Traceroute Probe Method and Forward IP Path Inference

Matthew Luckie
Department of
Computer Science
University of Waikato
Hamilton, New Zealand
mjl@wand.net.nz

Young Hyun
CAIDA
University of California
at San Diego
La Jolla, CA
youngh@caida.org

Bradley Huffaker
CAIDA
University of California
at San Diego
La Jolla, CA
bradley@caida.org

ABSTRACT

Several traceroute probe methods exist, each designed to perform better in a scenario where another fails. This paper examines the effects that the choice of probe method has on the inferred forward IP path by comparing the paths inferred with UDP, ICMP, and TCP-based traceroute methods to (1) a list of routable IP addresses, (2) a list of known routers, and (3) a list of well-known websites. We further compare methods by examining seven months of macroscopic Internet topology data collected by CAIDA's Archipelago infrastructure.

We found significant differences in the topology observed using different probe methods. In particular, we found that ICMP-based traceroute methods tend to successfully reach more destinations, as well as collect evidence of a greater number of AS links. UDP-based methods infer the greatest number of IP links, despite reaching the fewest destinations. We hypothesise that some per-flow load balancers implement different forwarding policies for TCP and UDP, and run a specific experiment to confirm this hypothesis.

Categories and Subject Descriptors

C.4 [**Performance of Systems**]: Measurement techniques

General Terms

Measurement

Keywords

Traceroute, Macroscopic Internet Topology Discovery

1. INTRODUCTION

Traceroute is one of the most widely used measurement tools, operationally useful for diagnosing problems on Internet paths, and vital to researchers trying to develop or validate models of Internet reachability, performance, structure, and growth. Van Jacobson [1] first implemented traceroute in 1988, but since then various developers have added optional extensions, typically in support of operational infrastructure management and troubleshooting needs. For example, NANOG traceroute [2] has extensions that support probing different hops in parallel, path MTU discovery [3], as well as the capture of information about MPLS, AS paths, and TOS fields. Other probe methods have been developed to extend the visibility and accuracy of gathered IP path data: tcptraceroute [4] and paratrace [5] can provide visibility beyond a firewall that blocks another probe method from reaching a destination; Paris traceroute reduces the likelihood of anomalies such as reporting false IP paths [6].

We compare six different probe methods in three scenarios: to a random list of routable IP addresses; to the top 500 websites as reported by Alexa [7]; and to a list of routers that recently responded to one of our probes. The first scenario reveals differences in effectiveness of various probe methods for large-scale Internet topology mapping. The second and third scenarios illustrate what an operator might encounter when using a specific probe method to debug a fault. We publicly release our dataset[1].

We further compare two of the methods by examining the macroscopic Internet topology data collected by CAIDA's Archipelago infrastructure. This data complements the data collected for the comparison of six methods by providing a comparison over a longer time period (7 months), a larger number of vantage points (21), and a larger number of destinations (every routed /24, or 7 million /24's).

1.1 Overview of traceroute probe methods

Van Jacobson's original traceroute tool [1] sends UDP probes to high-numbered destination ports. It increments the destination port number with each probe so that it can match up responses to probes. Traceroute uses the reception of an *ICMP port unreachable* message to indicate that the destination has been reached and probing is complete. The use of high-numbered ports minimises the chances of accidentally probing an open service on a machine, which would prevent an ICMP port unreachable message from signalling completion to traceroute. Three other causes for traceroute to terminate are: when an *ICMP destination unreachable* message is received in response to a probe; when a pre-specified number of hops are probed; or when the user cancels execution. This probe method is the default method used by LBL traceroute, derivatives of which are found on most Unix systems including MacOS X, FreeBSD, and popular distributions of Linux. By default, this UDP-based method starts with a destination port value offset from

IMC'08, October 20–22, 2008, Vouliagmeni, Greece.

[1] http://imdc.datcat.org/publication/1-06NL-T

32768 + 666 + 1 (33435) and increments the port value for each probe. A weakness of this probe method is that some firewalls block these probes from reaching their intended destination as a product of blocking unknown traffic by default, reducing the utility of traceroute. However, some firewalls by default explicitly permit UDP probes in the order of 100 ports above 33435 to pass, allowing traceroute to continue through [8].

A second approach is to use *ICMP echo request* probes in place of UDP probes. Since ICMP does not have the concept of ports, matching probe responses to outgoing probes uses a different technique: a unique ICMP id/sequence pair in each outgoing (and thus responding) probe. ICMP is the default probe method used by traceroute on Microsoft Windows, and is also the method that was used by CAIDA's skitter project [9]. The ICMP method takes advantage of the ICMP protocol facility intended to support lightweight network management; processing an ICMP echo request is simpler than sending an ICMP destination unreachable message in response to UDP or other probes. The disadvantage of the ICMP-based method – like UDP probes to high-numbered ports – is that ICMP echo request probes are also thought likely to be blocked by firewalls [10].

A third approach is to use TCP SYN probes to a well-known port, such as the default port for a web server – port 80. First implemented in tcptraceroute [4], this method sends each probe with a unique IP-ID value to match responses with sent probes. The advantage of this approach is that TCP SYN probes to port 80 cannot easily be distinguished from normal connection requests to web servers, and so are less likely to be blocked by a firewall, and thus more likely to reveal more of the forward IP path. However, some firewalls may be configured to block TCP packets that do not belong to an established TCP conversation [5, 11].

Paris traceroute [6] implements two additional variations of the ICMP and UDP probe methods, which we refer to as ICMP-Paris and UDP-Paris. Paris traceroute avoids measurement anomalies caused by load balancing routers by ensuring the first four bytes of the IP payload – which are often used by a load balancing device to select the next-hop – are the same for each probe in a single traceroute sequence. Convincing a load balancer that two packets should be sent along the same path reduces the likelihood of reporting false IP paths and encountering false loops.

For UDP probes to belong to the same flow, and be treated as such by load balancers, the UDP source and destination ports must be identical. As this constraint prevents the use of the UDP destination port to match responses with sent probes, Paris traceroute manipulates the UDP payload so that the UDP checksum value can be used to perform this matching. Since some systems use the UDP checksum field for temporary storage when processing a UDP packet [12], and NAT boxes irreversibly change the checksum when rewriting packet headers anyway, Paris traceroute also uses the IP-ID field as a backup key in the matching process. For ICMP probes to belong to the same flow, the ICMP type, code, and checksum values must remain the same. The ICMP id and sequence fields continue to be used as the probe identifier, and the ICMP payload is constructed so the checksum is the same for each probe.

There are additional traceroute probe methods and techniques that we do not consider in this paper. For example, tcptraceroute can send various combinations of TCP flags in addition to the SYN case described. Paratrace [5] uses an existing TCP session with a target to send TTL-limited retransmissions of data packets to the destination. As the retransmissions used for measurement cannot easily be distinguished from genuine TCP retransmissions, the tcptraceroute probes in paratrace are able to traverse firewall systems that would otherwise block these probes. This technique was first used for Internet topology discovery in Sidecar [11]. Unfortunately the requirement to have an existing TCP connection over which to execute the measurement renders this method difficult to apply to macroscopic Internet topology discovery.

1.2 Contributions of this work

This paper compares the pictures of Internet topology that emerge by using different traceroute probe methods popular in the Internet research and operations communities. In doing so, we corroborate the benefits of Paris traceroute reported in "Avoiding traceroute anomalies with Paris traceroute" [6], particularly the benefits of reducing the number of false loops inferred. We identify the probe methods most useful for inferring IP and AS links, and introduce techniques for identifying spoofed traceroute responses and inference of firewall positioning.

The rest of this paper is organised as follows. Section 2 describes the specific traceroute probe methods evaluated, the evaluation metrics considered, the probing tool, and probing sites. Section 3 examines the agreement of the forward IP paths inferred per probe method. In doing so, we provide data on firewall placement in relation to the destinations probed, as well as data on spoofed responses. Section 4 analyses seven months of macroscopic Internet topology data collected by the Archipelago infrastructure. Finally, Section 5 discusses related work, and Section 6 concludes.

2. METHODOLOGY

2.1 Traceroute probe methods

The first five traceroute probe methods described in Section 1.1 are implemented in scamper [13], a parallelised Internet measurement utility similar to skitter [9] that is optimised for large-scale Internet topology discovery. Rather than consider one traceroute task at a time, scamper aims to fill a specified packets-per-second rate and conducts measurements to multiple destinations in parallel as required.

