Avoiding traceroute anomalies with Paris traceroute

Brice Augustin*, Xavier Cuvellier*, Benjamin Orgogozo†, Fabien Viger†,
Timur Friedman*, Matthieu Latapy†, Clémence Magnien‡, and Renata Teixeira*

* Université Pierre et Marie Curie – CNRS, Laboratoire LIP6
† Université Denis Diderot – CNRS, Laboratoire LIAFA
‡ Ecole Polytechnique – CNRS, Laboratoire CREA

ABSTRACT

Traceroute is widely used, from the diagnosis of network problems to the assemblage of internet maps. However, there are a few serious problems with this tool, in particular due to the presence of load balancing routers in the network. This paper describes a number of anomalies that arise in nearly all traceroute-based measurements. We categorize them as "loops", "cycles", and "diamonds". We provide a new publicly-available traceroute, called *Paris traceroute*, which controls packet header contents to obtain a more precise picture of the actual routes that packets follow. This new tool allows us to find conclusive explanations for some of the anomalies, and to suggest possible causes for others.

Categories and Subject Descriptors: C.2.3 [Computer Communication Networks]: Network Operations

General Terms: Measurement.

Keywords: traceroute, load balancing.

1. INTRODUCTION

Jacobson's *traceroute* [1] is one of the most widely used network measurement tools. It reports an IP address for each network-layer device along the path from a source to a destination host in an IP network. Network operators and researchers rely on traceroute to diagnose network problems and to infer properties of IP networks, such as the topology of the internet.

This paper describes how traceroute fails in the presence of routers that employ load balancing on packet header fields. The failures lead to incorrect route inferences that may mislead operators during problem diagnosis and result in erroneous internet maps. We provide a new publicly-available traceroute, called *Paris traceroute* [1], which controls packet header contents to obtain a more precise picture of the actual routes that packets follow.

[1]Paris traceroute is free, open-source software, available from `http://www.paris-traceroute.net/`.

IMC'06, October 25–27, 2006, Rio de Janeiro, Brazil.

This paper highlights the problems of using classic traceroute for route inference by examining a number of topology artifacts that arise in traceroute-based measurements. We show that, using measurements from a single source tracing toward multiple destinations, one may observe anomalies that we categorize as "loops", "cycles", and "diamonds". We explain how many instances of these anomalies result from load balancing routers, and disappear when one uses Paris traceroute. We explain most other instances using additional information provided by Paris traceroute. Finally, we suggest possible causes for the remaining instances.

2. BUILDING A BETTER TRACEROUTE

This section first describes the deficiencies of the classic traceroute in the face of load balancing. Then we present our new traceroute, Paris traceroute, which avoids some of these deficiencies, notably the ones induced by per-flow load balancing.

2.1 Traceroute and load balancing

Network administrators employ load balancing to enhance reliability and increase resource utilization. They do so through the intra-domain routing protocols OSPF [2] and IS-IS [3] that support *equal cost multipath*. An operator of a multi-homed stub network can also use load balancing to select which of its internet service providers will receive which packets [4].

Routers can spread their traffic across multiple equal-cost paths using a per-packet, per-flow, or per-destination policy [5, 6]. In *per-flow load balancing*, packet header information ascribes each packet to a flow, and the router forwards all packets belonging to a same flow to the same interface. A natural flow identifier is the classic *five-tuple* of fields from the IP header and either the TCP or UDP headers: Source Address, Destination Address, Protocol, Source Port, and Destination Port. We found through our experiments that routers use various combinations of these fields, as well as three other fields: the IP Type of Service (TOS), and the ICMP Code and Checksum fields. We leave an exhaustive study of which header fields serve for load balancing, and in precisely which ways, to future work.

Per-flow load balancing ensures that packets from the same flow are delivered in order. *Per-packet load balancing* makes no attempt to keep packets from the same flow together, and focuses purely on maintaining an even load. *Per-destination load balancing* could be seen as a coarse form of per-flow load balancing, as it directs packets based upon the destination IP address. But, as it disregards source in-

formation, there is no notion of a flow per se. As seen from the measurement point of view, per-destination load balancing is equivalent to classic routing, which is also per destination, and so we will not explore it here.

