UNIVERSITÀ DEGLI STUDI DI NAPOLI FEDERICO II

Scuola Politecnica e delle Scienze di Base

Corso di Dottorato di Ricerca in Ingegneria Informatica ed Automatica
XXVI Ciclo
Dipartimento di Ingegneria Elettrica e delle Tecnologie dell'Informazione

Issues, Methodologies and Solutions for Internet Path Tracing

Pietro Marchetta

Ph.D. Thesis

Tutor
Prof. Antonio Pescapé

Coordinator
Prof. Francesco Garofalo

March 2014

A Valeria.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In this chapter, we introduce the scenario in which our study was conducted, outlining the motivations stimulating our research work. Afterwards, we briefly describe the contributions provided by this thesis while the ending part of the chapter outlines its organization.

1.1 The Internet scenario

The Internet is the largest distributed system ever built by the human kind: this critical infrastructure significantly contributes to the economic wealth of advanced and emerging societies, serving applications used by 2.1 billion of users worldwide according to recent studies [1].

While we increasingly depend on the Internet for our professional, personal and political lives, our understanding of its underlying structure, performance limits, dynamic, and evolution appears today still largely inadequate [2]. An accurate and exhaustive knowledge of how this infrastructure actually operates is of the utmost importance to guarantee high operational standards and to determine a positive future evolution of such an increasingly important communication system. However, gathering this knowledge is a particularly challenging task due to several factors: the Internet network is characterized by (i) high level of heterogeneity – in terms of network equipments and communication technologies, (ii) high level of dynamicity – in terms of how the traffic exchanged between two nodes of the network flows across the physical infrastructure, (iii) numerous networking protocols and their implementations variously interacting to make feasible the communication between end points of the network. An additional factor dramatically

increasing the complexity of gathering a clear understanding of the current operational status of the network is the radically distributed ownership of the Internet among its constituent parts. From the architectural point of view, Internet is partitioned in thousands of private and public independent networks called Autonomous Systems (hereafter simply AS). An AS is defined as a domain in which routers and hosts are managed by a single administrative authorities (a university, a company, or an Internet Service Provider – ISP), that expose a clearly defined routing policy to the rest of the Internet [3]. An AS has full control and visibility on its infrastructure but has no control or visibility on the infrastructure managed by other ASes. The current evolution of the Internet is largely determined by the forced cooperation and competition between these building blocks.

In this challenging scenario, researchers have developed over the years monitoring and measurement methodologies and techniques in order to investigate global and local aspects of the Internet, thus improving our visibility and comprehension of this critical yet largely opaque ecosystem of interconnected networks [4]. Through Internet measurements, we can better understand how the system behaves in practice, its deficiencies and vulnerabilities as well as its evolution over time.

1.2 On the importance of tracing Internet paths

We focus our attention on the methodologies and techniques designed to trace Internet paths. The goal of these techniques is to infer the network path followed by the traffic exchanged between two nodes of the network. In this thesis, we define an Internet path as the sequence of routers and links traversed by the traffic sent from a source node to a destination node.

An Internet path is determined by the combined action of intra- and inter-domain routing. The routing is in charge of disseminating routing information between routers and dictates how this information must be used to forward traffic to its destination. The inter-domain routing determines the sequence of ASes the traffic will traverse to reach its destination while the intra-domain routing dictates how the traffic is forwarded within each AS network, i.e. the sequence of routers the traffic must traverse to the destination or the exit point of the AS towards the next AS along the path.

Reconstructing the sequence of routers and links traversed by the traffic exchanged between two network nodes by inspecting the information related to the network routing

is not practically feasible because limited or no information is available respectively about the inter- and intra-domain routing as we explain in the following. The mostly adopted inter-domain routing protocol is the Border Gateway Protocol (hereafter BGP) that allows ASes to advertise their *best routes* to the neighboring ASes: a route contains several information including the sequence of ASes that will be traversed to reach a given block of addresses (i.e. a network prefix). By properly configuring the border routers of their infrastructure, an AS can implement high-level policies typically driven by economic constraints: essentially, an AS may accept or decline routes advertised by another AS, and advertise or not its best routes depending on the neighboring AS. Since BGP routes do not enclose any information about the internal infrastructure of each AS, observing the inter-domain routing messages provide only a coarse-grained information about the network path of interest: no information is provided about the traversed routers. To make matters worse, we only have a suboptimal access to the inter-domain routing information. Indeed, an exhaustive view of the global inter-domain routing requires the deployment of specific authorized network equipments (i.e. monitors) in any location of the Internet where ASes exchange inter-domain routing messages. Currently, research projects systematically collecting inter-domain routing messages such as RouteViews [5], RIPE RIS [6], PCH [7], etc., own a forcedly reduced amount of monitors: although located in privileged positions like Internet Exchange Points (i.e. large facilities typically owned by private companies where ASes may easily connect to each other), the information collected provides just a partial visibility on the global inter-domain routing.

As for the intra-domain routing, an AS can deploy any routing policy or combination of policies independently from the other ASes. However, without a direct access to the infrastructure managed by the AS, it is not possible to observe the intra-domain routing messages exchanged between the routers. Since the Internet operational climate based on the competition and cooperation among ASes generally discourages sharing data with researchers – ASes typically consider strictly confidential any information related to the managed infrastructure – no information is available to the research community about the intra-domain routing of the ASes composing the Internet.

Hence, researchers and network operators willing to understand the sequence of routers and links traversed by the traffic exchanged between any two end points of the network cannot rely on the partial or absent information on the inter- and intra-domain routing. Furthermore, even a theoretical exhaustive knowledge about the global routing in the

Internet would not be helpful when investigating specific characteristics of the network path such as its performance (e.g. delay) or in network troubleshooting operations aiming at detecting network failues (e.g. physical link disconnection, software errors, router misconfiguration, etc.).

For these reasons, since the dawning of the Internet, researchers developed measurement techniques purposely designed to trace Internet paths (we detail these techniques in the next chapter). Internet path tracing techniques have been extensively used to both gather fundamental knowledge about essential properties of the Internet and as core components of large distributed systems (e.g. overlay networks). We provide in the following an overview of the applications enabled by path tracing techniques. This overview is not intended to be exhaustive but to provide an overall idea of the importance of tracing Internet paths for the research community.

1.2.1 Assessing network topology

Tracing Internet paths proved to be extremely helpful especially when the final goal is to reverse engineer the topology of the Internet, since it provides essential information about the network topology.

The Internet topology has attracted the interest of the research community for decades [8]. The research on this theme is concerned with the study of the various types of connectivity structures that are enabled by the layered architecture of the Internet [9]: structures related to the physical infrastructures (such as routers, switches, etc. and interconnections among them) and logical structures as the ones that can be defined at the several layers of the TCP/IP stack (IP-level graph, AS-level graph, etc.). An accurate and extensive knowledge of the topology of the Internet is of the utmost importance for the community (i) to design and evaluate applications and innovative routing protocols through realistic network emulation and simulation; (ii) for improving and testing novel network-engineering practices; (iii) to manage large-scale complex and highly dynamic networks seeking for up-to-date information on the current status of the network; (iv) to verify, correct, and improve various desirable aspects of the global Internet including its robustness, reliability, efficiency and security. More in general, an accurate and exhaustive knowledge of the network topology is an important aspect for a deep understanding of the complex and ever-evolving ecosystem the Internet is.

The topology of Internet is typically investigated at several levels of abstraction and

tracing Internet paths is the key operation to gather this knowledge. Indeed, the most common approach is to sample the topology by performing large scale measurement campaigns tracing a large amount of Internet paths. As we will deepen in the next chapter, path tracing techniques report the traversed path essentially in terms of a sequence of IP addresses: typically, one address (a network interface) for each traversed router is reported. By tracing a large amount of Internet paths from multiple vantage points toward a large amount of destinations, researchers can obtain a first rough representation of the topology of the Internet by intersecting the collected sequences of IP addresses, i.e. the IP-level topology. Since a router may have dozens of interfaces, different IP addresses belonging to the same router may potentially appear as different nodes in the IP-level graph: for this reason, *alias resolution* techniques [10] are applied to merge those addresses owned by the same network device. With alias resolution, researchers transform the IP-level topology in a router-level topology. Furthermore, by geographically aggregating the routers, the router-level topology can be transformed into a Point of Presence-level (PoP-level) topology. Finally, by associating to each IP address the owner AS, one can also infer the presence of traffic exchanged between ASes, thus the IP-level topology can be used to gather visibility on the AS-level connectivity (AS-level topology): an approach widely adopted to complement the partial information about the AS-level connectivity that can be derived from the BGP routes [11, 12, 13, 14].

Essentially, tracing Internet paths is at the basis of the reverse engineering of the topology of the Internet: exploring large amount of Internet paths with the state of the art techniques provides the basilar information to reconstruct the topology at all the abstraction levels, an approach employed by several active research projects [2, 15, 16, 17]. On the other hand, the accuracy and completeness of the reconstructed topologies critically depend on the accuracy and precision of the adopted path tracing techniques.

1.2.2 Assessing network routing

There has been a large amount of scientific works demonstrating the utility of Internet path tracing as an invaluable source of information on the routing in the Internet useful for disparate purposes.

Monitoring over time Internet paths allowed researchers to answer fundamental questions about the Internet and its internal mechanisms: for instance, thanks to the seminal work of Paxons [18] recently reappraised by Cunha *et al* [19], we learned that routes in

the Internet remain stable for long period of time but with short-lived periods of instabilities. By tracing paths between pairs of network nodes, we also learned that routing in the Internet is mainly asymmetric [18, 20, 21]: the network path connecting A to B is often different from the path connecting B to A.

Other researchers relied on path tracing techniques to investigate and pinpoint abnormal behaviors of the network routing [18, 22, 23]. For instance, Spring *et al.* [22] used path tracing to identify the root causes of path inflations, i.e. Internet paths significantly longer than strictly necessary, discovering AS peering policies and latency-sensitive intra-domain routing as the two most important causes. About ten years later, Gupta *et al.* [23] discovered how path inflations still exist especially in African developing countries where path tracing revealed how local routes often detour through Europe apparently due to the lack of interconnectivity between local ISPs. Beside routing pathologies, researchers systematically exploited path tracing to detect and monitor transient and permanent network failures exposed by reachability problems, i.e. the inability of data packets to reach network prefixes marked as reachable according to the available information on the routing [24, 25, 26, 27].

Tracing network paths proved helpful also to predict other non-measured Internet routes [28, 29, 30]: for instance, Madhyastha *et al.* [28] proposed to stitch together path segments extracted from previously traced paths to predict the route between arbitrary pairs of network nodes as well as the latency of the communication. Tracing Internet paths helps also when building and maintaining efficient overlay networks [31, 32, 33]: for instance, one can easily verify if two apparently disjoint overlay paths share or not common underlying links [34]. Similarly, content distribution networks continuously trace Internet paths and their properties to select the best content server to serve the users [35].

Assessing the routing by tracing Internet paths allowed also to quantify the importance of individual countries and their policies (e.g. censorship) on the flow of international traffic: Karlin *et al.* [36] observed a primary role for the international reachability for US, Great Britain and Germany, while only a marginal role for other countries such as China and Iran. Similar studies relied on path tracing to directly locate the censorship in the Internet [37]. Note that the path tracing-derived knowledge about the routing can also be used to perform network attacks such as the link-flooding attacks whose goal is to disconnect entire portions of the network from the Internet by targeting a limited amount of network links [38, 39].

Several other works exploited the information collected by tracing Internet paths to (i) accurately geolocate resources and services over the Internet [40, 41, 42]; (i) to detect violations of the traditional destination-based routing scheme [43]; (iii) to shed light on the intra-domain routing of certain ASes [44]; (iv) to quantify the deployment in the Internet of specific technologies and network engineering practices such as Multiprotocol Label Switching (MPLS) [45]; or (v) to identify the AS causing inter-domain route changes [46].

The brief overview reported above is by no means exhaustive but it demonstrates the existence of a large amount of applications relying on Internet path tracing techniques.

1.2.3 The need for research

The applications reported in the previous section demonstrate the great utility of path tracing techniques not only for a deep understanding of such a complex and largely opaque ecosystem of networks but also for managing the network, troubleshooting connectivity problems, building efficient overlay systems, locating the censorship over the Internet, etc. Unfortunately, despite the numerous applications, the currently available path tracing techniques suffer from several severe limitations potentially causing the information obtained about the paths under investigation to be inaccurate (e.g. the traced path may not perfectly correspond to the actual path) and incomplete (e.g. the traced path may miss traversed links or routers).

Some limitations are intrinsic to the path tracing techniques: for example, when using state of the art path tracing techniques, users are well aware that no information is provided about traversed devices implementing up to the second layer of the TCP/IP stack such as switches, bridges, etc. Other limitations, however, are not so straightforward and may cause the users to draw wrong conclusions about the path under investigation: for instance, path tracing techniques may suggest that the traffic is flowing across an AS that is not actually traversed on the path to the destination.

We observed that some limitations attracted large interest from the research community with significant advancements. However, other important limitations with potentially large impact on the applications relying on Internet path tracing have been largely ignored. The lack of measurement methodologies and techniques designed to detect, quantify, and resolve these limitations as well as the potential impact on the applications relying on Internet path tracing motivated the research activity at the base of this thesis.

1.3 Thesis contributions

In this section, we provide an overview of the contributions presented in this thesis.

Addressing well-known unresolved path tracing limitations

We designed, implemented and evaluated a set of active probing techniques (i.e. techniques injecting into the network probe packets – also referred to as probes – purposely crafted to study the reaction of the network and draw conclusions about its properties), designed to detect, quantify and possibly resolve or mitigate, important unresolved limitations affecting state of the art path tracing techniques. The proposed techniques are built on top of an innovative measurement traffic composed by probe packets equipped with optional headers of the Internet Protocol (IP). These particular probe packets proved to collect additional valuable information about the traversed paths, information potentially useful to address limitations in Internet path tracing.

More precisely, we propose active probing techniques and methodologies to (1) detect and locate *hidden routers* in traced paths, i.e. devices configured to be transparent to the path tracing techniques having a great impact on the inferred network topological properties; to (2) detect *third-party addresses*, an important source of inaccuracy especially when the final goal is to identify the ASes traversed towards the destination; (3) accurately dissect the RTT experienced along the path under investigation overcoming the misleading and inaccurate information provided by the path tracing techniques. We also present in this thesis our attempt to explore (4) an innovative path tracing approach completely alternative to the universally adopted mechanism with the goal of finding complementary solutions to mitigate the limitations of path tracing.

Assessing new limitations in Internet path tracing

For the first time in literature, we report in this thesis two new limitations of the state of art path tracing techniques we experimentally observed. We demonstrate how path tracing techniques may induce one to (i) overestimate the number of equal cost paths towards the destination and to (ii) infer non-existing changes in the network routing. Essentially, we demonstrate that the representation of the path obtained through state of the art path tracing technique may potentially be a strongly biased representation of path under investigation.

An architecture for Internet path tracing on demand

Finally, by reviewing the literature, we observed how researchers, network operators and systems relying on path tracing are often interested in exploring and tracing particular Internet routes to obtain information about the specific phenomenon or aspect of interest (e.g. the topology of a particular AS, the connectivity between two ASes, etc.). However, the ability to explore particular Internet routes strongly depends on (i) the availability of multiple vantage points well-distributed in the Internet (i.e. the machines issuing path tracing measurement) but also (ii) the ability of identifying which specific vantage point to use and destination to target in order to explore the route of interest. Regarding the first challenge, researchers might use the vantage points made available by several experimental testbeds. However, the great heterogeneity of interfaces and internal mechanisms of these testbeds represents a great disincentive: as a consequence, researchers tend to use always the same testbed and its relatively small amount of vantage points. Regarding the second point, to the best of our knowledge, there are no available methodologies to identify which particular path tracing measurements to issue in order to explore a particular path of interest.

For this reason, we designed a general architecture and implemented and evaluated a first implementation of this architecture called PANDA. PANDA is designed to offer a path tracing on-demand service and to satisfy complex user queries requesting path tracing measurements for Internet routes with a priori known characteristics. This goal is reached by also aggregating under a unique interface vantage points made available by multiple experimental testbeds, thus masking the great heterogeneity of the interfaces of the aggregated experimental testbeds.

1.4 Thesis organization

The rest of this thesis is organized as follows. In Chapter 2, we discuss the path tracing techniques documented in literature as well as their limitations. We also discuss the two open challenges researchers and network operators must often deal with when relying on path tracing: the need for vantage points and how to select the path tracing measurements to perform in order to explore the particular paths of interest. In Chapter 3, we detail our research activities addressing important largely-ignored limitations of the state of the art techniques (i.e. hidden routers, third-party addresses, misleading intermediate RTT

values) and we also present an innovative path tracing solution totally alternative to the approach universally adopted. Successively, in Chapter 4, we document for the first time in literature two additional experimentally-observed limitations in Internet path tracing. In Chapter 5, we describe the general path tracing architecture and a first its implementation, we named PANDA, designed to support researchers, network operators and systems relying on Internet path tracing: our system is able to explore Internet routes with a priori known characteristics from the vantage points made available by multiple experimental testbeds. Finally, Chapter 6 ends the thesis with concluding remarks.

Chapter 2

Tracing Internet paths

In this chapter, we first detail the techniques proposed in literature for tracing Internet paths. Then, we discuss the limitations affecting these techniques and the partial solutions documented in literature. Finally, we conclude the chapter by highlighting two open challenges researchers and operators using path tracing techniques often must deal with: the need for multiple vantage points and the problem of selecting which path tracing measurement to perform in order to explore the Internet paths of interest.

2.1 Techniques

How to trace Internet paths in the lack of control over the infrastructure has gathered very early the interest of the research community since this operation proved helpful for monitoring and managing the network. In this section, we describe the techniques proposed in literature for tracing Internet paths. The goal of these techniques is to shed light on the path followed by the traffic sent toward a network destination: more precisely, the objective is to obtain accurate information about network routers and links traversed along the path. The available techniques provide an IP-level view of the path reconstructed essentially in terms of a sequence of IP addresses, one address of each traversed router.

2.1.1 The Record Route IP option

Researchers working on the standardization of the IP protocol recognized already in 1981 the importance of an embedded mechanism able to trace Internet paths. Among the optional headers of the IP protocol (commonly referred to as IP options), researchers

introduced an header specifically designed to trace the Internet path: the Record Route option (hereafter simply RR option) [47]. When a probe packet is equipped with the RR option, each router along the path is requested to insert one of its address in a pre-allocated area of the IP header of the packet. In this way, the packet collects one address for each router traversed along the path towards the destination: by inspecting the addresses registered within the RR option, one can reconstruct the traversed path essentially in terms of a sequence of IP addresses. Since we use this IP option for part of our contribution, we provide much more details about the RR option such as the format in Chapter 3.

A positive feature of using the RR option is the ability to trace the path by using only one packet compared to other approaches that rely on the injection of multiple packets potentially experiencing completely different network conditions and paths. On the other hand, this approach is affected also by some relevant limitations: due to space constraints (the space allocated for IP options is limited to 40 bytes), no more than nine addresses can be registered by the RR option. Since network paths consist of 15 hops on average [18], a significant portion of the path is potentially missed when using this approach discouraging a wide adoption of this approach as a stand-alone tool for tracing Internet paths.

2.1.2 Van Jacobson's Traceroute

Traceroute [48] together with its variants and optimizations represents today the standard de facto for tracing Internet paths. By injecting purposely crafted probe packets, Traceroute is able to trace the path from the machine under control (hereafter also referred to as Traceroute originator or vantage point) toward any network destination since it does not require the control on the targeted device: this universality is a key factor of its large success.

According to the RFC1812 [49], each router forwarding a packet must first decrease the value of the time-to-live (TTL) field of the IP header. When the TTL expires (i.e., reaches zero), the involved router states that the packet has consumed enough network resources along its travel: the packet is dropped and an ICMP Time Exceeded message is sent back to the Traceroute originator. This basic conservative mechanism of the network can be used to infer the network path as proposed by Van Jacobson [48] with Traceroute. The basic mechanism of Traceroute is depicted in Fig. 2.1: Traceroute injects into the network a sequence of UDP probe packets sent towards the destination. A UDP probe

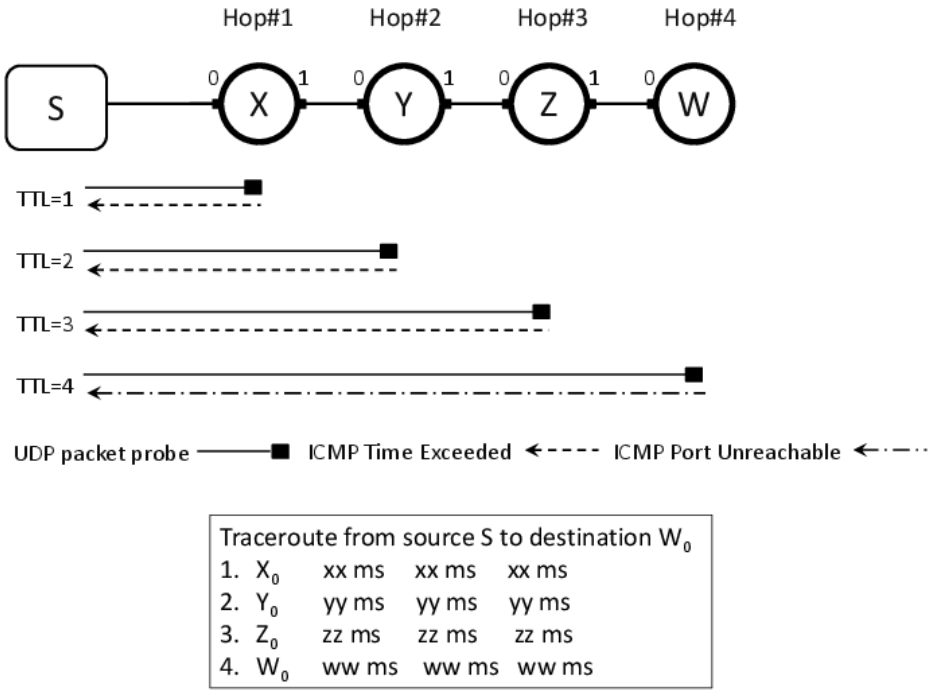


Figure 2.1: Traceroute basic mechanism – Traceroute reports the source addresses of the ICMP Time Exceeded messages solicited by issuing TTL-limited probe packets towards the targeted destination. Traceroute also reports for each hop the length of time it takes to send the probe packet and collect the ICMP Time Exceeded message.

packet is an IP packet carrying as transport protocol a UDP header with the destination port set to a high and presumably unused port: a similar packet is intended to solicit an ICMP Port Unreachable message from the targeted destination. The UDP probe packets are injected with limited TTL values, each time incrementing the TTL from an initial value of one. In this way, the technique solicits ICMP Time Exceeded messages from the routers encountered along the path. By extracting the source address from the collected ICMP error messages, Traceroute reports an IP address (i.e. an interface) for each traversed router. Essentially, a probe packet with an initial TTL value of i solicits a ICMP Time Exceeded reply from the i-th hop decreasing the TTL along the path towards the destination. The path tracing process stops as soon as an ICMP Port Unreachable message is received (i.e., the destination has been reached). The TTL values of the injected probe packets is gradually increased since the length of the path under investigation is not a priori known and it is important to avoid the overloading of the destination with multiple UDP probe packets. In addition, Traceroute reports the hop-by-hop Round

```
traceroute to www.google.com (173.194.35.48), 30 hops max, 60 byte packets
 1  143.225.170.254 0.283 ms  0.296 ms  0.325 ms
 2  143.225.190.98  0.251 ms  0.245 ms  0.238 ms
 3  143.225.190.146 0.339 ms  0.354 ms  0.370 ms
 4  193.206.130.9  0.461 ms  0.438 ms  0.431 ms
 5  90.147.80.169  24.505 ms  23.748 ms  25.929 ms
 6  90.147.80.165  18.272 ms  71.520 ms  23.073 ms
 7  90.147.80.62  15.193 ms 11.980 ms 31.134 ms
 8  90.147.80.17  17.860 ms 16.605 ms 17.821 ms
 9  90.147.80.73  31.818 ms 26.633 ms  31.029 ms
10  193.206.129.130  26.588 ms 15.512 ms  30.485 ms
11  209.85.241.67  15.978 ms  17.905 ms  15.883 ms
12  173.194.35.48  16.611 ms  17.571 ms  16.583 ms
```

Figure 2.2: A sample output of the standard UNIX implementation of Traceroute toward google.com.

