths, incorrect edges can be inferred (Figure 2). The traceroute-inferred network incorrectly links router B to router E because the third probe took a different path. However, the first entry in the RR array in the third probe changed from A2 to A3, exposing the new path and providing feedback that more probes are necessary to discover the entire topology. With record route, the route changes problematic for traceroute become a benefit because they permit the discovery of more topology information.

2.3 Router Behavior Inference

Not every router has the same record route behavior. Specifically, with each additional TTL, our probes may record zero, one, or many new record route addresses. In this section, we list six distinct router implementations and describe the rules we use to classify individual routers into their respective implementations. By classifying routers, we are able to match traceroute and record route addresses on the same router, thus enabling alias resolution and topology discovery.

The router implementation variants we have discovered are:

Type A routers are common: they record the outgoing interface address only if that interface transmits the packet (not when it expires at the router) as described in the simplified examples above. The prevalence of this behavior is consistent with belonging to Cisco routers.

Type B routers are less common: they record the outgoing interface address the packet would have taken even when the packet expires at that router. Because we infer Type B behavior within Abilene [1], we believe it to be consistent with Juniper routers.

Hidden routers never decrement TTL, but always mark record route. Such routers are discovered only by record route probing. We believe these routers are typically part of an MPLS tunnel where decrementing TTL is considered optional.

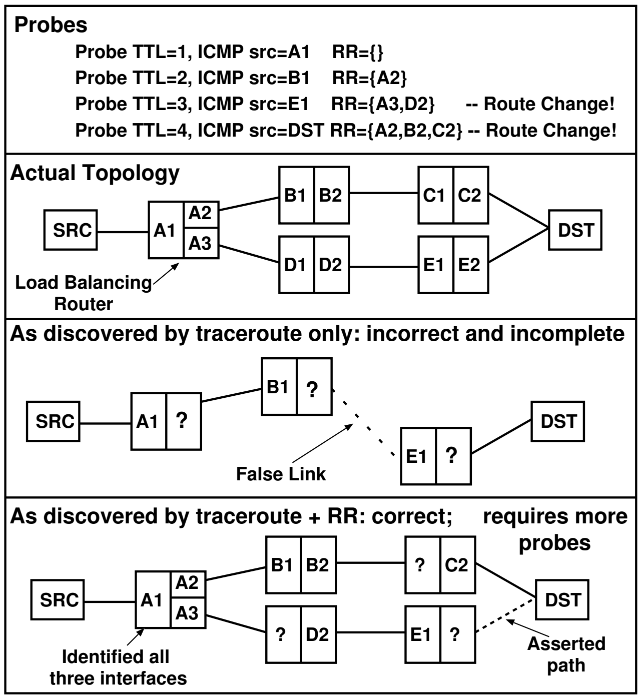
Probes
Probe TTL=1, ICMP src=A1 RR={}
Probe TTL=2, ICMP src=B1 RR={A2}
Probe TTL=3, ICMP src=E1 RR={A3,D2} -- Route Change!
Probe TTL=4, ICMP src=DST RR={A2,B2,C2} -- Route Change!
Actual Topology
SRC
A1
A2
A3
B1
B2
D1
D2
C1
C2
E1
E2
DST
Load Balancing Router
As discovered by traceroute only: incorrect and incomplete
False Link
As discovered by traceroute + RR: correct; requires more probes
Identified all three interfaces
Asserted path

Figure 2: Multi-path route detection with TTL-limited record route ("A3" denoted the third interface of router A, etc.).

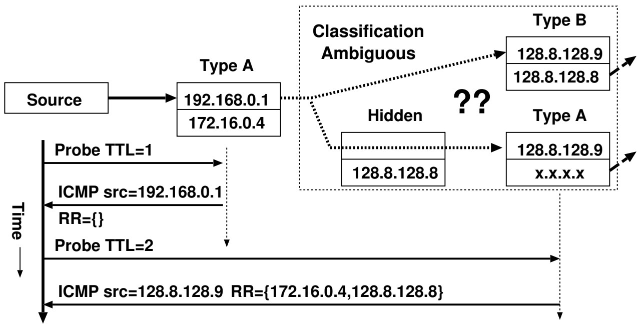

Figure 3: Variations in router implementations allow different topologies to generate the same trace, creating ambiguity.

Type N routers never mark record route packets but always decrement TTL. We believe this to be a configurable option.

Lazy routers do record the outgoing interface, but decrement the TTL only of packets lacking the record route option. We speculate that slow-path processing omits the TTL-decrementing step.[1]

Flaky routers sometimes, but not always, append record route entries. We speculate that they omit processing when under load.