Where there is load balancing, there is no longer a single route from a source to a destination. In the case of per-packet load balancing, a given packet might take any one of a number of possible routes. With per-flow load balancing, the notion of a single route persists for packets belonging to a given flow, but different flows for the same source-destination pair can follow different routes. Designing a new traceroute able to uncover all routes from a source to a given destination would be a significant improvement.

Classic traceroute is not adequate to the task, as it cannot definitively identify one single route from among many. It suffers from two systematic problems: it fails to discover true nodes and links, and it may report false links. These problems arise because traceroute discovers hops along a route with a series of probe packets, and the fact that a a load-balancing router, or *load balancer*, can direct these probes along different paths.

Our explanation of the problems draws on the example in Fig. 1. Here, L is a load balancer at hop 6 from the traceroute source, S. On the left, we see the true router topology at hops 6 through 9. Circles represent routers, and each router interface is numbered. Black squares depict probe packets sent with TTLs 6 through 9. They are shown either above the topology, if L directs them to router A, or below, if L directs them to router B. On the right, we see the topology that would be inferred from the routers' responses.

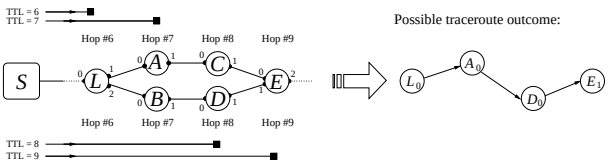

Figure 1: Missing nodes and links, and false links.

Missing nodes and links. Because routers B and C send no responses, nodes B_0 and C_0 are not discovered, and links such as (L_0, B_0) and (B_0, D_0), cannot be inferred.

False links. L directs the probe with initial TTL 7 to A and the one with initial TTL 8 to B, leading to the mistaken inference of a link between A_0 and D_0.

Classic traceroute sends three probes per hop, rather than one, but these problems do not go away. On a traceroute through the example topology, with purely random load balancing, the probability is $0.5^3 \times 2 = 0.25$ that one of the two devices at hop 7, A or B, goes undiscovered. Similarly, the probability is $0.75 + 0.25 \times 0.75 = 0.9375$ that two devices are discovered at hop 7 or hop 8, or both, making it ambiguous which links are true and which are false.

In the general case, there may be more than two next-hop interfaces following a load balancer. For instance, the newer Juniper routers permit up to sixteen equal-cost paths. It is certain that classic traceroute misses many nodes, and the possibility to infer many links, and that it has many ways to infer false links.

Topology inference systems based on traceroute handle these problems in different ways. Huffaker et al. recognize the problem for *skitter* [7], but they do not report a solution. In practice, the *arts++* tool for reading skitter data reports only the first address obtained for each hop. With *Rocketfuel* [8], Spring et al. attribute a lower confidence level to links inferred from hops that respond with multiple addresses. Still, they include all these links in the database from which they construct a network's topology. Their hope is that a subsequent alias resolution step will eliminate at least some of the false links. However, this only works if load balancing takes place over multiple links between the same pair of routers.

Where there is per-flow load balancing, traceroute's own behavior causes the problems. When sending UDP probes, it systematically varies the Destination Port field. When sending ICMP Echo probes, it varies the Sequence Number field. It does so because it needs to match routers' responses with the probes that elicited them. A router that sends an ICMP Time Exceeded response encapsulates the IP header of the packet that it discarded, plus the first eight octets of data [9, p.5], which, in the case of UDP or ICMP Echo probes, means the transport-layer header. A unique value in the probe header ensures a uniquely tagged response. Unfortunately, as we have found in our experiments, varying any field in the first four octets of the transport-layer header amounts to changing the flow identifier for each probe. The Destination Port field is in the first four octets of the UDP header, and, though the Sequence Number field is in the second four octets of the ICMP Echo header, varying this field varies the Checksum field, which is in the first four octets.