Trip Time (RTT) computed as the length of time between sending the data packet and receiving the corresponding ICMP Time Exceeded reply. Typically, Traceroute injects three probe packets for each TTL value.

Fig. 2.2 reports an example of the output of the standard UNIX implementation of Traceroute used to trace the path from our laboratory at the University of Napoli towards the domain google.com: the traced path consists of 12 hops. For each hop, Traceroute reports an address and the corresponding RTT values: for instance, injecting probe packets with TTL values set to 10 solicit ICMP Time Exceeded replies with source address 193.206.129.130.

According to a recent survey among the members of the North American Network Operators' Group (NANOG) [50], together with the tool Ping, Traceroute is the most widely adopted network diagnostic technique for monitoring, managing and troubleshooting the network, and definitely the number one approach for tracing Internet paths [51]. Every operative system provides an implementation of this technique.

2.2 Limitations

Extensive literature demonstrated that, despite the great number of applications relying on path tracing, the techniques used to trace Internet paths are affected by several limitations. Essentially, Traceroute, the most widely adopted approach for tracing Internet paths, may provide incomplete or inaccurate information about the paths under investigation. In this section, we describe the sources of inaccuracy, their impact and the solutions documented in literature: many of the described limitations appear not definitively resolved and motivate the continuous interest of the research community on this theme. In the next Chapter, we detail our contribution to investigate and resolve several among

the most severe limitations affecting Internet path tracing techniques while in Chapter 4, we experimentally demonstrate the existence of two new additional limitations in path tracing not recognized before.

2.2.1 Anonymous routers

The information obtained when tracing Internet paths can be inaccurate and incomplete. A severe source of incompleteness is represented by unresponsive routers, also known in literature as anonymous routers. Anonymous routers are devices configured to discard packets with an expiring TTL without generating the ICMP Time Exceeded response [52]. The presence of routers configured to not reply in case of path tracing heavily impacts the reconstruction of the network topology and makes harder in practice to tell if the lack of responses from the network is actually caused by ongoing network failures. Anonymous routers also further complicates the geolocation of network resources and services and the ability to infer properties of Internet routes such as the traversed ASes towards the destination.

When soliciting unresponsive routers, Traceroute may indefinitely wait for the response. To deal with this problem, implementations of Traceroute normally employ a mechanism based on timeouts: if the response is not collected within a given length of time, the corresponding hop along the path is marked as unresponsive (typically indicated with the symbol asterisk – "*") and a new probe packet with an increased TTL value is injected into the network. Fig. 2.3 reports a path tracing scenario including anonymous routers.

It is worth noticing that routers may also become unresponsive for reduced amount of time in case of ICMP rate limiting: indeed, routers can be configured such that when ICMP messages are solicited with rate exceeding specific thresholds, this is interpreted as a potentially malicious behavior and the ICMP replies are temporarily disabled.

The incompleteness caused by anonymous routers represents a severe source of uncertainty for many applications relying on path tracing especially when the final goal is to reconstruct the topology of the network since it is not trivial to state if multiple unresponsive routers observed along different paths are determined by the same unique unresponsive router (see Fig. 2.4).

The solutions documented in literature are mainly focused on mitigating the impact of anonymous routers on the reconstructed Internet topologies. A typical approach is to

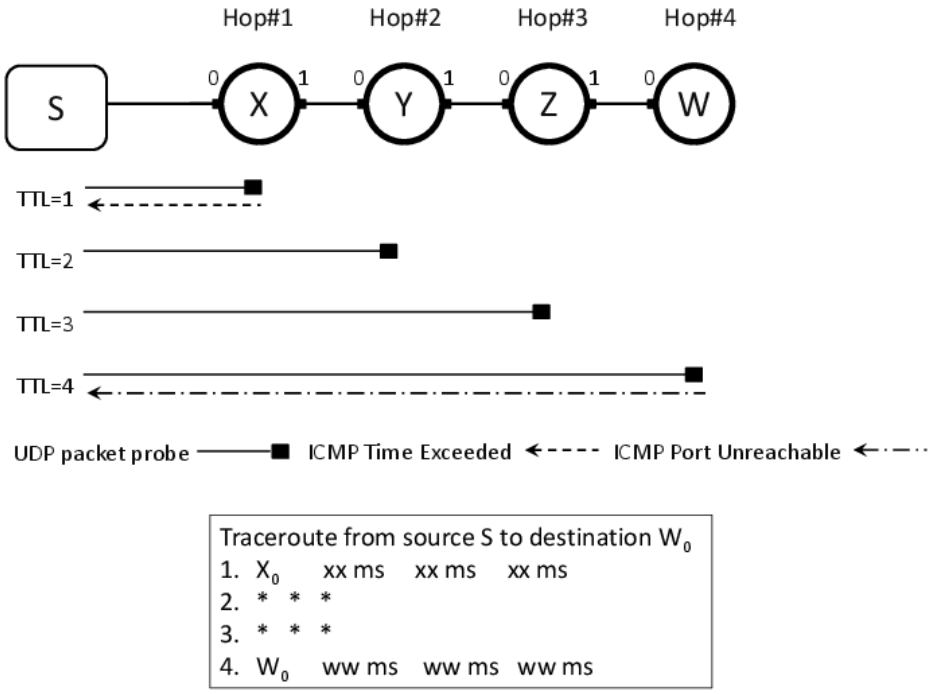
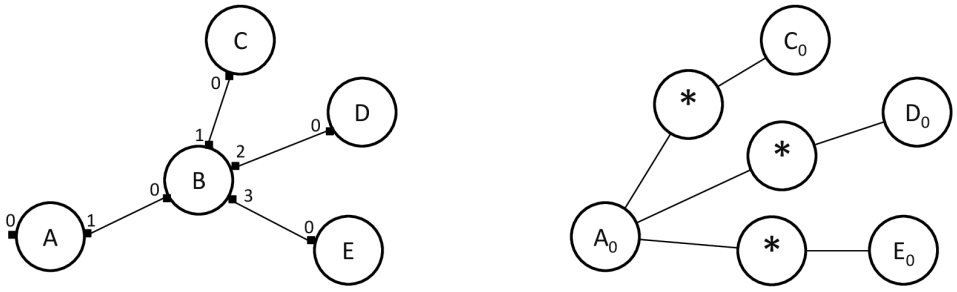


Figure 2.3: Traceroute and anonymous routers – The second and third hop along the path are anonymous routers discarding the Traceroute probe packets and not providing ICMP Time Exceeded messages. The Traceroute originator is not able to identify these nodes that appear marked with the symbol "*" in the trace.

analyze the IP level graph obtained by performing multiple Traceroute measurements: in this graph each symbol "*" in a trace is treated as an independent node. The techniques investigate the properties of the graph to identify the "*" caused by the same anonymous router to be merged. Yao *et al.* [53] formulates the problem of identifying the same anonymous routers appearing in multiple traced paths as an optimization problems whose objective is to find the minimum size topology obtained by merging the "*" under specific consistency constraints. The authors demonstrated that this problem is NP-complete and proposed a computationally expensive heuristic to solve the problem ($O(n^5)$ with n indicating the number of anonymous routers observed in the collected paths). Jin *et al.* [54] proposed an ISOMAP based dimensionality reduction approach to solve the problem with a complexity $O(n^3)$. Finally, Gunes *et al.* [52] proposed a graph-based induction approach to identify common structures within the topology graph and use them to resolve anonymous routers achieving a significant lower complexity than the previous approaches. All these solutions try to mitigate the impact of anonymous routers



(a) Actual topology. (b) A possible inferred IP-level topology.

Figure 2.4: The impact of anonymous routers when inferring the topology of the network – B is an anonymous routers causing the inferred IP-level topology to be biased.

on the inferred topology map. However, when the final goal is to infer the properties of a single Internet path, none of these approaches appears useful, i.e. anonymous routers still cause incomplete information about the path under investigation.

To the best of our knowledge, only the authors in [55, 56] tried to assign an IP address to anonymous routers while tracing Internet paths by exploiting the RR option since an anonymous router may register in the IP-option-equipped probe packet one of its address. The authors jointly exploited TTL-limited probe packets and RR option to trace the Internet paths. Unfortunately, besides the limited exploring range of the RR option, aligning the Traceroute and the RR traces proved to be a particularly complex and not definitively solved problem [56]: authors proposed disjunctive programming logic, an extremely computationally expensive approach hard to replicate [10]. To the best of our knowledge, there are no available solutions able to solicit replies from routers configured to not generate ICMP Time Exceeded replies.

2.2.2 Hidden routers

One of the most severe source of both incompleteness and inaccuracy when tracing Internet paths is represented by hidden routers: devices invisible when the paths are traced with Traceroute. The most important impact of hidden routers is related to the inference of the network topology. However, anonymous and hidden routers also greatly increase the complexity of managing and troubleshooting the network since the obtained information about the current status of the network is incomplete and thus potentially misleading.

Hidden routers are network devices configured to not decrement the TTL of the for-

warded packets. As a consequence, these devices are totally invisible to Traceroute. According to [56], "*hidden routers are caused by certain configurations of multi-protocol label switching (MPLS) and result in missing nodes and incorrect link inferences*". In addition, very recently, researchers also observed that *middleboxes* may also act as hidden routers, thus being non-reported by Traceroute. A middlebox is defined as any intermediary box performing functions apart from standard functions of an IP router on the data path between a source host and destination host [57] such as NAT, Firewalls, etc. For instance, the CISCO Firewall Adaptive Security Appliance [58] may be purposely configured to not appear in Traceroute traces while other middlboxes may refresh the TTL of the incoming packets [59] thus preventing Traceroute to explore the remaining portion of the path. Fig. 2.5 reports the path tracing scenario containing hidden routers. For instance, considering again the sample trace reported in Fig. 2.2, the traffic sent towards the destination might have traversed many more devices including hidden routers potentially affecting the experienced intermediate delays and the topological properties we may infer by observing this traced path. Note that, since Traceroute is often the only mechanism we have to explore the path towards the destination, we cannot estimate if the traced path is accurate or not, without a ground truth of the topology very rarely available due to the lack of collaboration with the ASes.

The impact of hidden routers is potentially disruptive when the goal is to to reconstruct the topology of the network. A clear example of this impact is reported in Fig. 2.6. Fig. 2.6a reports the actual router-level topology: a router B is connected to the routers A, C, D and E. Let us assume that the router B is an hidden router: Fig. 2.6b reports an IP-level topology we could infer by using Traceroute as path tracing technique in a large scale experimental campaign designed to discover all the possible links among these routers. Comparing Fig. 2.6b to Fig. 2.6a, we can easily evaluate the impact of an hidden router: the inferred topology is (a) incomplete due to missing routers (the router B) and links (the links between A and B, B and C, B and D, B and E) but also (b) inaccurate since the inferred links do not actually exist. Hence, hidden routers not only do not appear in the reconstructed topology but also impact the inferred properties of the topology such as the degree distribution (note the overestimation of the degree for the router A).

While other limitations of Traceroute have been investigated for years, the real magnitude of the phenomenon related to hidden routers has been largely ignored. Sherwood *et al.* [55, 56] proposed a solution based on a novel Traceroute enhanced by the RR op-

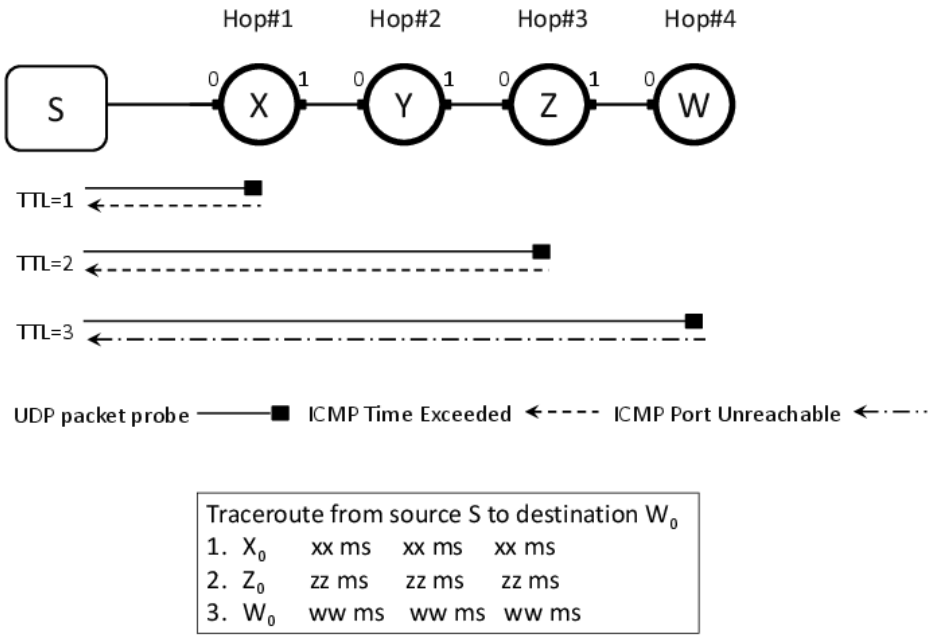
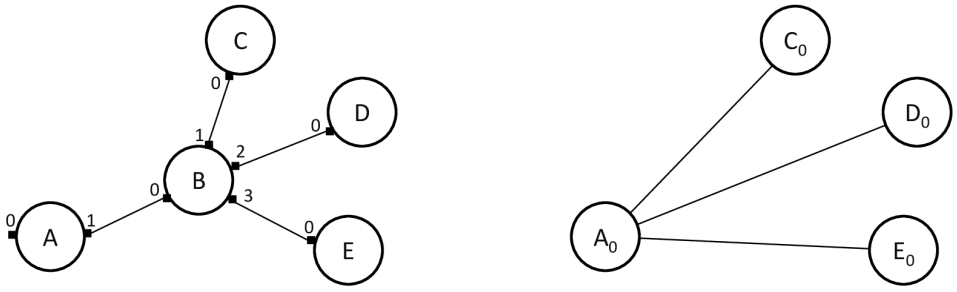


Figure 2.5: Traceroute and hidden routers – The second hop along the path is an hidden router forwarding the Traceroute probe packets without decrementing the TTL. Traceroute does not report any information about this device.

tion to identify load balancers, anonymous routers, addresses owned by the same device and, possibly, hidden routers. Thanks to the RR option, the injected probes register along the path additional IP addresses potentially revealing IPs of devices not appearing in the Traceroute trace. Authors observed 329 hidden routers, about 0.3% of all the discovered devices. Unfortunately, as explained earlier, the approach proposed in [55, 56] suffers from the nine-hops limited exploring range of the RR option and the not definitively solved problem of aligning the Traceroute and RR traces. Middleboxes may also act as hidden routers. The most recent work from this point of view is the one proposed by Detal *et al.* [60]. Authors proposed *tracebox*, an extension of Traceroute that (i) sends IP packets containing TCP segments with limited TTL values, and (ii) analyzes the packet encapsulated in the solicited ICMP Time Exceeded packets in search of any modifications potentially revealing the presence of middleboxes. However, the approach proposed in [60] does not allow one to discover hidden devices that do not modify the header or the content of the forwarded packets.



(a) Actual topology. (b) A possible inferred IP-level topology.

Figure 2.6: The impact of hidden routers when inferring the topology of the network – B is an hidden router causing the inferred IP-level topology to incomplete (i.e. missing nodes and links) and inaccurate (i.e. containing non-existing links).

2.2.3 Third-party addresses

Another severe source of inaccuracy largely ignored when tracing Internet paths is represented by third-party (TP) addresses, i.e. addresses reported by Traceroute but associated to interfaces not actually traversed by the traffic sent towards the destination. TP addresses may potentially induce one to conclude that the traffic sent towards the destination is traversing an AS or a network link that is not actually traversed.

According to the RFC1812 [49], the source address of an ICMP error packet must correspond to the outgoing interface of the ICMP reply, rather than the interface on which the packet triggering the error was received (i.e. the incoming interface). Hence, the addresses reported by Traceroute may correspond to the interface used by the intermediate routers to forward the packets back to the Traceroute originator. While it is commonly believed that routers provide to the Traceroute originator the incoming interface [51], routers implementing this portion of the RFC1812 exist such as the CISCO 3660 routers running IOS 12.0(7)XK1 [61]. When a router exposes to Traceroute an address not associated to the interfaces traversed by the issued traffic, Traceroute may potentially suggest that the traffic sent towards the destination is flowing across an AS that is actually not traversed.

For instance, the trace from S to D in Fig. 2.7 contains the sequence (a, b, c) of IP addresses, where a and b are associated to the incoming interfaces of routers A and B respectively, and c is the interface used by router C to send ICMP Time Exceeded messages to the Traceroute originator. The IP c is a TP address since it is associated – in

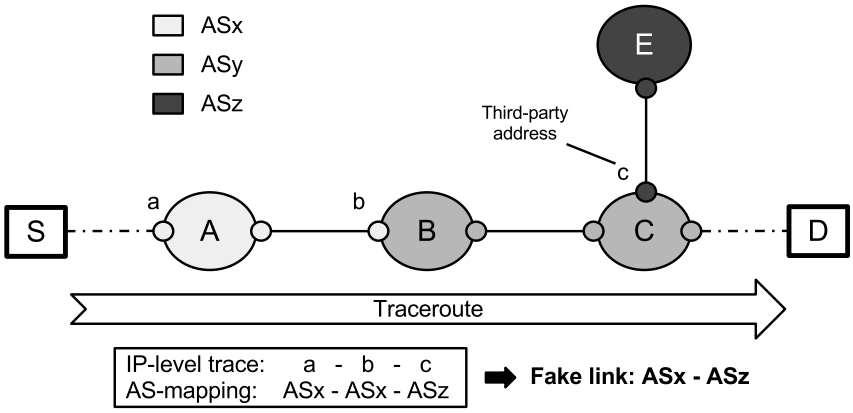


Figure 2.7: TP addresses inducing the inference of false AS links.

this specific trace – to an interface not effectively traversed by the packets sent from S to D. The same scenario also explains why Traceroute reporting TP addresses may suggest that the traffic is flowing across not actually traversed ASes. Consider again Fig. 2.7: if the IP address b belongs to ASx, and c belongs to the ASz addressing space, then a correct IP-to-AS mapping of the addresses contained in the trace may induce one to conclude that the traffic is traversing ASz while the actual traversed AS is ASy. In addition, when Traceroute is used to infer the connectivity between ASes [11, 12, 13], one may wrongly conclude that there is a link between ASx and ASz, thus potentially causing a distortion of the inferred AS-level map. While several other causes may impact the accuracy of AS links derived from Traceroute– such as divergence between data and control paths, anonymous routers, unmapped hops, Internet exchange points (IXPs), multi-origin AS prefixes, and AS siblings – TP addresses (when shared between peering AS neighbors) were recently defined by Zhang et al. [62] as "the last and the most difficult cause to be inferred" and as "a huge obstruction towards the accuracy of Traceroute measurements". Fig. 2.7 also shows an additional problem caused by TP addresses. When all the solicited routers expose the incoming interface in the output of Traceroute, one may use this information to clearly identify also the traversed links (for instance, the interfaces a and b identify two actually traversed network links). With TP addresses this assumption is not valid any more: indeed, the address c, while correctly identifying a router traversed on the path towards the destination, does not identify the traversed link. The potential inability to identify the traversed links further complicates operations like network troubleshooting and may strongly weaken those network attacks designed to isolate entire portion of the network by concentrating a large amount of traffic on network links discovered thanks to

large-scale Traceroute-based campaign [38, 39].

Several works, by using heuristic methods, tried to mitigate the impact of TP addresses with different objectives: to explain the mismatches between BGP- and Traceroute-derived AS paths [62, 12], or to complement the AS-level topology inferred from BGP repositories [13, 11, 12]. However, to the best of our knowledge, only two works tried to isolate and study the phenomenon of TP addresses in order to quantify their impact, achieving totally different conclusions. By adopting a heuristic method based on IP-to-AS mapped Traceroute traces, Hyun *et al.* [63] concluded that TP addresses mostly appear at the border of multi-homed ASes and cannot be a significant source of AS map distortion. On the other hand, by using pre-computed AS-level graphs and pre-acquired knowledge about routers interfaces, Zhang *et al.* [62] concluded that TP addresses cause 60% of mismatches between BGP- and Traceroute-derived AS paths, where mismatches affect from 12% to 37% of the paths depending on the vantage point.

All the solutions proposed in literature are based on heuristics or on not always available pre-acquired information about the network topology. To the best of our knowledge, there are no techniques in literature able to classify as TP address or not the addresses reported by Traceroute. Accordingly, the actual magnitude of the phenomenon is essentially unknown and the conflicting conclusions documented in literature appear not helpful.

2.2.4 Misleading intermediate RTTs

Traceroute reports for each discovered hop statistics about the RTT computed as the length of time it takes to send the TTL-limited data packet and receive the corresponding ICMP Time Exceeded response: unfortunately, these values may represent a misleading information if interpreted as contributions to the overall RTT experienced by the traffic sent towards the destination. As clear example, it is not uncommon to observe intermediate RTT values higher than RTT of the targeted destination. For instance, Fig. 2.2 exposes a clear inconsistency: the 9-th hop expose an average RTT higher than RTT of the destination. Several explanations exist for this phenomenon: (i) first of all, each RTT value is computed by taking advantage of different probe packets that may experience totally different network conditions (e.g. congestion). In addition (ii) due to asymmetric routing, the hop discovered by Traceroute as part of the forward path to the destination may be not part of the reverse path from the destination: hence, the delay experienced in the reverse path of the intermediate hop is not a contribution of the delay experienced

along the reverse path from the destination. More in general, the reverse paths of two consecutive hops may differ causing the inferred link latency to be inaccurate. Finally, (iii) when estimating the intermediate and the overall RTT, two different network nodes are solicited: these nodes may potentially employ different amount of time to generate the response. Although hop-by-hop RTTs reported by Traceroute proved helpful, for instance, to predict the latency of end-to-end paths [28] or to geolocate network target [42] or aggregate routers and interfaces in PoP [15], all the factors described above generates a great uncertainty about the accuracy of the reported RTT values. Such an uncertainty strongly impacts the possibility of profitably exploiting the hop-by-hop delays to isolate the contributions of each portion of the network to the overall experienced RTT along the entire path: a valuable information for operations like network performance troubleshooting.

2.2.5 Load balancing

Load balancing may cause Traceroute to report incomplete or inaccurate information about the path [64].

Network operators make use of load balancing in order to improve the reliability of their network, the robustness of the communication and the utilization of the managed resources [65]. By properly configuring the intra-domain routing protocols and routers [66, 67], network operators can install in their network *equal cost multipath*: routers may not have any more a unique next hop for the traffic issued toward a given destination, but may be instructed to split the traffic across multiple equal cost paths.

