Measuring Load-balanced Paths in the Internet

Brice Augustin, Timur Friedman, and Renata Teixeira
Laboratoire d'Informatique de Paris 6 (LIP6)
Université Pierre et Marie Curie and CNRS

ABSTRACT

Tools to measure internet properties usually assume the existence of just one single path from a source to a destination. However, load-balancing capabilities, which create multiple active paths between two end-hosts, are available in most contemporary routers. This paper proposes a methodology to identify load-balancing routers and characterize load-balanced paths. We enhance our traceroute-like tool, called Paris traceroute, to find all paths between a pair of hosts, and use it from 15 sources to over 68 thousand destinations. Our results show that the traditional concept of a single network path between hosts no longer holds. For instance, 39% of the source-destination pairs in our traces traverse a load balancer. Furthermore, this fraction increases to 70% if we consider the paths between a source and a destination network.

Categories and Subject Descriptors: C.2 [Computer Communication Networks]: Network Operations; Network Architecture and Design

General Terms: Measurement.

Keywords: traceroute, load balancing, multipath, path diversity.

1. INTRODUCTION

The traditional model of the internet assumes just one single path between a pair of end-hosts at any given time. Internet applications, network simulation models, and measurement tools work under this assumption. However, most commercial routers have load balancing capabilities [1, 2]. If network administrators turn on this feature, then a stream of packets from a source to a destination will no longer follow a single path. This paper reports on a measurement study of load-balanced paths in the internet, which should prompt the research community to revisit the concept of an "internet path".

Load balancing routers (or *load balancers*) use three different algorithms to split packets on outgoing links: *per destination*, which forwards all packets destined to a host to the same output interface (similar to the single-path destination-based forwarding of classic routing algorithms, but this technique assigns each IP address in a prefix to an outgoing interface); *per flow*, which uses the same output interface for all packets that have the same *flow identifier* (described as a 5-tuple: IP source address, IP destination address, protocol, source port, and destination port); or *per packet*, which makes the forwarding decision independently for each packet (and which has potentially detrimental effects on TCP connections, as packets from the same connection can follow different paths and be reordered). Our earlier work [3] showed that it is possible to control the paths that packets take under per-flow load balancing by controlling the flow identifiers in packet headers. Our tool, *Paris traceroute*, controls paths in just this way.

This paper uses Paris traceroute to perform the first measurement study of load-balanced paths in the internet. We make the following contributions:

1. **A tool to expose load-balanced paths**. Paris traceroute's *Multipath Detection Algorithm* (*MDA*) finds, with a high degree of confidence, all paths from a source to a destination under per-flow and per-packet load balancing. We extend it here to cover per-destination load balancing.

2. **A characterization of load-balanced paths between 15 sources and over 68 thousand destinations.** We quantify the load-balanced paths observed from the RON nodes [4] to a large number of destinations. In our data set, the paths between 39% of source-destination pairs traverse a per-flow load balancer, and 70% traverse a per-destination load balancer. We characterize these paths in terms of their length, width, and asymmetry.

3. **A methodology to measure RTTs of load-balanced paths.** RTT measurements take into account the delays on both the forward and the return paths, and therefore could be affected by a load balancer in any of the paths. We develop a technique to also maintain the flow identifier on the reverse path for a more accurate comparison of RTTs.

This paper proceeds as follows. After a discussion of previous work in Sec. 2, Sec. 3 presents our tool to measure load-balanced paths under each type of load balancer. Sec. 4, describes our measurement setup and characterization metrics. Sec. 5 characterizes the load balancers found in our

traces, and Sec. 6 studies the properties of load-balanced paths. Sec. 7 characterizes round-trip times under per-flow load balancing. Sec. 8 ends this paper with a discussion of the implications of our findings.

2. RELATED WORK

Early work on path diversity in the internet [5, 6] looked at the known topology of the large internet service provider (ISP) Sprint and the paths between points of presence (PoPs) in Sprint's network. It found that between any given pair of PoPs there were typically several link-disjoint and several PoP-disjoint paths. It also looked at topologies inferred from traceroute-style probing conducted by Rocketfuel [7] and CAIDA [8], concluding that, while there is evidence of significant path diversity in the core of the network, the measurements are particularly sensitive to errors that were inherent to active probing techniques at that time. Furthermore, when looking at path diversity in the router-level graph, the measurements are sensitive to insufficiencies in alias resolution techniques, which infer router-level graphs from IP-level information. The traceroute errors, of missing links and the inference of false links, have since been largely corrected by Paris traceroute, which we use here. We work purely at the IP-level, making no attempt to resolve the router-level graph. In this work, we observe the actual load-balanced paths taken by packets, rather than looking at the overall topology as an undirected graph in which packets could take any imaginable path.

A typical ISP builds redundancy into its physical infrastructure. To use the infrastructure efficiently, the ISP will split traffic load across multiple links, which introduces much of the path diversity that we measure here. The research community has looked at the question of how best to design load-balancing routers, for instance to adaptively split the traffic according to network conditions [9, 10, 11, 12]. We have not systematically looked for adaptive load balancing, but our familiarity with our own data leads us to believe that most current routers use a static mapping of flows to load-balanced paths. Other studies focus on the network operator's interest in path diversity. Giroire et al. [13] show how to exploit an ISP's underlying physical diversity in order to provide robustness at the IP layer by having as many disjoint paths as possible.

Commercial interests guide today's internet routing policies in ways that often yield inferior end-to-end paths, as measured by delay, loss rate, or bandwidth. Savage et al. [14] demonstrated that alternate paths, taking two or more end-to-end hops between hosts, could often outperform the default direct paths provided by the internet. Andersen et al. [4] have since proposed *RON*, an overlay network that exploits this insight. As opposed to looking at the diversity that can be obtained by composing multiple end-to-end paths, our work examines the diversity that exists in what has previously been regarded as individual end-to-end paths.

3. MEASURING LOAD-BALANCED PATHS

This section describes the Multipath Detection Algorithm (MDA), which Paris traceroute uses to discover load-balanced paths.[1] Sec. 3.1 describes our prior work [15] on enumerating all paths between a source and a destination in the pres-

_____
[1]Note that our technique detects load sharing performed by routers. It is not our goal to measure load balancing

ence of per-flow load balancing. Then, Sec. 3.2 introduces a simple extension that allows the MDA to take into account per-destination load balancers.

3.1 The Multipath Detection Algorithm

Our initial work on Paris traceroute [3] largely fixed the problem of the false paths reported by classic traceroute. The problem was that classic traceroute systematically varies the flow identifier for its probe packets. By maintaining a constant flow identifier, Paris traceroute can accurately trace a path across a per-flow load balancer. However, this early version only traced one path at a time.

Our subsequent work [15] suggested a new goal for route tracing: to find the entire set of load-balanced paths from source to destination. We showed that the classic traceroute practice of sending three probes per hop is inadequate to have even a moderate level of confidence that one has discovered load balancing at a given hop. Our Multipath Detection Algorithm (MDA) uses a stochastic approach to send a sufficient number of probes to find, to a given degree of confidence, all the paths to a destination.

The MDA proceeds hop by hop, eliciting the full set of interfaces for each hop. For a given interface r at hop $h-1$, it generates at random a number of flow identifiers and selects those that will cause probe packets to reach r. It then sends probes with those identifiers, but one hop further, in an effort to discover the successors of r at hop h. We call this set of interfaces, $s_1, s_2, \ldots, s_n$ the *nexthops* of r.

If we make the hypothesis that r is the interface of either a per-flow or a per-packet load balancer that splits traffic evenly across n paths, we can compute the number of probes k that we must send to hop h to reject this hypothesis with a degree of confidence $(1-\alpha)\times100\%$. If the MDA does not find n interfaces, it stops. Otherwise, it hypothesizes that there are $n + 1$ nexthop interfaces, and sends additional probes.