2.2 A new traceroute

We introduce Paris traceroute, a new traceroute designed for networks with load balancing routers. Its key innovation is to control the probe packet header fields in a manner that allows all probes towards a destination to follow the same path in the presence of per-flow load balancing. It also allows a user to distinguish between the presence of per-flow load balancing and per-packet load balancing. Unfortunately, due to the random nature of per-packet load balancing, Paris traceroute cannot perfectly enumerate all paths in all situations. But it can do considerably better than the classic traceroute, and it can flag those instances where there are doubts.

Maintaining certain header fields constant is challenging because traceroute still needs to be able to match response packets to their corresponding probe packets. Paris traceroute does this by varying header fields that are within the first eight octets of the transport-layer header, but that are not used for load balancing. For UDP probes, Paris traceroute varies the Checksum field. This requires manipulating the payload to yield the desired value, as packets with an incorrect checksum are liable to be discarded. For ICMP Echo probes, Paris traceroute varies the Sequence Number field, as does classic traceroute, but also varies the Identifier field, so as to keep constant the value for the Checksum field.

Paris traceroute also sends TCP probes, unlike classic traceroute, but like Toren's variant *tcptraceroute* [10]. For TCP probes, Paris traceroute varies the Sequence Number field. No other manipulations are necessary in order to maintain the first four octets of the header field constant.

Paris traceroute's TCP probing is not innovative in the same way as its UDP and ICMP Echo probing, as tcptracer-

oute already maintains a constant flow identifier. In order to more easily traverse firewalls, tcptraceroute by default sets probes' Destination Port field to 80, emulating web traffic, and varies the Identification field in the IP header. However, no prior work has examined the effect, with respect to load balancing, of maintaining a constant flow identifier for probe packets.

Fig. 2 summarizes the IP, UDP, ICMP Echo, and TCP header fields that are used by load balancers, classic traceroute, tcptraceroute, and Paris traceroute. We do not label the ICMP Type field as used for load balancing, because, in our experiments, routers have not sent responses to ICMP probes of any type other than ICMP Echo. Based on our experiments with UDP, TCP, and IPSec probes, we nonetheless are inclined to believe that routers blindly employ the first four octets in the transport-layer header for per-flow load balancing.

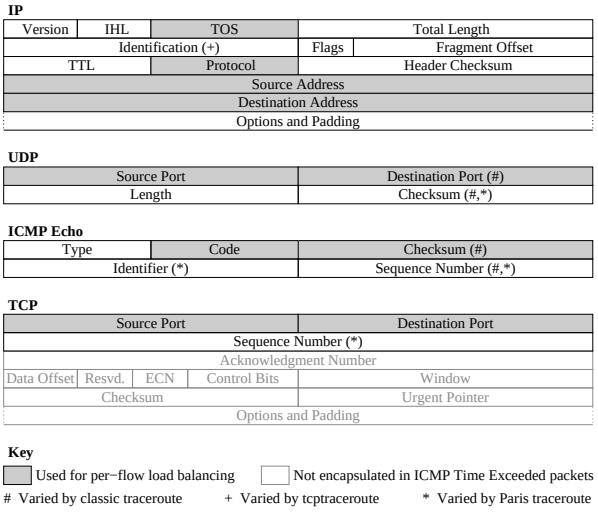

Figure 2: The roles played by packet header fields.

In addition to fixing the flow identifier problem, Paris traceroute provides additional information that helps a user recognize certain other traceroute anomalies. The *probe TTL* is the TTL that is found in the IP header of the probe packet that is encapsulated in the ICMP Time Exceeded response. This value corresponds to the probe's TTL when the router received it and decided to discard it. Under normal traceroute behavior, this value is one. A value other than one signals an anomaly. The *response TTL* is the TTL from the IP header of the Time Exceeded response itself. This value, available also through classic traceroute, helps us infer the length of the return path. Finally, the *IP ID* is the Identification field from the IP header of the Time Exceeded response. This field is set by the router with the value of an internal 16-bit counter that is usually incremented for each packet sent. The IP ID can help identify the multiple interfaces of a same router, as described in the *Rocketfuel* work [8], or uncover different routers and hosts hidden behind a firewall or a NAT box, as described by Bellovin [11].