In this work, we infer the forward IP path by sending up to two probes per hop and halt probing a destination for any of four reasons: upon receiving a response with a probed destination address as the source of the reply unless that response is a time exceeded message; upon receiving an ICMP destination unreachable message; upon receiving a response with an address that appears earlier in the path unless the reply sequence indicates a case of zero-ttl forwarding [6]; or after five unresponsive hops. When no matching response is received from five consecutive hops, we send up to two last-ditch probes with an IP-TTL value of 255, attempting to ascertain if anything further down the path might answer. Any response to these final two probes is stored as ancillary data, but no further probing of the path takes place, and the recorded reason for halting traceroute (five consecutive unresponsive hops) is not changed.

We experimented with a sixth probe method we call UDP-Paris DNS which sends 130-byte UDP packets from source

Table 1: Traceroute probe methods compared

	Method	Specifics
1.	UDP	udp-dst-port 33435 + index, udp-src-port ephemeral
2.	UDP-Paris	udp-dst-port 33435, udp-src-port ephemeral, ip-id and udp-checksum as index
3.	UDP-Paris DNS	udp-dst-port 33435, udp-src-port 53, ip-id and udp-checksum as index, DNS payload
4.	ICMP	icmp-sequence as index, icmp-checksum variable
5.	ICMP-Paris	icmp-sequence as index, icmp-checksum constant
6.	TCP port 80	tcp-dst-port 80, tcp-src-port ephemeral, ip-id as index

port 53 and includes a well-formed DNS response payload. The motivation of UDP-Paris DNS is to extend the reach of traceroute past firewalls that might permit specific protocols inwards, such as DNS, while blocking others. The payload is derived from the response packet solicited by requesting an A record for localhost from `a.root-servers.net`. The first two bytes of the UDP payload, corresponding to the DNS ID field [14], change with each traceroute probe so that the UDP checksum can be manipulated to be the probe's sequence number.

Table 1 summarises our six traceroute probe methods. We randomly order the sequence of probe methods for each destination to avoid persistently biasing the results of any particular method. In order to avoid biasing the next-hop decision a load-balancing router might make when forwarding a set of our traceroute measurements to a single destination, we also randomise the ephemeral source port chosen for each UDP, UDP-Paris, and TCP traceroute to one of 16 values, and randomise the ICMP checksum chosen for each ICMP-Paris traceroute to one of 16 values. A fixed delay of five seconds is inserted between completing one traceroute method and starting the next. Finally, scamper is configured to probe at 100 packets per second.

2.2 Traceroute method evaluation metrics

This section describes four metrics we use to evaluate the effectiveness of the various probe methods.

2.2.1 Destinations reached

We consider a destination *reached* if:

- an ICMP port unreachable message is received in response to a UDP or TCP probe,
- a TCP packet is received from the destination in response to a TCP probe,
- an ICMP echo reply packet is received from the destination in response to an ICMP echo request packet, or
- an ICMP destination unreachable message is received with a source address matching the destination address probed.

Reaching a destination is a more efficient form of halting than timing out due to a lack of responses. Also, by reaching the destination, we can obtain the RTT and forward IP path length.

Note that some hosts will spoof the source address of an ICMP response, leading to an overestimation of the number of destinations reached. It is not possible to identify all instances of source address spoofing, though we can identify some behaviour indicative of spoofing in our datasets, which we discuss in Section 3.

2.2.2 Complete IP paths

An IP path is *complete* if the destination is reached and there is a response from all intermediate hops; that is, there are no gaps in the path. Complete paths are desirable because they reduce uncertainties in topology analysis. Complete paths are also more time efficient to infer than paths with unresponsive hops.

2.2.3 Unique IP links

This metric counts the unique IP links (that is, pairs of adjacent IP hops) seen in traces. All other things being equal, the more IP links we have, the more accurately and comprehensively we can infer router-level connectivity and AS relationships. Thus one indication of a method's power is its ability to accurately infer IP links. In this work, however, we do not explicitly validate the inferred links but simply use the raw quantity of coverage as the measure of success.

We expect to see variation in each method's ability to infer IP links. For example, some paths load balance UDP packets on a per-flow basis but not ICMP packets [15]. Through the re-probing of common path segments between destinations, UDP and TCP probes may reveal alternative IP paths. Similarly, some methods will vary in their ability to infer IP hops where a firewall discards particular protocols and packet types.

The number of IP links and the number of complete IP paths are related metrics, but they address different concerns. Although a complete IP path is desirable, completeness is not a requirement for many analyses, and a traceroute method may produce a large number of IP links without necessarily producing a large number of complete paths because of the stricter conditions for complete paths.

2.2.4 Unique AS links

This metric counts the unique AS links inferred from measured IP paths. To derive AS links, we first convert the IP paths to AS paths by mapping IP addresses to ASes using a BGP table dump obtained from RouteViews [16]. Although complete validation of AS link inferences requires communication with the owner of the ASes in question, a probing method is typically considered superior if it is able to infer more unique AS links consistent with publicly available BGP data [17].

2.3 Traceroute vantage points

Table 2 shows the eight hosts used as traceroute vantage points. The host names consist of a 3-letter airport code and a 2-letter country code. These hosts, located at geographically and topologically diverse points in the Internet, are a part of CAIDA's Archipelago (Ark) measurement infrastructure [18]. One of Ark's tasks is to conduct active probing for CAIDA's macroscopic Internet topology mapping project.

Table 2: Traceroute vantage points used. These hosts are distributed across the globe; most are hosted at educational institutions.

Host	Location
cbg-uk	University of Cambridge Cambridge, England
nrt-jp	Asia-Pacific Advanced Network (APAN) Tokyo, Japan
syd-au	AARNet Sydney, Australia
bcn-es	Universitat Politècnica de Catalunya Barcelona, Spain
hel-fi	Helsinki University of Technology (TKK) Espoo, Finland
cjj-kr	KREONet2 Daejeon, Korea
iad-us	ARIN Bethesda, Maryland
san-us	CAIDA San Diego, California

Table 3: Traceroute methods used by various topology mapping projects. Most use combinations of ICMP and UDP methods. Ark used UDP prior to Nov 2, 2007 before switching to ICMP-Paris. iPlane uses ICMP-Paris from PlanetLab nodes. iPlane also uses traceroute measurements obtained from public traceroute servers which use undisclosed methods, though most likely UDP or ICMP.

Project	Method
CAIDA Ark/Scamper	ICMP-Paris, UDP
CAIDA Skitter	ICMP
DIMES	ICMP primary, UDP backup
iPlane	ICMP-Paris, UDP, ICMP

2.4 Destination address lists

We review the three different destination address lists used in our probing experiments.

2.4.1 *Random routable IP address list*

The first address list consists of 261,530 random IP addresses contained in advertised prefixes found in RouteViews BGP tables [16]. The objective of using this list is to compare the utility of various traceroute probe methods to large-scale Internet topology mapping projects. Table 3 lists several well-known topology measurement projects and their traceroute methods. CAIDA's topology mapping project, DIMES, and iPlane all derive destination lists from advertised BGP prefixes.

This list is actually composed of several different lists. We generated a different set of destinations for each vantage point consistently, using the following procedure. We first derived a set of BGP prefixes by merging daily snapshots of 'sh ip bgp' on RouteViews [16] from 19-25 March 2008. We chose the median file size snapshot each day to avoid truncated or otherwise non-representative tables. We then extracted IP prefixes from this set using straightenRV [19].

In prior experience we have found that prefixes that appear in only one or two snapshots tend to be anomalous in some way, such as being unreachable most of the time, so we selected prefixes that appear in at least three snapshots. Of the initial set of 257,504 base prefixes, 255,981 (99.4%) appear in at least three snapshots, and represent the equivalent of 110.3 /8's. 251,452 prefixes (97.6%) appear in all seven snapshots.

We then divided this list of 255,981 prefixes into two sets. The first set, *roots.le16*, consists of all root prefixes (prefixes not enclosed in any other prefix) with a prefix length of 16 or shorter. This set contains 10.0k prefixes covering 89.8 /8's. The second set, *prefs.gt16.le24*, consists of the remaining prefixes, those of length greater than 16 and less than or equal to 24. This set represents 240.6k prefixes covering 29.9 /8's. Note that these two sets of prefixes intentionally overlap in their coverage of the address space.