To rule out the initial hypothesis that $n = 2$, the MDA, operating at a 95% level of confidence, sends $k = 6$ probes. As we have seen load balancing across as many as 16 load-balanced interfaces, the MDA may ultimately need to send a total of $k = 96$ probes to find all the nexthops of an interface r.

If the MDA discovers more than one nexthop interface, it sends additional probes so as to classify r as belonging to either a per-flow or a per-packet load balancer. It makes the hypothesis of per-packet load balancing and attempts to disprove this by sending a number of probes, all with the same flow identifier that is known to reach r. If two different interfaces respond, the hypothesis is confirmed. Six probes are required, all returning the same nexthop interface, to reject the hypothesis with a 95% level of confidence, and conclude that r belongs to a per-flow load balancer. If no per-packet load balancers are encountered, once the destination is reached, the MDA has effectively enumerated all of the end-to-end paths. If there were per-packet load balancers, the MDA will not be able to discover all of the true paths, but it will nonetheless have found all the interfaces at each hop and been able to trace those parts of the paths that are not affected by per-packet load balancing.

3.2 Extending the MDA

When tracing towards a single destination with the MDA,

_____
at server farms, where dedicated boxes distribute incoming requests to a set of replicated servers.

Paris traceroute is naturally incapable of detecting instances of per-destination load balancing. In Fig. 1, for example, L might be a per-destination load balancer, sending traffic destined for T along the upper path, and traffic for T' along the lower path. When Paris traceroute uses the MDA to trace to T, it only discovers the upper path. We generalize the
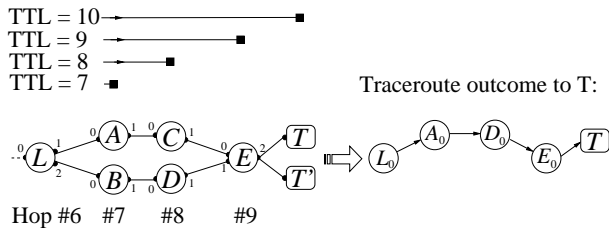
Figure 1: Traceroute and per-destination load balancing

MDA to enumerate all of the paths from a source to a given address prefix rather than simply to a given destination. In this example, the generalized MDA detects both paths, and flags L_0 as the interface of a per-destination load balancer.

We achieve this goal by refining the techniques previously described. When testing the hypothesis that there are n nexthops for an interface r, the MDA initially chooses flow identifiers that differ only in their destination address. It chooses destination addresses that share a long prefix with the destination of interest. Two addresses sharing a prefix longer than /24 are unlikely to have different entries in a core router, so any path differences should purely be the result of load balancing. The MDA initially chooses addresses that share a /29 prefix, allowing the choice of up to 8 different addresses. As before, the MDA sends anywhere from 6 to 96 probes (for the 95% confidence level) one hop past r in order to enumerate its nexthops. If 96 different destination addresses are required, they can all share a /25 prefix. As this nexthop enumeration technique is designed to work when r belongs to a per-destination load balancer, it will *a fortiori* also work when r belongs to a per-flow or a per-packet load balancer.

Then, if the MDA has found two or more nexthop interfaces, it hypothesizes, as before, that r belongs to a per-packet load balancer. For the 95% confidence level, it sends up to 6 probes with the same flow identifier, and rules out the hypothesis only if all 6 probes go to the same nexthop interface. If it rules out per-packet load balancing, then the extended MDA hypothesizes that r belongs to a per-flow load balancer, and again sends up to 6 probes. These all go towards the same destination, but with flow identifiers that nonetheless differ. (The MDA varies the destination port number.) Only if all of these probes go to the same nexthop interface does the MDA conclude that r is a per-destination load balancer.

3.3 Limitations

The Multipath Detection Algorithm may return inaccurate results in the following cases:

MPLS: MPLS represents a challenge for all traceroute-like measurements, because some ISP networks deploy MPLS tunnels in which routers do not necessarily decrement the IP TTL of packets. Under this configuration, the TTL will never expire while in a tunnel and traceroute will observe the path through the tunnel as a single link, causing an underestimation of the length of load-balanced paths. Furthermore, if a load balancer splits traffic across several MPLS paths sharing the same entry and exit points, the MDA will not detect it.

Nonresponding routers: When routers do not respond to probes even after retransmissions, we cannot accurately enumerate a given nexthop set. This is a fundamental limit to traceroute-style measurements, and the amount of load balancing will be underestimated in these instances.

Uneven load balancing: If a load balancer distributes load with nonuniform probability across its nexthop interfaces, the algorithm risks not discovering a low-probability nexthop interface. The solution, if we expect probabilities to be possibly skewed up to some maximum extent, is to send more probes, in order to regain the desired degree of confidence. Despite having seen some examples in which a router does not distribute load evenly, our informal experience tells us that this is rare, and we have not adjusted the MDA to catch all such cases, leading to another small source of under-estimation of load-balanced paths.

Routing changes: Routing changes during a traceroute can lead to the inference of false links. They may cause an overestimation of load balancing, or the incorrect classification of a routing change as per-packet load balancing. Fortunately, routing changes are relatively infrequent [16], especially on the time scale of an individual route trace. The MDA could potentially reprobe a path to try to determine if the route has changed, but we do not currently implement such an extension.

4. METHODOLOGY

This section describes our measurement setup and introduces the metrics we use to characterize load balancing.

4.1 Measurements

We ran our measurements from 15 sources: 13 RON nodes (the other RON nodes were not available), plus a host at our laboratory in Paris and another in Bucharest. Eleven of the sources are in the US, the others in Europe. Table 1 summarizes the locations and upstream providers of each source. Although the sources do not exhibit great geographic diversity (most of them are on the US east and west coasts), they connect to the internet through many different providers.

We used two destination lists. The first, which we call MIT, contains 68,629 addresses. It was generated by researchers at MIT from the BGP table of a router located there. They randomly selected a couple of addresses from each CIDR block of the BGP table, and ran classic traceroute from MIT towards each address. The basis of the MIT list consists of the last responding hop from each trace. From this, they removed addresses that appeared in any of several blacklists, as well as any host regarding which they received complaints during their experiments. We updated this list by adding all our source nodes.

Since the first list doubtless includes targets that are routers or middleboxes rather than end-hosts, we also used a second list for which we were certain that we could trace all the way through the network. This consists of the 500 most popular websites, as reported by the commercial service, Alexa.[2]

[2]See `http://www.alexa.com/site/ds/top_sites?ts_mode=global&lang=none`

We call this the WEB list. We could only use it from the paris node, as the RON acceptable use policy forbids tracing towards arbitrary destinations.

We collected our data over the months of February to April 2007, using Paris traceroute adapted to run in 32 parallel threads of a single process. We limit the overall bandwidth to 200 probes per second. Each thread takes the next address d in the destination list, and uses the MDA to enumerate all of the paths to d. We use the following parameters: 50 ms of delay between each probe sent, abandon after 3 consecutive unresponsive hops, a 95% level of confidence to find the nexthops of an interface. We use UDP probes. We avoided ICMP probes because some per-flow load balancers do not perform load balancing on ICMP packets, thus hiding part of the load-balanced paths. We did not use TCP probes to avoid triggering IDS alarms. We collected data from all 15 sources, but due to disk space restrictions we were able to collect per-destination load balancing data from only 11 of them.