3. MEASUREMENT SETUP

To study traceroute anomalies, we conduct side-by-side measurements with classic traceroute and Paris traceroute. These experiments form a case study showing how Paris traceroute finds more accurate routes and yields knowledge concerning the causes of each anomaly. As observations from other vantage points towards other destination sets would reveal different numbers, the particular values from this study cannot be considered statistically representative.

Our measurement source is located at the LIP6 laboratory of the Université Pierre et Marie Curie, in Paris, France. The university has only one connection to the internet via the French academic backbone, Renater. We measured during one month, from 12 June to 12 July, 2006.

Our destination list consists of 5,000 randomly chosen pingable IPv4 addresses, without duplicates, and in random order. We only consider pingable addresses so as to avoid the artificial inflation of traceroute anomalies in our results that would come from tracing towards unused IP addresses [12].

We launch 32 parallel processes that each probe 1/32 of the destination list. Each process selects a destination d from its portion of the list and proceeds as follows:

1. Trace a route to d with Paris traceroute, sending one UDP probe per hop, and waiting up to 2 sec. to receive a reply at one hop before sending a probe to the subsequent hop. For the entire sequence of probes to d, we set an unchanging five-tuple, with Source and Destination Port values chosen at random from the range [10,000, 60,000]. We set the minimum TTL to two, to skip the routers inside the university network.

2. Trace a route to d using an instance of classic traceroute, using the same timing parameters as Paris traceroute. We use NetBSD traceroute version 1.4a5, also sending one UDP probe per hop. As per traceroute's default behavior, the Source Port value is set to the running process identifier (PID) plus 32,768, and the initial Destination Port value is set to 33,435, and is incremented with each probe sent.

No trace contains more than eight consecutive non-responses or extends further than 39 hops. Receipt of an ICMP Destination Unreachable response terminates a trace immediately.

One round of measurements to all destinations takes approximately one hour and eleven minutes, at a rate of approximately 27.3 seconds for both a Paris traceroute and a classic traceroute to a given destination. We perform consecutive rounds of measurements, and the results presented here are based on the 556 rounds that we completed during our observation period.

For the 90 million responses that contain valid IP Source Address values, we map the address to an AS number using Mao et al.'s technique [13]. Our traces cover 1,122 different ASes, which corresponds to five percent of the ASes in the internet today. Our traces traverse all nine tier-1 ISP networks and 64 of the one hundred top-20 ASes of each region according to APNIC's weekly routing table report [2]. Invalid IP addresses account for 19 thousand responses. The probes that did not receive a response are called *stars*, as they appear as such ('*') in traceroute output. Stars typically appear at the ends of routes, with just 2.6 million appearing in the midst of responses.

[2]APNIC automatically generates reports describing the state of internet routing tables. It ranks ASes for each of five world regions according the number of networks announced.

4. ANOMALIES

We focus our observations on three topology anomalies that we observe in measured routes, which we call loops, cycles, and diamonds. We will see that these anomalies largely disappear when routes are measured with controlled header fields. We observe load balancing, and the changes that come from tracing with Paris traceroute, in seven of the nine tier-1 ASes, and in 17 of the 64 top regional ASes through which we trace.

Throughout this section, we define a *measured route* to be the ℓ-tuple $R = (r_0, \cdots, r_\ell)$ where r_0 is the source address, and, for each i, $1 \leq i \leq \ell, r_i$ stands either for the IP address received when probing with TTL i, or for a star if none was received. Our formal definition, or *signature*, of each anomaly is in terms of the addresses seen in measured routes. We may see multiple instances of a loop, cycle, or diamond with a given signature.