The presence of load balancers is typically recognized when multiple addresses appear at the same hop-distance from the Traceroute originator. According to [65], there are three different load balancing schemes: per-packet, per-flow, per-destination load balancing. Routers performing per-packet load balancing split the traffic on a per-packet basis (e.g. round robin fashion). Routers adopting a per-flow load balancing scheme, instead, forward along the same path all the packets belonging to the same traffic flow. The traffic flow is determined by analysing particular fields of the packet. Finally, routers performing per-destination load balancing apply a rough version of the per-flow load balancing that is exclusively based on the destination field.[1]

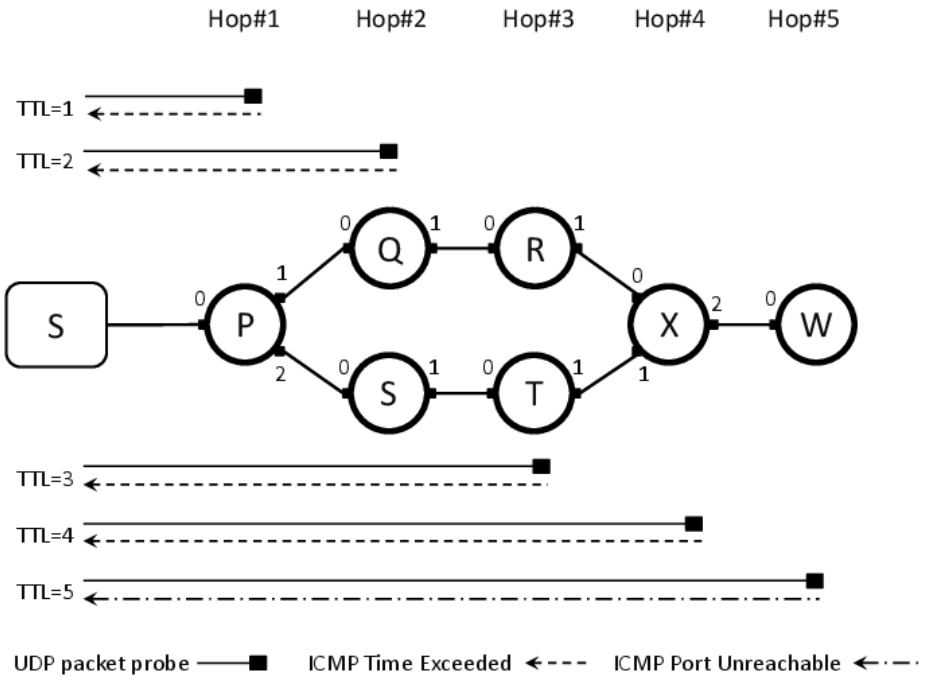
[1]Per-destination is similar but different from classic destination-based routing: classic routing forwards along the same path all packets issued towards the same network subnet, while per-destination load balancing split packets along different paths sent toward different addresses part of the same network subnet.

As largely discussed in literature [64, 65, 68], the presence of per-packet and per-flow load-balancing routers along the path may strongly impact the accuracy of the path inferred by using Traceroute. We report in Fig. 2.8 a scenario to highlight the phenomenon [65]. The figure shows the actual topology and a corresponding potential Traceroute outcome: essentially, the probe packets injected by Traceroute may follow completely different paths due to the load balancing: in this case, the traced paths appears incomplete (nodes and links are not traced) but also inaccurate (the inferred links do not actually exist). This may happen not only in case of per-packet load balancing, but also in case of per-flow load balancing. Indeed, in order to assign each newly discovered address to a given hop of the path, Traceroute needs to match each injected probe packet with its reply. For this operation, Traceroute assigns an identifier to each probe packet that is then recovered from the solicited ICMP Time Exceeded response. The classic Traceroute implementation encodes the identifier in header fields normally used by the routers to perform per-flow load balancing (e.g. the UDP destination port): by modifying these fields every time, the injected probe packets appear as part of different flows and are forwarded along different paths.

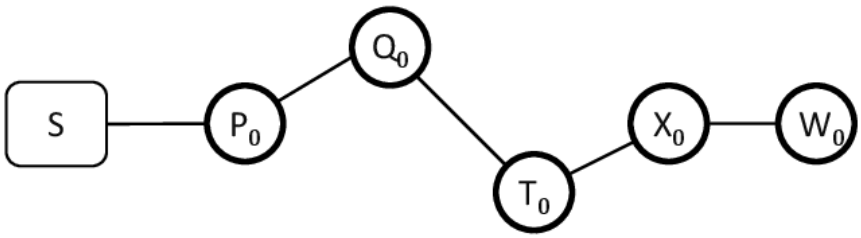
The application most affected by this inaccuracy is the inference of the network topology since Traceroute may expose non-existing links potentially causing a distortion in the reconstructed map of the topology [64, 69]. More in general, the inaccurate information provided by Traceroute make more complex to investigate the properties of Internet routes such as symmetry, stability, inflation, routing, etc. and also locate and resolve network failures and outages. For instance, the same IP address may appear multiple times along the path wrongly suggesting an abnormal routing behaviors (i.e. routing loop) when the traffic normally reaches its destination.

The impact of load balancing on the inferred path is one of the most investigated issues by the research community in the recent years.

Augustin *et al.* [64] categorized the artefacts in Traceroute measurements caused by load balancing and proposed a solution called Paris-Traceroute: this new variant of Traceroute injects probe packets as part of the same flow in order to avoid the artefacts caused by per-flow load balancing routers. This goal is reached by encoding the probe packet identifier in packet header fields different from those used by balancing routers to identify the traffic flow (e.g. UDP checksum).



(a) Actual scenario – Traceroute probe packets follow different network paths due to a load balancer.



(b) A possible outcome of Traceroute. Note the inferred non-existing link between the router Q and T.

Figure 2.8: Inaccuracy caused by load balancing – probe packets follow different paths causing Traceroute to report a false link.

By purposely injecting probe packets as part of different traffic flows, researchers demonstrated how to trace all the equal cost multipath toward a destination: authors in [70] proposed the Multipath Detection Algorithm (MDA), an enhancement of Paris-Traceroute designed to systematically find the entire set of load-balanced paths that probes can follow in the presence of per-flow load balancing.

Finally, authors in [68] quantified the magnitude of the phenomenon related to load balancing in the Internet: load balancers are involved in about 39% of the investigated Internet paths with a higher concentration in commercial networks compared to academic networks. Authors also observed that multipath routes mainly span over a limited number of hops typically concentrated inside the same AS.

Although not as common as per-flow load balancers [68], all the solutions proposed in literature to deal with load balancing still fail in case of per-packet load balancers: in this case, all the inconsistencies observed and investigated in [64] and successive works [70, 68] still represent a potential source of inaccuracy for Traceroute measurements. Furthermore, since a well-known problem in tracing Internet paths and accurately reconstructing the topology is the lack of ground truth, validating the findings documented in literature appears particularly hard.

2.2.6 Concurrent routing change

Routing in the network is reactive and may change any time, for example, because of connectivity disruption between ASes. Routing changes occurring while tracing Internet paths may cause Traceroute to report misleading results such as false links. When not properly recognized, routing changes may introduce a distortion in the inferred topology of the network. Tracing an Internet path with Traceroute can be significantly time-consuming since the technique incrementally learns the path length: routers are discovered one-by-one by increasing the TTL at each step. The process stops as soon as a response from the destination is collected. The time required might be significant (dozens of seconds) and larger is the amount of time required to trace the path, higher is the chance of a routing change in the network. Routing changes cause Traceroute to report the similar inconsistencies caused by load balancing: since probe packets injected before and after the routing change may follow completely different paths. While tracing again the path may potentially solve the observed inconsistencies, recognizing that a routing change occurred when it is not obvious is a non-trivial task. Some solutions documented in literature are

available to reduce the time required to trace the path [71, 72]. One possibility is to issue multiple probe packets with different TTL values at the same time: it is unclear however the exact number of probe packets to inject in order to trace the entire path. Authors in [71] proposed the adoption of scouting probe packets to solve this issue: a preliminary step is performed to estimate the number of hops contained in the path and thus the exact amount of probe packets to inject. These probe packets can be injected at the same time to strongly reduce the time required to trace the path. Authors proposed to estimate the length of the path with the the adoption of UDP probe packets: they compare the initial TTL value set in the injected packets with the one arrived at the destination and contained in the probe packet brought back in the payload of the solicited ICMP Port Unreachable reply. A similar approach has been also employed in [72]. However, when the destination is unresponsive, it is not possible to estimate the length of the path and thus the amount of probes to inject. In this case, the solutions proposed in literature to speed up the tracing process cannot be applied.

2.2.7 Probing overhead

Tracing large amount of Internet paths by taking advantage of multiple vantage points proved helpful to infer the topology of the Internet [2, 73]; to detect, monitor, locate and repair network failures or outages [25]; to assess changes in the global routing and so on. However, this type of experimental campaign imposes a potentially significant overhead on the network. Reducing this overhead is of the utmost importance since Internet measurements could be easily misinterpreted as large-scale Distributed Denial of Service (DDoS) attacks.

As a consequence of the routing in the network, all the paths originated by a given vantage point typically form a tree-like graph where the nodes located in the proximity of the vantage points are involved in the paths toward multiple destinations. When tracing these paths with Traceroute, these nodes are overloaded of Traceroute probe packets. When the goal is to infer the topology of the network, re-discovering every time these routers do not provide any additional topological information. Similarly, the paths originated by different vantage points targeting the same destination form a tree-like graph where the routers located close to the destination are overloaded by Traceroute probe packets: again, re-discovering those routers do not provide any additional information about the topology under investigation.

Few solutions have been proposed to reduce the Traceroute probing overhead on the nodes located in the proximity of vantage points and targeted destinations in large-scale measurement campaigns designed to discover the topology of the network [74, 72]. These solutions rely on (i) adapting the Traceroute exploring direction by increasing or decreasing the TTL values of the injected probe packets and (ii) stopping the tracing process as soon as no more topological information can be obtained compared to the previously collected paths.

For instance, in Doubletree [74], the tracing process start at the middle of the path and proceeds towards the destination (by increasing the TTL value) as long as an interface discovered by another vantage point towards the same destination is reached: the process is interrupted because the remaining path has been already discovered by the other vantage point. Similarly, from the middle of the path the exploring proceeds backward to the vantage point (by decreasing the TTL) and it is interrupted as soon as an interface contained in one of the paths previously traced by this vantage point toward any other destination: the missing portion of the path from the vantage point has been already traced in the past. One weak aspect of this solution is the need for communication and coordination between vantage points. Authors proposed the adoption of Bloom filters [75] to decrease the bandwidth consumption but false positives may cause the tracing process to be prematurely interrupted.

Other approaches tried to reduce the probing overhead when the goal is to investigate routing change by limiting the frequency of repeated measurements by taking into account the stability of the routes [19] but it is non trivial to deal with the trade-off between reducing the amount of repeated measurements while still recognizing all the ongoing routing changes in the network.

2.2.8 Unresponsive destinations and filtering policies

Filtering policies and unresponsive destinations cause the lack of responses determining an incompleteness and an impact similar to the one caused by anonymous routers.

Unresponsive destinations do not provide any response to the Traceroute originator. Accordingly, Traceroute marks the corresponding hop as anonymous and injects additional probe packets with increased TTL values. This operation, however, does not produce any result since the destination will discard all the successively injected packets: in this case, the traced path ends with multiple anonymous routers.

Exactly the same result happens when network operators apply filtering policies: by properly configuring their network (e.g. with also the deployment of Firewalls), network operators can easily disable Traceroute measurements in their infrastructure. A common approach is to configure the border routers of the network to drop any ICMP Time Exceeded messages sent toward addresses out of the managed IP space. When filtering policies are employed, Traceroute waits for the ICMP Time Exceeded responses until the expiration of the internal timeouts and causing the process to be extremely slow and the traced path to contain multiple hops marked as anonymous routers.

An approach proposed in literature to deal with unresponsive destinations is to inject probe packets carrying different transport protocols. Indeed, different transport protocols are intended to solicit different reaction from the targeted device thus potentially increasing the chance to trigger a response [72]. Furthermore, Luckie *et al.* [76] experimentally observed that the properties of the path inferred by relying on Traceroute depends on the adopted transport protocol: ICMP probe packets tend to successfully solicit replies from more destinations while UDP probe packets tend to reach less destinations but explore the highest number of links along the path.

Unfortunately, independently from which specific transport protocols is adopted, Traceroute is able to trace the path as long as ICMP Time Exceeded replies are collected from the network: filtering policies discarding these messages prevent Traceroute to trace the entire path. To the best of our knowledge, there are no solutions documented in literature able to deal with this problem.

2.2.9 Lack of visibility on the reverse path

An important limitation affecting Internet path tracing technique is related to the inability to trace the reverse path. By using Traceroute, one can trace the path from the Traceroute originator towards the destination, however, no information is available on the path taken by the traffic from destination back to the Traceroute originator. Several applications of path tracing are impacted or strongly limited by this issue [77]: for instance, network failures along the reverse path are much more complex to be located.

Katz-Bassett *et al.* [77] proposed *reverse traceroute*, a system to trace the reverse path from any destination back to any node of the network. To reach this goal, authors exploited different methods to incrementally discover reverse path hops and stitch them together in a unique path. More precisely, authors make use of (i) an atlas of Internet

paths traced with Traceroute, (ii) IP options such as the RR option and the Timestamp option, and (iii) *spoofed* probe packets, i.e. packets crafted with the source address of a different machine. The very basic mechanism used to trace the reverse path from a given destination back to a given source is first to identify a vantage point located no farther than 8 hops from the destination. This vantage point, then, launches an ICMP Echo Request packet equipped the RR option towards the destination spoofing as the source: in this way, the packet arrives at the destination and triggers an ICMP Echo Reply response sent towards the source. Since the destination typically replicates the incoming RR option in the response, the RR option starts registering addresses along the reverse path. These addresses are extracted from the source and used to incrementally build the reverse path. Authors demonstrated the great utility of tracing reverse paths by investigating AS-level connectivity, link latency and path inflations. However, though effective, the resulting system appears particularly complex compared to the simple stand-alone ready-to-use solution represented by the traditional Traceroute. In addition, since spoofing is often used to perform network attacks, this practice may easily trigger alarms and expose the injected traffic to filtering policies.

2.2.10 Other limitations

Other limitations affect Internet path tracing.

Layer-2 clouds. Path tracing techniques are not able to provide information about layer-2 devices such as switches, bridges, hubs, etc. Also MPLS tunnels may not be reported by path tracing techniques [45]. The most important impact of this limitation is related to the ability of entirely and accurately reconstructing the topology of the Internet. In this case, clouds of layer-2 devices cause the inferred topologies to be incomplete but also inaccurate with an impact very similar to hidden routers [9]. Also, network failures are harder to be detected and located when using path tracing techniques if they involve layer-2 devices.

Not routable addresses. Traceroute may also report non-routable addresses due to router and interface misconfigurations. Since these addresses can be used by different routers in different ASes, their presence in the trace causes uncertainty especially when investigating the global properties of the network such as the topology: in this case, researchers are forced to treat these addresses as anonymous routers and face them with the same methodologies and techniques proposed for that limitation.

2.3 Open challenges

Network operators and researchers often rely on path tracing techniques to explore and monitor Internet paths with the goal of collecting useful information about the particular phenomenon under investigation (e.g. the topology [8, 9], the current routing [22, 23], etc.). This methodological approach proved to be very useful and effective as demonstrated by the applications described in Section 1.2. However, by analysing the literature we observed how network operators and researchers must often deal with two open challenges when adopting this approach.

2.3.1 The need for vantage points

Investigating aspects or phenomena ongoing in the Internet typically requires the ability to explore particular Internet routes. One way to address this issue is to exploit a single vantage point and trace Internet paths toward multiple network destinations. Unfortunately, a unique vantage point is very limited in terms of Internet routes that can be explored: for instance, tracing the path from the vantage point A towards the network destination B does not provide any useful information about the path followed by the traffic sent from the node B back to the node A and tracing the paths toward any destinations in the Internet from the vantage point A will not provide more insight about this specific Internet route. This happens since asymmetric routing is predominant in the Internet [21], i.e. one cannot simply assume that the path from the node A to the node B is equal to the path from B to A. As a consequence, there is the possibility that the Internet routes of interest cannot be measured by using the adopted vantage point.

Using multiple vantage points is definitively helpful. In order to investigate large-scale phenomena in the Internet from multiple observation points, the research community has built over time several experimental testbeds providing vantage points located at universities [78, 79, 80], homes [81, 82] and, recently, also at mobile devices [83, 84, 78]. From these privileged observation points, researchers and network operators are potentially able to explore a large number of Internet routes, thus potentially collecting the information required to deeply understand the particular phenomenon under investigation. Unfortunately, these vantage points are made available through a set of highly heterogeneous interfaces: each testbed typically requires the user to (i) join a community (ii) obtain credentials (iii) access the vantage points through the specific mechanisms provided by the

testbed management platform. In addition, each testbed may also impose particular constraints on the managed vantage points: for instance, RIPE ATLAS [6] provides a set of well distributed vantage points capable of performing basic measurements such as Traceroute, Ping and DNS queries. However, each measurement has a cost in terms of credits and users gains credits over time according to how many active vantage points they host.

In this scenario, network operators and researchers are not encouraged to use the vantage points made available by different testbeds because this operation would require a deep knowledge of the policies, constraints and interfaces of each specific testbed. As a consequence, we observed in literature that most researchers exploited always the relatively small amount of vantage points of the same testbed, thus obtaining visibility on a limited amount of Internet routes providing potentially only partial information about the phenomenon under investigation.

2.3.2 Tracing paths with a priori known characteristics

Another open challenge network operators and researchers need to deal with is the ability of current systems to satisfy only one simple query: *given the possibility of issuing Traceroute measurements from the vantage point A, what is the path taken by the traffic from the vantage point A to the network destination B?*

While tracing Internet paths is helpful to investigate several aspects of the Internet as largely demonstrated by the scientific literature, researchers and operators need to address much more complex queries. These queries typically imply the knowledge of at least some properties for the subset of paths of interest: for example, researchers are interested in Internet paths crossing a given AS to investigate its topology [85, 86, 73], intra-domain routing [44] or performance [87] and more in general to infer the properties of the routes crossing this specific network. Researchers are interested in the paths traversing two given consecutive ASes to demonstrate the existence of traffic exchanged between them and thus potentially infer an additional AS-level link not already considered in the partial AS-level topology one can derive from inter-domain routing [13, 11, 12]. More in general, requesting Internet paths traversing consecutive and non-consecutive ASes is potentially useful to investigate circuitous routes and path inflations [22]. Finally, traversing a given AS on the path to reach a destination located in another AS is helpful since it may provide relevant information about the the reverse path from any destination in the Internet back to any vantage point [77]: we already acknowledged how the lack of visibility of Traceroute

on the reverse path is one of the major limitations of path tracing. The above-mentioned categories of advanced queries is by no means complete, but illustrates that (1.) advanced queries requiring tracing Internet paths are very common in literature and (2.) often users are willing to trace Internet paths compliant with specific a priori-known requirements (like the ASes to traverse) while they do not care about which specific vantage point and destination need to be considered to reach this goal.

Unfortunately, since current systems are able to simply trace the path from a given source toward a given destination, researchers are forced to identify the vantage points to use and the destinations to target in order to trace the paths of interest. To deal with this non-trivial issue, researchers and network operators usually (1) make use of a given testbed; select some vantage points and destinations according to their expertise, experience and the specific phenomenon they want to investigate; trace the selected paths and verify if the measurements satisfy their needs, otherwise, new vantage points or destinations are selected. This solution is time-consuming and provides just a suboptimal access to the routing information. Users might not easily identify all the possible routes of interest, thus collecting only a partial information required for a deep understanding of the phenomenon under investigation. Other solutions are (2) to adopt a bruteforce approach performing large-scale measurement campaigns using as many vantage points as possible to issue Traceroute measurements toward as many network destinations as possible or (3) to exploit the information made publicly available by large research projects adopting bruteforce approach by monitoring millions of Internet paths every day [2, 15]. These solutions are not only network resource-consuming but do not necessarily provide all the desired information: although a large amount of paths is traced, this set of paths may not necessarily contain the routes of interest. Furthermore, since tracing millions of paths is a time-consuming process, the collected information might be also potentially stale due to routing changes.

2.4 Final remarks

Techniques designed for Internet path tracing inject into the network purposely crafted probe packets to reconstruct the Internet path essentially in terms of as a sequence of IP addresses, i.e. they report an address associated to each traversed router on the path towards the targeted destination. According to a recent survey among network opera-

tors [50], Traceroute is the most widely adopted approach to trace Internet paths particularly helpful for managing and troubleshooting the network. At the same time, tracing Internet paths proved to be extremely useful also for scientific purposes as demonstrated by the numerous applications described in the previous chapter.

Unfortunately, despite the numerous applications, path tracing techniques are affected by several important limitations causing inaccuracy (i.e. the output of path tracing does not correspond to the actual traversed paths or the provided information is potentially misleading) and incompleteness (i.e. the path tracing technique misses traversed nodes or links) with an impact on the applications relying on path tracing.

In this chapter, we described the available path tracing techniques and provided an overview of the limitations and the partial solutions discussed in literature. Some limitations attracted great interest from the research community (e.g. load balancers), others have been largely ignored (e.g. hidden routers) motivating part of the research activities at the base of the contributions described in Chapter 3. In Chapter 4, we will also document and investigate two additional experimentally observed limitations affecting Internet path tracing.

Finally, we noticed that researchers, network operators and systems using path tracing techniques are often interested in exploring specific Internet paths. However, the ability of exploring the Internet paths of interest strongly depends on (i) the available vantage points and (ii) the selection of the vantage point to use and the network destinations to target. These two factors currently represent open challenges: even if several experimental testbeds providing vantage points exist, the great heterogeneity of the interfaces to access these vantage points induced researchers to use always the same testbed and its relatively limited amount of vantage points. In addition, there are no methodologies to identify the vantage point and the destination to select in order to explore a particular Internet path with a priori known characteristics (e.g. the ASes to traverse). These open challenges may potentially cause a suboptimal access to the paths of interest. In Chapter 5, we will describe a general architecture and an its implementation designed to address both the open challenges described in this chapter.

Chapter 3

Augmenting Internet path tracing

Several applications rely on Internet path tracing to investigate particular phenomena or aspects of the global network. Unfortunately, the techniques widely adopted to this end suffer from severe limitations causing the collected information about the paths under investigation to be potentially inaccurate or incomplete. Despite the efforts of the research community, several limitations (e.g. hidden routers, third-party addresses, etc.) appear not definitively solved or exhaustively investigated.

In this chapter, we describe innovative methodologies and techniques developed for augmenting Internet path tracing. Our goal is to detect, quantify and possibly resolve or mitigate unresolved limitations affecting the path tracing techniques.

At the basis of most of the contributions described in this chapter, there is the adoption of probe packets equipped with the IP optional headers (the IP options): as we demonstrate in the following sections, this innovative type of measurement traffic has the great potential to collect additional information about the traversed path. This additional information is extremely useful to address the limitations affecting path tracing techniques. We first briefly describe the IP options adopted for augmenting Internet path tracing and how routers proved to manage this particular type of traffic. Then, we detail and experimentally evaluate the methodologies and techniques designed for augmenting Internet path tracing: more precisely, we designed and developed methodologies and techniques for detecting hidden routers and third-party addresses, and for accurately dissecting the RTT in the traced paths. Finally, we explore a totally alternative path tracing approach exposing additional information about the traversed paths when compared to the classic Traceroute-based approach.

3.1 Measurement traffic enhanced by IP options

In this section, we describe the particular measurement traffic we adopted for augmenting Internet path tracing. This traffic consists of probe packets equipped with IP options with the potential to collect additional information about the traversed path: later in this chapter, we demonstrate how using this particular traffic one can detect and possibly mitigate or resolve important limitations affecting Internet path tracing techniques.

3.1.1 Classic probe packets in active probing

Measurement techniques based on active probing injects into the network purposely crafted probe packets: by observing how the network treats the injected traffic, researchers can infer properties of the network such as the topology, the available bandwidth, the link capacity, and so on. In this operation, researchers may have or not the control on both the end points of the path. Tracing Internet paths, however, typically implies the lack of control on the targeted destinations since any path on the Internet might be the one of interest. Accordingly, active probing performed for Internet path tracing typically exploits a set of probe packets crafted in order to solicit replies from devices not under the control of the user.

The traditional probe packets used in path tracing are briefly reported in the following:

- UDP [88] probe packets are crafted with a high and presumably unused destination port in order to trigger an ICMP Port Unreachable message from the targeted destination. By default, this is the transport protocol used by Traceroute.
- ICMP [89] refers to ICMP Echo Request packets created to trigger ICMP Echo Reply messages from the destination.
- TCP [90] probe packets refer to packets carrying the TCP protocol with the SYN flag set crafted with a high and presumably unused destination port in order to trigger TCP Reset responses.

Note how these probe packets are intended to impose a minimum load on the targeted destination since the goal is to investigate the properties of the path and the process ends as soon as a message from the destination is collected: for instance, TCP probe packets with the SYN flag set are the very first packet of the multistep TCP handshake [90].

Since during this process resources are allocated by the destination for the ongoing communication, it is important to avoid that the targeted device allocates resources for a communication that is not going to happen. This is why the destination port is set to a presumably unused port.