A round towards all destinations in the MIT list takes between 10 and 15 hours, depending upon the source. Our traces with the MIT list, for all sources, cover 9,506 ASes, including all nine tier-1 networks and 96 of the one hundred top-20 ASes of each region according to APNIC's weekly routing table report.[3]

4.2 Metrics

This section describes the metrics we use to characterize load balancing. Fig. 2 illustrates these metrics. This is a real topology we discovered when tracing from a US source, S, towards a Google web server, T. We use the following terminology in the context of IP-level directed graphs generated by the MDA:
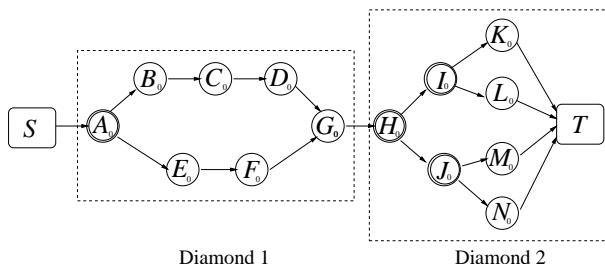
Diamond 1   Diamond 2

Figure 2: Two diamonds in a set of paths to a destination

Load balancer. A node with out-degree $d > 1$ is an interface of a load balancer. For instance, A_0, H_0, I_0 and J_0 are interfaces of load balancers.

Diamond. A diamond is a subgraph delimited by a *divergence point* followed, two or more hops later, by a *convergence point*, with the requirement that all flows from source to destination flow through both points. Fig. 2 has two diamonds, shown in dashed boxes. Note that this differs from definitions of diamonds we have employed in other work, in which we restricted their length to two hops, or allowed just a subset of flows to pass through them (as between I_0 and T in Fig. 2).

---

[3]APNIC automatically generates reports describing the state of the internet routing table. It ranks ASes per region according to the number of networks announced.

Diamond width. We use two metrics to describe the width of a diamond. The *min-width* counts the number of link-disjoint paths between the divergence and convergence points. This gives us a lower bound on the path diversity in a diamond. For instance, diamonds 1 and 2 in Fig. 2 have the same min-width of 2, although diamond 2 appears to offer greater diversity. Thus, in addition, we also use the *max-width* metric, which indicates the maximum number of interfaces that one can reach at a given hop in a diamond. In our example, diamond 1 has a max-width of 2 whereas diamond 2 has a max-width of 4.

Diamond length. This is the maximum number of hops between the divergence and convergence points. In our example, diamond 1 has length 4 and diamond 2 has length 3.

Diamond symmetry. If all the parallel paths of a diamond have the same number of hops, we say that the diamond is symmetric. Otherwise, it is asymmetric. The *diamond asymmetry* is the difference between the longest and the shortest path from the divergence point to the convergence point. Diamond 1 has an asymmetry of 1, since the longest path has 4 hops and the shortest one has 3 hops. Diamond 2 is symmetric.

5. LOAD BALANCERS

This section characterizes load balancers. We show that per-flow and per-destination load balancing are very common in our traces. This high frequency is due to the fact that per-flow and per-destination load balancers are located in core networks, and thus are likely to affect many paths. We also observe that the majority of load balancing happens within a single network.

5.1 Occurrences of load balancing

Per-destination load balancers are the most common in our traces: the paths between 70% of the 771,795 source-destination pairs traverse a per-destination load balancer. This percentage is still considerable for per-flow load balancers, 39%, but fairly small, only 1.9%, for per-packet load balancers. Our measurements for per-flow and per-packet load balancers had 1,010,256 source-destination pairs in total. (This difference is because our dataset for per-destination load balancers uses only 11 sources, whereas the per-flow and per-packet dataset uses 15 sources). The fraction of per-flow load balancers generalizes the results of our preliminary study [3], in which we found that per-flow load balancing was common from the paris source. This result comes from the widespread availability of load balancing in routers. For instance, Cisco and Juniper routers can be configured to perform any of the three types of load balancing [1, 17, 2]. Even though per-packet load balancing is widely available, network operators avoid this technique because it can cause packet reordering [18].

Table 2 breaks down these results for each source. The frequency of per-flow and per-destination load balancers varies according to the source (from 23% to 80% and from 51% to 95%, respectively), whereas the frequency of per-packet load balancers is more stable across all sources (around 2%). The frequency of per-flow and per-destination load balancers depends on the location and upstream connectivity of the source. For instance, the roncluster1 and speakeasy sources, which are in the same location and have the same upstream connectivity, observe the same fraction of per-flow load bal-

Source	Location	Upstream provider
am1-gblx	Amsterdam, NL	557 ASes
bucuresti	Bucharest, RO	Sprint, Level3 + 6 others
chi1-gblx	Chicago, IL	482 ASes
coloco	Laurel, MD	XO comunications
cornell	Cornell, Ithaca, NY	Level3 + 5 others
digitalwest	San Luis Obispo, CA	NTT, Qwest, Level3, GBLX + 4 others
intel	Berkeley, CA	AT&T
lon1-gblx	London, UK	553 ASes
msanders	CA	NTT, UUNET, GBLX, Level3 + 28 others
nyu	New York, NY	30 AS, most tier-1s, Abilene
paris	Paris, FR	RENATER
roncluster1	Cambridge, MA	Sprint, Level3, Cogent + 2 others
speakeasy	Cambridge, MA	Sprint, Level3, Cogent + 2 others
vineyard	Vineyard Haven, MA	Qwest, Savvis

Table 1: Locations and upstream providers of our measurement sources

Source	per-flow	per-packet	per-dest	any
MIT list				
am1-gblx	23%	2.1%	63%	83%
bucuresti	25%	2.6%	60%	82%
chi1-gblx	27%	2.3%	62%	82%
coloco	27%	2.0%	n.a.	n.a.
cornell	80%	2.0%	74%	97%
cybermesa	25%	1.7%	61%	83%
digitalwest	54%	2.0%	70%	89%
intel	31%	1.9%	95%	97%
lon1-gblx	26%	2.1%	n.a.	n.a.
msanders	39%	2.2%	93%	93%
nyu	64%	1.9%	82%	92%
paris	30%	1.9%	81%	93%
roncluster1	51%	2.8%	51%	89%
speakeasy	51%	2.8%	n.a.	n.a.
vineyard	40%	2.0%	n.a.	n.a.
all	**39.5%**	**2.1%**	**72%**	**89%**
WEB list				
paris	35%	0%	n.a.	n.a.

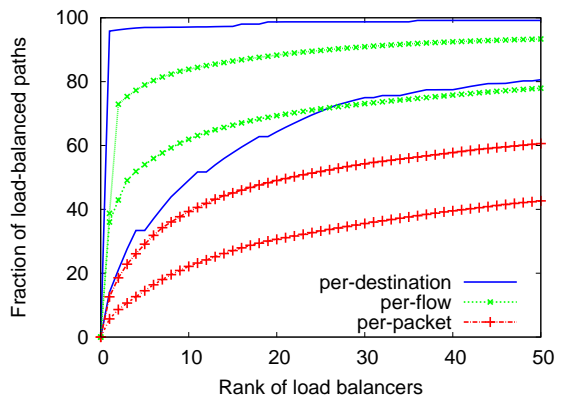**Table 2: Fraction of paths affected by load balancing**

Figure 3: Cumulative fraction of paths affected by the 50 most frequently traversed load balancers. For each type of load balancer, we plotted the two sources which represent the extreme results.

ancers. On the other hand, the frequency of per-packet load balancing depends mostly on the destination list used–always around 2% for the MIT list and zero for the WEB list. Furthermore, it is relatively constant from all our sources, which suggests that per-packet load balancers tend to be close to destinations.

We now study how load balancers affect paths to verify whether there are a few routers responsible for most load balancing. In a typical MIT round, we find from each source around 1,000 distinct per-flow load balancers, 2,500 to 3,000 per-destination load balancers, and 500 per-packet load balancers.