Finally, we created a custom destination list for each vantage point using the following procedure:

- select one random address in each /16 covered by the set *roots.le16*,
- select one random address in each prefix of the set *prefs.gt16.le24*,
- never select more than one destination in any /24,
- exclude addresses in bogon prefixes [20],
- exclude addresses in CAIDA's do-not-probe list.

CAIDA's do-not-probe list consists of 975 prefixes covering the equivalent of 1.14 /8's, largely due to a single /8 on the list. This procedure selects as many destinations in a /16 as there are more specific prefixes for that /16, but never more than one destination per /24. This constraint may allow any possible divergent routing introduced by the more specific prefixes to be captured. This procedure yields 261,530 addresses for each vantage point, with minimal overlap in addresses between vantage points.

2.4.2 *Alexa top 500 websites*

The second address list is composed of the top 500 websites as reported by Alexa. We chose this address list for three reasons: (1) the targets are destination hosts reachable on TCP port 80, and as popular web servers they probably receive traceroute traffic as users troubleshoot connectivity problems; (2) the targets are likely to have a firewall in front of them, which allows us to compare which probing methods are more likely to reach the destination, and possibly infer the approximate location of the firewall; (3) the Alexa list is studied in other work [15, 21].

The actual number of IP addresses included in the website list is 422, as some websites in the top 500 have IP addresses in common with other websites. For example, several top 500 websites hosted by Akamai resolve to the same IP addresses. Similarly, some nationalised instances of Google (e.g., `google.co.nz`, `google.co.uk`) and Ebay share the same sets of IP addresses, and we include them only once.

Many of these websites have multiple A records (IP addresses) returned in a single DNS response. In these cases, we add to the address list a random address from the returned list not previously included. For these reasons, we

Table 4: Median rate of traceroute method halt reason per technique. ICMP-Paris reaches 80% more destinations than UDP-Paris.

	Reached	ICMP-Unreach	Loop	GapLimit
ICMP-Paris	9.3%	12.7%	6.7%	71.5%
TCP	8.5%	11.7%	6.3%	73.7%
UDP-Paris	5.3%	11.2%	6.4%	77.2%

probe all four addresses returned from an A query to the 58 nationalised instances of Google ranked by Alexa in the top 500 websites. The IP addresses returned by DNS for popular websites can differ depending on the vantage point. We did all DNS lookups from the same network as the host from which we subsequently performed traceroute measurements. Because we probed this list from a single host, the vantage-point dependency of DNS mappings should not affect our conclusions.

2.4.3 Router list

Finally, we probe an address list composed of 2000 random router IP addresses previously discovered with traceroute. We chose this list because the targets are known to be routers that have recently responded with an ICMP time exceeded message, and should therefore be reachable. The objective of probing this list is to determine if any particular traceroute probe method is likely to be more useful when probing destinations that are routers.

3. RESULTS

3.1 Random routable probing results

3.1.1 Overall results

Figure 1 shows the distribution of traceroute halt reasons for all methods per vantage point for the random routable IP address list. The vantage points cbg-uk and nrt-jp used all six probing methods, while the other six vantage points used three probing methods: UDP-Paris, ICMP-Paris, and TCP traceroute. These six vantage points used a reduced set of methods due to the limited additional utility of the other methods; UDP-Paris DNS reached few additional destinations compared to UDP-Paris, and the traditional UDP and ICMP methods are known to infer false loops at a greater rate due to not maintaining a constant flow identifier between probes [6]. The data collection was staggered between Aug 6th and 16th 2008; the collection for the six vantage points that used three probing methods completed in approximately two and a quarter days; the collection for the two vantage points that used six methods completed in approximately twice as much time. Table 4 shows the median rate of halt reason for the three methods which were used on all eight vantage points.

We now examine the results for each halt reason, starting with the lowest bars in Figure 1 and working up. The lowest bars, *reached*, indicate the percentage of traces where the destination was considered reached according to the definition in Section 2.2.1. The rate of reachability is between 9.0% and 9.4% with the ICMP-Paris method, between 8.2% and 8.7% with the TCP method, and between 5.1% and 5.4% with the UDP-Paris method – depending on the vantage point. The reachability statistics for each method vary an average of 0.4% across vantage points; an earlier data collection varied less than 0.1% when it was collected across all vantage points simultaneously, so we believe that the variation in reachability in this scenario is caused by the variation in when the data was collected.

The second lowest bars, *ICMP-unreach*, report the percentage of traces whose halt reason was an ICMP destination unreachable message with a source address not matching the destination address probed. The median percentage across vantage points for this halt reason was 12.7% for ICMP-Paris, 11.7% for TCP, and 11.2% for UDP-Paris. Notably, the rank order of methods for both the reached and ICMP-unreachable cases are the same, indicating that firewalls are less likely to silently discard ICMP probes than they are UDP probes, rendering ICMP-Paris a more productive probing method than UDP-Paris for random destinations.

The next set of bars, *loop*, indicate the percentage of traces that halted because an IP address appeared for the second time in the same trace and was not a case of zero-ttl forwarding [6]. This halt reason varied more per vantage point than per probe method, consistent with the fact that loops are routing issues independent of probe method. The median rate of observed loops was 6.7% for ICMP-Paris, 6.4% for UDP-Paris, and 6.3% for TCP.

Finally, the top bars, *gaplimit*, indicate the percentage of traces that probed five consecutive hops and obtained no responses for any. By far the most common halt reason for all methods, this unreachability is unsurprising given a random destination list, since it is well-established that most IP addresses do not respond to direct probes [22]. As a result of ICMP-Paris more effectively reaching destinations or triggering ICMP destination unreachable messages, the percentage of destinations that result in such unproductive probing is 2.2% more for TCP and 5.7% more for UDP-Paris.

Two of our vantage points, cbg-uk and nrt-jp, used three further probe methods; traditional ICMP, traditional UDP, and UDP-Paris DNS. UDP-Paris DNS is only slightly more effective than UDP-Paris; from a list of 261,530 addresses, it reaches 311 more destinations from cbg-uk and 356 more from nrt-jp. This negligible difference surprised us; we had believed that many more operators would configure their firewalls to allow packets with a source port of 53 through without requiring an initial matching request, which would have allowed UDP-Paris DNS through while blocking UDP-Paris, but this data does not support our initial hypothesis. We note that it could be possible that firewalls are configured to drop probes to the traditional traceroute port range while permitting other packets with a source port value of 53.

Our other two probe methods were the traditional UDP and ICMP techniques which do not send probes with a constant flow identifier as Paris traceroute does [6]. Table 5 shows the halt differences between a traditional traceroute technique and the Paris variation for vantage points cbg-uk and nrt-jp. Our results show that the UDP technique derives a greater reduction than ICMP in the number of loops detected, perhaps due to fewer routers that load balance ICMP packets per flow [15]. The UDP technique also achieves the greatest percentage increase in reached destinations compared to non-Paris probing. However, approximately 80% of the paths that inferred a false loop in the traditional case