Limiting the TTL of these probe packets as done in the most adopted path tracing techniques allows to elicit ICMP Time Exceeded messages from the routers located along the path, thus providing information about the traversed path.

3.1.2 Probe packets and IP options

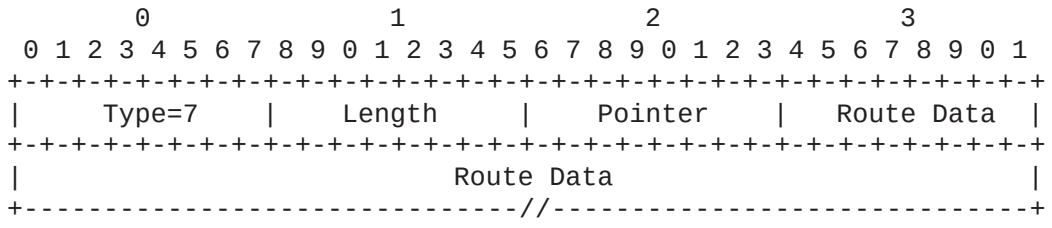
The classic probe packets reported above have been used for decades to trace Internet paths with Traceroute [76]. We demonstrate in this chapter that it is possible to strongly improve their ability to collect information about the paths by also using IP options.

The IP options of the protocol IP version 4 are a generic and simple way of transmitting per packet information related to network layer components like routers and hosts [47]. The options may appear or not in a packet and according to the standard they must be implemented by all the IP modules in the Internet. Since the IP protocol is in charge of delivering packets and involves all the network devices located along the path from the source towards the destination, when present, the option is inspected by all the network devices encountered by the probe packet in its travel. A single packet may carry multiple IP options and each IP option may have a variable length. There are two cases for the format of an option: (i) a single byte specifying the type of option and (ii) multiple bytes where the first specifies the option-type, the second the length of the option, and remaining bytes represent area allocated for the option-related data.

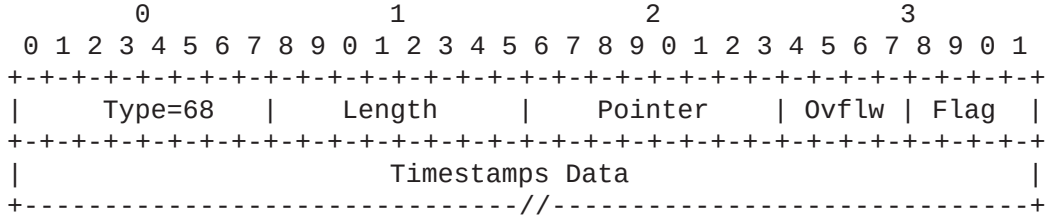
In this thesis, we exploit two specific IP options: the IP Timestamp option and the IP Record Route option. Later in this chapter, we demonstrate how probe packets equipped with these options provide much more information about the traversed path, thus augmenting Internet path tracing.

IP Record Route option - RFC791 [47]

As already discuss in Chapter 2, the IP Record Route (RR) option (type 7), provides a way to record the route traversed by a probe packet toward its destination and represents the first (and unique) path tracing approach included in the Internet standards. The format is reported in Fig. 3.1a. The option header consists of 3 bytes: besides the type



(a) IP Record Route option.



(b) IP Timestamp option.

Figure 3.1: Format of IP optional headers used in this thesis.

field, the *length* field counts the option size in terms of octets, while the *pointer* field indicates the first byte of the slot reserved for the next address to register. The minimum value of the pointer is 4. The *route data* area is initialized to zero and serves as a container for IPs registered along the path. When receiving a packet equipped with this option, a network device checks if the pointer does not exceed the option length (i.e. the option is not full), inserts an owned IP address, and increments the pointer value accordingly. If the option data is full, the packet is normally forwarded without inserting any address. Considering the maximum size of the IP header, the RR option cannot contain more than 9 address slots. For this reason, the RR option represents a valuable but also limited tool for tracing IP paths as already discussed in Chapter 2.

According to [56], the registered address is usually the one associated to the interface selected by the router to forward the traffic to the next hop along the path. However, authors also observed a limited but significant percentage of routers registering in the option data the address of the interface on which the packet arrived.

IP Timestamp option - RFC791 [47]

The IP Timestamp option (type 68) or simply TS option, has the format reported in Fig. 3.1b. Compared to the RR option, the TS option header is one byte larger and contains additional fields. As usual, the length field indicates the size of the option in bytes, while the pointer field is used by each traversed router to identify the first byte of

the slot to analyze in the option data. The minimum value of the pointer is 5 and it is possibly incremented by the router encountered along the path. When the pointer value is higher than the option length, the option is considered full.

The *flag* field defines the variant of the TS option and may assume three different values.

- **Flag 0:** each traversed router on the path is requested to insert inside the slot indicated by the pointer a 32-bit timestamp. After this operation, the router increments the pointer by 4 units and forwards the packet to the next router along the path.
- **Flag 1:** each router is requested to insert into the slot indicated by the pointer a 64-bit *(IP address, timestamp)* record, where *IP address* is an address owned by the router. In this case, the router increments the pointer value by 8 units before forwarding the packet.
- **Flag 3:** the sender can prespecify up to four addresses (and thus the devices) from which a timestamp is requested. In this case, the option data is initialized with a set of $(IP, 0)$ records. When an incoming packet is equipped with this variant of TS option, the router checks if the address contained into the slot currently indicated by the pointer is an owned address and only in this case a timestamp is inserted in the corresponding area. Otherwise the option is ignored and the packet is normally forwarded along the path. Whenever a timestamp is inserted, the pointer is incremented by 8 units. This TS option variant is also commonly referred in literature as *prespecified* TS option. Note that according to the described mechanism, the timestamp are requested in a specific order. Indeed, since each traversed router inspects the slot currently pointed by the pointer field, the second prespecified address cannot insert its own timestamp before the first prespecified address, and third address before the first two prespecified addresses and so on. Many of the contributions described in this chapter rely on the order of the prespecified addresses.

Independently from the adopted variant, every time a router cannot insert a timestamp because the option is full (i.e. the pointer exceeds the option length), the router is requested to increment by one the *overflow* field. Accordingly, this the 4-bit field counts the number of routers encountered along the path that could not insert a timestamp due to lack of space. According to the RFC, whenever possible, the timestamp value should

be inserted in a standard format, which represents the elapsed time in milliseconds since midnight UT (universal time). When such format cannot be respected, the highest order bit of the provided timestamps is set to one, indicating the use of a non-standard value.

Since the maximum size of an IP option is 40 bytes, the TS option can potentially contain no more than 9 timestamps (flag 0) or 4 (IP, TS) records.

3.1.3 A new research trend

According to the standard, the IP options must be implemented in any network device of the Internet (routers and hosts). Hence, they can be exploited by the users to request additional information about the routers encountered along the path. For instance, one may ask routers to identify them-self with the RR option or to provide a temporal information with the TS option. Although IP options are not universally supported and may potentially expose the measurement traffic to higher delay, jitter and packet loss [91, 92], a growing research trend demonstrated the great benefit of using IP options-equipped probe packets for Internet measurements.

In this section, we recall how IP options have been recently used to investigate different aspects of the Internet.

- *Advancing path tracing:* As already described in Chapter 2, one of the pioneer works demonstrating the utility of IP options in Internet measurements is the work proposed by Sherwood *et al.* [55]: by using Traceroute and the RR option, authors investigated the possibility to gather additional information about the traversed paths by potentially identifying routers performing load balancing, anonymous routers, multiple interfaces of the traversed routers, etc. Authors tried to face the non-trivial task of aligning the RR and Traceroute traces by also adopting disjunctive logic programming [56]. Although the proposed approach is computationally complex and hard to replicate, these two works attracted large interest from the community and clearly demonstrated the great potential benefit of using IP options in Internet measurements.

- *Reverse path tracing:* Recently, the RR and TS options as well as the adoption of multiple vantage points and spoofed probe packets proved helpful to trace the reverse path from any network destination back to a given source [77]. This approach tried to address one of the most severe limitations of current path tracing techniques and

proved a great potential when troubleshooting the network (e.g. path inflations), infer the AS-level connectivity and measuring the properties of the network links (e.g. one way link latency). The proposed approach received important awards in the most important conferences in the networking field.

- *Alias resolution:* The TS option proved to be helpful also when facing the alias resolution problem, i.e. the problem of gathering under a unique identifier those addresses part of the same network device. In particular, Sherry *et al.* [93] proposed a promising technique based on the prespecified variant of the TS option to identify the IPs belonging to the same network device. The proposed approach proved to identify a significant amount of addresses in alias not recognized by other techniques part of the state of the art.

- *Violations to the destination-based forwarding scheme:* RR option has been also recently used to assess violations to the destination-based forwarding scheme. Each router is supposed to select the next hop on the path towards the destination exclusively based on the destination of the packet. However, increasingly common mechanisms such as load balancing, MPLS and default routing represent a deviation from this paradigm. Flach *et al.* [43] exploited the great potential of the RR option to quantify such deviation discovering that surprisingly about 29% of observed routers violate the destination-based forwarding scheme.

- *Inference of router statistics:* Finally, researchers in [94] demonstrated how the IP TS option allows one to remotely infer router statistics. Authors use the prespecified TS option to bound the rate of UDP traffic carried by CISCO 3600-series routers and the start and finish of multicast traffic carried by 6500-series Catalysts by no requiring any control on the tested devices. This works demonstrate how using IP options may provide additional information on the current status of the router uncovering CPU-intensive operations like forwarding multicast traffic.

The works reported above proposed innovative promising measurement techniques relying on the adoption of probe packets equipped with IP options: this approach allowed to investigate, mitigate or resolve long-lasting unresolved issues like tracing the reverse path, resolving the alias resolution, quantifying the deviation from destination-based forwarding scheme and remotely inferring the current status of network devices. All these

achievements were made possible thanks to the adoption of IP options.

Inspired by these works, we designed, implemented and evaluated innovative methodologies and techniques enabled by IP options with the goal of investigating, characterizing and possibly mitigating or resolving largely ignored limitations in Internet path tracing. We provide more details about these methodologies and techniques in the rest of the chapter.

3.1.4 On the IP Prespecified Timestamp option

Part of the contributions highlighted in this chapter are based on the prespecified TS option. This variant of the TS option has been already used by several works in literature to investigate relevant aspects of the Internet [77, 93, 94]. Thanks to a large scale measurement campaign targeting 1.7M addresses [95], we tried to infer how routers manage probe packets equipped with this particular variant of TS option. Hereafter, we adopt the following notation: PROBE A|BCDE refers to a probe packet type PROBE sent towards the destination A equipped with a prespecified TS option where the ordered sequence of addresses BCDE is prespecified. In our analysis, we targeted each destination A with classic probe packets (ICMP, UDP, TCP) equipped with A|AAAA, i.e. the destination is requested to insert four timestamps in the option data. The destinations probed with ICMP and TCP probe packets typically replicate the received option inside the IP layer of the reply packet. When probed with UDP, instead, the destinations return an ICMP Port Unreachable error message containing the original probe packet including the TS option. Accordingly, when using ICMP and TCP, the returned option (if present) is extracted from the IP layer of the reply packet, while, for UDP, the option is extracted from the original probe packet carried back as payload of the ICMP Port Unreachable error packet.

We report in the following the main findings of this analysis.

Probe packet responsiveness. In line with similar studies [91, 92], we observed that IP options-equipped probe packets can be profitably exploited to explore the core of the network while filtering may occur at the edge of the network probably due to Firewalls and Intrusion Detection Systems. The most responsive type of probe equipped with the prespecified TS option is ICMP, followed by UDP and TCP probe packets. In particular, TCP probes solicit very limited amount of responses. ICMP (UDP) probes

solicited replies from about 50% (21%) of the targeted routers in the core of the network – as a reference, without the option, ICMP (UDP) probes solicited replies from about 79% (50%) of the targeted devices. We also observed that destinations may reply to one type of probe but not to another: this encourages the adoption of different types of probes to successfully solicit replies from the network.

Router behaviors. We analyzed the number of collected timestamps for each type of probes and isolated the following two router behaviors (these behaviors have been also recently documented in an on-line CAIDA report [96] describing the results of an independent study):

- *Per-interface stamping routers.* When processing the prespecified TS option, these routers insert one timestamp only when the probe packet passes through the interface associated to the prespecified address (see Fig. 3.2a and 3.3a). A destination A exposing this behavior provides between 0 and two timestamps when probed with ICMP A|AAAA and zero or one timestamp when probed with UDP A|AAAA. Indeed, when a per-interface stamping router owning the address A is targeted with ICMP A|AAAA, it provides (i) no timestamp, if the option (i.e. the ICMP Echo Request or the ICMP Echo Reply carrying the option) does not traverse the interface associated to the prespecified address A; (ii) one timestamp, if the option traverses the prespecified interface when entering or leaving the node; and (iii) two timestamps when the option enters and leaves the node through the prespecified interface. On the other hand, when using UDP probe packets, the option is brought back inside the payload of the ICMP Port Unreachable message: the solicited router inspects the option only when the probe packet enters the node. For this reason, UDP probe packets collect no more than one timestamp in case of per-interface stamping routers. According to [96], CISCO 6000 series routers show this particular behavior.

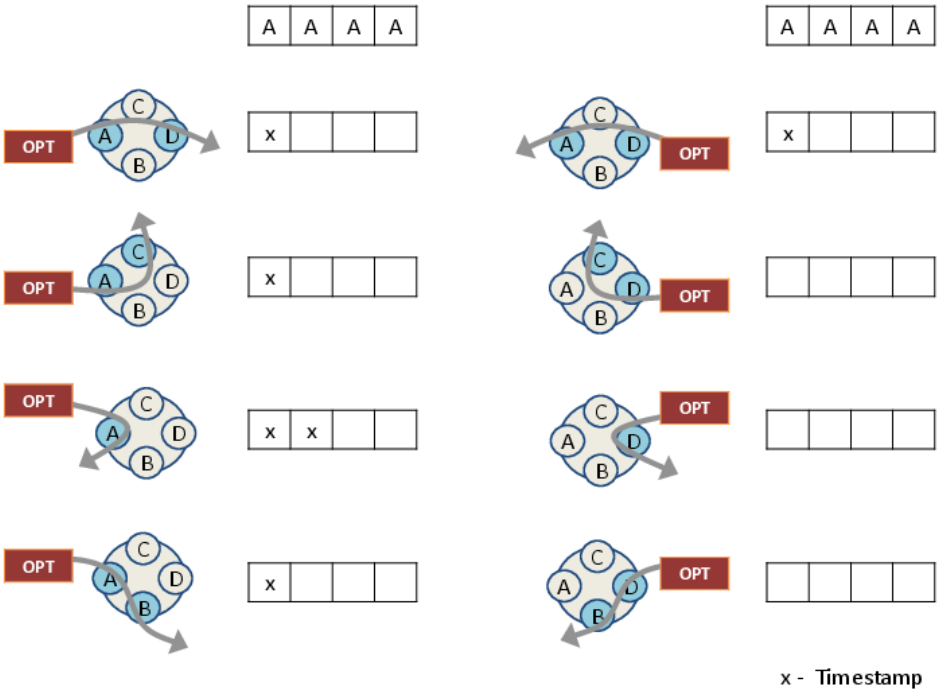
- *Any-interface stamping routers.* These routers insert all the requested timestamps when the prespecified address is associated to *any* owned interface (see Fig. 3.2b and 3.3b). For about 10% of the targeted IPs, we observed destinations providing all the four requested timestamps both for ICMP and UDP probe packets. We also employed IGMP probing [85] to fingerprint these devices: we discovered that

many Juniper routers show this particular behaviors. This finding is also confirmed by [96].

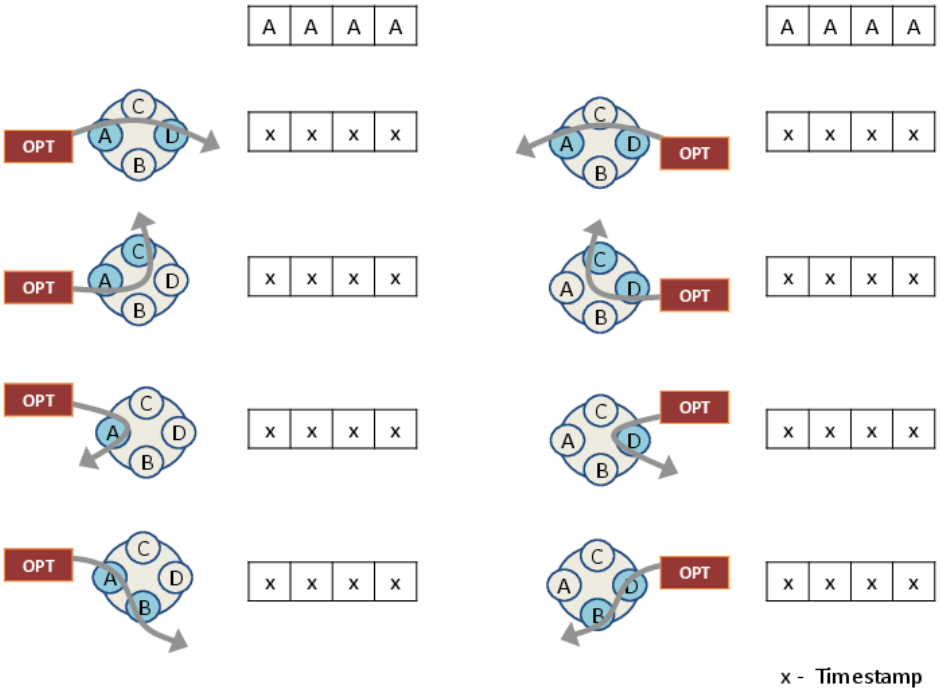
Since IP options are under specialized, there might be other router behaviors in the Internet. For instance, we also observed non RFC-compliant behaviors for about 2.25% of the probed destinations. We isolated the following abnormal behaviors: (i) some prespecified addresses may be overwritten; (ii) the destination may stamp the option by skipping one or more records (e.g. the second IP is stamped, but not the first one); (iii) the pointer field is inconsistent with the respect to the number of timestamps (e.g. it does not indicate the beginning of a valid record); (iv) the option returned in the payload of the ICMP error message is truncated; (v) the overflow field counts several *extra−stamps*, even if the option is not full (e.g. the number of inserted timestamps is less than four); (vi) the option data is entirely overwritten with part of the original packet header; finally (vii) some destinations provide timestamps even when it is not requested (extra-stampers) – this behavior has been also recognized in [93]. Although limited in number, these behaviors must be taken into account when experimenting with the prespecified TS option. We point the reader to [95] for a detailed analysis of these cases.

Timestamp values. According to the RFC791 [47], routers may provide both standard or non-standard timestamp values: a standard value represents the elapsed time in milliseconds since midnight UT and should always be lower than $86.4 * 10^5$ $(24h * 3600s * 1000)$, while a non−standard values should belong to the range $[2^{31}, 2^{32}]$ since in this case the most significant bit is set to one. Accordingly, the range $]86.4 * 10^5, 2^{31}[$ consists of non RFC−compliant values. Among the $660k$ destinations stamping at least once, we found timestamp values according to the following distribution: 87.6% standard, 11.3% non−standard, 1.15% non RFC−compliant. Surprisingly, we also found few destinations providing both standard and non-standard timestamps. Non-standard values are hard to interpret since they may indicate any time. Luckily, destinations providing standard values represent the vast majority and can be profitably exploited for measurement purposes.

The experience and knowledge acquired with the experimental campaign described above are the basis of many of the contributions described in the rest of the chapter.

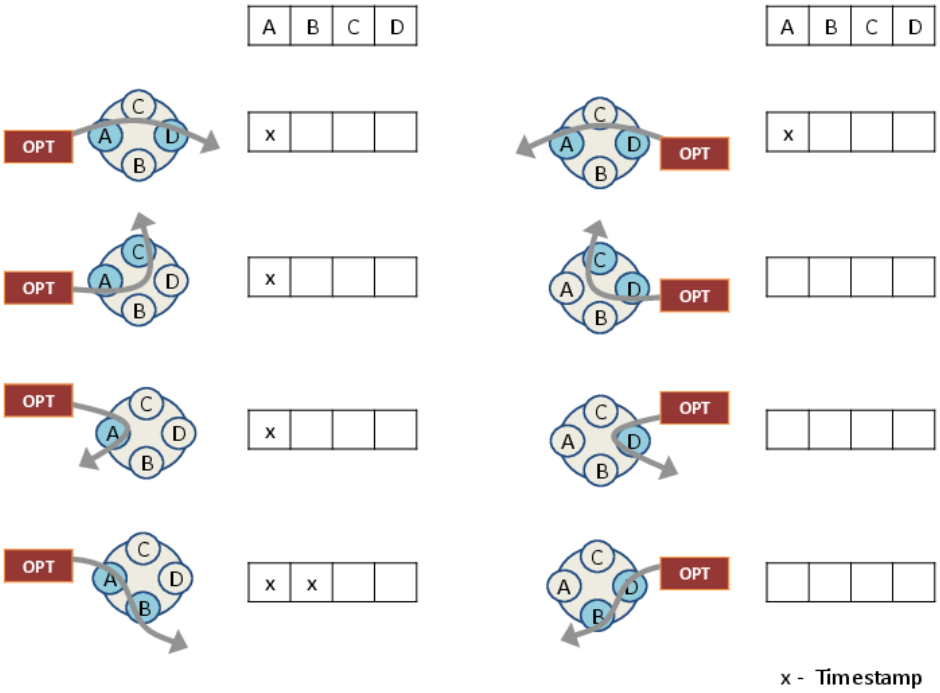


(a) Per-interface stamping router – a timestamp is provided only when the option traverses the prespecified interface.

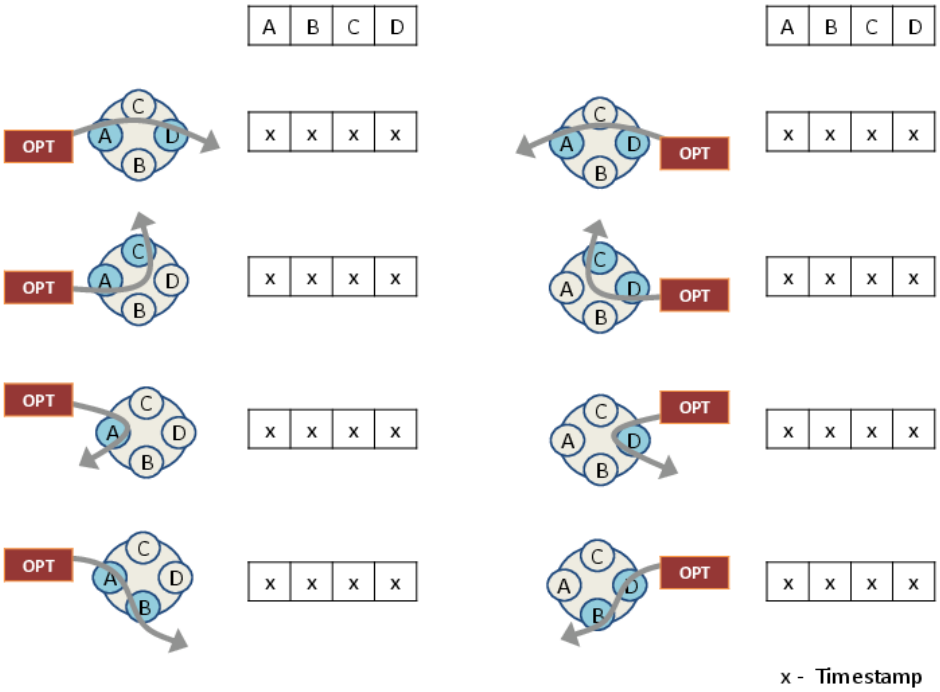


(b) Any-interface stamping router – all the requested timestamps for the owned inferfaces are provided independently of the traversed interface.

Figure 3.2: Inferred behaviors of routers managing the prespecified TS option when the same address is prespecified multiple times.



(a) Per-interface stamping router – no timestamps are collected when the option does not traverse the first prespecified interface due to the order between the prespecified addresses.