Fig. 3 shows the disparity between the relatively small number of load balancers and the large number of load-balanced paths. It presents the cumulative fraction of paths affected by the 50 most frequent load balancers (per-destination, per-flow and per-packet). Each curve represents the results for one type of load balancer and one source. We

only plotted the results for the two sources having the extreme results. The "per-flow" and "per-destination" load balancers curves show that the top-50 load balancers affect at least 78% of the paths that exhibit load balancing. For instance, the most frequent per-flow load balancer affects 38% of the load-balanced paths in the paris trace. We studied this load balancer in detail and found that it is a Level3 router that connects to RENATER.[4] Similarly, nearly all the paths from the intel source have per-destination load balancing caused by a load balancer in AT&T's network, which is intel's provider.

In contrast, we do not find any predominant per-packet load balancer in our traces. The 50 most frequently found ones affect at most 60% of the paths with per-packet load balancing. We find that the most frequently encountered per-packet load balancers are in Sprint's network. This finding is puzzling given that large ISPs often avoid per-packet load balancing for fear of the negative impact on TCP connections. We studied these load balancers more closely and verified that they are located at peering points between

---

[4]Level3 is a tier-1 ISP and is one of RENATER's providers.

Sprint and other domains. For instance, we found one per-packet load balancer between Sprint and the China169 backbone. The load-balanced interfaces after this load balancer all belong to the same router, and have DNS names such as `sl-china7-5-0.sprintlink.net`, a name that indicates that it is, indeed, at a peering point. We find similar situations at the edges of other tier-1 ISPs. If this is being done purposefully, perhaps it is a situation where high link utilization is especially important, such as when load balancing over a bundle of parallel low capacity links in preference to a single, more expensive, high capacity link. Some other instances may also correspond to misconfigured routers using the per-packet technique instead of the per-flow or per-destination one.

Most of the per-packet load balancers affect just a few paths, because they are located far from the source and close to the destination. Indeed, 85% of those load balancers are located at less than 3 hops from the destination.

5.2 Intra- and inter-AS load balancing

Load-balanced paths can be contained in one autonomous system (AS), which we define as *intra-domain load balancing*, or span multiple ASes, defined as *inter-domain load balancing*. Although forwarding in both cases is done in the same way, the routing mechanism behind them is very different. A router can install multiple intra-domain routes in its forwarding table, because of the equal-cost multipath capability of common intra-domain routing protocols such as IS-IS [19] and OSPF [20]. In this case, the paths will diverge after entering the AS and re-converge before exiting it.

On the other hand, BGP [21], the internet's inter-domain routing protocol, does not allow a router to install more than one next hop for a destination prefix. Given this restriction, there should be no inter-domain load balancing. However, some router vendors now provide BGP-multipath capabilities (for instance, Juniper [22] and Cisco [23]). If two BGP routes for a prefix are equivalent (same local preference, AS-path length, etc.) and the multipath capability is on, then BGP can install more than one next hop for a prefix. Another scenario in which we could observe inter-domain load balancing is when BGP routes are injected into the intra-domain routing protocol. Then, BGP routes would be subject to the OSPF or IS-IS equal-cost multipath mechanism. Injecting BGP routes into intra-domain routing is, we believe, rare, so this scenario should not often arise. However, injecting only the default route(s) to upstream provider(s) is a more practicable scenario which is often used by network operators.

To make the distinction between the two types of load balancing, we need to map each IP address in our traces to an AS. We use a public IP-to-AS mapping service [24]. This service builds its mapping from a collection of BGP routing tables. There are well-known issues with this type of mapping [25], so for one of the traces we manually verified each instance of supposed inter-domain load balancing.

Our automated classification does not consider the convergence or the divergence point of a diamond to label load balancers. In so doing, we avoid false positives (classification of intra-domain load balancing as inter-domain), but may generate false negatives. This technique is important because it is very common that an interface in the boundary between two ASes is numbered from the address space of one AS, but belongs in fact to the other. Fig. 4 illustrates this

scenario. It shows two domains, AS1 and AS2, and a load balancer, L. Square interfaces are numbered from AS1's address space, whereas circular ones belong to AS2's address space. We observe that the interfaces of the link Z-L are numbered from AS1's address space. A traceroute from S to T discovers the "square" interface of L. In this case, we could mistakenly label L as an inter-domain load balancer, because L belongs to AS1 and balances traffic to routers A and B, which belong to AS2. If we ignore the divergence point when computing the AS path in a diamond, then L would be correctly labeled as an intra-domain load balancer in AS2.
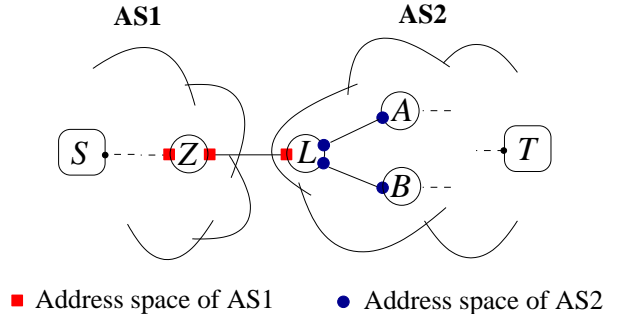
Figure 4: Domain boundary delimitation can be inaccurate.

We also ignore the convergence point because it may not be involved in load balancing. Indeed, the IP-level load-balanced path inferred by Paris traceroute may not correspond to the router-level load-balanced path in the real topology. Fig. 5 illustrates how this phenomenon arises. The left side represents the router-level topology and the right side the IP-level topology inferred with the MDA. The two load-balanced paths merge at two different interfaces of router C. The probing of the upper path reveals C_0 and the lower path reveals C_1. Since we do not conduct alias resolution, we treat those two interfaces as if they belonged to different routers. The consequences are twofold. First, the
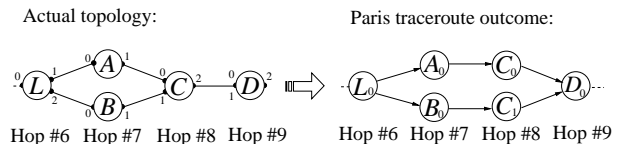
Figure 5: The IP-level multi-path inferred by Paris traceroute may not correspond to the router-level multi-path in the real topology.

length of the measured diamond does not reflect the length of load-balanced path in the router-level topology. Second, we may consider some parts of the topology as being involved in load balancing, whereas they are not. More precisely, the convergence point in the inferred topology, D_0, has actually nothing to do with load balancing. The left side of the figure shows that router D is not part of the real load-balanced path at all. As a result, we may misclassify some diamonds as inter-domain if router D belongs to a different autonomous system. Note that this bias arises because the parallel paths merge through different interfaces of a router. If they merge through a level 2 device such as

a switch, and then connect to a single interface, then the inferred topology maps to the router-level one. Although we do not perform systematic alias resolution on the discovered interfaces, our partial observations of IP Identifiers and DNS names indicated that all the penultimate interfaces of a diamond generally belong to the same router.

The manual verification step is very time consuming, so we only classified intra- and inter-domain load balancers for the paris MIT trace. In most cases, diamonds are created by intra-domain load balancing. From the paris vantage point, 86% of the per-flow diamonds fit in a single AS. Fig. 2 illustrates this case. Diamond 1 exactly spans Savvis's network and diamond 2 spans Google's network. The parallel paths in diamond 1 diverge at the entry point of Savvis's domain and then reconverge before they reach its exit point, because routers selected a single peering link between the two domains. We found rarer cases of diamonds crossing multiple ASes. Most of them involve two ASes, but extremely rare diamonds cross three networks. We found such diamonds in the paths towards 37 destinations. They always involved Level3 as the first domain, peering with Verizon, Bellsouth and some smaller networks like Road Runner. Thus, it seems that very few core networks enable BGP multipath capabilities in their routers.