As further diversity, some router configurations appear to not process RR options if the outgoing interface is part of an MPLS tunnel.

The variety of router types make router classification ambiguous. Because different topologies and router implementations can generate the same trace (Figure 3), a router may be misclassified, leading to mismatched addresses. Thus, "128.8.128.8" may be an alias for the router with address "128.8.128.9" (Figure 3, top topology) or an alias for a hidden router that does not appear in the traceroute data (Figure 3, bottom topology).

We describe the rules we use to classify routers from the available data.[2] In these rules, the *current* router originated the ICMP response we are attempting to classify, while *previous* and *next* refer to the routers one TTL closer and further. For clarity, we classify probes based on their RR *delta*: how many new RR entries were added since the previous TTL.

We evaluate the resulting inferences in Section 5.

A-to-B transition If a probe's delta is two, we classify the current router as Type B and the previous router Type A. The first new address belongs to the previous Type A, which it did not place in the previous TTL; the second new address belongs to the current Type B.

Types A and B transitivity If a probe's delta is one and the previous router is Type A or B, then we classify the current router as the same. Unfortunately, the transition from a type B router to another type B router in one hop is indistinguishable from a B-to-Hidden-to-A transition. We use the off-by-one rule to disambiguate.

Off-by-one Because address prefixes are assigned to networks, addresses that are numerically off-by-one are more likely to represent interfaces at either end of a point-to-point link (assigned a /31 prefix) than interfaces on the same router. This heuristic overrides the previous two rules and can assert the existence of hidden routers.

Double-zero The current router is Type N if the current and next routers' deltas are both zero.

Lazy detection The current router is Lazy if all probes *with* record route come from IP address X, all probes *without* from IP Y, $X \neq Y$, and all probes *without* for the next router also come from IP X.

As we apply these rules to Figure 3, the second probe has a delta of two, implying a transition from a Type A to Type B router. However, addresses 128.8.128.8 and 128.8.128.9 are "off-by-one," so because the off-by-one rule overrides the A-to-B transition rule, we declare that the bottom topology is most likely the correct topology.

We are unable to classify all traces: these rules can lead to ambiguity and contradiction, e.g., one trace might classify a router as Type A, when another trace could classify the same router as Type B. Using these simple rules, we can classify 65.4% of our 65 million traces without contradiction. In Section 5, we use the alias resolution tool *ally* [27] to evaluate the correctness of our classifications. Robust and complete router classification and address alignment is the subject of our continued work.

3. SIDECAR DESIGN

Sidecar is our engine for injecting probes, including TTL-limited packets, into TCP connections from user level without altering TCP behavior. Probes sent from within TCP connections can traverse and expose the firewalls and NATs that traceroute probing cannot.

[1]We have not confirmed that the TTL is decremented if the record route list is full. As such, we consider this a potential flaw in router software.

[2]Router OS fingerprinting, or similar additional probing may yield a more accurate classification; we avoided this because router addresses are often not routable.

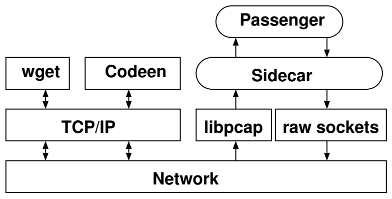

Figure 4: Design layout of TCP Sidecar and Passenger.

The Sidecar system comprises connection tracking, probe identification, RTT estimation, and rate limiting without requiring kernel (firewall or module) support. These design choices make it possible to transparently instrument TCP connections, even from the middle of the network. Figure 4 shows how the TCP embedding logic in TCP Sidecar is separated from the higher-level probe-generation driver, Passenger, allowing easy development of new TCP probing tools. Other Sidecar applications [25] include round trip time and bandwidth estimation.

Injecting probes into TCP without harming connections requires careful design. Sidecar records state and application data via libpcap for many connections in parallel. Sidecar probes take the form of replayed packets carefully crafted to look like retransmissions. Sidecar probes are thus transparent to connections because TCP is robust to packet reordering and duplication. Responses to these probes are either time-exceeded messages from routers, which are ignored by the kernel, or duplicate acknowledgments from the destination host. Because three successive duplicate acknowledgments serve as congestion notification event, Sidecar is careful to not send probes when data is outstanding. We accomplish this by delaying probes until the connection is idle for at least 500 ms.