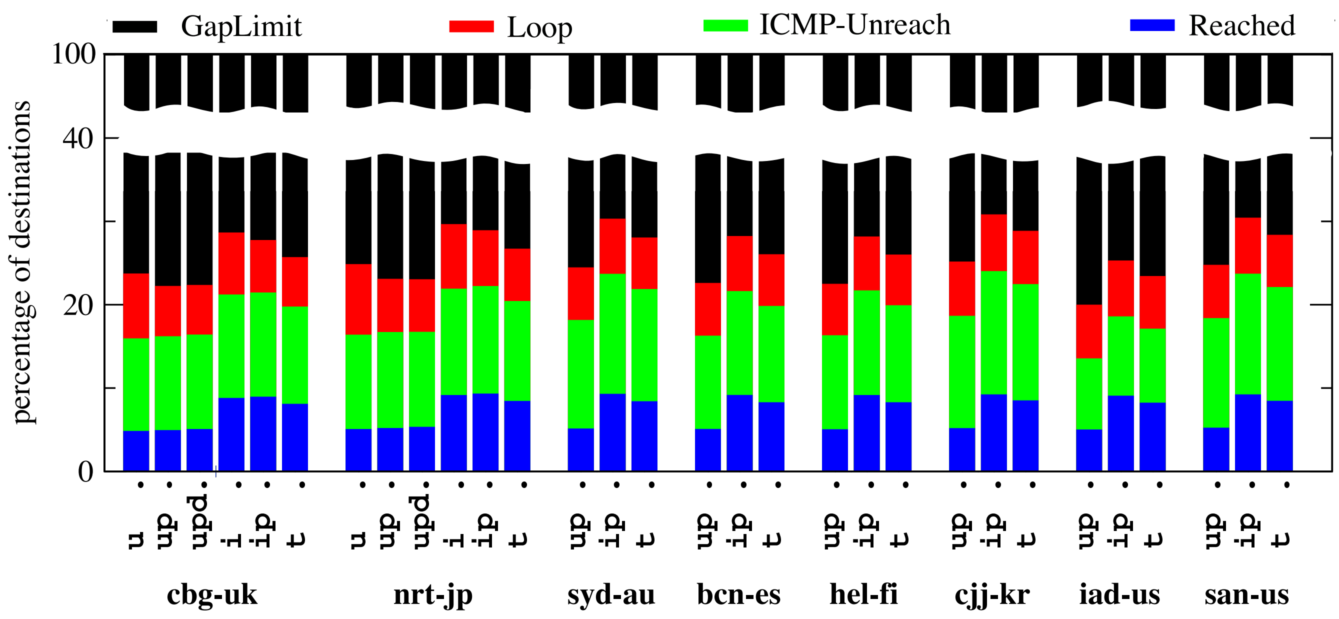

Figure 1: Traceroute method halt reasons for random routable IP address list. 9.3% of destinations were reached with ICMP-Paris, 8.5% with TCP, and 5.3% with UDP-Paris. (u=*UDP*, up=*UDP-Paris*, upd=*UDP-Paris DNS*, i=*ICMP*, ip=*ICMP-Paris*, t=*TCP*)

Table 5: Halt reason differences between Paris and traditional traceroute methods. There are significantly fewer loops inferred with Paris traceroute, though only a small increase in the number of destinations reached. Percentages give the relative increase/decrease in the number of traces with a given stop reason.

	UDP-Paris vs. UDP		ICMP-Paris vs. ICMP	
cbg-uk				
Reached	+274	(+2.1%)	+376	(+1.6%)
ICMP-Unreach	+383	(+1.3%)	+265	(+0.8%)
Loop	−4563	(−22.4%)	−3019	(−15.5%)
GapLimit	+3906	(+2.0%)	+2378	(+1.3%)
nrt-jp				
Reached	+308	(+2.2%)	+460	(+1.9%)
ICMP-Unreach	+553	(+1.9%)	+391	(+1.2%)
Loop	−5463	(−24.7%)	−2775	(−13.7%)
GapLimit	+4602	(+2.3%)	+1924	(+1.0%)

now halt in the Paris case due to the gaplimit condition being met.

3.1.2 *Reachability*

We next consider aspects of destination reachability for UDP-Paris, ICMP-Paris, and TCP. We consider these the canonical traceroute methods; their variations do not reach significantly more destinations. Figure 2 shows reachability intersection statistics across each method for all vantage points. The destination reachability statistics are similar for all probe sources, so we choose cbg-uk for detailed analysis. In total, 31439 destinations were reachable from cbg-uk by at least one of the three methods, while 32.5% of these were reachable by all methods. The ICMP-Paris technique reached 75.2% of these destinations. Using both TCP and ICMP-Paris methods, 97.7% of the destinations were reached. Put another way, of the 31439 destinations reached only 2.3% were reachable solely with the UDP-Paris method. If a goal of Internet topology discovery is to maximise discovery at the edges of the Internet, UDP-Paris is not a good probe method to use compared with the other methods evaluated.

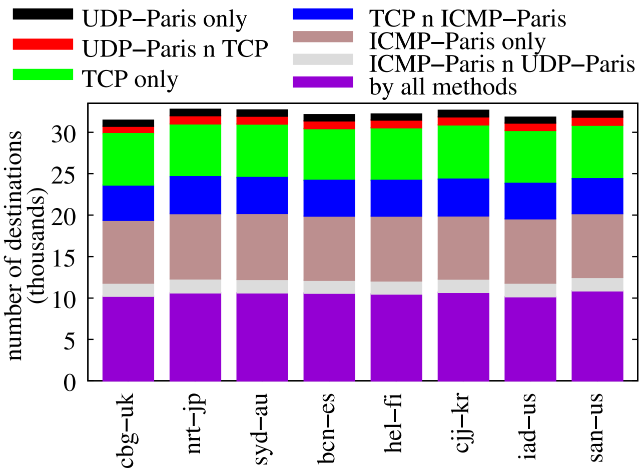

Figure 2: Uniqueness of reached destinations by combinations of methods for the random routable address list. Focusing on cbg-uk the total coverage for individual methods is ICMP-Paris 75.2%, TCP 68.6%, and UDP-Paris 42.7%.

As described in Section 2.2.1, there is more than one way to define a destination as being reached when using trace-

Table 6: Replies classed as reaching a destination from cbg-uk. At least 95% of the replies from a destination were of the expected type (shown with an asterisk).

	UDP-Paris	TCP port 80	ICMP-Paris
Reached	13411 (5.1%)	21576 (8.2%)	23638 (9.0%)
Expected type	96.0%	94.6%	98.8%
Prohib filter	156	156	56
Prohib host	163	44	2
Unknown host	0	0	1
Unreach net	8	5	8
Unreach host	208	225	176
Unreach proto	1	7	5
Unreach port	12875*	293	26
Echo reply	–	–	23364*
Syn-ack	–	11121*	–
Rst-ack	–	9238*	–
Rst	–	394	–
Other TCP	–	93	–

Table 7: Stop reason for methods that probed beyond an ICMP time exceeded message from the destination for the cbg-uk vantage point. ICMP-Paris is nearly 4 times as likely to obtain an expected type of response with continued probing than UDP-Paris.

	UDP-Paris	TCP port 80	ICMP-Paris
Reached, expected	225	787	1103
Reached, other	53	99	83
Loop	367	459	568
GapLimit	1368	1132	1002
Total messages	2013	2477	2756

route. Table 6 shows details of the replies that caused destinations to be reached using the UDP-Paris, TCP port 80, and ICMP-Paris methods from cbg-uk. The expected type of reply from a destination of a UDP-Paris traceroute is an ICMP port unreachable message; 96.0% of the replies classed as reaching the destination were of this type for this method. The expected type of reply packet for a TCP port 80 traceroute is a TCP SYN/ACK or RST/ACK packet, depending on if a service is listening on the destination port; 94.6% of the replies classed as reaching the destination were of this type for this method. The expected type of reply packet for an ICMP-Paris traceroute is an ICMP echo reply packet; 98.8% of the replies classed as reaching the destination were of this type for this method.

Table 6 excludes ICMP time exceeded replies with a source address matching the destination probed. In this case when such a reply was received by scamper it continued probing until there was a reason to halt probing in order to determine if a destination might be reached with further probes. Table 7 shows the result of probing further. The ICMP-Paris method is much more likely to then obtain an ICMP echo reply message (40.0%) than UDP-Paris is to obtain an ICMP port unreachable message (11.2%) in this scenario. Of the remaining ICMP destination unreachable messages received, all had a source address matching the destination address probed.

Since we use multiple methods to infer forward IP paths to each destination, we can use the combined information to implicate some replies as spoofed. For UDP traceroute methods, an ICMP port unreachable message should be sent by a destination host when the datagram cannot be delivered because the indicated port is not running an active process [23]. The port unreachable message may be from an address of a different interface on the same host as the destination address of the triggering packet; in fact, this artifact is used as a heuristic in alias resolution [24, 25]. However, if the source address of a port unreachable message in one IP path is also observed in a time exceeded message in one of the other IP paths to that destination, and at least one other time exceeded message follows from an address other than the destination's, then the port unreachable is inferred as not sent by the destination, since the pair of time exceeded messages in the second path indicate the destination has still not been reached. Of the 12875 destinations considered reached using the UDP-Paris method from cbg-uk based on an ICMP port unreachable response, 27 are inferred to be from an intermediate host rather than the destination itself. Although arguably negligible, this result does demonstrate that intermediate systems do send port unreachable messages on a destination's behalf, which can lead to false alias resolution and false inference of reachability – reachability that was previously thought to be unambiguous [11].

Similarly, if one method reveals an IP path to E of A B C D E, while a second method reveals a path of A B E, it is likely that the reply allegedly from E in A B E is spoofed, as this path suggests B and E are neighbours, while the presence of D in the first path rules out the possibility that C is merely an alias of E. Since a router should always prefer a directly connected path, we would not expect to see C and D in the middle of two neighbours (B and E) using any method – hence we conclude that B and E are not really neighbours and in the second method some other host is spoofing replies using E's address.