Most per-destination diamonds are also created by intra-domain load balancers (at least 80% in the paris trace), but we did not conduct any of the manual verification on this dataset.

6. LOAD-BALANCED PATHS

Having described the mechanisms behind load-balanced paths, we now study their properties and characterize them in terms of the widths and lengths of diamonds. The statistics presented here are for the MIT destination list.

6.1 Diamond width

We use two metrics defined in Sec. 4.2 to describe the number of paths available to a source-destination pair: a diamond's *min-width* provides a lower bound and the *max-width* provides an upper bound on this number. If there should be two or more diamonds in a path, we take the lowest min-width and the highest max-width. It is fairly common to see two diamonds in a path: 21% of the pairs have two per-flow diamonds and 24% have two per-destination diamonds. Any more than two is extremely rare; less than 1% of the paths.

Fig. 6 presents the min-width distribution for per-flow and per-destination load-balanced paths in the MIT dataset. Note that the Y axis is in log scale.

6.1.1 Narrow diamonds

This plot shows that load-balanced paths are generally narrow. For per-flow load balancing, 55% of the pairs encounter a diamond with only two link-disjoint paths, and 99% of the pairs encounter diamonds with five or fewer link-disjoint paths. For per-destination load balancing, the figures are 67% and 98%, and for per-packet load balancing (not shown), they are 60% and 90%.

The max-width distribution (not shown) is of course less skewed towards diamonds of width 2. Only 24% of per-flow load-balanced paths and 27% of per-destination load-balanced paths traverse a diamond with just two interfaces at the widest hop distance. Nonetheless the diamonds tend
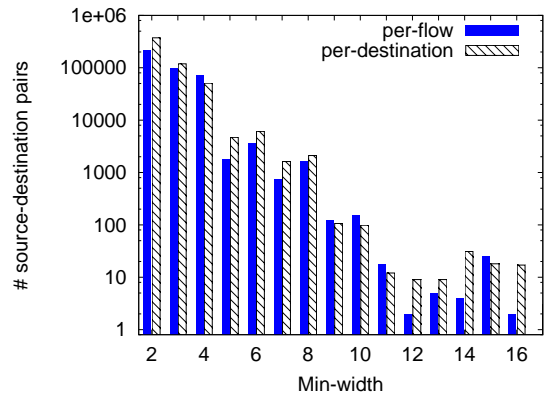

Figure 6: Min-width distributions (MIT).

to be narrow by this metric as well: 85% of the per-flow load-balanced paths and 90% of the per-destination load-balanced paths have five or fewer interfaces at the widest hop. Because most of per-packet load-balanced paths have a length equal to 2, their max-width distribution is similar to their min-width distribution.

6.1.2 Wide diamonds

The maximum width that we encounter, by either metric, is 16. For instance, we discovered a diamond of max-width 16 for per-flow load balancing at a peering point between a tier-1 and a Brazilian ISP. This may correspond to many low-capacity links which are bundled because the next higher capacity link is unavailable, unaffordable, or unsuitable. That we do not see anything wider can be explained by a built-in limit to the number of entries that a router can install in the forwarding table for a given prefix. For instance, Juniper [2] allows one to configure at most 16 load-balanced interfaces.

Almost all of the diamonds of width 10 and greater are two hops long. One obvious explanation for a diamond of this length is that we are seeing multiple parallel links between a pair of routers. As routers typically respond to traceroute probes using the address of the incoming interface [26], a pair of routers with parallel links will appear as a diamond of length two at the IP level. Fig. 7 shows an example with two parallel links.
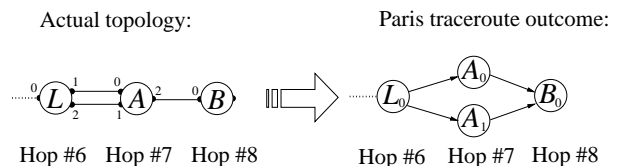

Figure 7: Load balancing over parallel links

There are rare cases (67 source-destination pairs in the MIT trace) of very wide and short per-packet diamonds at the ends of paths (i.e., close to the destinations). For instance, all load-balanced paths to a few hosts in Egypt traverse a per-packet diamond of length 2, having 11 interfaces in parallel. Alias resolution techniques (DNS names and checking the IP Identifier values returned by probes [7])

confirm that all 11 interfaces belong to the same router, and thus that the network operator configured 11 parallel links between two routers. Per-packet load balancing typically appears to take place at the boundary of a small AS and its provider. Customers may use such load balancers on access links for resilience and traffic engineering.

6.2 Diamond length

Recall that Sec. 4.2 defines the length of a diamond as the maximum number of hops between its divergence point and convergence point. We define the diamond length for a load-balanced path to be the length of the longest diamond found in that path.

Fig. 8 shows the distribution of the diamond lengths for load-balanced paths between all source-destination pairs in the MIT dataset. The Y axis, in log scale, represents the
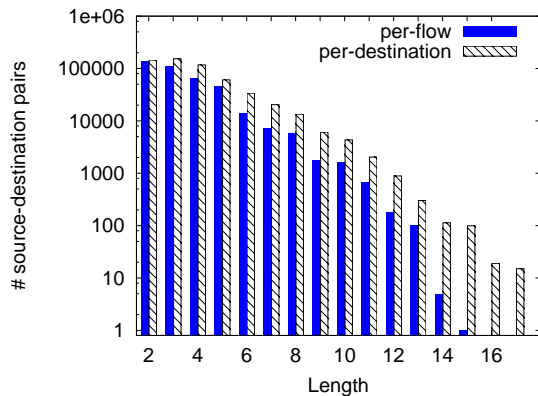

Figure 8: Diamond length distributions (MIT).

number of source-destination pairs that have a given diamond length. All sources have similar statistics, except for the nyu trace in which 56% of the per-flow load-balanced paths (26,016 out of the 43,112 per-flow load-balanced paths) traverse a diamond of length 4. This diamond, which is located in the Broadwing network, skews the nyu distribution. Overall, diamonds tend to be short, with a significant portion being of length 2.

6.2.1 Short diamonds

Of a total of 394,165 source-destination pairs with per-flow load balancing, 35% have a diamond of length two. Per-destination diamonds also tend to be short. Of 555,693 paths with per-destination load balancing, 26% of them have a diamond of length two. Diamond length is most skewed towards the short end for per-packet load balancing (not shown), with 90% of paths having a diamond of length two.

As discussed earlier, diamonds of length two should typically correspond to multiple links between two routers. Operators use load balancing between two routers not only for load sharing, but also as active backup in case of single link failures.

6.2.2 Long diamonds

Load-balanced paths with longer diamonds are less frequent. For instance, fewer than 1% of per-destination load-balanced paths have diamonds longer than 8. We observe per-flow diamonds of lengths up to 15 and per-destination
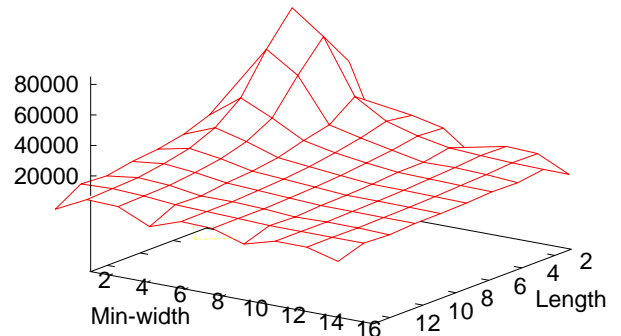

Figure 9: Diamond length and min-width of per-flow load-balanced paths.

diamonds with up to 17 hops. The longest per-packet load-balanced paths have lengths up to 6.

Per-destination diamonds tend to be longer than per-flow. Around 37% of load-balanced paths traverse a per-flow diamond of length greater than 3; this percentage is 46% for per-destination diamonds. There are few long per-packet diamonds (only 3% have a length greater than 3).