4.1 Loops

In some measured routes, the same node appears twice or more in a row: we call this a *loop*. Formally, a loop is observed on IP address r_i with destination d if there is at least one measured route towards d containing $\cdots, r_i, r_{i+1}, \cdots$ with $r_i = r_{i+1}$. The term 'address' implies that r_i is not a star. A loop's signature is a pair (r, d), where r is the address involved in the loop and d is the destination. In the normal course of routing, a router does not forward a packet back to the incoming interface. Hence, loops are most likely an artifact of the measurement itself.

4.1.1 Possible causes

One cause of loops is **load balancing** on paths with different lengths. Fig. 3 depicts an example where the load balancer L forwards the probes having initials TTL 7 and 8 to router A and the probe having initial TTL 9 to router B, resulting in two responses from the same router, E. (We assume, for this example, that both responses are generated from the same interface, E_0.) The same scenario could arise

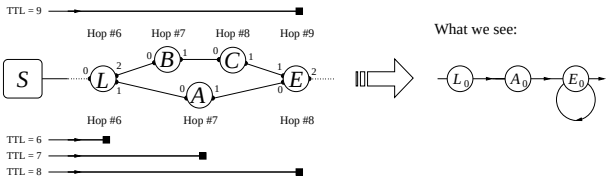

Figure 3: A loop caused by load balancing.

due to a **routing change** that forced packets from the path through A to the one through B in the middle of a traceroute. In particular, this occurs if the routing change happens between the time S receives the response to its probe with TTL 8 and the time that it emits the probe with TTL 9.

Not all loops are caused by load balancing or routing changes, however, and we found several other explanations:

Unreachability message. This type of loop arises when a router is unable to forward probes, perhaps due to failures or routing instabilities. The router treats as normal the first probe that it receives, with TTL equal to one, and generates an ICMP Time Exceeded reply. But a subsequent probe that arrives at the router, with TTL greater than one, causes an ICMP Destination Unreachable response. In such a circumstance, classic traceroute outputs the same address for each response. If not interpreted correctly, it appears to be a loop. Both classic traceroute and Paris traceroute will flag the second response with '!H' or '!N', for host or network unreachable, and then halt. When a loop appears at the end of a measured route, we look for these flags to identify this scenario.

Zero-TTL forwarding. This explanation is mentioned in the traceroute manual, which says that misconfigured routers forward packets with a TTL equal to zero, whereas they are supposed to drop them and generate an ICMP Time Exceeded response. Fig. 4 depicts an example, in which the faulty router is F. The probe sent with initial TTL 7 ought

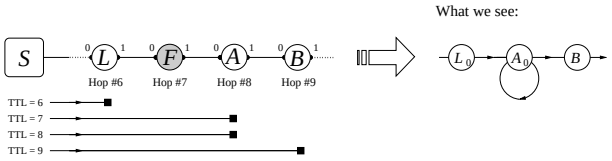

Figure 4: A loop caused by zero-TTL forwarding.

to be dropped by F, but instead it is forwarded to router A, which sends a response with probe TTL zero. (See the definition of "probe TTL" at the end of Sec. 2.2.) Then, in response to the subsequent probe, A sends a normal response, with probe TTL one.

When there is a loop, we check for zero-TTL forwarding by looking at the probe TTL. We check that the first of the two ICMP Time Exceeded responses that form a loop has a probe TTL equal to zero and the second a probe TTL of one. The examination of the IP IDs from the two reply packets should indicate that they come from the same router.

Address rewriting. Gateway routers, like NAT boxes and some firewalls, replace the Source Address field of all ICMP packets that originate within the subnetwork to which it is attached with a single IP address. Hence, all probes to a destination behind such a gateway will appear to come from the same router. In practice, we find such loops only at the end of measured routes.

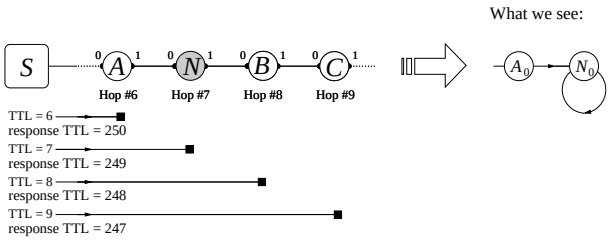

Figure 5: A loop caused by address rewriting.