Of the 21576 destinations that were classed as reached using the TCP method from cbg-uk, we used this reasoning to infer 221 replies that were spoofed. Of these, 158 replies were SYN/ACK packets, 61 were RST/ACK packets, and two were time exceeded messages. The TCP packets indicate the presence of an application-level gateway early in the path; because these replies are spoofed at least two hops prior to the destination, the TCP method will miss potentially valuable IP links at the edge of the network. In addition, because of our conservative requirement to visit at least two additional hops from a common parent hop before inferring a reply as spoofed, these values are likely lower bounds of spoofing activity. The other methods were inferred to spoof much less; with the UDP-Paris method 41 messages purporting to be from the destination were received, while there were only 14 such messages with the ICMP-Paris method.

3.1.3 Firewall placement

Related to the problem of reachability is the detection of firewalls along the probed path. Figure 3 shows the additional hops discovered by cbg-uk when one probe method reaches a destination but another does not, either because

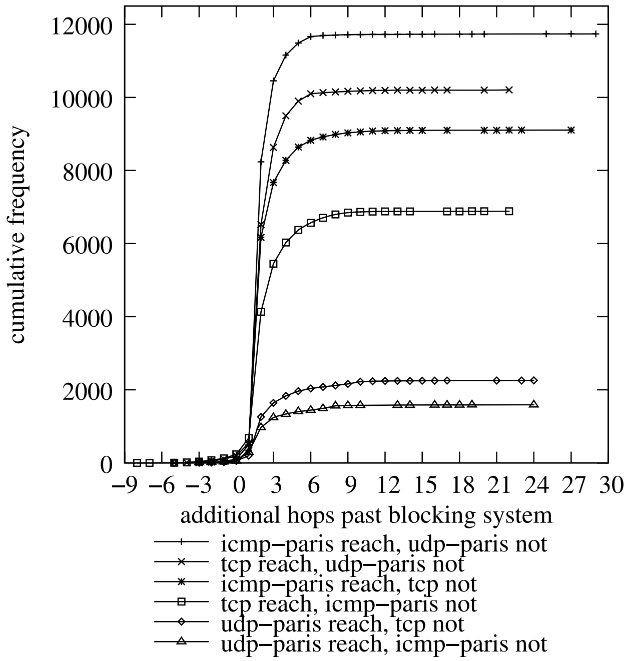

Figure 3: Cumulative frequency of additional hops past blocking system to reachable destination from cbg-uk. Most firewalls were observed to discard packets two hops from its destination.

Table 8: Responses to last-ditch probes collected at cbg-uk

	UDP-Paris	TCP port 80	ICMP-Paris
Gaplimit	203408	194371	189012
Replies to probe #1	1241	2152	1795
Replies to probe #2	320	373	380
Total	1561	2525	2175
Response rate	0.8%	1.3%	1.2%
Time exceeded	331	329	359
Prohib filter	135	166	226
Prohib host	6	2	0
Unreach net	32	35	33
Unreach host	719	794	790
Unreach port	338	12	2
Echo reply	–	–	765
TCP Syn-ack	–	370	–
TCP Rst-ack	–	807	–
Other TCP	–	10	–

an ICMP destination unreachable message was sent, or because no reply was received for five consecutive hops. In all cases in which one method reaches a destination but another fails, the most common number of additional hops discovered, including the destination, is two. However, this distribution of additional hops is long tailed; in most cases the number of additional hops discovered does not level out until after seven.

3.1.4 Results of last-ditch probing

When five consecutive unresponsive hops were encountered, traceroute halted due to the gaplimit condition. In this condition, we send up to two last-ditch probes with an IP TTL of 255 in order to determine if there is anything further in the path that may have responded if hop-by-hop probing had continued. Table 8 shows the response rate to the last-ditch probes and classifies the types of responses received for three probe methods at cbg-uk. Overall, the last-ditch probes have a low response rate, although the last-ditch response rate to the TCP method is 1.7 times that of the UDP-Paris method.

3.1.5 Similarity of paths observed

This section considers similarity of paths collected by each method for all vantage points. Pervasive deployment of load balancing [15] implies expected variation in IP paths collected to each destination. Figure 4 shows the uniqueness of a subset of paths captured at each vantage point for which the destination was reached and all intermediate hops inferred for all three probe methods. The total number of complete paths considered depends on the vantage point; bcn-es had the fewest considered with 1732, while syd-au had the most with 7814; in our data, the vantage point has more impact on the number of complete IP paths than the probe method does. As roughly the same number of destinations were reached from each vantage point, the variation is mostly due to the prevalence of unresponsive hops unique to each vantage point.

Figure 4 also suggests that the uniqueness of the paths depends mostly on the vantage point and the forwarding policies of its upstream paths. For example, the number of traces where the same IP path was inferred by all three methods ranges from 2.5% at nrt-jp to 48.3% at syd-au. While there is significant variance among vantage points, one trend emerges. Of the IP paths that were not the same for all three methods, the largest intersection is between the ICMP-Paris and TCP methods for all vantage points except nrt-jp. If per-flow load balancing were applied equally to TCP, UDP, and ICMP packets, the distribution of dissimilar paths would be equal across the three techniques.

Figure 5 shows the intersection of IP links per method for each vantage point. Depending on the vantage point, between 238k and 266k IP links were inferred, with 76% to 78% of the IP links inferred by all three probe methods. Despite reaching the fewest destinations, UDP-Paris allows inference of more links than any other method, while ICMP-Paris allows inference of the most links that are not inferred by the other methods. 97% of IP links are inferred by combining ICMP-Paris and UDP-Paris. The fact that TCP infers the fewest links reinforces the earlier result where application level gateways are inferred to intercept TCP connection requests at a greater rate, and tends to suggest some routers may have different forwarding policies depending on the transport protocol. We explore this further in Section 3.1.7.

Figure 6 shows the intersection of AS links inferred from each vantage point. Depending on the vantage point, between 17461 and 18065 unique AS links were inferred using the three methods, with 82% in common across the three methods. While UDP-Paris inferred the greatest number of IP links, it inferred the fewest AS links, suggesting that most of the additional IP links inferred with UDP-Paris are internal to the ASes inferred. The IP links UDP-Paris misses

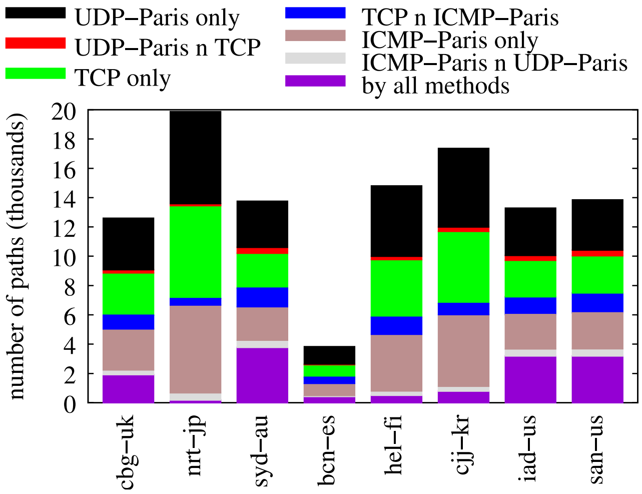

Figure 4: Uniqueness of complete IP paths by the combinations of methods that see them for the random routable address list. The vantage point has more impact than the method on the number of complete IP paths observed and their uniqueness to each method.

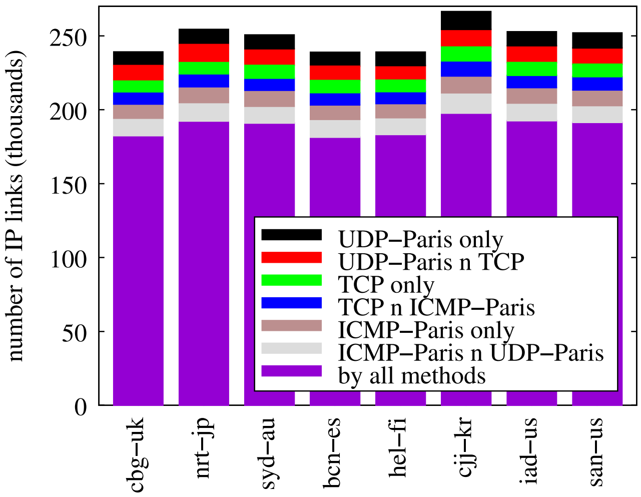

Figure 5: Uniqueness of IP links by the combinations of methods that see them for the random routable address list. UDP-Paris infers the most IP links despite reaching the fewest destinations. Looking at cbg-uk: UDP-Paris 89.1%, ICMP-Paris 88.7%, and TCP 87.4%.

due to reaching the fewest number of destinations (Figure 1) restricts its inference of AS topology. 99% of the AS links inferred using our data are inferred by combining the AS paths inferred with ICMP-Paris and TCP.