By changing the TTL of replayed packets, Sidecar is able to accomplish traceroute-like functionality in a TCP stream. When UDP-based traceroute reaches the destination host, the type of response changes from “time exceeded” to “port unreachable”: an unambiguous sign that the destination host has been reached. With Sidecar, the destination’s equivalent response to a replayed packet probe is a duplicate acknowledgment. Because TCP acknowledgments are cumulative and do not identify the specific segment/probe that triggered them,[3] Sidecar cannot distinguish multiple responses from the destination. For efficiency, Passenger sends low TTL probes that will probably not reach the destination in parallel and higher TTL probes that might reach the destination serially.

If the connection closes before probes can be sent, Sidecar can replay the final FIN/ACK packet if the destination is in the “time-wait” state. FIN/ACK probing is not ideal, since the local TCP stack may generate unnecessary RSTs in response to receiver ACKs.

Sidecar permits trivial NAT detection. If Sidecar receives a “time exceeded” message from the destination IP address of the probe, we conclude that a node behind a NAT expired the packet and the source address of the error was rewritten by the NAT.[4] The destination’s distance behind the NAT can be determined by incrementing the TTL until receiving a redundant ACK.

Sidecar parses ICMP extensions [6] allowing detection of MPLS tunnels that support them [5]. Although the utility of knowing MPLS labels is unclear, these extensions proved helpful in debugging the effects of MPLS on router classification.

[3]The DSACK extension [9] does identify the duplicated segment but it does not appear widely deployed [21]. Identifying DSACK support and using it to match multiple probes and responses is future work.

[4]We have found an exception to this rule: a firewall near a PlanetLab source in China would forge “time-exceeded” responses as if from the distant destination address.

4. PASSENGER DESIGN

While Sidecar is the underlying engine for embedding probes into TCP streams, Passenger performs the higher level topology discovery probe generation (Figure 4). Our evaluation goals in exploring Passenger are to: (a) show that embedding record route probing within TCP connections is feasible, (b) quantify how much of network topology record route discovers, and (c) demonstrate reasonable correctness in address alignment over a variety of paths. To construct a dataset for this evaluation, we allow Passenger to observe and trace within the TCP connections of two applications: a web crawler and the CoDeeN web proxy.

4.1 Passenger Logic

Passenger implements the logic of our traceroute and record route probing. Sidecar determines the type of packet to send, determines the round trip time, and returns responses; here we are concerned only with the logic of the measurement. Passenger starts as soon as the completed connection has been idle for one half second. Because web-like connections terminate soon after becoming idle, we try to compress the traceroute into as little time as possible. Passenger remembers the addresses it probes so that it will not repeat a trace for the same source/destination pair.

Passenger traces have two phases. Let *safettl* represent an estimate of the number of hops that probes can be sent into the network without reaching destination firewalls or IDSs. We set *safettl* to the minimum of eleven or three less than the TTL of the destination, as estimated from observing the TTL hops remaining of incoming packets. In the first phase, Passenger sends probes in parallel for all TTLs between 1 and *safettl* with record route set, and then waits for one RTO for them to return. Passenger repeats this process six times, alternating probes with and without the record route option. In the second phase, like traceroute, passenger sends three probes per hop starting at TTL= *safettl* +1 until it reaches the destination or TTL=30 is reached. In this way, Passenger records traceroute data for the entire path and record route data for TTL=1 to *safettl*.

4.2 Data Sources

CoDeeN CoDeeN [30] is a network of partially-open Web proxies deployed on PlanetLab. We ran Passenger for a week (May 17–24, 2006) observing CoDeeN servers. Although CoDeeN is installed on 671 hosts, because some were inaccessible, rebooted during that week, or had too little disk, we only recorded complete data from 234 sources. Passenger monitored connections on port 3128 to CoDeeN users, not proxied connections to origin servers. We collected 13,447,011 traces.

Web Crawler We connect to every web server we could discover. In the first phase, we ran the Larbin [2] web crawler, seeded with http://slashdot.org to find 316,094 websites. We then removed duplicate IP addresses to arrive at 166,745.[5] In the second phase, we ran a multi-threaded Web client from each available PlanetLab node to each address, using Passenger to instrument the connection. Our Web client holds each connection open for 30 seconds, as HTTP persistent connections would, to allow the measurement to complete.[6] The client retrieved the robots.txt file from each server. We collected 51,742,928 traces.

[5]We initially failed to consider virtual hosting, leading to reports of abuse.

[6]If the remote server closes the connection earlier, it remains in TCP’s “time-wait” state, allowing further measurement.