(b) Any-interface stamping router – all the requested timestamps for the owned interface are provided independently of the traversed interface.

Figure 3.3: Inferred behaviors of routers managing the prespecified TS option when different addresses are prespecified.

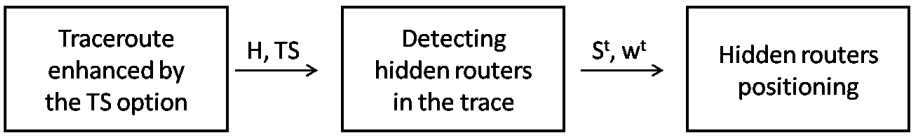


Figure 3.4: The three steps performed by DRAGO.

3.2 Detecting and locating hidden routers

In this section, we use IP options-equipped probe packets to detect and locate hidden routers when tracing Internet paths.

3.2.1 Motivation

As described in Chapter 2, hidden routers are network devices configured to not decrement the TTL value when forwarding the packets. As a consequence, these devices are totally invisible when path tracing techniques is performed with Traceroute. Hidden routers represent a severe source of inaccuracy and incompleteness especially when researchers use Traceroute to infer the topology of the network. Indeed, the presence of hidden routers causes the inferred topology to be inaccurate (e.g. it contains non-existing links) and incomplete (e.g. it misses network nodes and links). Due to the lack of measurement methodologies and techniques, the actual magnitude of the phenomenon is practically unknown. This in turn raises doubts about the overall accuracy of the topology commonly inferred by relying on Traceroute. This scenario motivated the research activities at the basis of the contribution detailed in this section.

3.2.2 The proposed solution

In this section, we describe DRAGO, our multi-step technique designed to detect and locate hidden routers along the traced paths.

Basic idea

The key mechanism used to detect hidden routers is based on the comparison between the number of routers managing the TS option and those decrementing the TTL: there is an evidence of hidden routers on the path every time the number of routers managing the option is higher than the ones decrementing the TTL. This simple approach is applied to any portion of the traced path to detect and locate hidden routers [97].

The algorithm

As depicted in Fig. 3.4, DRAGO performs the following three main steps: (*1.*) a novel Traceroute enhanced by the TS option is launched towards the destination; (*2.*) a procedure to detect and approximately locate hidden routers is applied to the Traceroute trace; (*3.*) a last procedure is applied to reduce, as much as possible, the uncertainty on the position of the detected hidden routers starting from the output of the previous step. To highlight each step, we also refer to the sample scenario reported in Fig. 3.5.

Step 1: Traceroute enhanced by the TS option. This step aims at counting at the same time the number of devices managing the TS option and those decrementing the TTL along the path toward a destination. To reach this goal a novel Traceroute is used: the injected TTL limited Traceroute probes are also equipped with the TS option (flag 0). In this way, all the routers along the path are requested to insert a timestamp in the option data or to increment the overflow field. This enhanced Traceroute normally collects ICMP Time Exceeded messages from the routers along the path. From each collected ICMP Time Exceeded reply, our Traceroute extracts the source address as usual but also the number of devices managing the TS option up to the replying router. The latter information is computed by inspecting the TS option brought back in the payload of the ICMP Time Exceeded error message. Indeed, the original probe packet (TS option included) triggering the ICMP error is inserted by the solicited router inside the payload of the ICMP error message. The number of routers managing the TS option is computed as the number of timestamps inserted in the option data plus the overall number of overflow

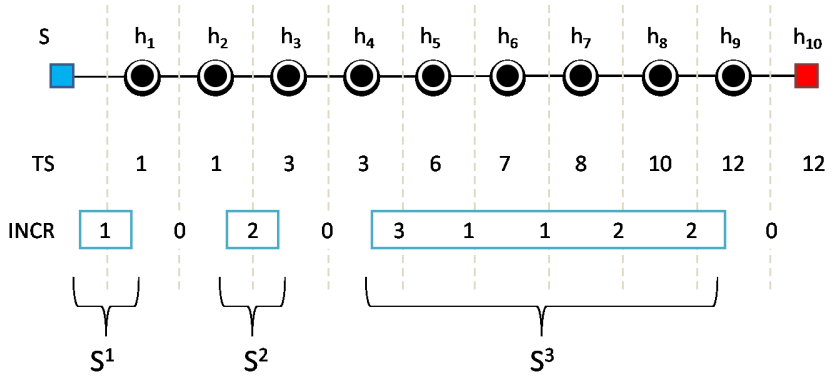


Figure 3.5: H, INCR and S^t in a sample scenario.

increments: our probe packets can count up to 24 routers managing the TS option (9 timestamps plus 15 overflow increments).

Hereafter, we adopt the following notation (see Tab. 3.1 as a reference for the notation introduced across the section): let h_i be the i-th router discovered by our enhanced Traceroute along the path towards the destination; H is a generic Traceroute trace made by the n routers $h_1h_2...h_n$; let TS_i be the number of timestamps plus the overflow increments contained in the TS option brought back in the payload of the ICMP Time Exceeded reply provided by h_i. Basically, TS_i represents the number of routers which actively managed the TS option up to the TTL-decrement performed in h_i. Accordingly, TS_n represents the overall number of routers managing the TS option registered in the path. Since Traceroute uncovered $n+1$ devices in the path (considering also the source machine), there is an evidence of hidden routers in the path every time $TS_n > n+1$: in this case, the path contains more devices managing the option than those decrementing the TTL. TS is obviously monotonically non-decreasing with i.

In the sample scenario reported in Fig. 3.5, our enhanced Traceroute discovered 10 routers towards the destination. We would like to understand if there are other devices than the ones reported by Traceroute in this path. By inspecting the TS option brought back in the payload of the collected ICMP Time Exceeded messages, we stored in TS the number of routers managing the option. In our sample scenario, TS=[1 1 3 3 6 7 8 10 12 12]: for instance, the ICMP Time Exceeded provided by the 7-th hop contained 8 timestamps, i.e. between the source and the 7-th router decrementing the TTL we observed exactly 8 routers managing the option. Since TS_n is 12 and the path contains 11 devices (including the source machine), we can already argue that the path contains more devices than those exposed by Traceroute. Note that $TS_n > n+1$ is not a necessary condition: the path may still contain hidden routers also when $TS_n \leq n+1$.

The trace H and the associated TS represent the input of the second step.

Step 2: Detecting and quantifying hidden routers This step aims at detecting the presence of hidden routers in the trace H pointing out also the portion of the trace in which those devices lie. To reach this goal, DRAGO applies the condition $TS_n > n+1$ to portions of the traced path.

To explain how the step works, we introduce a new variable called *INCR*:

$$INCR_i = \begin{cases} TS_1 & \text{if i = 1} \\ TS_i - TS_{i-1} & \text{otherwise} \end{cases} \tag{3.1}$$

Each element in *INCR* reports the number of routers managing the TS option between two consecutive routers decrementing the TTL: $INCR_i = z$ indicates that there are exactly z routers managing the option between the two routers decrementing the TTL h_{i-1} and h_i (i.e. in the transition h_{i-1} h_i). Note that $INCR_1$ refers to the routers managing the option between the adopted vantage point and the first router decrementing the TTL on the path.

Once *INCR* is computed, the p longest subsequences $S^1, ...S^p$ of consecutive non-zero elements in *INCR* are extracted. The t-th subsequence S^t contains exactly s^t elements and it is related to a specific portion of the trace made by $s^t + 1$ routers decrementing the TTL. The subsequence S^t contains hidden routers every time the following condition is verified:

$$\sum_{i=1}^{s^t} S_i^t > s^t + 1 \tag{3.2}$$

Basically, there are hidden routers in a subsequence when the number of involved routers decrementing the TTL is lower than the routers managing the TS option in the associated portion of the trace. In particular, in the subsequence S^t there are exactly w^t hidden routers:

$$w^t = \max \left(\quad 0 \quad ; \quad \sum_{i=1}^{s^t} S_i^t - (s^t + 1) \quad \right) \tag{3.3}$$

Accordingly, the overall number of hidden routers W contained in the trace is:

$$W = \sum_{t=1}^{p} w^t \tag{3.4}$$

In the sample scenario of Fig. 3.5, by applying the Eq. 3.1 on TS, we first determine *INCR*=[1 0 2 0 3 1 1 2 2 0]. Then, all the p longest subsequences of non-zero consecutive elements contained in *INCR* are extracted. In our example, p=3 with S^1=[1], S^2=[2], S^3=[3 1 1 2 2]. The Eq. 3.2 is applied on the extracted subsequences to detect hidden routers: only S^3 reveals hidden routers. By applying the Eq.3.3, we can count the exact number of hidden routers contained in S^3, i.e. $w^3 = 3$. We discovered that there are three

hidden routers located somewhere in the portion of trace associated to S^3, i.e. between the routers reported by Traceroute h_4 h_5 h_6 h_7 h_8 h_9.

The set of subsequences S^t containing hidden routers is the input of the next step.

Step 3: Positioning of hidden routers The goal of this step is to reduce, as much as possible, the uncertainty about the position of the w^t hidden routers detected in the subsequence S^t: up to now, all that we know is that w^t hidden routers are located somewhere in the subsequence S^t. Especially when the final goal is to accurately map the network topology, such level of accuracy is not enough. We would like to clearly identify in which specific transition between routers managing the TTL the hidden routers lie.

We define the *position range* of an hidden router as the number of transitions in which it is potentially located. The higher is the position range, the higher is the uncertainty on the exact position of the hidden router. Initially, the position range for all the hidden routers detected in S^t is s^t: at the end of the second step, the position range of the three hidden routers detected in S^3 of our sample scenario is 5. DRAGO performs the third step to reduce as much as possible such uncertainty.

A first possibility is to analyze the elements of the subsequence S^t one-by-one. Note

Table 3.1: DRAGO: Adopted notation.

Notation	Description
H	h_1 h_2 .. h_n , vector where h_i is the i-th hop discovered by Traceroute along the path.
TS	TS_1 TS_2 .. TS_n , vector where TS_i is the number of routers managing the TS option up to h_i.
$INCR$	$INCR_1$ $INCR_2$ .. $INCR_n$, vector where $INCR_i$ is the number of routers managing the TS option in the transition h_{i-1} h_i.
p	total number of subsequence of consecutive non-zero elements contained in $INCR$.
S^t	t-th subsequence of consecutive non-zero elements extracted from $INCR$. It contains s^t elements.
w^t	Hidden routers contained in S^t.
$S^t_{i,j}$	S^t_i, .. , S^t_j vector of elements in the subsequence S^t .
$w^t_{i,j}$	Hidden routers contained in $S^t_{i,j}$.
W	Total number of hidden routers contained in the trace.

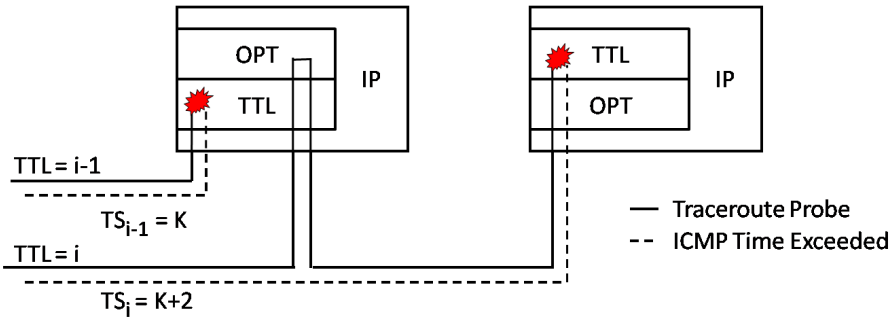


Figure 3.6: Different implementations of the TCP/IP stack and their impact on TS and $INCR$.

that, while $S_i^t > 2$ definitively uncovers hidden routers (and also their exact position in the trace), $S_i^t = 2$ is hard to interpret: it may suggest the presence of a hidden router but this is not always the case. Indeed, both the TTL-decrement and the TS option management are performed at the IP layer of the TCP/IP stack. In distinct implementations, the two operations may be performed in a different order and such circumstance has an impact on $INCR$ and the extracted S^t. For example, Fig. 3.6 shows the (i-1)-th and i-th hops discovered towards the Traceroute destination as well as the order in which the TTL and the TS option are managed: the first hop manages the TTL before the TS option while the opposite happens in the second hop. In this scenario, $INCR_i$ is 2 but there are no hidden routers in this portion of the trace. More in general, analyzing one-by-one the elements in S^t may not uncover all the hidden routers contained in the trace: indeed, each node may manage at most once the TS option and its contribution should count no more than once. Hence, analyzing entire portions of S^t may reveal additional hidden routers.

Hereafter, we use $S_{i,j}^t$ to refer to the elements S_i^t,...,S_j^t in the subsequence S^t. In each portion $S_{i,j}^t$, there are exactly $w_{i,j}^t$ hidden routers:

$$w_{i,j}^t = \max\left(\quad 0 \quad ; \quad \sum_{k=i}^{j} S_k^t - (j - i + 2) \quad \right) \tag{3.5}$$

To accurately locate the hidden routers, the technique should count the number of hidden routers contained in all the possible portions of the subsequence S^t. To explore only a subset of all the possibilities, the technique makes use of a binary tree. Each node in the tree is related to a specific portion $S_{i,j}^t$ and it is labelled with the corresponding $w_{i,j}^t$. The root node in the tree is related to the entire subsequence S_{1,s^t}^t and it is labelled with w^t,

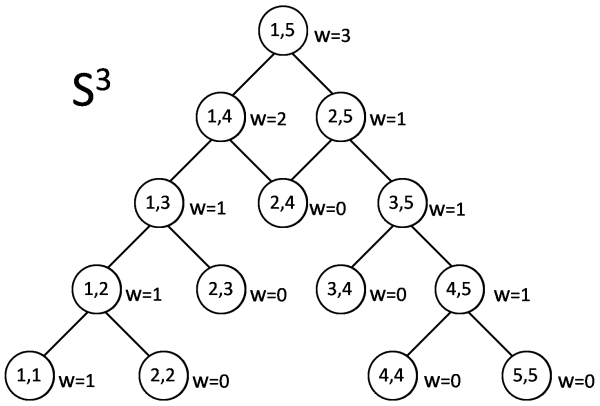


Figure 3.7: The binary tree for the the subsequence S^3 of Fig. 3.5.

the total number of hidden routers contained in S^t.

At the beginning, the tree contains only the root node. The technique generates the two child nodes of a generic node $S^t_{i,j}$ only if $w^t_{i,j} > 0$. When this occurs, the technique *explodes* the node $S^t_{i,j}$ by generating the two child nodes $S^t_{i,j-1}$ and $S^t_{i+1,j}$: a child node is associated to the sequence of its parent shortened of either the first or the last element. The ratio behind this choice is that a portion of trace (a node in the tree) must be further investigated (exploded) only if there are still evidences of hidden routers. At the end of this process, the tree contains several levels (in the worst case, s^t levels). All the nodes at the same level are related to specific portions of the subsequence with the same size: the higher is the level the lower is the size of the portion associated to the nodes. The paradigm adopted to build the binary tree is *depth-first*. The exploration of a branch ends when one of the following conditions is verified:

- The node to explode is associated to a portion made by a unique element $S^t_{i,i}$: the $w^t_{i,i}$ hidden routers are exactly located.
- Both the child nodes $S^t_{i,j-1}$ and $S^t_{i+1,j}$ of the last exploded node $S^t_{i,j}$ do not contain hidden routers, i.e. $w^t_{i,j-1} = 0$ and $w^t_{i+1,j} = 0$: hidden routers are visible at the parent node $S^t_{i,j}$ ($w^t_{i,j} > 0$) but disappear in the child nodes. In this case, we conclude that $w^t_{i,j}$ hidden routers are contained in the portion of the trace associated to $S^t_{i,j}$ but it is not possible to locate such devices with a higher accuracy.

The last phenomenon may also affect a subset of the hidden routers: this happens when the parent node in the tree contains more hidden routers then the ones visible in

the two child nodes. For those hidden routers, the technique is not able to point out their position in the trace with an accuracy higher than the portion associated to the parent node. The position range of an hidden router located in $S^t_{i,j}$ is $j-i+1$: the hidden router may be located in one of $j-i+1$ different transitions. When the hidden router is exactly located in $S^t_{i,i}$, its position range is 1.

Let us apply the third step of DRAGO on the sample scenario of Fig. 3.5. To reduce the uncertainty on the location of the three hidden routers detected in S^3, the third step builds the binary tree reported in Fig. 3.7. At the beginning, the binary tree is made just by the root node. This node is associated to the entire subsequence $S^3_{1,5}$ containing $w^3 = w^3_{1,5} = 3$ hidden routers. Since $w^3_{1,5} > 0$, the node $S^3_{1,5}$ is exploded in $S^3_{1,4}$ and $S^3_{2,5}$. The technique implements a depth-first exploration. Hence, the next considered node is $S^3_{1,4}$ and $w^3_{1,4} = 2$ is computed by applying the Eq. 3.5. In turn, the node $S^3_{1,4}$ is exploded in $S^3_{1,3}$ ($w^3_{1,3} = 1$) and $S^3_{2,4}$ ($w^3_{2,4} = 0$). Then, $S^3_{1,3}$ is exploded in $S^3_{1,2}$ ($w^3_{1,2} = 1$) and $S^3_{2,3}$ ($w^3_{2,3} = 0$). The node $S^3_{1,2}$ is further exploded in $S^3_{1,1}$ ($w^3_{1,1} = 1$) and $S^3_{2,2}$ ($w^3_{2,2} = 0$). The exploration of this branch is terminated: we reached the leaf of the binary tree and we found the exact position ($S^3_{1,1}$) of a hidden router (in the transition h_4 h_5). In addition, note that we count 2 hidden routers in $S^3_{1,4}$ but the child nodes provided details only about one hidden router. We forcedly conclude that another hidden router is located somewhere in $S^3_{1,4}$ but we could not better identify its position. According to the depth-first paradigm, the technique analyzes $S^3_{2,3}$ and then $S^3_{2,4}$: both nodes are not exploded since they do not contain hidden routers. Then, node $S^3_{2,5}$ is exploded and the exploration continues as before until the $S^3_{4,5}$ is exploded in $S^3_{4,4}$ and $S^3_{5,5}$. While $w^3_{4,5} = 1$, either $w^3_{4,4} = 0$ either $w^3_{5,5} = 0$: a hidden router visible in $S^3_{4,5}$ disappeared in the lower level of the tree. We can conclude that a last hidden router lies somewhere in $S^3_{4,5}$ and the technique stopped.

At the beginning of the process, all that we knew was that three hidden routers exist somewhere in the portion of the trace associated to S^3 with a position range of 5 (i.e. somewhere between the nodes h_4 h_5 h_6 h_7 h_8 h_9). By applying the third-step of our technique, we concluded that (*a.*) a first hidden router is *exactly* located in $S^3_{1,1}$ with a position range of 1 (i.e. between h_4 and h_5); (*b.*) a second hidden router is located somewhere in $S^3_{1,4}$ with a position range of 4 (i.e somewhere between the nodes h_4 h_5 h_6 h_7 h_8); (*c.*) the last hidden router is located somewhere in $S^3_{4,5}$ with a position range of 2 (i.e somewhere between the nodes h_7 h_8 h_9).

This result is achieved by inspecting 14 out 15 portions of the subsequences of S^3.

Advantages and drawbacks.

A first limitation of our solution is that DRAGO is not able to distinguish if $INCR_i = 1$ is caused by one of the hops traced by Traceroute or it is caused by a hidden router. From this point of view, DRAGO estimates a lower bound of the hidden routers in the path.

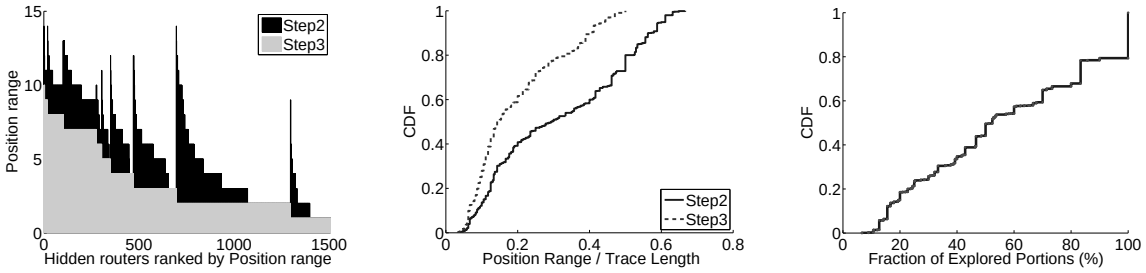
Compared to the techniques proposed in literature, DRAGO shows both advantages and drawbacks. Considering the RR option-based solution proposed in [56], using the TS option provides an almost three times larger exploring range (24 hops against 9 hops). In addition, while the disjunctive logic programming is a computationally complex solution [56], our technique is very light and easy to replicate. On the other hand, DRAGO is not able to assign any address of the detected hidden router and the same hidden router could be acknowledged multiple times in different paths. Compared to tracebox [60], our technique is potentially able to uncover also devices that do not modify the forwarded probe packets but manage the TS option.

3.2.3 Experimental analysis

In this section, we describe the methodology adopted to evaluate DRAGO and the results of the evaluation.

To evaluate DRAGO, we selected $25K$ destinations in distinct ASes among the addresses showing stable responsiveness to ICMP Echo Request probe packets according to the PREDICT project [98]. These addresses have been selected by using the IP-to-AS mapping service provided by the Team Cymru [99]. We launched DRAGO towards these destinations from our laboratory at the University of Napoli. To deal with load balancers, the enhanced Traceroute launched during the first step was instructed to generate probe packets as part of the same flow by replicating the internal mechanism adopted in Paris Traceroute [64]. After having removed the filtered traces and those affected by loops, the final dataset consisted of $22K$ traces containing more than $45K$ addresses.

From the traces of the dataset, we extracted $49,956$ unique transitions $h_{i-1}h_i$ not involving anonymous routers and the corresponding $INCR_i$ value. Besides few exceptions, all the transitions showed a stable number of intermediate routers managing the TS option, i.e. every time the transition $h_{i-1}h_i$ appears in a trace, our enhanced Traceroute reported always the same $INCR_i$ value. About 0.2% of the extracted transitions already exposed hidden routers, i.e. these transitions are characterized by $INCR_i$ values higher than 2.



(a) Position range size. (b) Position range size as fraction of the Traceroute trace. (c) Explored portions in each subsequence.

Figure 3.8: Step 3 - Hidden routers positioning.

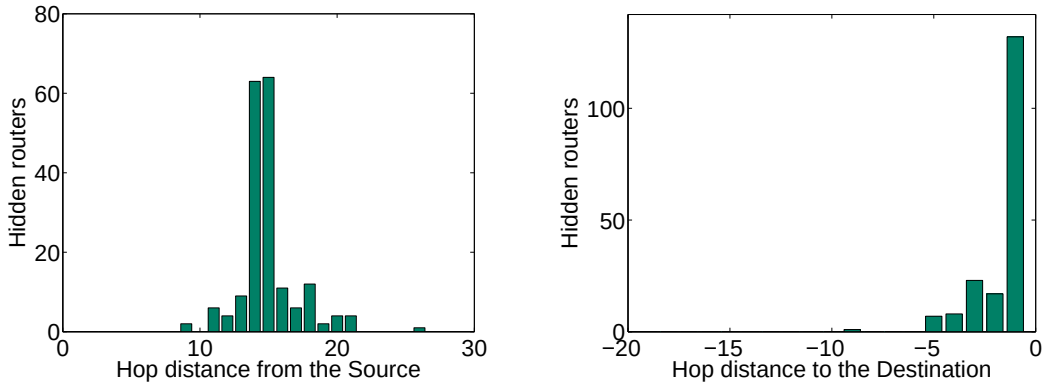
Surprisingly, we observed transitions containing up to 4 consecutive hidden routers: when tracing Internet paths with Traceroute, the traced paths may miss not only single devices but also entire portions of the network.