3.1.6 *Probe generation*

We examined the workload that various probing methods generated in terms of the number of probes sent to the network. Since scamper runs at a constant packets per second rate regardless of probing method, the probe count is a reasonable approximation of the time a measurement will take. For cbg-uk, ICMP-Paris sent the fewest packets with 6,943,071, followed by TCP at 7,033,384 and UDP-Paris with 7,122,459. ICMP-Paris had a slight edge, sending 2.5% fewer packets than UDP-Paris.

3.1.7 *Enumeration of all hops per method*

As the UDP-Paris, ICMP-Paris, and TCP probe methods reveal different sets of IP links, we became curious about the extensiveness of forwarding policies for different protocol types. To investigate further we implemented a per-flow load-balancer traceroute in scamper similar to that of Augustin, *et al.* [15] to enumerate all links that could be observed between a source and a destination. The three methods implemented were UDP and TCP traceroute methods which vary the source port but keep the destination port constant, and an ICMP traceroute method which varies the ICMP checksum. In the UDP and TCP cases, we kept the destination port constant in order to receive consistent treatment by firewalls in the path.

We probed with the three methods from san-us to 500 addresses chosen with the procedure of Section 2.4.1. We used the same stochastic approach as Augustin, *et al.* [15] to send sufficient probes with varying flow identifiers to reach 99% confidence we had observed all links forward from a hop. We did not probe beyond any unresponsive hops or load balancers we inferred to forward on a per-packet basis.

Figure 7 shows the intersection of IP links observed by each method. In total, we enumerated 3899 IP links; 68% of these links were in common for all three methods, rising to 82% in common between TCP and UDP. In total, we detected 619 load-balancer interfaces; the next-hop for 222 (36%) of these varied with IP protocol. This confirms our hypothesis that the sequence of hops visited will vary based on the IP protocol used.

3.2 Website probing results

Table 9 lists the halt reasons for all methods using the website list from san-us. At the time of probing, 95.3% of webservers were reachable with tcptraceroute. However, less than half of these webservers were reachable when probing with a UDP method, and only three-quarters of them with an ICMP method. In these cases, the proportion of traceroutes that halt after five consecutive unresponsive hops increases significantly. That is, a firewall was silently discarding traceroute probes from reaching the intended destination.

3.3 Router probing results

Table 10 lists the halt reasons for all methods using the router list from san-us. As with the website list, most of the targets should be up; in this case we expect them to be active because we recently received an ICMP time exceeded message from them. ICMP-based methods are most effective for this router destination list, reaching approximately 84% of the list. The UDP and TCP probe methods both reach approximately 68% of the routers in the list, with the UDP methods reaching slightly more than the TCP method.

4. ARK DATA ANALYSIS

In this section, we analyse the reachability and loop rates observed for UDP and ICMP-Paris traceroutes in the first seven months of IP topology data collected on Ark [18].

Table 9: Traceroute method halt reasons for top 500 website address list. 422 addresses were probed.

	Reached	ICMP-unreach	Loop	Gaplimit
UDP	182 (43.0%)	18 (4.3%)	14 (3.3%)	209 (49.4%)
UDP-Paris	182 (43.0%)	15 (3.5%)	10 (2.4%)	216 (51.1%)
UDP-Paris DNS	196 (46.3%)	11 (2.6%)	10 (2.4%)	206 (48.7%)
ICMP	323 (76.4%)	10 (2.4%)	11 (2.6%)	79 (18.7%)
ICMP-Paris	324 (76.6%)	8 (1.9%)	9 (2.1%)	82 (19.4%)
TCP port 80	404 (95.5%)	0	9 (2.1%)	10 (2.4%)

Table 10: Traceroute method halt reasons for router address list. 2000 addresses were probed.

	Reached	ICMP-unreach	Loop	Gaplimit
UDP	1383 (69.2%)	117 (5.8%)	34 (1.7%)	466 (23.3%)
UDP-Paris	1399 (70.0%)	117 (5.8%)	16 (0.8%)	468 (23.4%)
UDP-Paris DNS	1364 (68.2%)	119 (6.0%)	16 (0.8%)	501 (25.1%)
ICMP	1690 (84.5%)	118 (5.9%)	27 (1.4%)	165 (8.2%)
ICMP-Paris	1702 (85.1%)	116 (5.8%)	16 (0.8%)	166 (8.3%)
TCP port 80	1342 (67.1%)	133 (6.7%)	14 (0.7%)	511 (25.6%)

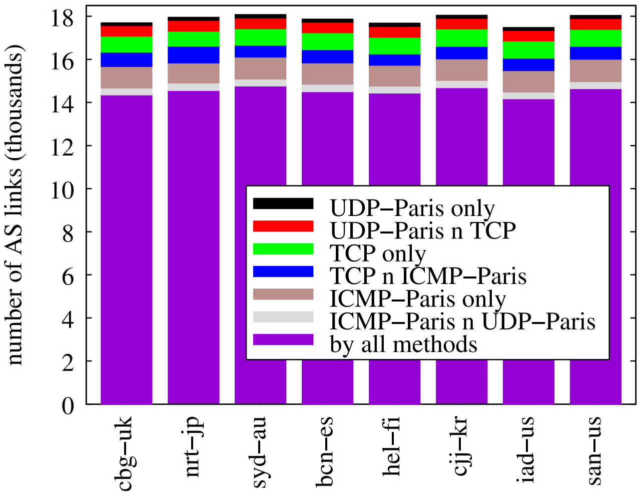

Figure 6: Uniqueness of inferred AS links by the combinations of methods that see them for the random routable address list. ICMP-Paris infers the most AS links due to reaching the most destinations. Looking at cbg-uk: ICMP-Paris 92.3%, TCP 92.2%, and UDP-Paris 87.2%

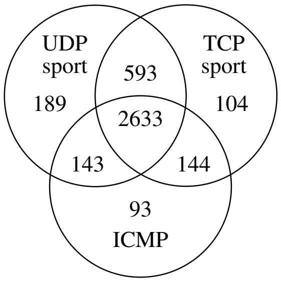

Figure 7: Intersection of IP links inferred for 500 random addresses using three load-balancer traceroute techniques.

4.1 Ark background

Ark is CAIDA's newest active measurement infrastructure, the next generation of the skitter infrastructure CAIDA operated for nearly a decade. The distinguishing feature of Ark is its focus on *coordination*. Coordination, broadly speaking, is concerned with planning, executing, and controlling an ensemble of distributed computations [26, 27]. Coordination is the meta-activity that surrounds a computation.

To facilitate coordination, Ark provides a new implementation, called Marinda, of the well-known tuple-space coordination model first introduced by David Gelernter in his Linda coordination language [28, 29]. A tuple space is a distributed shared memory combined with operations, including an operation to retrieve tuples by simple pattern matching.

4.2 Dataset

We use Marinda to perform coordinated large-scale topology measurements on the Ark infrastructure using a process we call *team probing*. In team probing, monitors dynamically divide the work of probing to a random destination in every routed /24. This parallelisation allows us to obtain a traceroute measurement to all routed /24's in a relatively short period of time: about 48 hours for a team of 13 monitors probing 7 million /24's at 100pps. One pass through every routed /24 is called a *cycle*, and a team continues to the next cycle when it has finished probing all /24's. In a given cycle, each /24 receives a traceroute from only a single monitor per team. This probing approach differs from our past skitter measurements, in which each monitor probed every destination on each cycle.

The Ark project currently has two teams active. Each team independently probes the same set of routed /24's, but to different random destinations within each /24, and typically to different /24's at any given moment in time. We probe the set of /24's themselves, as well as the destinations within each /24, in random order[2] to avoid patterns in

[2] The random ordering of /24's is the same across all cycles.