We looked at the 25 cases of per-packet diamonds of length 5 and 6 in detail. Most of them appear in core networks in Asian ISPs. Given the general practice of avoiding per-packet load balancing in core networks, perhaps these are cases of misconfigured load balancing. If so, then we see how Paris traceroute could help operators detect such misconfigurations.

6.2.3 Length and width

We now study the relationship between the min-width and length. Fig. 9 presents the number of per-flow load-balanced paths in the MIT dataset with a given diamond length and min-width. The vertical axis represents the number of source-destination pairs whose diamond length and min-width are given by the horizontal axis.

As discussed in Sec. 6.1, there may be several diamonds for the same source-destination pair. If so, we select the min-width and length of the diamond with the smallest min-width. There is a clear peak in the number of load-balanced paths with length and min-width equal to two (load-balanced paths between 17% of the 394,165 source-destination pairs with per-flow load balancing are in this category). Load-balanced paths between 53% of source-destination pairs traverse a diamond with a length less or equal to 2 and min-width 2 or 3. This result confirms that the vast majority of the diamonds are both short and narrow.

There are no wide and long diamonds. There is a bipartition of the remaining diamonds into two categories. The first category contains wide but short diamonds. It is extremely rare to observe wide diamonds (whose width is greater than 2) with more than 3 hops. The second one corresponds to narrow but long parallel paths. In this case, the min-width is always 2. Wide but short diamonds probably correspond

to multiple links between routers. Operators may introduce new links between routers to upgrade capacity. Long and narrow diamonds likely correspond to paths between the ingress and egress routers in a network, which are useful for traffic engineering.

6.3 Diamond asymmetry

We say that a diamond is asymmetric when one can reach its convergence point with different hop counts. There might be some concern that asymmetric diamonds are the result of misconfiguration. But the equal-cost multipath mechanisms of OSPF and IS-IS require only that paths have the same cost in terms of link weight, not hop count [19, 20]. Network operators can configure two paths of different hop counts to have the same sum of link weights. In addition, some new mechanisms [27] allow load-balancing over paths with small cost differences.

From the point of view of performance, asymmetry might cause delay differences. We study this correlation in Sec. 7. Asymmetry might also be related to path reliability, with longer paths, traversing as they do more IP interfaces, being potentially more susceptible to failure. We do not study this.

Fig. 10 presents the number of source-destination pairs in the MIT dataset that have per-flow and per-destination load-balanced paths with a given asymmetry. The Y axis is
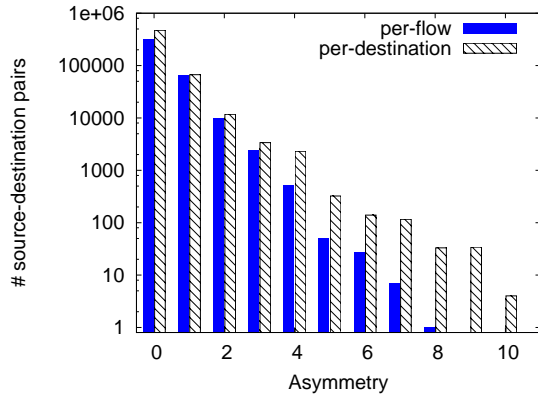

Figure 10: Asymmetry distributions (MIT).

in log scale.

Most paths with per-flow load balancing, 79%, traverse symmetric diamonds. Paths under per-destination load balancing, are slightly more symmetric: 85% of the 555,693 paths under per-destination load balancing traverse symmetric diamonds. That still leaves a significant fraction of destinations that can be reached with different numbers of hops.

On the other hand, over 95% of the paths with per-packet load balancing (not shown) traverse a symmetric diamond. This is consistent with the observation that the majority of such diamonds are short, and thus have less possibility for asymmetry. Nonetheless, the 5% of such paths with asymmetry are a concern for TCP connections, in so far as asymmetry implies different delays and thus greater chances for packet reordering.

6.3.1 Low asymmetry

When asymmetry is present, it is typically low. For instance, out of 84,222 per-flow load-balanced paths with asym-

metric diamonds, 69,334 (82%) only differ by one hop. For per-destination load-balanced paths, this fraction is 79% (66,721 out of 84,744 pairs with asymmetric diamonds), and for per- packet, 65% (923 out of 1,406 pairs).

6.3.2 High asymmetry

Per-flow diamonds with more than 3 hops of difference are extremely rare (614 source-destinations pairs). For instance, the paris trace contains only 63 such diamonds, and the chi1-gblx trace, 162. We found 2,549 such per-destination, and only 11 such per-packet load-balanced paths.

We examined the per-flow diamond with the maximum asymmetry, which has 8 hops difference. One path traverses 8 routers between the divergence and convergence points while the other directly reaches the end of the diamond. We believe that the latter path is an MPLS tunnel, maybe even traversing the same routers as the one traversed on the first path. This hypothesis is supported by the observation that routers in the diamond append MPLS extensions [28] to the ICMP responses. This example suggests that some of the shorter diamonds may also result from MPLS tunnels.

For per-destination load balancing, there are 71 cases of asymmetry between 8 and 10. We examined some of these diamonds with very distinct paths. For instance, there is one asymmetric diamond that spans the US and Europe in Cogent's network. By inspecting the interface names, we concluded that the parallel paths each traverse different numbers of points of presence (PoPs), which causes the asymmetry. Depending upon which address is probed inside the destination prefix, packets either cross the ocean through a link via London or another via Paris.

7. DELAYS

This section characterizes round-trip times (RTTs) under per-flow load balancing. We focus on per-flow load balancers because we cannot control the paths under per-packet splitting and we cannot strictly compare delays to different destinations under per-destination load balancing.

RTTs measure the forward and the return path combined; therefore we need to ensure that all probes to measure delays of paths in the forward direction follow the same return path (even if they traverse a load balancer). Otherwise, our conclusions could be mistaken: delay differences between load-balanced paths in the forward direction could be due to delay variation in the reverse direction. This section first describes how to control the return path under load balancing. Then, it presents our measurement setup and results.

7.1 Controlling the return path

Classic traceroute fails to maintain the flow identifier on the reverse path, so probe responses may take different paths when they traverse a per-flow load balancer. Probe responses are ICMP messages (usually, Time Exceeded or Port Unreachable) from routers or end-hosts. Per-flow load balancers use the ICMP checksum, which depends on the ICMP header and data, as part of the flow identifier for ICMP packets [15]. Routers construct an ICMP Time Exceeded message with the IP and UDP headers of the probe, but usually not the UDP data [29]. Classic traceroute systematically varies the checksum of the probe responses, because it varies the UDP data of the probe, which in turn impacts the UDP checksum of the probe. Ping also fails to measure accurate delays under load balancing, because it varies the

ICMP sequence number, which has an impact on the ICMP checksum. Ping probes may take different paths not only on the forward path, but also on the reverse path.

Unlike ping and classic traceroute, Paris traceroute maintains the forward flow identifier, but its original implementation [3] varies the return flow identifier. We develop a method to control the return path:

1. Build a probe with the desired forward flow identifier (for UDP, we set the UDP port numbers);

2. Predict the ICMP checksum value by constructing the predicted ICMP response; and

3. Find the appropriate value we have to put in the UDP data of the probe to yield the desired ICMP checksum.

Unfortunately, there are some challenges that may prevent us from correctly predicting the flow identifier of the probe responses. First, to predict the response content we have to know the TTL value of the probe when it reaches the router. The TTL is generally equal to 1, but in some cases it can have a different value (for instance, in the presence of routers that forward packets whose TTL has reached the minimum value [3]). Second, we found some routers which fill some ICMP fields with junk bytes (fortunately, these are always the same byte values for a given router), whereas they should be filled with zeros. Since the ICMP checksum also depends on those fields, we cannot know the content of those fields until we have probed the router. Third, we found rare routers that include IP (security related) options in the ICMP responses. We solve the first two issues by sending a first "scout" probe to discover the parameters one has to use to build the next probe. The third issue cannot be addressed, because per-flow load balancers behave like per-packet when they forward packets with IP options, which prevents any attempt to control the path responses may take.