While we already observed some hidden routers by simply analysing the single transitions, DRAGO discovers additional hidden routers by analysing entire subsequences extracted from *INCR*. Tab. 3.2 shows the number of distinct subsequences and traces containing a specific number of hidden routers. We extracted 29,756 unique subsequences from the dataset: 1,348 (4.5% of the total) contain at least one hidden router. From the trace point of view, almost 6% of all the Traceroute traces in the dataset contains at least one hidden router. Taking into account that the phenomenon has been largely ignored, such a value appears surprisingly high suggesting that hidden routers are not uncommon and may heavily affect the accuracy of the results achieved by classic topology discovery approaches based on Traceroute [8, 9, 2, 15, 16].

Regarding the location of these devices, after the second step, the position range for each hidden router coincides with the size of the subsequence: during the third step,

Table 3.2: DRAGO: Inferred hidden routers.

Hidden Routers	Unique Subsequences	Involved Paths
0	28,408	20,603
1	1,222	1,211
2	98	91
3	23	22
4	5	5



(a) Relative to the source. (b) Relative to the destination.

Figure 3.9: Positions of hidden routers exactly located.

DRAGO tries to reduce such uncertainty.

Fig. 3.8a shows the position range of each detected hidden router after the second and third step. The black portion in the figure is the gain achieved in accuracy: while on average, the position range of a hidden router decreased from 5.3 to 3.3 (-37%), the hidden routers exactly located grew from 7% to 14%. Fig. 3.8b shows the position range as a fraction of the length of the Traceroute trace. From the second to the third step, this fraction decreased on average from 0.32 to 0.19, i.e the final area identified by DRAGO as affected by hidden routers represents on average less than one fifth of the Traceroute trace containing these devices. These results were achieved efficiently thanks to the support of the binary tree: the positioning of the detected hidden routers did not require the inspection of all the possible portions in each subsequences. Fig. 3.8c reports the distribution of the fraction of explored portions for the subsequences containing at least 2 elements: on average, only 57% of all the possible portions were explored.

Finally, for the subset of hidden routers exactly located we computed the hop distance from the Traceroute source (Fig. 3.9a) and destination (Fig. 3.9b): 70% of these devices appeared just one hop far from the destination. These results support the idea that a portion of the detected hidden routers may be middleboxes located in the proximity of the destination network.

3.2.4 Summary and discussion

In this section, we addressed a first long-lasting limitation of Internet path tracing techniques: hidden routers. More precisely, thanks to the adoption of probe packets equipped

with the TS option, we designed and experimentally evaluated a multi-step technique able to detect and locate along the traced path these devices totally invisible to Traceroute. The experimental analysis provided a first quantification of the magnitude of the phenomenon: hidden routers appear in at least 6% of the investigated paths and are mostly located in the proximity of the destinations, thus potentially representing middleboxes located at targeted networks.

3.3 Detecting third-party addresses

In this section, we investigate another severe and largely ignored source of inaccuracy when tracing Internet paths. More specifically, we design and evaluate an active probing technique build on top of IP options-equipped probe packets able to identify third-party (TP) addresses in Internet paths traced with Traceroute. We also investigate the impact of this limitation when the final goal is to infer the ASes traversed on the path towards the destination.

3.3.1 Motivation

As explained earlier in Chapter 2, TP addresses are addresses listed by Traceroute as part of the path under investigation but they are associated to interfaces that are not actually traversed by the traffic sent towards the Traceroute destination. Traceroute typically reports one IP address for each router encountered along the path: this address is commonly believed to be associated to the incoming interface of the router, i.e. the interface on which the Traceroute probe packet has arrived. However, the RFC1812 [49] dictates that the source address of an ICMP error message must be set as the address of the interface chosen by the router to send the packet back to the Traceroute originator, i.e. the outgoing interface. When the solicited router implements this part of the standard and the incoming and outgoing interfaces differ, the address reported by Traceroute is a TP address. The presence of TP addresses in the traced paths represent a severe yet largely ignored source of inaccuracy for several applications relying on Internet path tracing such as the discovery of AS-level connectivity [62, 11, 12, 13], the investigation of network outages and routing anomalies [100, 101, 25] or the assessment of ISP performance [87]. For instance, TP addresses may potentially induce one to wrongly conclude that the traffic sent towards the destination is flowing across an AS that is not actually traversed.

The topic has been only marginally considered in literature and the few available works [62, 63] reported conflicting results: as for hidden routers, the magnitude of the phenomenon related to TP addresses is essentially unkown, mainly due to the lack of measurement techniques able to recognize this anomaly in the traced paths. Identifying the TP addresses is of the utmost importance: for instance, when using Traceroute to infer the AS-level connectivity, researchers may easily improve the accuracy of the reconstructed AS-level topology by isolating the subset of AS links affected by TP addresses that require additional measurements to be confirmed.

3.3.2 The proposed solution

In this section, we describe our active probing technique designed to identify TP addresses in the Internet paths traced with Traceroute. Our technique requires the injection into the network of only two additional probe packets in order to classify an IP address discovered by Traceroute as associated to a traversed interface (i.e. an on-path address – OP) or if it is a TP address [102, 103].

Basic idea

Our technique is based on the IP prespecified TS option [47]. As already described in the Sec. 3.1, this option allows to prespecify in a single packet up to four IP addresses from which a timestamp is requested. We adopt again the notation PROBE A|BCDE, where PROBE is the probe type, A is the targeted destination and BCDE is the ordered list of prespecified IPs from which a timestamp is requested. We remind that the order implies that B cannot insert its own timestamp before A, C before B and A, and so on.

The technique is able to classify an IP address when it is part of a per-interface stamping router (see Sec. 3.1): these routers provide a timestamp only if the probe packet traverses the prespecified interface. The very basic idea behind the technique is that, if an address reported by Traceroute is owned by a per-interface stamping router, one can easily state if this address is on the path or not by prespecifying it inside a probe packet sent towards the destination: if the address inserts a timestamp, the probe packet has traversed that interface, therefore the address is on the path; otherwise, the address is a TP address. To reach this goal, however, it is necessary to remove from the classification process the routers exposing non-compliant behaviors such as the any-interface stamping routers.

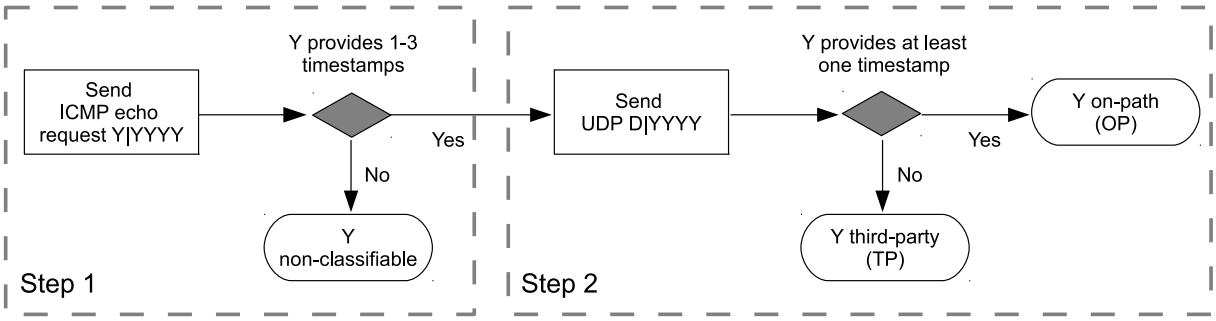


Figure 3.10: Classification of the hop Y discovered by Traceroute towards D.

The algorithm

In order to understand if the hop Y discovered by Traceroute towards the destination D is a TP address, the proposed technique performs the following steps (see Fig. 3.10): (1.) it targets Y with an ICMP Echo Request Y$\vert$YYYY probe to verify if it is classifiable or not (see below); (2.) if Y is classifiable, it targets D with a UDP D$\vert$YYYY probe packet toward a high and presumably unused port to solicit from the targeted destination an ICMP Port Unreachable error message: if the TS option brought back into the payload of the ICMP Port Unreachable message contains at least one timestamp, Y is classified as OP, otherwise it is a TP address. Note that by inspecting the option returned inside the ICMP error message we can observe the status of the option as affected exclusively by the forward path: indeed, since the option is inside the payload of the packet, none of the routers along the reverse path can interfere by modifying the content of the option.

The first step is necessary to remove routers not exposing a per-interface stamping behaviors since these routers may lead to erroneous results. During the first step, we also remove all those destinations exposing abnormal behaviors (see Section 3.1.4). By adopting a conservative approach, a Traceroute hop Y is considered *non−classifiable* every time there is no a clear evidence that its router has a per-network interface stamping behavior, as in the following circumstances:

- *Private addresses*: Y is part of a private addressing block [104] and it may be unreachable by the ICMP Echo Request message or it may be employed in different networks along the path towards the destination. In the latter case, a timestamp in the ICMP Port Unreachable message may be inserted by a router different from the one under investigation.

- *Unresponsive addresses*: no reply to ICMP Echo Request Y|YYYY is received, thus either the targeted device dropped the probe or the reply was filtered along the path. In both cases, it is not possible to clearly assess if the this device manages or not the option and if it is a per-interface stamping router.

- *Addresses removing the option*: Y replies to the ICMP Echo Request Y|YYYY with an ICMP Echo Reply messages not containing the TS option, thus either the targeted hop did not replicate the option inside the reply or the option was removed along the path.

- *Addresses providing no timestamps*: Y does not provide any timestamp when probed with ICMP Echo Request Y|YYYY. Either, the targeted device simply ignores the TS option or it is a per-interface stamping router but the probe packet has not travelled across the prespecified interface. Since we are not able to distinguish this two cases from the collected reply, we conservatively consider these addresses as non-classifiable.

- *Any-interface stamping routers*: the targeted device provides 4 timestamps when probed with ICMP Echo Request Y|YYYY. Such behavior is exposed by any-interface stamping routers (e.g. Juniper routers), which insert their timestamp when the prespecified address is associated to *any* owned interface [95]. Hence, for these routers, the presence of a timestamp in the ICMP Port Unreachable message obtained during the second step would not allow to clearly classify Y as a crossed interface or not.

In conclusion, a Traceroute hop Y is considered *classifiable* only if it provides one or two timestamps when directly probed with ICMP Echo Request Y|YYYY.

3.3.3 Experimental analysis

In this section, we describe the large scale measurement campaign conducted to evaluate the proposed technique as well as the main achieved findings.

Methodology

To evaluate our technique, we selected more than $327K$ destinations in $14K$ ASes among the ones showing stable responsiveness to both ICMP Echo Request probe packets, ac-

Table 3.3: Root cause analysis of IPs non−classifiable as third-party addresses.

Category	IPs	%IPs
Private address	9,428	2.2
Unresponsive addresses	72,775	16.4
Addresses providing no timestamps	64,641	14.6
Addresses removing the option	18,039	4
Any-interface stamping routers	45,963	10.4
Multiple Behaviors	9	∼0
Non−classifiable IPs	**210,885**	**47.6**

cording to the PREDICT project [98], and UDP probes carrying the TS option according to a preliminary experimental campaign conducted from our laboratory at the University of Napoli. To perform a large scale measurement campaign, we used 53 PlanetLab nodes [80] located in different ASes as vantage points. In particular, each node was instructed to (1) send UDP probes towards the destinations and select those which reply and preserve the TS option; (2) launch UDP Paris-Traceroute towards the selected destinations; (3) launch an ICMP Echo Request Y|YYYY toward each intermediate hop Y; (4) select the classifiable hops as the ones providing one or two timestamps; (5) send a UDP probe towards the Traceroute destination prespecifying each time a different classifiable hop collected on the path.

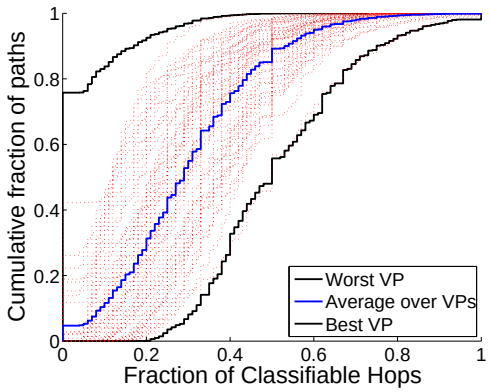
In order to avoid ambiguities caused by load balancers, the UDP probes used to classify the hops and the ones generated by Traceroute are crafted as part of the same flow [64]. After removing the traces affected by filtering, our final dataset consists of about 12M traces for a total number of 443K addresses.

Experimental results

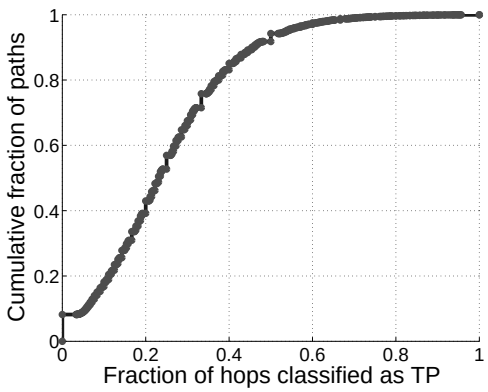
Since each vantage point traced IP paths towards the same destinations, a specific IP address may be discovered by multiple vantage points: this happens especially for those addresses located close to the targeted destinations. More than 96% of IPs were captured by at least two different vantage points, while about a half were captured by more than 35 vantage points.

Hop classification. When an IP address is captured by multiple vantage points, each node independently states if it is classifiable or not according to the collected times-

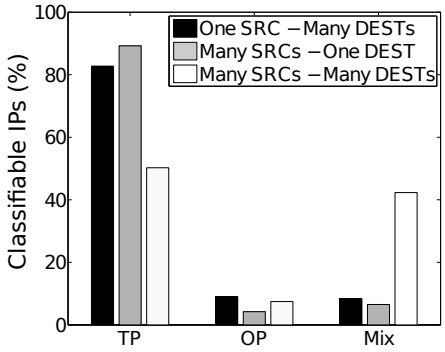
tamps. We removed from the following analysis few addresses showing non-RFC compliant behaviors (see Section 3.1).



(a) Classifiable hops per trace.



(b) TP addresses per trace.



(c) IPs by classification.

Figure 3.11: Third-party (TP) and on-path (OP) addresses in Internet paths traced with Traceroute.

Our vantage points unanimously agreed about more than 97% of IPs labelling 51% of addresses as classifiable and 47.6% as non−classifiable. Conflicting verdicts regarded a limited number of IPs (1.4%) and were mainly caused by the removal of the TS option on a subset of reverse paths. Tab. 3.3 reports a breakdown of non−classifiable IPs per category: our technique was unable to classify such IPs mostly because of devices not replying (16.4%), ignoring the TS option (14.6%), or exposing an any-interface stamping behavior (10.4%).

More than a half of IPs in the dataset were classifiable by our technique. Adopting a per-trace point of view, Fig. 3.11a shows the fraction of classifiable hops per trace (*i*) for each vantage point and (*ii*) over the entire dataset: on average 4%, 52% and 30% hops are classifiable in each trace respectively by the most filtered node (Worst vantage point),

the less filtered one (Best vantage point) and over the entire dataset. Fig. 3.11b reports the distribution of the percentage of hops classified by the technique as TP addresses in each path: more than 90% of the traced Internet paths contain this anomaly and about 21% of hops on average have been classified as TP addresses. This result reveals how the phenomenon related to TP addresses is not as uncommon as suggested by previous works [63]. In addition, most classifiable hops appeared in several paths from multiple vantage points toward multiple destinations and have been classified each time. Fig. 3.11c shows the percentage of classifiable IPs always classified as TP or OP and those classified as both (Mix), on the paths in which they appeared. Such paths are aggregated in three different ways: paths originated (1.) by the same vantage point toward multiple destinations, (2.) by multiple vantage points toward a single destination, (3.) by multiple vantage points toward multiple destinations. The obtained results highlight an unexpected general trend: a classifiable address is labelled by the technique more often as TP than OP address. According to this result, routers often reply to the Traceroute originator by using an interface different from the ones traversed by the packets sent to the targeted destination. For both the aggregations 1 and 2, most of addresses were always classified as TP or OP. However, some IPs were also variably classified and this phenomenon is much more important in the aggregation 3. Such an evidence allows to conclude that *the same address discovered with Traceroute may lie or not on the IP path depending on the (i) originating node and (ii) the targeted destination, essentially due to both inter- and intra-domain routing.*

AS-level transitions affected by TP addresses While $224K$ IPs were classified at least once as TP address, not all the TP addresses impact the AS-level links derived from Traceroute. Mapping each hop to the owner AS [99], we identified in our dataset $14,783$ different ASes. In order to avoid ambiguities caused by the presence of IXPs, we removed from our traces the hops associated to them according to the datasets provided by *peeringDB* [105] and *PCH* [7] as already done in literature [12]. From the resulting $34,414$ AS-level links, we removed 38 links involving sibling ASes according to [106, 107].

Taking into account that the same AS link may appear in several traces toward distinct destinations and may be exposed each time by different IP addresses, a single AS link may be associated to multiple classifications according to how the two involved IPs were classified each time by our technique. In order to deal with this phenomenon, we applied the following methodology: (1.) if both the involved IPs were classified as OP at least once,

we are confident that the corresponding AS link actually exists; otherwise, by adopting a conservative approach, (2.) if both the involved IPs were non−classifiable by our technique at least once, we consider the link as possible; finally, (3.) the AS links which always involved at least one TP address are considered potentially false (see the link ASx−ASz in Fig. 2.7). We counted $1,897$ existing links and $25,990$ possible links. On the other hand, we found $6,299$ potentially false AS links corresponding to about 17% of the links extracted from the dataset.

AS-level loops affected by TP addresses False AS links caused by TP addresses may also generate bogus AS-level loops. In our dataset, we registered $587,126$ traces normally reaching the destination, in which an AS-level loop appeared. Among these traces, about $4,144$ loops involved sibling ASes. Thanks to our technique, we discovered that TP addresses are involved in at least 37% of such loops[1]: $105K$ and $149K$ loops respectively started or ended with a TP address, while $6,083$ loops involved a sequence of consecutive TP addresses. For instance, considering the sequence AS1 AS2 AS3 AS1, if AS2 and AS3 are associated to TP addresses, one possibility is that the corresponding path is entirely contained in AS1, thus generating a bogus loop.

Implications of the achieved results. The surprisingly high value of AS links affected by TP addresses can represent a significant source of AS maps distortion. This conclusion confirms the one drawn by Zhang et al. [62] and is totally different from the one reached by Hyun et al. [63]. Here, we investigated the basic reasons of such contradiction.

According to the heuristic method proposed by Hyun *et al.*, a *candidate* TP address is an intermediate hop that resolves to an AS that differs from the ASes of both adjacent IPs in the same path. The method takes into account also path stability, AS ownerships and hostnames. On the one hand, applying the Hyun's method on our dataset, $7,457$ IPs were classified as candidate TP addresses. Such addresses appeared in 56,595 different IP1 IP2 IP3 sequences where all the IPs were mapped to different ASes and IP2 represents the candidate TP address. Each sequence appeared in multiple traces and each time the involved IPs were classified by our technique[2]: (i) 166 sequences resulted as real AS1 AS2 AS3 transitions, since all the three IPs were classified at least once as OP; (ii) although the candidate TP address was non−classifiable by our technique in $39,824$ sequences, in

[1]Since we used a conservative approach, the real impact may be potentially wider.

[2]As described above, the address identified by Hyun as candidate TP address may effectively lie or not on the IP path depending on the source and the destination.

15,850 of them we recognized as TP address the previous or the next hop, which could be the real responsible of a false AS link; (iii) in the remaining 16,605 sequences, our technique always classified the central address as TP in 85% of cases (the two techniques validate each other in such cases) and as OP in 14% of sequences (in contradiction to the response of the Hyun's method). In the last case, we also found 52 sequences classified as both TP and OP depending on the Traceroute destination and the vantage point used.

At the same time, only 1.5% of the TP addresses identified by our technique is detected by the Hyun's method. The main reason is that a TP address is such independently from the AS point of view. In addition, a Traceroute path may contain multiple consecutive TP addresses – a possibility considered *remote* in [63]: by inspecting our data, we registered $680K$ unique sequences where about 25% were isolated TP addresses, but more than a half consisted of more than 3 consecutive TP addresses. As for the ASy in Fig. 2.7, if a Traceroute path only crosses border routers exposing TP addresses mapped to other ASes, consecutive TP addresses may entirely hide an AS traversed along the path.

3.3.4 Summary and discussion

In this section, we presented and evaluated – to the best of our knowledge, for the first time in literature – an active probing technique able to identify TP addresses in Traceroute traces. Differently from most previous works, our technique does not rely on information provided by BGP monitors and it allows to conclude that TP addresses are very common affecting more than 90% of the observed paths. Thanks to a large scale measurement campaign, we draw the following general conclusions: (i) the same address may be a TP address or not depending on the originating host and the targeted destination; (ii) TP addresses may also be responsible for bogus AS-level loops. We further observed that our technique was able to classify more than half of the total discovered IPs and, surprisingly, about 17% of Traceroute-derived AS-level links involved TP addresses, being thus potentially false. Finally, our results confirmed the conclusion drawn by Zhang *et al.*[62] on the severity of this phenomenon and allowed to explain why such conclusion conflicts with the one achieved by Hyun *et al* [63]: on our dataset, their heuristic method was able to discover only 1.5% of the TP addresses recognized by our technique.

Since TP addresses cause inaccuracy when inferring the ASes traversed on the path towards the destination, the surprisingly high concentration of TP addresses suggests

caution especially when Traceroute is used to assess the AS-level connectivity.

3.4 Dissecting RTT in Internet path tracing

When tracing Internet paths with Traceroute, state of the art implementation of this technique also provides statistics about the RTT experienced for each intermediate hop. Unfortunately, as described in Section 2.2.4, the practical utility of this information is limited since for several reasons (e.g. asymmetric routing) these intermediate RTT values cannot be easily interpreted as correct contributions to the overall RTT experienced along the path. In this section, we describe our contribution to solve this limitation. More precisely, we detail an active probing technique based on the TS option able to dissect the RTT experienced towards the destination in two different chunks determined by different well-identified portions of the traced network path.

3.4.1 Motivation

A common metric used to estimate the delay over a network path is the RTT [108], defined as the length of time it takes to send a data packet toward a destination and receive its response. Monitoring RTT provides useful information about the network status when managing testbeds and operational networks [50]. However, an RTT sample comprises all the delays experienced by the data packet and its response along the forward and reverse path respectively, and it also includes the time the destination takes to inspect the incoming packet and generate the proper response. As a consequence, it can be difficult to interpret RTT values or tease apart the contributing factors.

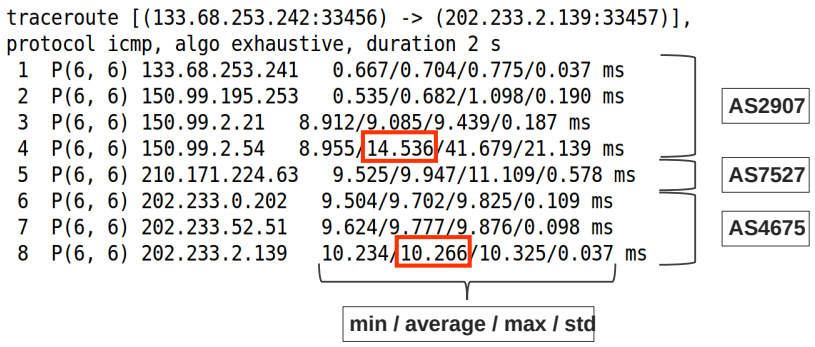
From this point of view, dissecting the RTT into chunks related to specific portions of the network path may be helpful, making it possible to evaluate the relative impact of each subpath on the total experienced RTT. This approach is particularly useful in several scenarios. In a home network, one could isolate the impact of the home network on the RTT experienced toward a destination of interest, such as a website or network service. A large corporation with multiple providers may want to evaluate the impact of its access networks when considering performance optimization and traffic engineering. Service providers may be interested in assessing if the ISP of a particular user has a great impact on the RTT, thus potentially representing the main cause of poor performance perceived by the user.