	PlanetLab		Web crawler		CoDeeN	
Autonomous Systems Traversed	331		8,739		891	
Total traces	151,688		51,742,928		13,447,011	
- Unclassified due to contradiction	35,450	(23.3%)	20,324,192	(39.2%)	1,616,079	(12.0%)
IP Addresses discovered	13,048		375,851		22,428	
- Found by Traceroute only	7,293	(55.9%)	298,455	(79.4%)	14,261	(63.6%)
- Found by Record Route only	2,059	(15.8%)	61,672	(16.4%)	3,268	(14.6%)
- Found by both	3,696	(28.3%)	15,724	(4.2%)	4,899	(21.8%)
% end-hosts and routers 9 hops from a PlanetLab Node	87.6%		98.5%		93.0 %	
% links found or confirmed with Record Route	59.5%		69.1%		65.8%	

Table 1: Summary of experimental results.

PlanetLab We also collect a PlanetLab-to-PlanetLab data set using the same web client and PlanetLab hosts as servers. This data set is a strict subset of the web crawler data, but manageable in size. We collected 151,688 traces.

4.3 Safety

We limit probes to 500 kbits per second; above this rate, or the rate at which the raw socket accepts new packets, we skip connections to trace rather than significantly delay the probes of traces in progress. To run within CoDeeN and ensure little interference, we used kernel resource limits to prevent unexpected, excessive memory and processor consumption. We also fetched result data often to reduce disk consumption. We tested our Web crawler and CoDeeN instrumentation from a local machine, grew to a few PlanetLab nodes, and only eventually to all of them. This approach prevented early implementation errors from causing undue havoc.

5. RESULTS

Our evaluation focuses on feasibility, coverage of the topology with record route, and correctness of address alignment (alias resolution). We also comment on the intrusiveness of the technique as measured anecdotally by the absence of abuse reports. Our experiments traversed 8,817 ASes (Table 1) and generated 65,189,939 unique traces.

Approximately 16% of IP addresses in our experiment were discovered by record route alone (Table 1, row three). These routers are either anonymous (do not respond to traceroute), use MPLS to avoid decrementing the TTL, or were interfaces not crossed by traceroute. Of the remaining IP addresses, 55.9–79.4% were discovered by traceroute only.

5.1 Intrusiveness

We designed Sidecar to discover topology without the abuse reports of traceroute. Our CoDeeN experiment probed 22,428 IP addresses over a week with no incidents. Our web crawler experiment, however, generated ten abuse reports across 166,761 destinations. All reports noted frequent, unexpected, and synchronized accesses to robots.txt. One noted ICMP "time-exceeded" messages, but only after being alerted from web logs. We take this as indication that the Sidecar probing technique is in fact unobtrusive, but our custom web crawler needs improvement. We received no reports generated by automated intrusion detection systems. We describe our experiences in more detail in another paper [25].

5.2 Record Route Coverage

The record route option holds at most nine addresses; record route adds nothing to topology discovery further than nine hops from vantage points. To use record route for a broad topology discovery effort requires a measurement platform with sufficient network diversity that most of the Internet is within nine hops of at least one vantage point. By virtue of its geographic diversity, we believe that PlanetLab is, or is becoming, such a platform. Here we attempt to evaluate how well PlanetLab "sees" the broader Internet through record route, to show that record-route-based topology discovery is feasible with current infrastructure.

Record route provides additional information (aliases) or confidence in that information (multi-path detection) for nodes and links within nine hops of a vantage point. From the PlanetLab testbed, we found that 87.6–98.5% of end-hosts and routers in the data set are within nine hops of at least one PlanetLab node (Table 1, second to the last row). Of the links discovered in our topology 59.6–69.1% were found (or confirmed) by RR. Perhaps surprisingly, the fraction of addresses and links reachable in nine hops *increases* with the number of IP addresses. We speculate that the larger measurement set more completely explores the network that is near to PlanetLab nodes, while the measurement of the intra-PlanetLab topology discovers several paths too long for record route and little else. A full characterization of addresses outside of nine hops remains the subject of future work.

5.3 Correct Alias Resolution

Correct alias resolution in the context of record route requires accurate classification of routers in the path. Because the rule set is ambiguous (Figure 3), many traces, especially those experiencing multi-path routing or hidden routers, may be incorrectly classified. Rather than incorporate faulty data into the analysis, we remove any trace that results in a classification contradiction because the cost of erroneous data is impermissibly high. Due to the linear nature of the inference heuristics, one misclassified router may corrupt an entire trace. We believe that a more formal inference engine combined with active probing would reduce the number of ambiguous traces significantly.