We use this technique to maintain the return flow identifier for our RTT measurements. In future work, we plan to vary the return flow identifier in a controlled manner to detect load-balanced paths on the reverse path.

7.2 Measuring RTTs

We measure RTTs using the version of Paris traceroute that maintains the return flow identifier. Per-packet load balancers can still make the return path vary, but as discussed in Sec. 5, this type of load balancer is less frequently seen (less than 2% of the paths in our traces). Furthermore, we looked at the paths between all our sources (recall that we updated the MIT list by adding all our source nodes) and did not observe any per-packet load balancers. Given that they tend to be deployed in edge networks, we believe that there were no per-packet load balancers close to our sources, thus our measurements were not affected by per-packet load balancing on the return path.

We take the following steps to conduct our measurements:

Selection of destinations: We launch Paris traceroute from 10 sources towards all destinations in the MIT list. For each source, we select the first 5,000 destinations reachable through a per-flow load balancer.

Selection of flow identifiers: We use the MDA in a first phase as an input to guide the second phase of the experiments. For each destination, we select at most n flow identifiers belonging to different flow classes (a flow class is a set of flows which take exactly the same IP-level forward path). We chose $n = 6$, because the results in Sec. 6 show that only 4% of the per-flow diamonds have a max-width larger than 6.

Selection of interface to probe: We probe intermediate hops instead of the destinations themselves. We select the last responding interface that all the selected paths to the destination have in common. Instead of probing this interface directly, we use UDP probes to the destination (with a limited TTL value) to force the probed router to generate ICMP error messages.

Computation of RTTs: We compute the RTT by subtracting the time of the reception of the ICMP error message from the time the probe is sent. For each selected flow class, we send 100 probes (at a constant rate, one probe every 50 ms) to this interface, using the associated flow identifier (i.e., UDP source and destination ports). We then keep only the lowest RTT. This eliminates, we hope, any over-estimate of the RTT due to the variability of queuing delays. Probes and responses can be lost, and routers limit their ICMP packet response rate, so we ignore any flow class for which we have fewer than 90 responses.

Comparison of delays: We use the minimum RTT for each flow class as representative of the delay over the corresponding load-balanced path. We set the threshold for declaring that two flow classes have different end-to-end delays at a difference of one millisecond in their representative delays. To be sure that we are not declaring differences where none are to be found, we batched delays in groups of ten and looked at the variance of the minimum from each group. These minimum delays have a variance of 0.5 ms or less (often less than 0.2 ms), indicating that a difference of 1 ms can be considered as significant.

7.3 Results

We measure RTTs between 43,889 source-destination pairs with per-flow load balancing. We had to trace 146,296 pairs to find them, given that only 30% of the pairs traverse a per-flow load balancer. We obtained enough RTT samples (i.e., replies to more than 90% of our probes) for 21,791 pairs. Despite the high fraction of pairs with insufficient measurements, no doubt due to routers that rate-limit ICMP messages, we were still left with a large number of paths to study.

For most of the source-destination pairs, there is no significant RTT difference among load-balanced paths. Only 12% of pairs (or 2,644) have a delay difference of more than 1 ms. Fig. 11 shows the distribution of the ratio, for source-destination pairs, between their shortest- and longest-delay paths. These results are an indication of the time an end-host might save by selecting the appropriate flow identifier to reach a particular destination. The Y axis is logarithmic and plots the number of source-destination pairs with delay ratio given by the X axis. Most pairs (59%) can save at least 2% with respect to the longest delay, but only 2% of pairs can save 10% or more. There are some rare cases with a difference of 29% in the delay between the longest- and shortest- delay paths. For instance, roncluster1 can reach a user of the MindSpring ISP through the Road Runner domain either with a 22 ms RTT or a 31 ms RTT.

We study the correlation between diamonds' lengths (respectively asymmetry) and delays, as shown in Fig. 12 left (resp. right). Each group of bars represents all source-des-
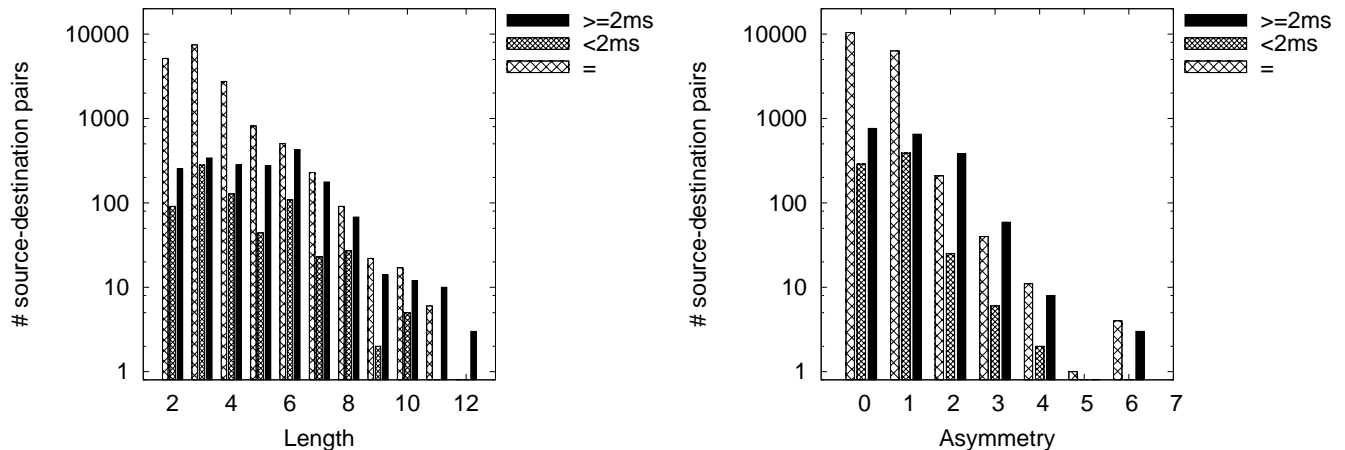
Figure 12: Left plot: correlation between diamond length and delays. Each group of vertical bars represents all the source-destination pairs traversing a diamond whose length is given by the X axis. Each individual bar represents the number of those pairs having paths with either equal or different delays to the destination. Right plot: correlation between diamond asymmetry and delays, using the same conventions.
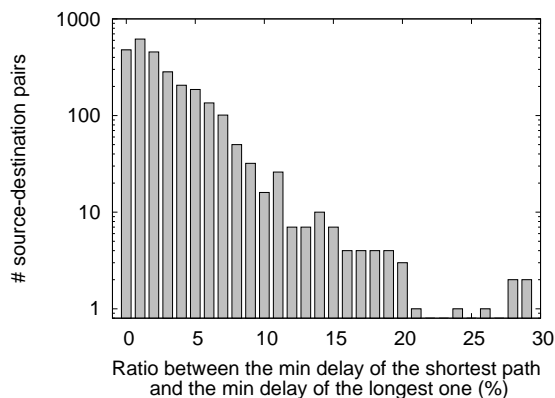

Figure 11: Delay difference distribution.

tination pairs traversing a diamond with a length (resp. an asymmetry) given by the X axis. Then each group has three bars, one for pairs with all paths having equal delays, one for pairs with a delay difference (between the longest path and the shortest one) lower than 2 ms, and the third for pairs with a difference greater than 2 ms.