Unfortunately, accurately dissecting RTT in a traced path is not a trivial task. As described in Chapter 2, one possibility is to rely on the RTTs reported by Traceroute, i.e. the time it takes to send the TTL-limited probe and receive the ICMP Time Exceeded reply. However, it is not uncommon to observe the RTT of intermediate hops to be higher than the RTT of the destination as reported in the sample trace of Fig. 3.12a. The figure shows the output of a state of art implementation of Traceroute (Paris-Traceroute) launched from a vantage point of the Planetlab testbed: the traced path is stable and there are no routers performing load balancing towards the destination, yet the 4-th hop shows an average RTT higher than the one of the targeted destination.

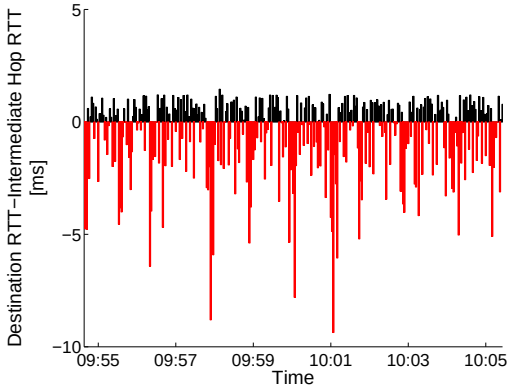
As a possible alternative approach, we could use the Ping command to monitor both the RTT to an intermediate hop and to the destination: Ping estimates the RTT related to a network destination as the length of time it takes to send an ICMP Echo Request packet and receive the ICMP Echo Reply response. Let us assume that our goal is to evaluate the contribution of the provider, AS2907 (SINET-AS), to the overall RTT experienced towards the destination. We monitored the RTTs up to the last hop within AS2907 (150.99.2.54) and the destination by issuing pairs of ICMP Echo Request probe packets closely in time with the Ping command. We launched one probe pair every 200 ms for 10 minutes and computed the average RTT obtained in one second bins. Finally, we computed the difference between the average RTT to the destination and to the intermediate hop: Fig. 3.12b shows the results. For about half of the bins, the intermediate hop had an average RTT higher than the RTT of the destination, making it hard to understand how the intermediate hop contributes to overall delay. Further analysis suggests that this problem holds even for sophisticated advanced variants of the Ping command that injects probe packets as part of the same traffic flow [109]. Dissecting the RTT by relying on the information provided by the path tracing techniques or by network diagnostic tools such as Ping provided inconsistent results.

The inaccuracy of the two methods described above may be caused by (i) asymmetric routing [20], (ii) different network conditions experienced by the different exploited probe packets; (iii) the different amount of time required by the solicited devices to generate the response. Furthermore, when using Ping, the forward path up to the intermediate hop may not represent a subpath of the forward path towards the destination, since the forwarding in the network is typically destination-based.

In conclusion, the RTT values reported by Traceroute cannot be used to accurately



(a) A Traceroute trace from planet1.pnl.nitech.ac.jp: the RTT of the 4-th hop is higher than the RTT of the destination.



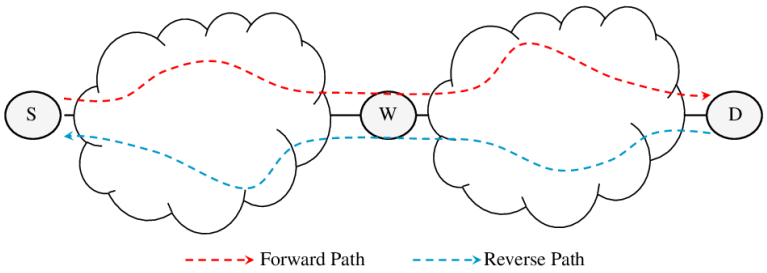
(b) Difference between the average RTTs up to the destination and up to the last hop within AS2907 computed with Ping.

Figure 3.12: On the inaccuracy of traditional approaches useful for RTT dissecting.

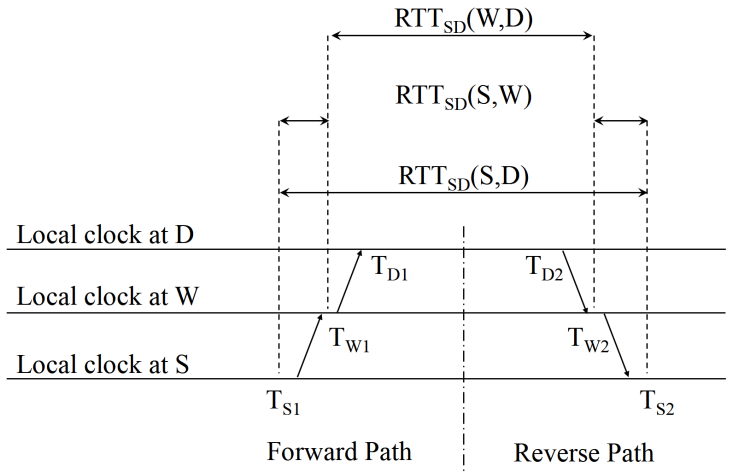
estimate intermediate delays experienced along the path under investigation and alternative approaches such as using Ping seems not being helpful to this end. In this section, we introduce a novel approach to isolate the contributions to the RTT experienced in a traced path. We dissect the RTT into two distinct chunks, using a single purposely crafted probe packet to avoid the complications described above.

3.4.2 The proposed solution

In this section, we describe an innovative active probing technique to dissect the RTT in chunks when tracing Internet paths. The proposed technique is based on the prespecified TS option and relies on an intermediate router that honors the option and appears on both the forward and reverse paths [110]. In these cases, the technique dissects the RTT into (a) the time the probe spends between the source and an intermediate router (in



(a) Baseline scenario (S: source - W: compliant node - D: destination). The proposed technique is able to dissect the overall RTT in two chunks: the time spent by the packets between S and W (both directions) and between W and D (both direction).



(b) Timestamps collected with D|WDDW and the extracted RTT chunks.

Figure 3.13: Dissecting RTT in Internet path tracing.

both directions) and (b) the time the probe spends between the intermediate router and the destination (in both directions).

Algorithm

Our technique makes it possible to dissect the RTT in a path traced toward a network destination that (i) provides at least one timestamp when probed with ICMP Echo Request D|DDDD and (ii) does not expose abnormal behaviors such as extra-stamping [93], i.e. it does not provide more than one timestamp when probed with ICMP Echo Re-

quest D|DXXX where X is an IP address surely not involved on the traversed path. On these paths, we can dissect the RTT into chunks by exploiting a *compliant* router located along the path (see Fig. 3.13a): a compliant node W (i) is part of both the forward and reverse path under investigation; (ii) honors the TS option and provides standard timestamps [47], i.e milliseconds since midnight UT; (iii) provides timestamps both on the forward and reverse path.

Hereafter we adopt the following notation: $\mathrm{RTT}_{S,D}(\mathrm{X}, \mathrm{Y})$ is the time taken by probes sent from the source S to the destination D to travel from X to Y on the forward path and from Y to X on the reverse path. This is a portion of the RTT of the entire path, i.e. $\mathrm{RTT}_{S,D}(\mathrm{S}, \mathrm{D})$. Let W be a compliant node between the source S and the destination D (see Fig. 3.13 as a reference). Besides $\mathrm{RTT}_{S,D}(\mathrm{S}, \mathrm{D})$, our approach estimates $\mathrm{RTT}_{S,D}(\mathrm{S}, \mathrm{W})$ and $\mathrm{RTT}_{S,D}(\mathrm{W}, \mathrm{D})$ by using the same single-probe packet. To this end, we send a ICMP Echo Request D|WDDW probe from S to D. Once S receives the reply, six timestamps are available: (a) the sending and receiving time at the source (T_{S1} and T_{S2}); (b) the timestamp provided by W along the forward (T_{W1}) and reverse path (T_{W2}); (c) the two timestamps provided by the targeted destination D (T_{D1} and T_{D2}). These timestamps allow us to easily compute the RTT chunks (see Fig. 3.13b as reference): $\mathrm{RTT}_{S,D}(\mathrm{S}, \mathrm{D})$ as T_{S2}-T_{S1}, $\mathrm{RTT}_{S,D}(\mathrm{W}, \mathrm{D})$ as T_{W2}-T_{W1} and $\mathrm{RTT}_{S,D}(\mathrm{S}, \mathrm{W})$ as $\mathrm{RTT}_{S,D}(\mathrm{S}, \mathrm{D})$-$\mathrm{RTT}_{S,D}(\mathrm{W}, \mathrm{D})$.[3] When the destination provides only one timestamp when probed with D|DDDD, we send probe packets formatted like D|WDWW, rather than D|WDDW, to dissect the RTT.

The slow path. Packets can traverse a router either through the *fast* (hardware) or the *slow* (route processor/software) path. The IP option on our probe packets causes routers to inspect them and process them on the slow path. Previous work showed that IP options traffic experiences higher delay, jitter, and packet loss, compared to traffic without IP options [111]. Ferguson *et al.* [94] recently observed that the processing time of packets with the TS option depends on the status of the router (traffic and CPU load). Accordingly, the estimated RTTs provide insight into the current condition of network links and routers, a particular view of network path performance.

[3]Note how it would be possible to estimate also several one way delays: from S to D (T_{D1}-T_{S1}), D to S (T_{S2}-T_{D2}), S to W (T_{W1}-T_{S1}), W to D (T_{D1}-T_{W1}), D to W (T_{W2}-T_{D2}) and W to S (T_{S2}-T_{W2}). However, unlike the RTT considered in this section, one way delays are potentially biased if clocks at the various nodes are not properly synchronized, a common case in the Internet.

Accuracy concerns. Concerns about the accuracy of the estimated RTTs may arise since we exploit timestamps provided by distinct network nodes potentially not synchronized. However, we compute each RTT using only the timestamps provided by a single router's clock. Accordingly, any clock offsets do not affect the estimated RTTs. Our measurements are subject to local clock drift, but we assume this impact is negligible over the short duration of a typical RTT.

3.4.3 Experimental analysis

In this section, we first describe the results of an experimental campaign aiming at evaluating the applicability of the proposed approach. Then, we describe two use cases to explore the utility of the proposed approach.

Degree of Applicability

We conducted a study to evaluate how many nodes per path will allow our approach to dissect the RTT (i.e. are compliant). To identify compliant nodes on a path between a source S and a destination D, we first need to discover all the nodes along the path. To this end, we launch an ICMP Traceroute from S toward D. Let us suppose that the destination D provides two timestamps when probed with ICMP Echo Request D|DDDD. For each discovered address Y, we send two probe packets ICMP Echo Request D|YDDY and D|DYYY: if D|YDDY collects four timestamps, then Y is a compliant node. Indeed, four timestamps imply that Y inserted the first timestamp along the forward path (otherwise, D would not have been able to insert its own timestamp), and Y inserted its second timestamp along the reverse path (because the destination D inserted its timestamp before). Non-compliant nodes (i) simply ignore the TS option (D|YDDY and D|DYYY collect none and one timestamp, respectively) or (ii) provide a timestamp only on the forward path (D|YDDY and D|DYYY collect between two and three timestamps and one timestamp respectively) or (iii) provide a timestamp only on the reverse path (D|YDDY and D|DYYY collect one and more than one timestamp, respectively). We refer to the latter two cases as *forward* and *backward stampers*. Forward stampers are nodes that do not appear on the reverse path while backward stampers are more challenging to explain: these nodes are discovered along the forward path but insert a timestamp only when traversed on the reverse path. Load balancing and third-party addresses [112, 63, 102] may explain this behavior. When the destination provides only one timestamp, we make

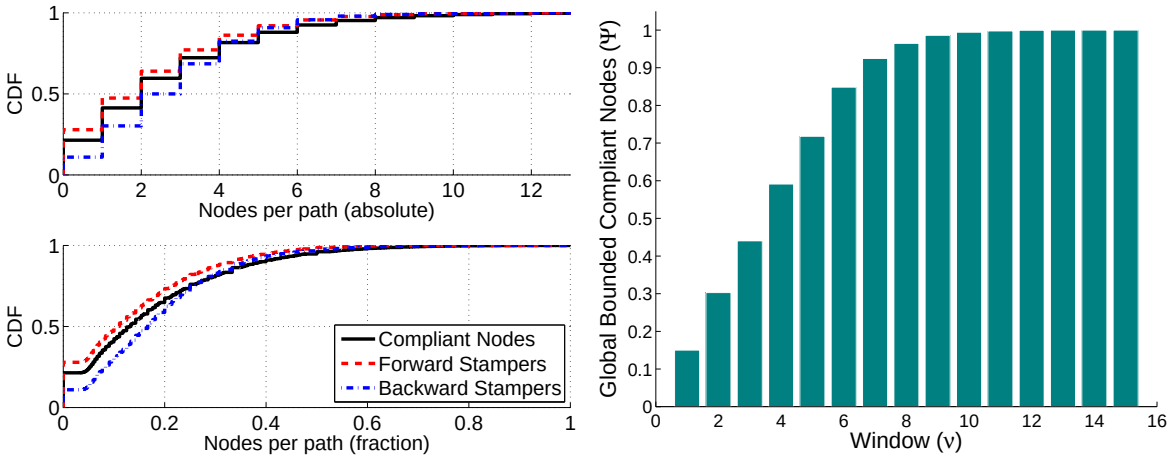
use of ICMP Echo Request D|YDYY probes instead of D|YDDY. In this case, a node is compliant when D|YDYY collects at least three timestamps.

To generate a hitlist of suitable destinations, we extracted the addresses that provided at least one timestamp when probed with ICMP Echo Request D|DDDD in a large-scale experimental campaign from our previous work [95]. Of 1.7M IP addresses probed, 36% replied providing timestamps. From these addresses, we randomly selected one representative IP for each AS [99]. The final hitlist comprises 3,133 distinct ASes, including all Tier-1 ISP networks[4] and 35 out of 50 top-10 ASes for each region, according to the APNIC weekly routing table report. We then performed another experimental campaign using 116 PlanetLab nodes [80] as vantage points (VPs). Each VP made the following steps for each destination of the hitlist: first, it sent two probes, ICMP Echo Request D|DDDD and D|DXXX, to check if the destination is still responsive and is not an extra-stampers. Second, it performed a Traceroute towards the destination. Third, for each address Y discovered along the path, it sent a D|YDDY (or D|YDYY depending on the number of timestamps provided by the destination) and D|DYYY. After removing paths toward extra-stamping and unresponsive destinations, our final dataset comprises 223,548 distinct paths.

Fig. 3.14a reports the compliant nodes observed per path. Ideally, we would like all the intermediate routers to be compliant, in order to split the RTT into all the available chunks. On the other hand, just a single compliant node (W) allows us to split the RTT into $\text{RTT}_{S,D}$(S, W) and $\text{RTT}_{S,D}$(W, D), thus providing much more information on the network status than a classic RTT estimation. We found that about 77.4% of the paths contain at least one compliant node: dissecting the RTT thanks to our approach is possible on a significant amount of investigated paths. In addition, 27.3% of the paths contain more than four compliant nodes potentially allowing a dissection of the RTT in multiple chunks. Finally, on average, we observed 2.5 compliant nodes, 2.1 forward stampers, and 2.7 backward stampers per path: this result implies that, on average, about 17% of the nodes in each scanned path are compliant.

Since compliant nodes represent meeting points between the forward and reverse path and most paths in the Internet are asymmetric at the router level [20, 21], we expect most compliant nodes to appear close to the source or the destination. Our experimental results partially confirm this hypothesis. Let Ω be the set of Traceroute traces and p

[4]http://en.wikipedia.org/wiki/Tier_1_network#List_of_tier_1_networks. August 1, 2013.



(a) Quantity. (b) Relative position.

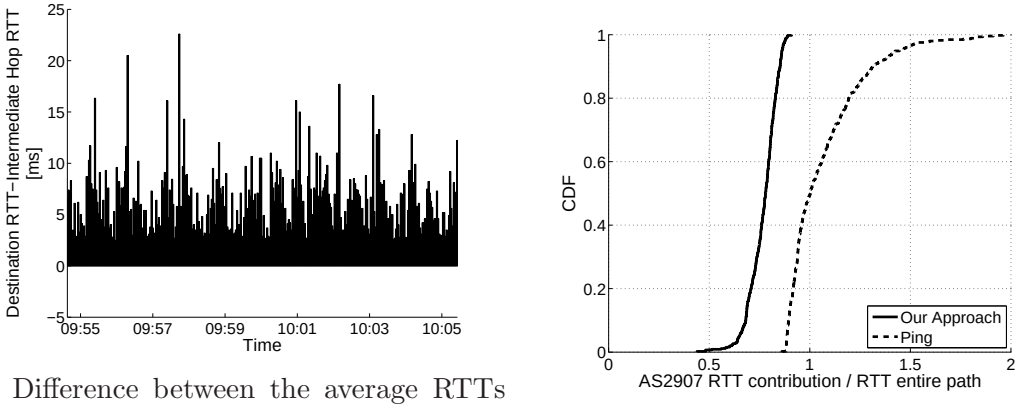
Figure 3.14: Compliant nodes per path.

a particular trace comprising n nodes (a_1, .. , a_i, .. , a_n). Also, let C be the overall number of compliant nodes contained in the dataset. To investigate the position of the compliant nodes, we used a *window* ν to compute the *bounded compliant nodes* $\Phi(p, \nu)$ representing the number of compliant nodes on the path p appearing within ν hops from the source *or* the destination, i.e the compliant nodes contained in (a_1, .. a_ν) and ($a_{n-\nu}$, .. a_n). The *global bounded compliant nodes* $\Psi(\nu) = \frac{\sum_{p \in \Omega} \Phi(p,\nu)}{C}$ represent the global fraction of compliant nodes contained within ν hops from the source or the destination when considering all the paths. Fig. 3.14b depicts how the global bounded compliant nodes varies with ν: if the hypothesis is true, then the global bounded compliant nodes should quickly tend to one. The figure shows an evident though not sharp growth: about 72% of all the compliant nodes occur within 5 hops from the source or the destination, with about 15% appearing just one hop after the source or before the destination. These results confirm that the majority of the compliant nodes are located near the two end points of the paths but there is also a significant percentage of compliant nodes in the middle of the paths.

Applications

We now report two use cases of the proposed approach augmenting Internet path tracing.

Per-Autonomous System RTT contribution. Our approach can isolate the contribution of entire ASes to the overall experienced RTT. To this end, as the first step, we use



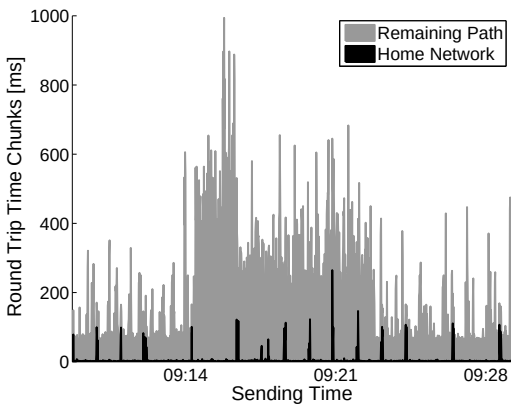
(a) Difference between the average RTTs up to the destination and up to the last hop within AS2907.

(b) AS2907 RTT contribution as fraction of the entire RTT.

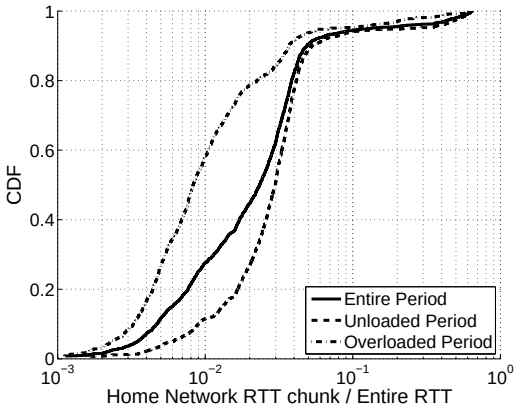
Figure 3.15: Isolating the RTT contribution of AS2907 over the path of Fig.1(a).

path tracing technique to trace the particular path of interest. Consider again the trace in Fig. 3.12a. Our goal is to isolate the RTT contribution of the provider network, AS2907. We cannot rely on the RTT values reported by Traceroute since they are inconsistent: apparently, packets spend more time inside the network of AS2907 compared to the time required to reach the destination and come back to the Traceroute originator. To this end, we monitored the path by using both the Ping command and our approach (the last hop within AS2907, 150.99.2.54, is a compliant node). As described earlier in this section, when using Ping to estimate the RTT up to the last hop within AS2907 and up to the destination with probe packets sent closely in time, we observed again inconsistent results, as reported in Fig. 3.12b: often, the average RTT up to the intermediate hop is higher than the RTT up to the destination (see the negative difference values in Fig. 3.12b). Our approach, instead, always provides coherent results as shown in Fig. 3.15a: the estimated contribution of the AS2907 is always a fraction of the whole RTT. Results obtained with Ping do not provide any meaningful information about the impact of the AS2907 on the end-to-end performance. As shown in Fig. 3.15b, according to Ping, the AS2907 RTT contribution represents on average 106% of the whole RTT, an unreasonable result. On the other hand, thanks to our approach, we can conclude that the AS2907 RTT contribution on the slow path is on average 76.8% of the whole RTT. The probe packets spent more than two-third of the time within the provider network for this specific end-to-end communication.

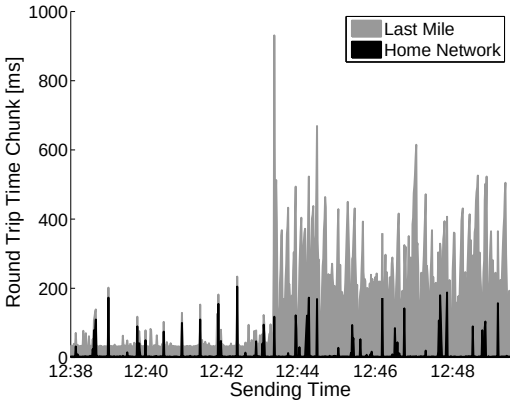
Our approach allows also to isolate the RTT contribution of a target AS network when



(a) RTT chunks over time. Another host transferred large files from 9:14 to 9:23.



(b) Home network RTT contribution as a fraction of the entire RTT.



(c) Home network RTT contribution over the last mile.

Figure 3.16: Home network RTT contribution toward repubblica.it monitored through a wireless link and an ADSL connection.

the first hop within this AS is a compliant node. In the dataset collected to evaluate the applicability, the last hop within the provider AS (the first hop within the targeted AS) is a compliant node in 44,846 (22,236) paths, about 20% (9.95%) of the paths.

Home network contribution to the RTT. The proposed approach can be also used to investigate the impact of home networks on Internet performance, a topic that has recently attracted an increasing interest from the research community [113, 82, 81].

When the home gateway behaves as a compliant node, our approach allows us to evaluate the RTT toward any destination, and, at the same time, the contribution of

the home network, by using a single probe packet.[5] As a case study, we monitored the RTT toward a top-ranked Italian journal website (repubblica.it). The monitored home network is connected to the Internet via an ADSL connection provided by Telecom Italia. The laptop in charge of monitoring is connected via Wi-fi to a NETGEAR DGN2200v3, a common commercial modem-router compliant with our approach. To monitor the RTT, we used ICMP Echo Request D|WDDW probe packets where W is the private address of the modem-router: We approximate the home network contribution as $\mathrm{RTT}_{S,D}(\mathrm{S}, \mathrm{W})$.

Fig. 3.16a shows the trend over time of the RTT chunks. In the beginning, the home network is unloaded. However, from 9:14 to 9:23, another Wi-fi connected host started downloading and uploading large files through the Internet. During the overloaded period, the RTT grows in median by 356% (from 69.8 ms to 249 ms) but the home network played just a marginal role (see Fig. 3.16b). On average, packets spent 4.7% and 2.6% of the entire RTT within the home network during the unloaded and overloaded period, respectively. At the same time, we observed spurious latency spikes inside the home network probably caused by the packet-by-packet impact of contention-induced transmission delays over the wireless link (these spikes disappear on the wired connection). In the worst cases, the spikes represent more than 60% of the total RTT experienced in both the unloaded and overloaded period. These results suggest that the stable performance degradation observed during the overloaded period is not caused by the home network but by congestion of the last mile.[6] Indeed, by replicating the experiment while monitoring the RTT on the last mile and isolating the home network contribution, we observed that downloading and uploading large files through the Internet does not affect the intra-home network delay while it determines a dramatic growth of the delay on the last mile (see Fig. 3.16c).