We detect address rewriting using the response TTL (see the definition at the end of Sec. 2.2) and the IP ID of all packets involved in the loop. All ICMP messages generated by a router will have the same initial TTL (most routers use the default TTL for ICMP, which is 255). If the response TTL is different at each hop, then the responses are likely to come from different routers. The example in Fig. 5 presents a network with a NAT box N, and routers B and C behind this NAT box. We assume that all routers use the default

initial TTL and we indicate the response TTL for each hop below the respective probe. Even though the responses to probes with initial TTLs 7, 8, and 9 all indicate N_0, the response TTL decreases because the routers are indeed further away. We can verify that a loop corresponds to a NAT box using the IP ID.

4.1.2 Statistics

Loops appear to be surprisingly common: 5.3% of the measured routes contained at least one of them. Moreover, loops were observed on routes towards 18% of the destinations, and 6.3% of all the addresses discovered in our traces were in a loop at least once. These last two values will grow if the measurements last longer, because new loops constantly appear, even after a month of probing. This effect is related to the observation that many loop signatures are rare: 18% of them were observed on only one round of measurements among 556.

We compared the traces obtained with classic traceroute and with Paris traceroute. The loops that are caused by per-flow load balancing are absent from the latter, and we estimate the proportion of such loops to be 87%. This value is necessarily imprecise, as routing dynamics may cause each traceroute to see slightly different paths, with different quantities of loops of each possible type. For instance, a small portion of loops, equivalent in quantity to 0.25% of the loops seen by classic traceroute, were seen only by Paris traceroute. By conducting the two traceroutes for each destination close together in time, we hope to have minimized the impact of routing dynamics.

Having filtered out loops due to per-flow load balancing, it was possible to ascribe other causes to many of the rest. Zero-TTL forwarding explained 6.9%, Unreachability messages explained 1.2%, and address rewriting explained 2.8% of the loops. We suppose the remaining 2.5% of loops to be caused by per-packet load balancing, but as of yet we cannot verify such an assertion.

4.2 Cycles

Formally, a measured route R is said to be *cyclic* on an IP address r if it contains r at least twice, separated by at least one address r' distinct from r. This distinction ensures that we do not misinterpret possible loops as cycles. A *cycle*'s signature is a pair (r, d) such that at least one measured route towards d is cyclic on r.

4.2.1 Possible causes

Just as loops can be generated by load balancing over paths that have a length difference of one, **load balancing** is also a possible cause for cycles when the difference in path lengths is greater than one. Similarly, an **unreachability message** (described in Sec. 4.1) issued by a router that has already appeared in the measured route can be responsible for a cycle. Finally, packets may truly follow cyclic routes when they are caught in a **forwarding loop**, which may happen during routing convergence.

We identify cycles caused by load balancing and unreachability messages in manners similar to the ways described in Sec. 4.1. To identify cycles that were induced by forwarding loops, we looked for some periodicity in the measured routes: we should repeatedly observe a fixed sequence of addresses. Paris traceroute also allows us to infer that all appearances of a given address indeed come from one router, by checking that the IP ID contains values that increment, and by a relatively small amount, with each cycle. We can infer in the same way that different addresses come from different routers.

4.2.2 Statistics

Cycles are less common than loops: they appear on only 0.84% of the measured routes. On the other hand, they appear on a broader range of the nodes: we observe cyclic measured routes towards 11% of the destinations, and the cycles concern 3.6% of the IP addresses discovered during our experiment. They are rarer events than loops: we observe 30% of cycle signatures on only one measurement round among 556. Overall, we observe each cycle signature in only 6.8 measurement rounds on average, or in 1.2% of the rounds.