Alias resolution, as reported in Rocketfuel [27] and confirmed by Teixeira et al. [29] is typically error prone. We use Rocketfuel's ally tool [27] to validate Passenger's asserted aliases (Table 2, "Alias pairs"), but we use only the IP identifier and common source address techniques because others are too error prone [26]. We traced the false aliases (Table 2, "Ally: no") as reported by ally to B-to-Hidden-to-A transitions in which the Hidden interface address is falsely associated with the Type A router. We are seeking ways of resolving this ambiguity from within the trace to reduce the false positive rate further.

Many of these address pairs cannot be confirmed or disproven because at least one address is unresponsive or unroutable. Ally was only able to confirm or deny aliases for 50.6%, 18.4%, and 40.3% ("Ally:yes"+"Ally:no"/"Alias Pairs") of asserted aliases for the PlanetLab, Web crawler, and CoDeeN data sets. We report the false positive rate as the fraction of aliases disproven ("Ally: no") divided by aliases responsive ("Ally: yes"+"Ally: no").

	PlanetLab	Web crawler	CoDeeN
Alias pairs	6,789	108,870	9,233
Ally: yes	3,389	17,972	3,291
Ally: no	46 (1.3%)	2041 (10.2%)	432 (11.6%)

Table 2: Router alias pairs as compared to *Ally*.

5.4 MPLS Results

We use MPLS ICMP extensions to discover MPLS usage. Between PlanetLab hosts, we identify 2,546 distinct routers that advertise MPLS ICMP extensions, across 38 different ASes. Among CoDeeN hosts, there were 7,730 routers and 112 different ASes. Sidecar was not instrumented with MPLS detection at the time of the web crawler experiment. Further investigation of the possible uses of this information is the subject of future work.

6. CONCLUSION AND FUTURE WORK

IP's record route option has the potential to provide more detailed topology information than previously available through traceroute. The diversity and size of the PlanetLab platform makes this primitive practical for topology measurement. We have found 16% more addresses and discovered IP aliases that cannot be resolved by the techniques of prior mapping efforts, but at the same time, uncovered an interesting problem of inferring the aliases between the addresses discovered by traceroute and record route.

We believe TTL-limited record route packets can substantially improve the efficiency of methods like Doubletree [8] that attempt to reduce the number of probes required to collect a topology.

Sidecar is a new tool for network measurement. Its use of existing TCP connections enables unintrusive measurement and its support for IP options and ICMP extensions makes it potentially useful in developing new measurement techniques.

The code for Sidecar and Passenger, as well as the data generated from this experiment are available from http://www.cs.umd.edu/projects/sidecar.

The combination of Sidecar and record route as topology discovery tools provides many potential avenues of future work. The low rate of abuse reports opens the door for an in depth longitudinal study of network behavior. While we demonstrate that our heuristic solution for address assignment can classify approximately 65% of traces, we seek a more formal treatment of the problem, with solutions that can be verified. We also hope to investigate how the new information exposes anonymous routers [31] that do not generate ICMP responses for traceroute, how record route performance affects other traceroute-like measurements like RPT [13] or pathchar [14], how to assign ownership to routers with this new address information [20], how to adapt the frequency of record route probes to manage load on intermediate routers, how to use congestion control information to limit probe rates, how address alignment would benefit from other data sources like DNS or active probing, and many other questions.

Acknowledgments

We owe Mark Huang and the rest of the PlanetLab staff for their help and patience in the early phases of debugging Sidecar and its tools. We would also like to thank Vivek Pai and the rest of the CoDeeN project for allowing us access to their slice, and Fritz McCall and the UMIACS staff for their timely assistance. Last, we would like to thank our shepherd, Gianluca Iannaccone, and the anonymous reviewers for their helpful comments.