The figure shows that the longer the load-balanced paths, the more we find delay differences. 94% of the paths with short diamonds have paths of equivalent delays. Diamonds longer than 4 lead to delay differences greater than 2 ms in nearly 40% of the cases.

90% of the paths with symmetric diamonds (i.e., an asymmetry equal to 0 in the figure) have delays that we consider to be equal. This proportion decreases for asymmetric diamonds. We found that 61% of the paths with an asymmetry equal to 2 have delay differences greater than 2 ms. Interestingly, beyond an asymmetry of 2, an increase in the asymmetry of diamonds correlates with a decrease in the number of source-destination pairs with different delays. An explanation might be a lack of significance in this range due to

the small size of the dataset, since we have only 134 pairs traversing a diamond with an asymmetry greater than 2, in comparison to the 11,391 pairs with a symmetric diamond. Another possibility is the presence of hidden MPLS paths that shorten some of the paths in a diamond, as discussed in Sec. 6.3.2.

Our results clearly show that one can achieve RTT gains towards some internet hosts simply by controlling the ports used by an application. However, this phenomenon is not widespread enough (only 12% of the source-destination pairs with per-flow load balancing) to be employable on a wide scale for end-to-end performance improvements.

8. CONCLUSION

This paper makes three main points. First, Paris traceroute's Multipath Detection Algorithm can now measure the load-balanced paths under per-destination load balancing, in addition to per-flow and per-packet load balancing. Second, load-balanced paths are common in the internet, at least as seen from our 15 vantage points. Third, in order to ensure accurate RTT measurements under per-flow load balancing it is important to maintain both the forward and the return flow identifier.

Given the high fraction of paths that traverse a load balancer, a natural question is whether applications can use the native path diversity thus provided. In our dataset, there are not many opportunities for applications that want fully-disjoint paths or paths with very different properties (such as hop count or delay). Often, paths are disjoint just for a few hops and have comparable delays. In cases where partially-disjoint paths suffice, applications can easily take advantage of multiple paths under per-flow load balancing. It only requires selecting the appropriate flow identifier from the list provided by a utility such as Paris traceroute. Our characterization of load-balanced paths, however, shows that the fraction of paths that traverse per-flow load balancers and their characteristics depends on the location and the upstream connectivity of the source node. Therefore, applica-

tions that want to explore this type of path diversity should first use a utility such as Paris traceroute from their hosts to evaluate the potential benefits. Per-destination path diversity is more prevalent, but it is not as simple to use. One could imagine giving multiple addresses to a host, or having hosts connected to the same local network collaborate to use the path diversity from a remote source.

In future work, we plan to improve the MDA to work under uneven load balancing; develop a tool to find the available bandwidth under per-flow load balancing; and explore our technique to control the return flow identifier to characterize load balancing on the return path.

Acknowledgments

9. REFERENCES

[1] Cisco, "How Does Load Balancing Work?" from the Cisco Documentation, see http://www.cisco.com/warp/public/105/46.html.

[2] Juniper, "Configuring Load-Balance Per-Packet Action," from the JUNOS Policy Framework Configuration Guideline.

[3] B. Augustin, X. Cuvellier, B. Orgogozo, F. Viger, T. Friedman, M. Latapy, C. Magnien, and R. Teixeira, "Avoiding traceroute anomalies with Paris traceroute," in *Proc. ACM SIGCOMM Internet Measurement Conference*, October 2006.

[4] D. Andersen, H. Balakrishnan, M. F. Kaashoek, and R. Morris, "Resilient Overlay Networks," in *Proc. 18th ACM Symposium on Operating Systems Principles*, October 2001.

[5] R. Teixeira, K. Marzullo, S. Savage, and G. M. Voelker, "Characterizing and Measuring Path Diversity of Internet Topologies," in *Proc. ACM SIGMETRICS*, June 2003.

[6] ——, "In Search of Path Diversity in ISP Networks," in *Proc. ACM SIGCOMM Internet Measurement Conference*, October 2003.

[7] N. Spring, R. Mahajan, and D. Wetherall, "Measuring ISP topologies with Rocketfuel," in *Proc. ACM SIGCOMM*, August 2002.

[8] B. Huffaker, D. Plummer, D. Moore, and k claffy, "Topology discovery by active probing," in *Proc. Symposium on Applications and the Internet (SAINT)*, January 2002.

[9] S. Kandula, D. Katabi, B. Davie, and A. Charny, "Walking the Tightrope: Responsive Yet Stable Traffic Engineering," in *Proc. ACM SIGCOMM*, August 2005.

[10] S. Sinha, S. Kandula, and D. Katabi, "Harnessing TCPs Burstiness using Flowlet Switching," in *Proc. 3rd ACM SIGCOMM Workshop on Hot Topics in Networks (HotNets)*, November 2004.

[11] C. Villamizar, "OSPF Optimized Multipath (OSPF-OMP)," IETF Internet Draft, February 1999.

[12] A. Elwalid, C. Jin, S. H. Low, and I. Widjaja, "MATE: MPLS adaptive traffic engineering," in *Proc. IEEE Infocom*, April 2001.

[13] F. Giroire, A. Nucci, N. Taft, and C. Diot, "Increasing the Robustness of IP Backbones in the Absence of Optical Level Protection," in *Proc. IEEE Infocom*, March 2003.

[14] S. Savage, A. Collins, E. Hoffman, J. Snell, and T. E. Anderson, "The end-to-end effects of internet path selection," in *Proc. ACM SIGCOMM*, August-September 1999.

[15] B. Augustin, T. Friedman, and R. Teixeira, "Multipath Tracing with Paris Traceroute," in *Proc. Workshop on End-to-End Monitoring (E2EMON)*, May 2007.

[16] V. Paxson, "End-to-end internet packet dynamics," *IEEE/ACM Trans. Networking*, vol. 5, no. 5, pp. 139–154, September 1999.

[17] Cisco, "Cisco 7600 Series Routers Command References," from the Cisco Documentation.

[18] J. Bellardo and S. Savage, "Measuring packet reordering," in *Proc. ACM SIGCOMM Internet Measurement Workshop*, November 2002.

[19] R. Callon, "Use of OSI IS–IS for Routing in TCP/IP and Dual Environments," IETF RFC 1195, December 1990.

[20] J. Moy, "OSPF Version 2," IETF RFC 2328, April 1998.

[21] Y. Rekhter, T. Li, and S. Hares, "A Border Gateway Protocol 4 (BGP-4)," IETF RFC 4271, January 2006.

[22] Juniper, "Configuring BGP to Select Multiple BGP Paths," JUNOS Software Documentation.

[23] Cisco, "BGP Best Path Selection Algorithm," from the Cisco Documentation, see http://www.cisco.com/warp/public/459/25.shtml#bgpmpath.

[24] Team Cymru, "IP to BGP ASN Lookup and Prefix Mapping Services," see http://www.cymru.com/BGP/asnlookup.html.

[25] Z. M. Mao, D. Johnson, J. Rexford, J. Wang, and R. H. Katz, "Scalable and Accurate Identification of AS-level Forwarding Paths," in *Proc. IEEE Infocom*, March 2004.

[26] Z. M. Mao, J. Rexford, J. Wang, and R. H. Katz, "Towards an accurate AS-level traceroute tool," in *Proc. ACM SIGCOMM*, August 2003.

[27] Cisco, "How Does Unequal Cost Path Load Balancing (Variance) Work in IGRP and EIGRP?" from the Cisco Documentation.

[28] R. Bonica, D. Gan, D. Tappan, and C. Pignataro, "ICMP Extensions for Multiprotocol Label Switching," IETF RFC 4950, August 2007.

[29] D. Malone and M. Lucky, "Analysis of ICMP Quotations," in *Proc. Passive and Active Measurement Workshop*, April 2007.