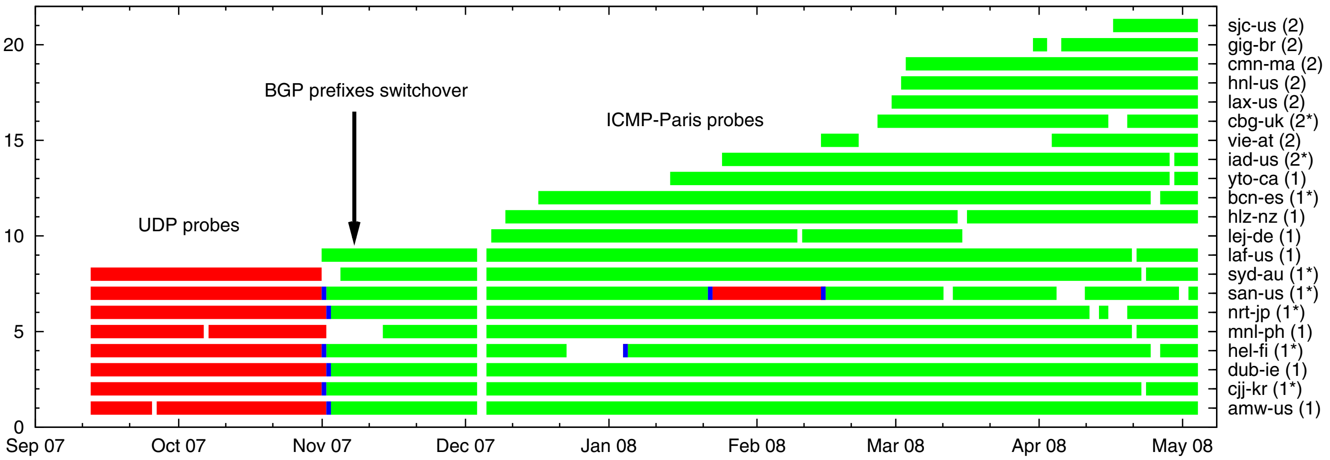

Figure 8: IPv4 Routed /24 Topology Dataset collection on all Ark monitors. The bars indicate the availability of traceroute data for each monitor. The probing method was UDP (red/dark) until around Nov 2, 2007, when it changed to ICMP-Paris (green/light). The san-us monitor also probed with UDP between Jan 22 and Feb 15, 2008. We updated the BGP prefixes used to dynamically compute the destination list around Nov 7, 2007. Monitors are grouped into teams 1 and 2 as indicated in parentheses; each team independently probes the entire set of routed /24's. To support this study, we ran multiple traceroute methods on the monitors marked with asterisks.

probing that may elicit complaints. Randomness in probing also more broadly distributes measurement traffic topologically in order to reduce potential bias or measurement gaps caused by packet loss or transient routing problems on common links.

The end product of team probing is the CAIDA IPv4 Routed /24 Topology Dataset [30], which is available for researchers to download. We have collected this data from September 13, 2007 to present, using between 8 and 21 monitors located worldwide (see Figure 8). In the first 7 months (up to April 30, 2008), we collected 748 million traceroutes and completed 107 cycles.

4.3 Dataset pruning

For the purposes of comparing UDP and ICMP-Paris traceroutes, we exclude a small portion of the dataset that we know is non-representative in some way. In particular, we exclude all traces collected by vie-at and gig-br, and dub-ie traces collected between Apr 10–30, 2008. This excludes 22.3 million traces, or about 3% of the original 748 million. All tables and analyses in the remainder of this section are based on this pruned dataset.

For the excluded traces, the loop rate is far higher than for the other Ark monitors, due to MPLS-induced loops in an immediate upstream provider serving these monitors. These loops are caused by improper decrementing of the IP TTL by MPLS egress routers. These loops start at hop 3 for vie-at, hop 4 for gig-br, and hop 5 for dub-ie. For dub-ie, these loops affect as much as 28% of the traces gathered on a given day. Determining the loop rate is harder for vie-at and gig-br, since on these monitors we run scamper with a special option to probe past these loops.[3] We can say that for one set of 50k destinations, vie-at had a loop rate of 50%.

Table 11: Traceroute method halt reasons for the Routed /24 Dataset.

	UDP		ICMP-Paris	
All traces	111.98M		613.90M	
Reached	4.25M	3.79%	45.67M	7.44%
ICMP-Unreach	11.52M	10.29%	63.62M	10.36%
Loop	5.97M	5.33%	21.23M	3.46%
Gaplimit	90.24M	80.59%	483.38M	78.74%
Reached	4.25M		45.67M	
Time exceeded	745k	17.54%	4.03M	8.83%
Unreach port	3.50M	82.46%	31k	0.07%
Echo reply	–		41.60M	91.10%

4.4 UDP vs. ICMP-Paris analysis

Table 11 gives the breakdown of halt reasons for this dataset. The reachability rate is 3.79% for UDP and 7.44% for ICMP-Paris, which are both lower in this dataset than for the earlier random routable address list, which reaches an average of 5.1% of UDP and 9.0% of ICMP-Paris traces. This result is consistent with the fact that this list is constructed by subdividing the routable prefixes into smaller chunks, many of which are not occupied.

This dataset shows the same noticeable difference in reachability between UDP and ICMP-Paris as in the random routable address list. The difference with this dataset is more pronounced, as ICMP-Paris has 1.96 times the reachability rate of UDP, whereas earlier the ICMP-Paris rate reached 1.75 times more destinations than UDP.

As expected, the ICMP-Paris loop rate of 3.46% is lower than the UDP rate of 5.33%. Both loop rates are lower than those observed in the random routable address list, which has an average loop rate of 7.8% for UDP and 6.3% for ICMP-Paris.

[3] We used the '-L 1' option. The difficulty arises because this scamper option will lead to the under-counting of loops in paths that do not cross these nearby MPLS loops.

Table 12: Traceroute halt reasons for the Routed /24 Dataset for traces collected before and after the set of BGP prefixes used to generate destinations was updated around Nov 7, 2007. The UDP column describes san-us traces for Sep 13–Nov 1, 2007 (old) and Jan 22–Feb 15, 2008 (new). The ICMP-Paris column describes traces from 7 monitors collecting data on Nov 1–6, 2007 (old) and the same 7 monitors on Nov 10–13, 2007 (new).

	UDP		ICMP-Paris	
Old prefixes				
Traces	13.38M		7.77M	
Reached	505k	3.78%	531k	6.84%
ICMP-Unreach	1.54M	11.51%	924k	11.89%
Loop	703k	5.26%	250k	3.22%
Gaplimit	10.63M	79.46%	6.06M	78.04%
New prefixes				
Traces	6.33M		7.63M	
Reached	260k	4.10%	546k	7.16%
ICMP-Unreach	619k	9.78%	846k	11.09%
Loop	320k	5.05%	268k	3.52%
Gaplimit	5.13M	81.07%	5.97M	78.23%

4.5 Impact of BGP prefixes

Across two intervals we used two different sets of BGP prefixes to generate destination addresses for probing. The first set we used from Sep 13 to about Nov 7, 2007, and the second set thereafter. We obtained the first set from a single RouteViews BGP snapshot taken on Nov 14, 2006 (during the early stages of Ark development). This prefix list provided 6.45 million /24's for probing after applying our do-not-probe list. The second set of prefixes was derived from the union of seven RouteViews snapshots taken Oct 14–20 2007, one per day. We excluded prefixes that appeared in only one or two snapshots. This provided 7.02 million /24's after applying our do-not-probe list and bogon filtering.

Table 12 lists the reachability statistics for a subset of UDP and ICMP-Paris traces collected before and after the BGP prefix list switch-over on Nov 7th. Both methods show small but noticeable trends. With the new prefix list, reachability increases slightly, about 8.5% for UDP and 4.7% for ICMP-Paris. The gaplimit rate also increases slightly, which seems unsurprising, given the additional half a million /24's in the new list. Overall, the change in prefix sets did not have a dramatic impact on reachability, so the significant differences in reachability between UDP and ICMP-Paris shown in Table 11 are indicative of real differences rather than solely artifacts of the change in BGP prefix list.

4.6 Impact of vantage points

We now examine the amount of variation that exists across vantage points by looking at the reachability statistics for the 19 monitors that collected ICMP-Paris traces on and after Nov 10, 2007. As we can see from Table 13, there is little variation in the percentage of reached destinations despite the widely differing geographic and topological distribution of monitors. This suggests that the main cause of variation for reachability (for a single method) may be something close to the destination, such as the policy used in firewalls. Or said differently, this result suggests reachability is determined more by the choice of the destination than the choice of the vantage point, even though different vantage points can encounter significantly different numbers of unreachable responses and loops.