By comparing the traces obtained with classic traceroute and Paris traceroute, we estimate that 78% of the cycles seen by classic traceroute are due to per-flow load balancing. Forwarding loops explained 20%, and unreachability messages explained 1.2% of the cycles. We suppose the remaining 1.1% of loops to be caused by the use of fake IP addresses and by per-packet load balancing, but as of yet we do not have a reliable method to prove these assertions.

4.3 Diamonds

Whereas loops and cycles can appear when we probe, as we do, with only one probe per hop, diamonds can only arise if probing involves multiple probes per hop. To study diamonds, we created two graphs for each of the 5,000 destinations: one composed from all the classic traceroutes towards that destination, and the other from the Paris traceroutes. Within a graph, a *diamond*'s signature is a pair (h, t) of IP addresses, such that there are $k \geq 2$ IP addresses $r_1, \ldots, r_k$ seen on measured routes of the form $\ldots, h, r_i, t, \ldots$.

Fig. 6 presents, on the left, a network with a load balancer L that splits packets along three different paths. On the right, we see one possible outcome from classic traceroute sending three probes per hop. For clarity, we omit the probe packets. The ordered pairs (L_0, D_0), (L_0, E_0), (A_0, G_0) and (B_0, G_0) are diamonds. Note that (C_0, G_0) is not a diamond because there is only one interface, D_0, between C_0 and G_0. Space does not permit us here to describe our observations of

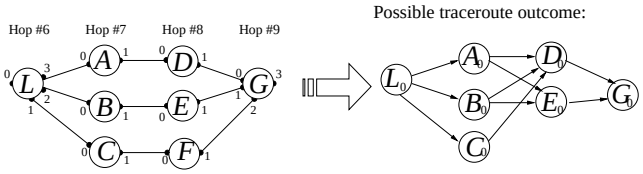

Figure 6: An example of several diamonds.

more complex diamond-like anomalies, such as those having greater lengths.

4.3.1 Possible causes

The main cause for the apparition of diamonds is **load balancing**. As presented in Sec. 2 (see Fig. 1), load balancing can cause traceroute to report false links, and we can easily have cases in which we falsely detect all possible links between interfaces at successive hop levels, inducing the appearance of diamonds. So although a small portion of the diamonds can be attributed to the true topology of

a load-balanced network, we expect most diamonds to be a consequence of the false links inferred from classic traceroute, and to reflect an erroneous view of the network.

4.3.2 Statistics

Diamonds are quite frequent. They are observed towards 79% of the destinations. Overall, we see 16,385 diamonds in our traces. When comparing this with the traces obtained with Paris traceroute, we estimate that 64% of the diamonds seen by classic traceroute are due to per-flow load balancing. Per-packet load balancing could explain most of the remainder, though we have as yet been unable to verify this.

5. RELATED WORK

The principal variants on Jacobson's traceroute [1] are Gavron's *NANOG traceroute* [14] and Toren's tcptraceroute [10]. NANOG traceroute labels IP addresses with the numbers of the ASes to which they belong. Tcptraceroute sends TCP probes, rather than the classic UDP or ICMP Echo probes, using Destination Port 80 to emulate web traffic and thus more easily traverse firewalls. As described in Sec. 2.2, this has the effect of maintaining a constant flow identifier. However, no prior work has looked at the use of a constant flow identifier to avoid traceroute anomalies.

Although there is an extensive literature on internet maps, and much work that uses traceroute, there have been few studies of anomalies as seen from the perspective of traceroute. Paxson's work on end-to-end routing behavior in the internet [15] uses traceroute to study routing dynamics, including "routing pathologies". Although some of these pathologies do relate to the traceroute anomalies we discuss in this paper (for instance, "routing loops" are one cause of "cycles", and "fluttering" is one cause of "diamonds"), his work focuses on the routing aspect of the anomalies and not on traceroute's deficiencies. Huffaker et al. [7] recognize that load balancing is an issue for constructing internet maps. Their work does not elaborate on the errors induced by load balancing or on a solution for the problem. Teixeira et al. [16] examine inaccuracies introduced into ISP maps obtained by Rocketfuel [8]. Their paper quantifies differences between the true and the measured topologies, and identifies routing changes in the midst of individual traceroutes as being responsible for a portion of the false links in the measured topologies, but does not touch on load balancing.