7. REFERENCES

[1] Abilene. http://abilene.internet2.edu.
[2] S. Ailleret. Larbin: Multi-purpose web crawler. http://larbin.sourceforge.net/.
[3] D. Applegate and E. Cohen. Making intra-domain routing robust to changing and uncertain traffic demands: Understanding fundamental tradeoffs. In *ACM SIGCOMM*, 2003.
[4] H. Balakrishnan, *et al.* TCP behavior of a busy Internet server: Analysis and improvements. In *INFOCOM*, 1998.
[5] R. P. Bonica and D.-H. Gan. ICMP extensions for multiprotocol label switching. Internet Draft (work in progress): draft-ietf-mpls-icmp-05, 2006.
[6] R. P. Bonica, *et al.* Modifying ICMP to support multi-part messages. Internet Draft (work in progress): draft-bonica-internet-icmp-08, 2006.
[7] k. claffy, T. E. Monk, and D. McRobb. Internet tomography. *Nature, Web Matters*, 1999. http://www.nature.com/nature/webmatters/tomog/tomog.html.
[8] B. Donnet, P. Raoult, T. Friedman, and M. Crovella. Efficient algorithms for large-scale topology discovery. In *ACM SIGMETRICS*, 2005.
[9] S. Floyd, J. Mahdavi, M. Mathis, and M. Podolsky. An extension to the selective acknowledgement (SACK) option for TCP. IETF RFC-2883, 2000.
[10] R. Fonseca, *et al.* IP Options are not an option. Tech. Rep. UCB/EECS-2005-24, EECS Department, University of California, Berkeley, 2005.
[11] R. Govindan and H. Tangmunarunkit. Heuristics for Internet map discovery. In *INFOCOM*, 2000.
[12] M. Gunes and K. Sarac. Analytical IP alias resolution. In *IEEE Int'l Conference on Communication*, 2006.
[13] N. Hu, O. Spatscheck, J. Wang, and P. Steenkiste. Locating Internet bottlenecks: Algorithms, measurements, and implications. In *ACM SIGCOMM*, 2004.
[14] V. Jacobson. Pathchar. ftp://ftp.ee.lbl.gov/pathchar/.
[15] V. Jacobson. Traceroute. ftp://ftp.ee.lbl.gov/traceroute.tar.Z.
[16] D. Kaminsky. Paratrace. http://www.doxpara.com/read.php/docs/paratrace.html, 2002.
[17] K. Keys. iffinder. http://www.caida.org/tools/measurement/iffinder/, 2006.
[18] K. Lai and M. Baker. Nettimer: A tool for measuring bottleneck link bandwidth. In *USITS*, 2001.
[19] A. Lakhina, J. Byers, M. Crovella, and P. Xie. Sampling biases in IP topology measurements. In *INFOCOM*, 2003.
[20] Z. M. Mao, J. Rexford, J. Wang, and R. Katz. Towards an accurate AS-level traceroute tool. In *ACM SIGCOMM*, 2003.
[21] A. Medina, M. Allman, and S. Floyd. Measuring the evolution of transport protocols in the Internet. *ACM CCR*, 2005.
[22] V. N. Padmanabhan, L. Qiu, and H. J. Wang. Passive network tomography using bayesian inference. In *IMW*, 2002.
[23] J.-J. Pansiot and D. Grad. On routes and multicast trees in the Internet. *ACM CCR*, 28(1):41–50, 1998.
[24] L. Peterson, T. Anderson, D. Culler, and T. Roscoe. A blueprint for introducing disruptive technology into the Internet. In *HotNets*, 2002.
[25] R. Sherwood and N. Spring. A Platform for Unobtrusive Measurements on PlanetLab. In *USENIX Workshop on Real, Large Distributed Systems (WORLDS)*, 2006.
[26] N. Spring, M. Dotcheva, M. Rodrig, and D. Wetherall. How to resolve IP aliases. Tech. Rep. 04–05–04, University of Washington Dept. CSE, 2004.
[27] N. Spring, R. Mahajan, and D. Wetherall. Measuring ISP topologies with Rocketfuel. In *ACM SIGCOMM*, 2002.
[28] N. Spring, L. Peterson, A. Bavier, and V. Pai. Using PlanetLab for network research: Myths, realities, and best practices. *ACM SIGOPS Operating Systems Review*, 40(1):17–24, 2006.
[29] R. Teixeira, K. Marzullo, S. Savage, and G. Voelker. In search of path diversity in ISP networks. In *IMC*, 2003.
[30] L. Wang, *et al.* Reliability and security in the CoDeeN content distribution network. In *USENIX Annual Technical Conference*, 2004.
[31] B. Yao, R. Viswanathan, F. Chang, and D. Waddington. Topology inference in the presence of anonymous routers. In *INFOCOM*, 2003.
[32] Y. Zhang, L. Breslau, V. Paxson, and S. Shenker. On the characteristics and origins of Internet flow rates. In *ACM SIGCOMM*, 2002.
[33] Y. Zhang, M. Roughan, C. Lund, and D. Donoho. An information-theoretic approach to traffic matrix estimation. In *ACM SIGCOMM*, 2003.