Table 13: Traceroute halt reasons for the Routed /24 Dataset for ICMP-Paris traces collected on Nov 10, 2007 and after (599M total traces). This shows the variation across 19 monitors.

	min	max	avg	stddev
Reached	7.15%	7.63%	7.47%	0.11%
ICMP-Unreach	6.89%	12.35%	10.25%	1.11%
Loop	3.17%	4.28%	3.44%	0.28%
Gaplimit	76.86%	81.98%	78.84%	1.10%

5. RELATED WORK

Although the field of Internet topology research is at least a decade old, with 317 citations in Citeseer, we know of no other work that has formally compared the myriad of available forward IP path probing methods. Recently Heidemann, *et al.* compared ICMP and TCP methods in a ping-based census of the IPv4 address space [22], and although they did not attempt to probe entire forward paths, they gave up early on using TCP after learning it triggered more abuse complaints. We note that during our data collection, we received a single complaint which was in response to TCP probes sent to routers in the router address list.

Bush, *et al.* [31] uses traceroute (but does not specify which method) to test the reachability of new address space. Rocketfuel [32] trades breadth of topology collection for more detail within individual networks. Recent revisions of Paris traceroute [15] make the same trade: rather than macroscopic collection of Internet paths, they focus on enumerating all possible IP paths between a source and a destination – which requires an order of magnitude more probes to accomplish.

Hubble [33] uses ongoing as well as targeted pings and traceroutes to diagnose reachability problems to edge networks. Traceroute paths are also used to find locations (router or AS) of reachability problems. Because low end-host responsiveness reduces the utility of active probing, they base their reachability diagnosis on being able to reach the origin AS of each announced BGP prefix.

6. CONCLUSION

Macroscopic Internet topology data is vital to researchers trying to develop or empirically validate models of Internet topology structure and growth. In this paper, we explored the utility of six traceroute probe methods in three scenarios. Our results indicate that researchers should be careful when selecting a probing method, or a topology dataset. While ICMP-Paris demonstrated superiority in metrics of reached destinations for both random destination and router scenarios, as well as in AS links inferred, it missed IP links inside each AS which UDP-Paris found, likely due to ICMP not being load-balanced as widely. We also demonstrated significant variation in the IP paths inferred; despite UDP-Paris reaching the fewest number of destinations, it inferred the most IP links, demonstrating its utility for obtaining the most detailed intra-AS topology.

Our experiments also showed that using more than one probe method will improve coverage of both AS and IP links. Using more than one probe method also allows the integrity of the IP paths to be tested in the face of systems which spoof their source address, and facilitates inference of next-hop selection based on the protocol type.

Acknowledgements

We thank the anonymous reviewers, shepherd Simon Leinen, and kc claffy. This work is supported by National Science Foundation grant CNS 0551542, DHS contract S/N 660001-08-C-2029, and the New Zealand Foundation for Research Science and Technology (FRST) contract UOWX0705.

7. REFERENCES

[1] Van Jacobson. traceroute. `ftp://ftp.ee.lbl.gov/traceroute.tar.gz`.

[2] Ehud Gavron. NANOG traceroute. `http://momo.lcs.mit.edu/traceroute/traceroute.c`.

[3] J. Mogul and S. Deering. Path MTU discovery. RFC 1191, November 1990.

[4] Michael Toren. tcptraceroute. `http://michael.toren.net/code/tcptraceroute/`.

[5] Dan Kaminsky. paratrace. `http://www.doxpara.com/read.php/docs/paratrace.html`.

[6] Brice Augustin, Xavier Cuvellier, Benjamin Orgogozo, Fabien Viger, Timur Friedman, Matthieu Latapy, Clémence Magnien, and Renata Teixeira. Avoiding traceroute anomalies with Paris traceroute. In *IMC '06: Proceedings of the 6th ACM SIGCOMM conference on Internet measurement*, pages 153–158, Rio de Janeiro, Brazil, October 2006.

[7] Alexa. `http://www.alexa.com/`.

[8] R-fx Networks. Advanced policy firewall (APF). `http://www.r-fx.ca/downloads/apf-0.9.6-3.tar.gz`.

[9] Bradley Huffaker, Daniel Plummer, David Moore, and k claffy. Topology discovery by active probing. In *2002 Symposium on applications and the Internet (SAINT 2002)*, pages 90–96, Nara City, Japan, January 2002.

[10] S. Savage. Sting: a TCP-based network measurement tool. In *Proceedings of USITS '99: The 2nd USENIX Symposium on Internet Technologies and Systems*, pages 71–79, Boulder, CO, October 1999.

[11] Rob Sherwood and Neil Spring. Touring the Internet in a TCP sidecar. In *IMC '06: Proceedings of the 6th ACM SIGCOMM conference on Internet measurement*, pages 339–344, Rio de Janeiro, Brazil, October 2006.

[12] Matthew Luckie. [patch] sys/netinet/udp_usrreq.c modifies received udp checksum, May 2007. `http://www.freebsd.org/cgi/query-pr.cgi?pr=112471`.

[13] Matthew Luckie. scamper. `http://www.wand.net.nz/scamper/`.

[14] P. Mockapetris. Domain names - implementation and specification. RFC 1035, November 1987.

[15] Brice Augustin, Timur Friedman, and Renata Teixeira. Measuring load-balanced paths in the Internet. In *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, pages 149–160, San Diego, California, USA, October 2007.

[16] Routeviews sh ip bgp snapshots. `http://archive.routeviews.org/oix-route-views/`.

[17] Z. Morley Mao, Jennifer Rexford, Jia Wang, and Randy Katz. Towards an accurate AS-level traceroute tool. In *Proc. ACM SIGCOMM*, pages 365–378, Karlsruhe, Germany, September 2003.

[18] Young Hyun. Archipelago measurement infrastructure. `http://www.caida.org/projects/ark/`.

[19] Patrick Verkaik. rv2atoms-0.4. `http://www.caida.org/funding/routing/atoms/download/rv2atoms-0.4/`.

[20] Team Cymru. Bogon bit notation list v4.0 25 JAN 2008. `http://www.cymru.com/Documents/bogon-bn.html`.

[21] A. Medina, M. Allman, and S. Floyd. Measuring the evolution of transport protocols in the Internet. *ACM SIGCOMM Computer Communication Review*, 35(2):37–52, April 2005.

[22] John Heidemann, Yuri Pradkin, Ramesh Govindan, Christos Papadopoulos, Genevive Bartlett, and Joseph Bannister. Census and survey of the visible Internet (extended). ISI-TR 2008-649, USC/Information Sciences Institute, February 2008.

[23] J. Postel. Internet control message protocol. RFC 792, September 1981.

[24] Ramesh Govindan and Hongsuda Tangmunarunkit. Heuristics for Internet map discovery. In *INFOCOM 2000*, pages 1371–1380, Tel-Aviv, Israel, Mar 2000.

[25] Jean-Jacques Pansiot and Dominique Grad. On routes and multicast trees in the Internet. *ACM SIGCOMM Computer Communication Review (CCR)*, 28(1), Jan 1998.

[26] David Gelernter and Nicholas Carriero. Coordination languages and their significance. *Commun. ACM*, 35(2):97–107, 1992.

[27] Sascha Ossowski and Ronaldo Menezes. On coordination and its significance to distributed and multi-agent systems. *Concurrency and Computation: Practice and Experience*, 18(4):359–370, 2006.

[28] David Gelernter. Generative communication in linda. *ACM Trans. Program. Lang. Syst.*, 7(1):80–112, 1985.

[29] Nicholas Carriero and David Gelernter. *How to write parallel programs: a first course.* MIT Press, Cambridge, MA, USA, 1990.

[30] CAIDA IPv4 routed /24 topology dataset. `http://imdc.datcat.org/collection/1-0360-J`.

[31] Randy Bush, James Hiebert, Olaf Maennel, Matthew Roughan, and Steve Uhlig. Testing the reachability of (new) address space. In *Sigcomm workshop on Internet network management*, August 2007.

[32] N. Spring, R. Mahajan, and D. Wetherall. Measuring ISP topologies with Rocketfuel. In *Proceedings of ACM/SIGCOMM '02*, pages 133–145, Pittsburgh, PA, August 2002.

[33] Ethan Katz-Bassett, Harsha V. Madhyastha, John P. John, and Arvind Krishnamurthy. Studying black holes in the Internet with Hubble. In *Networked Systems Design and Implementation (NSDI)*, Apr 2008.