6. CONCLUSION

This paper makes the following contributions: (i) identification of the roles that packet header fields play in load balancing, and their interactions with classic traceroute; (ii) description of Paris traceroute, a new traceroute tool that allows more precise measurements of a route, and helps determine the causes of a number of traceroute anomalies; (iii) definition and characterization of three anomalies that appear in individual or repeated traceroute measurements: loops, cycles, and diamonds; and (iv) enumeration of possible causes for each of these anomalies and elaboration of experiments that test our hypothesis.

This work should inspire future research in many directions. We believe that we can improve Paris traceroute through the addition of algorithms to automatically find all interfaces of a given load balancer, and to differentiate per-flow from per-packet load balancers. Then, a study with more sources and more destinations would allow us to characterize the prevalence and characteristics of load balancers in the internet. This data would also serve to better quantify anomalies, including others not discussed here, and the effects of Paris traceroute on internet maps.

Finally, this research opens up the possibility for applications to choose among alternate paths through the internet through careful control of their header fields. If these paths prove to have different characteristics, we will have uncovered some path diversity in the network itself, without the need for an overlay network.

Acknowledgments

We are grateful to our shepherd Dave Andersen, to Ehud Gavron, and to the anonymous reviewers for their thoughtful comments. We also thank Neil Spring and Ratul Mahajan for their explanations of how Rocketfuel handles hops that respond with multiple interfaces. The Paris traceroute tool was developed with financial support from the French CNRS, as part of its contribution to the European Commission-sponsored *OneLab* project. The research using the tool was conducted with financial support from the French MNRT ACI *Sécurité et Informatique* 2004 grant, through the *METROSEC* project.

7. REFERENCES

[1] V. Jacobson, "traceroute," February 1989, see ftp://ftp.ee.lbl.gov/traceroute.tar.gz.
[2] J. Moy, "OSPF Version 2," RFC 2328, April 1998.
[3] R. Callon, "Use of OSI IS–IS for Routing in TCP/IP and Dual Environments," RFC 1195, December 1990.
[4] B. Quoitin, S. Uhlig, C. Pelsser, L. Swinnen, and O. Bonaventure, "Interdomain traffic engineering with BGP," *IEEE Communication Magazine*, May 2003.
[5] Cisco, "How does load balancing work?" see http://www.cisco.com/en/US/tech/tk365/technologies_tech_note09186a0080094820.shtml.
[6] Juniper, "Configuring load-balance per-packet action," from the JUNOS 7.0 Policy Framework Configuration Guideline, see http://www.juniper.net/techpubs/software/junos/junos70/swconfig70-policy/html/policy-actions-config11.html.
[7] 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.
[8] N. Spring, R. Mahajan, and D. Wetherall, "Measuring ISP topologies with Rocketfuel," in *Proc. ACM SIGCOMM*, 2002.
[9] J. Postel, "Internet control message protocol," RFC 791, September 1981.
[10] M. Toren, "tcptraceroute," April 2001, see http://michael.toren.net/code/tcptraceroute/.
[11] S. Bellovin, "A technique for counting NATted hosts," in *Proc. ACM SIGCOMM Internet Measurement Workshop*, 2002.
[12] J. Xia, L. Gao, and T. Fei, "Flooding attacks by exploiting persistent forwarding loops," in *Proc. ACM SIGCOMM Internet Measurement Conference*, 2005.
[13] 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.
[14] E. Gavron, "NANOG traceroute," May 1995, see ftp://ftp.login.com/pub/software/traceroute/.
[15] V. Paxson, "End-to-end internet packet dynamics," *IEEE/ACM Trans. Networking*, vol. 5, no. 5, 1999.
[16] R. Teixeira, K. Marzullo, S. Savage, and G. M. Voelker, "In search of path diversity in ISP networks," in *Proc. ACM SIGCOMM Internet Measurement Conference*, October 2003.