3.4.4 Summary and discussion

In this section, we faced another important limitation in Internet path tracing: when Traceroute is used to isolate the contribution of specific portion of the network to the overall RTT experienced toward a network destination, the RTT values reported by Traceroute

[5]In these experiments, the precise border of the home network clearly depends on when and how the home router handles the IP option. For instance, if the home router inserts its own timestamp before putting the probe on an overloaded buffer (an instance of home network bufferbloat), such buffering delay is not included in the home network contribution.

[6]The physical connection between a customer's home and the DSLAM or the CMTS.

represent a misleading information. Alternative approaches based on Ping also proved to be no more reliable. In this section, we presented an approach using a single packet to accurately dissect the RTT of an end to end path under investigation. Our approach makes use of standard path tracing technique to first discover the path. Then, it exploits the prespecfied TS option to dissect the RTT in two chunks by relying on an intermediate compliant router along the path: compliant routers are deveices involved in both the forward and reverse path that honors the TS option. Thanks to a large-scale measurement campaign from 116 vantage points comprising 223K traced paths, we observed that the proposed approach can be applied on about 77.4% of the considered paths with more than 50% of the paths containing more than one compliant router, thus potentially allowing one to dissect the RTT in multiple chunks: on average, we observed 2.5 compliant routers per path. We also presented two case studies, showing how our approach allows us to isolate the RTT contribution of (i) an home network and (ii) an entire AS in a end-to-end communication.

3.5 Complementing Traceroute by using malformed IP options

In this section, we explore an innovative path tracing approach totally alternative to the classic TTL-based mechanism employed in Traceroute.

Our approach solicits ICMP Parameter Problem error messages instead of ICM Time Exceeded messages from the routers encountered along the path by injecting packets equipped with malformed IP options. The experimental analysis shows that routers reply to these solicitations and this alternative tracing solution provides complementary information about the traversed path when compared to the classic TTL-based Traceroute.

3.5.1 Motivation

Although state of the art implementations of Traceroute are much more powerful, accurate and robust than the original version proposed by Van Jacobson more than two decades ago [64, 114], as we explained in Chapter 2, several limitations like anonymous routers [52], hidden routers [56], third-party addresses [102], filtering policies, etc. still cause the traced paths to be potentially incomplete or inaccurate. By critically analysing the literature, we observed that all the proposed improvements represent just an evolution of the original

technique: after more than two decades, Internet paths are still discovered by relying on the TTL field of the IP header.

The idea behind the research activity documented in this section is that totally alternative path tracing solutions, i.e. not based on the TTL field, may complement the TTL-based tracing mechanism by providing additional information on the traversed paths: for the first time in literature, we explored this possibility by demonstrating that totally alternative path tracing mechanisms are actually possible. More specifically, we describe three novel active probing methods able to solicit ICMP Parameter Problem error responses instead of ICMP Time Exceeded replies from the routers located along the path. On top of these methods, we build PP Traceroute, an innovative path tracing technique able to emulate the traditional TTL-based Traceroute algorithm. We experimentally evaluate PP Traceroute to investigate its ability to provide additional information – in terms of additional discovered IP addresses (IPs) and circumvention of ICMP Time Exceeded based filtering rules – on the traversed path when compared to the TTL-based technique. Since the experimental results reveal that PP Traceroute is unable to express its full potential, we conduct a results-driven analysis to pinpoint the main factors affecting its effectiveness identifying non fully RFC-compliant stack implementations as one of the main issues. Finally, we discuss how PP Traceroute could be improved by applying the same improvements proposed over two decades for the traditional Traceroute and we outline the path to follow and the challenges to carefully consider for integrating TTL-based and PP-based mechanisms.

3.5.2 Emulating Traceroute with malformed IP options

In the following, we first briefly provide the background necessary to introduce the proposed approach. Then, we detail the three active probing methods proposed to solicit ICMP error messages from routers located along the path by not relying any more on the TTL field.

ICMP Parameter Problem

According to the RFC1349 [115], ICMP messages can be divided in three main classes: errors, requests and replies. The error class includes ICMP types 3 (Destination Unreachable), 4 (Source Quench), 5 (Redirect), 11 (Time Exceeded), and 12 (Parameter Problem). Up to now, not all the types of ICMP message have received the same attention from the

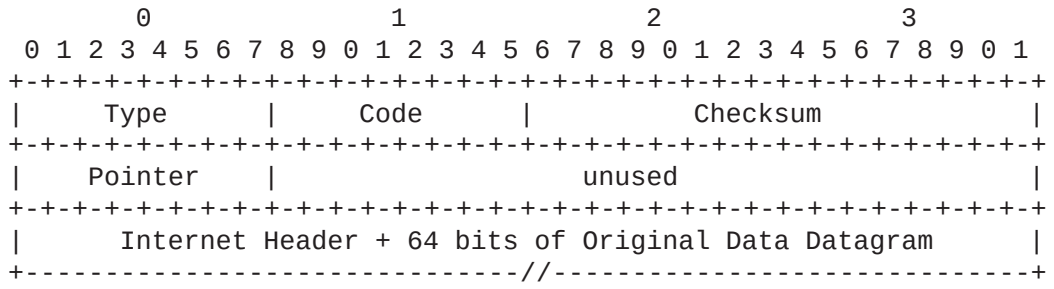


Figure 3.17: ICMP Parameter Problem format.

research community. For instance, in the error class most of attention has been focused on Destination Unreachable and Time Exceeded messages, while the Parameter Problem (PP) type has been totally ignored. In this section, we show how it is possible to solicit ICMP Parameter Problem to trace Internet paths.

The ICMP Parameter Problem format is described in RFC792 [89], while the behavior of routers and hosts with respect to this message is clarified in RFC1812 [49] and RFC1122 [116]: a router (host) *must* (*should*) send a notification to the source by using an ICMP Parameter Problem message when the incoming packet has to be discarded and no other ICMP message covers the detected problem.

Fig. 3.17 reports the ICMP Parameter Problem format as documented in RFC792 [89]. The type field is set to 12 while the code field can vary among 0 (invalid IP header), 1 (a required option is missing), and 2 (bad length). When code is 0, the pointer field identifies the octet where the error occurred. In fact, as usual in case of ICMP error messages, part of the original datagram which caused the error (i.e. the IP header plus the next eight octets of the original datagram) is carried back as payload.

Soliciting ICMP Parameter Problem messages using IP options

The novel active probing methods proposed in this section exploit the TS and RR IP options to solicit ICMP Parameter Problem replies. Here, we point out the error conditions causing the generation of ICMP Parameter Problem messages.

Considering the RR and TS options, the RFC791 explicitly states that, if the IP module fails to check the option of an incoming packet, an ICMP Parameter Problem message *may* be sent to the source in the following conditions:

- **RR option**: (i) there is *some room but not enough room* to insert a full IP address into the option data; or (ii) the route data area is already full.

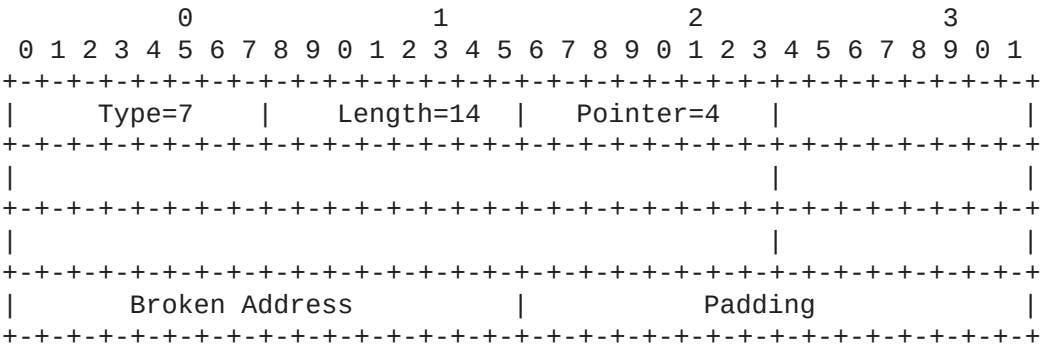


Figure 3.18: CRR probe packet crafted to solicit an ICMP Parameter Problem message from the 3^{rd} hop: the length is set such that not enough available space is allocated for the third address.

- **TS option**: (i) there is *some room but not enough room* to insert a full timestamp into the option data; or (ii) the overflow field counts itself overflows.

According to the standard, recreating the above conditions at a specific hop should cause the packet to be discarded and a notification to be sent to the source through an ICMP Parameter Problem message.

It is worth noticing a contradiction existing between RFC792 and RFC791. According to the former, an ICMP Parameter Problem message is only sent if the error caused the datagram to be discarded, while the latter considers the possibility to generate an ICMP Parameter Problem message when the route data area of a RR option is full, even if this eventuality does not cause the packet to be dropped.

A novel Traceroute based on ICMP Parameter Problem

In this section, we describe three novel indirect probing methods able to solicit ICMP Parameter Problem replies from routers along an IP path. Finally, we detail how we implemented a PP-based Traceroute starting from these methods. The methods we describe are indirect in that they target a network destination with the specific purpose of soliciting a response from an intermediate router whereas direct probing aims at soliciting a reply directly from the targeted destination.

PP indirect probing. The error conditions previously described can be profitably recreated by properly crafting probes to solicit ICMP Parameter Problem messages from network devices along a path.

Our ICMP Parameter Problem-based indirect probing approach includes three different methods as detailed in the following.

Cut Record Route (CRR). A router forwarding a probe equipped with the RR option, having *some room but not enough room* in the option data for a full IP address, should consider the datagram as damaged, discard it, and eventually send an ICMP Parameter Problem message to the source. Accordingly, to solicit an ICMP Parameter Problem message from the i^{th} hop on the path the CRR method sets the RR option length (RRLen) such that there is enough space in the option data for $i-1$ IPs, while only 3 bytes are available for the i^{th} one.

$$RRLen = RRHeaderLen + AddrSize \times (i-1) + BrokenAddr \tag{3.6}$$

Hence, RRLen is computed as reported in Eq. 3.6, where: RRHeaderLen is the RR header size (3 bytes); AddrSize is the size of an IPv4 address (4 bytes); BrokenAddr refers to a malformed slot of 3 bytes, thus unable to contain a full IP address. The pointer field value is initialized to 4, in order to point to the first slot in the RR option data. Thus, the first $i-1$ hops normally manage the option, while only the i^{th} hop detects the malformation, eventually notifying the error to the source. An example of CRR probe is reported in Fig. 3.18: this probe is crafted to solicit an ICMP PP reply from the third hop along the path. Indeed, since the length field is set to 14, the option data accounts for 11 bytes. Accordingly, there is enough space to let the first two hops insert their address, but only 3 bytes are available for the third one. Hence, only the third hop will recognize the malformation, discard the packet and notify the event to the source with an ICMP PP message. Note how two padding bytes are introduced to keep the packet consistent

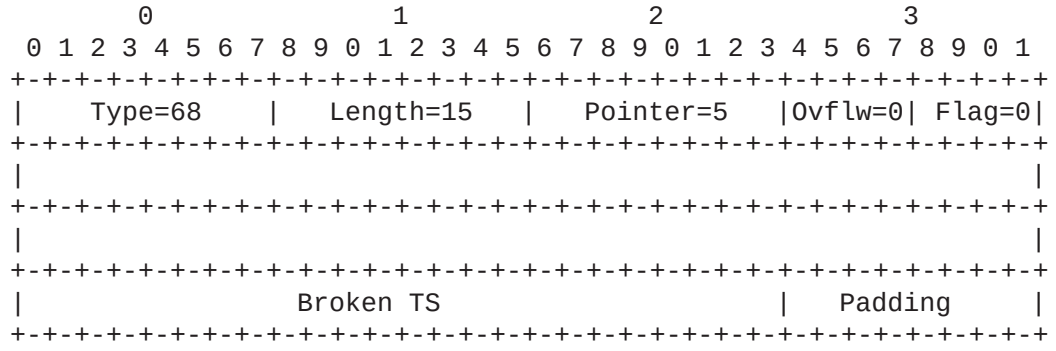


Figure 3.19: CTS probe packet crafted to solicit an ICMP Parameter Problem message from the 3^{rd} hop: the length is set such that not enough avaiable space is allocated for the third timestamp.

with the IP header length field, while just the RR option is malformed.[7] Since the RR option data cannot contain more than 9 slots, the range of the CRR method is limited to 9 hops.

Cut Timestamp (CTS). Similar, CTS solicits an ICMP Parameter Problem message from the i^{th} hop on the path by exploiting a TS option in which enough space is allocated just for $i-1$ timestamps, while only 3 bytes are available for the i^{th} one.

$$TSLen = TSHeaderLen + TSSize \times (i-1) + BrokenTS \tag{3.7}$$

The TS option length (TSLen) is computed as reported in Eq. 3.7, where: TSHeaderLen is the size of the TS header (4 bytes); TSSize is the size of a standard timestamp (4 bytes); BrokenTS refers to a malformed slot of 3 bytes, thus unable to contain a full timestamp. An example of CTS probe crafted to solicit an ICMP PP reply from the third hop along the path is reported in Fig. 3.19: while the pointer field value is initialized to 5, the overflow and flag fields are both set to 0, and the length field is set to 15. In this way, the option data is able to contain the two timestamps inserted by the first two hops. However, only three bytes are available for the third timestamp, causing the third hop to discard the packet and notify the event to the source with an ICMP PP message. Note how in this case, just one byte of padding is required to properly align the IP header to 32-bits words. Since the TS option data cannot contain more than 9 slots, the range of the CTS method is limited to 9 hops.

Overflow in Overflow (OV2). Unlike CTS, the OV2 method exploits the 4 bits overflow field of a full-size TS option. Once all the slots in the option data are filled, a probe equipped with the TS option can travel for at most 16 additional hops before being discarded. In fact, a router forwarding a probe with the TS option overflow field at 15 should detect an overflow exception on that field and discard the datagram, eventually sending an ICMP Parameter Problem message to the source. The OV2 method solicits such condition at the i^{th} router along the path by setting the pointer and overflow fields as reported in Eq. 3.8.

$$1 \leqslant i \leqslant 16 \begin{cases} pointer = TSLen + 1 \\ overflow = 16 - i \end{cases}$$

[7] Padding bytes are treated as End of Options list [47].

$$16 < i \leqslant 25 \begin{cases} pointer = TSLen - TSSize \times (i - 16) + 1 \\ overflow = 0 \end{cases} \tag{3.8}$$

If the target is reachable within 16 hops, OV2 relies just on the overflow field: the pointer is set such that the option appears already full, while the overflow value is set in order to cause the *overflow in overflow* condition after i increments. When the target is x hops far, with $x > 16$, the overflow value is set to zero and $n = x - 16$ slots are left available in the TS option data. Thus, the insertion of n timestamps and 16 increments of the overflow value cause the *overflow in overflow* event at the targeted hop. For example, an OV2 probe crafted to solicit an ICMP PP reply from the third hop contains a full-length TS option of 40 bytes, where the pointer is set to 41 (the option is full) and the overflow field is set to 13. In this way, the first two hops on the path increment the overflow respectively to 14 and 15. However, when the third hop tries to increment the overflow, it recognizes the overflow in overflow exception, discards the packet, and notifies the event the source with an ICMP PP message. Note how in this case, no padding bytes are required. Since the overflow field allows up to 16 increments and the TS option data can contain up to 9 slots, the range of the OV2 method is limited to 25 hops.

Algorithm 1 PP Traceroute based on the OV2 method.

```
Require: target IP address
{Prepare all the probes to send}
for i = 1 to 25 do
    probes[i] = UDPFlowPacket()
    probes[i].TTL = 100
    probes[i].TS.type = 68
    probes[i].TS.length = 40
    probes[i].TS.flag = 0
    if i ≤ 15 then
        probes[i].TS.pointer = probes[i].TS.length + 1
        probes[i].TS.overflow = 16 - i
    else
        probes[i].TS.pointer = probes[i].TS.length - 4 × (i - 16) +1
        probes[i].TS.overflow = 0
    end if
end for
replies = send(probes, timeout)
path = analyze(replies)
ordered_path = order_by_ttl(path)
```

PP-based Traceroute. In order to trace Internet paths with the indirect probing methods described above, it is necessary to correctly rebuild the IP path starting from the collected replies: this task is challenging. Indeed, crafting each probe with a progressive IP identifier (hereafter probe id) and ordering the replies according to this id is not sufficient, because the obtained trace would potentially result incomplete. A router ignoring the IP options always forwards the probe packets independently of the malformations and, therefore, it is not reported in the resulting trace. Furthermore, routers not correctly implementing the standards may potentially cause the resulting trace to be also inaccurate: we deepen this aspect later in this section. In order to correctly rebuild the IP path and to align the discovered routers with those traced by TTL Traceroute, we also take advantage of the TTL-based distance covered by each probe along its travel. Such distance is computed as the difference between the TTL value initially set into the probe and the one carried back by the payload of the ICMP Parameter Problem message. Accordingly, the resulting PP-based Traceroute (hereafter simply PP Traceroute) sorts the collected IPs by TTL distance first, and by probe id then.

Algorithm 1 reports the pseudo code describing PP Traceroute based on OV2. The technique starts by preparing all the 25 probes potentially able to solicit ICMP Parameter Problem replies along the path (line 2-15). More specifically, to deal with possible per-flow load balancers, each probe is generated as part of the same UDP flow (line 2) [64, 68]. The initial TTL value is set to 100, a custom value independent of the operative system (line 3) and the embedded TS option is properly configured (lines 5-7). Then, the pointer and the overflow field of the option are set according to the OV2 method (lines 8-14). Finally, all the probes are injected into the network (line 16), the collected replies are analyzed (line 17), and the extracted addresses are sorted based on the network distance covered by each probe as explained above (line 18).

A prototype of PP Traceroute written in Python is publicly available[8] to foster other researchers to experiment with our technique.

3.5.3 Experimental analysis

In this Section, we describe the experimental campaign conducted to evaluate PP Traceroute: our goal is to investigate the ability of this new technique to capture information about the traversed paths and complement the TTL-based Traceroute. In this analysis,

[8]http://traffic.comics.unina.it/pptr/

we do not consider aspects related to the performance of the two techniques (e.g. the probing efficiency) and we do not adopt for PP Traceroute all the improvements proposed in literature for the traditional Traceroute. Experimental results show that PP Traceroute is actually able to solicit replies from the network and to provide additional information when compared to TTL Traceroute. At the same time, although complementary, our solution seems unable to express its full potential and we deepen the motivations at the basis of these experimental evidences later in this section.

Methodology

In order to experimentally evaluate the proposed technique, starting from the list of addresses showing a stable responsiveness to ICMP Echo Request according to the PREDICT project [98] (one IP address for each available /24 subnet), we selected about 139 K destinations proved to be responsive to UDP probe packets sent from our laboratory at the University of Napoli. We further selected addresses responsive also to UDP probes to detect possible filtering policies triggered by IP options, as deepened later in this section. Our hitlist covers 19,760 ASes, including all Tier-1 ISP networks[9] and 96 of the one hundred top-20 ASes for each region, according to the APNIC's weekly routing table report. Fig. 3.20 shows the geographical distribution of the hitlist obtained with Maxmind [117].

We traced the IP paths towards the destinations of the hitlist from our laboratory with both PP and TTL Traceroute. We instructed the two techniques to generate probes as part of the same flow, i.e. the same flow generated by TTL Traceroute is replicated

[9]http://en.wikipedia.org/wiki/Tier_1_network#List_of_tier_1_networks. May, 2012.

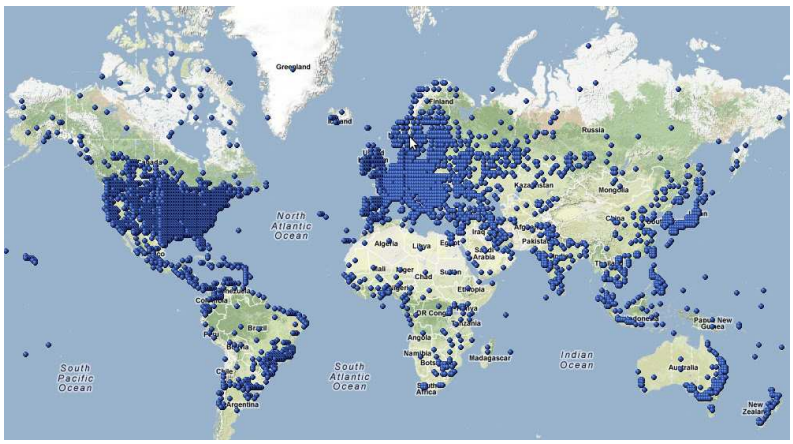


Figure 3.20: Hitlist geographical distribution.

Table 3.4: IPs providing ICMP Parameter Problem replies and also responsive to direct probing.

ICMP PP	ICMP	UDP	TCP	SKIP
78 K	86.1%	79.4%	69.9%	57.8%

in PP Traceroute by equipping the packets with TS and RR options according to the adopted method.

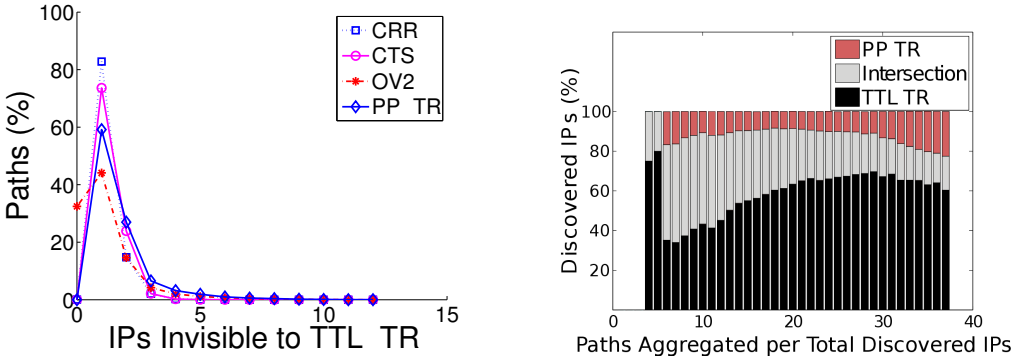
Experimental Results

Hereafter, when we refer to a generic PP Traceroute trace, we mean the trace obtained by merging the IP paths collected with CRR, CTS and OV2 towards the same destination.

Discovering additional IPs along the path. First of all, the experimental campaign confirmed that by injecting probes with malformed IP options, PP Traceroute is able to solicit ICMP Parameter Problem replies from the routers located along the path toward a destination. Jointly, the two tracing techniques discovered 118,242 IPs: 75,320 IPs were collected by both techniques, while 39,611 and 3,311 IPs were respectively discovered only by TTL and PP Traceroute. Targeting the IPs reported by our technique with direct probes such as ICMP Echo Request, TCP Syn, UDP, and SKIP, we observed that not all the IPs which provided ICMP Parameter Problem messages also replied when probed with traditional direct probing We use IP packets carrying a SKIP message to solicit ICMP Protocol Unreachable messages from the targeted destinations [95]. Tab. 3.4 reports the percentage of IPs discovered by PP Traceroute and responsive to a particular direct probe: globally, 4,236 IPs providing ICMP Parameter Problem replies did not reply to any direct probe and 307 of them were also invisible to TTL Traceroute. Hence, we observed that PP Traceroute is able to solicit replies from network devices/interfaces which are invisible to both traditional Traceroute and direct probing. We also notice that some RFC1812-compliant routers returned the entire probe packet in the payload of the ICMP Parameter Problem replies and not just the first 64 bits, i.e. ICMP Parameter problem is subject to *full ICMP* [60]. Based on this result, we believe that network diagnostic techniques like tracebox [60] may potentially benefit from this innovative tracing solution.

An important aspect to consider is the possibility that TTL and PP Traceroute report

different interfaces of the same routers. This possibility is concrete since a router may expose different interfaces to PP and TTL Traceroute. This problem affects also state of the art implementations of TTL Traceroute: for instance, as we describe in Chapter 4, multiple addresses discovered at the same hop may actually correspond to different interfaces of the same router [112]. Being aware of this phenomenon, we used an alias resolution technique, Ally [73], to investigate if the IPs discovered along the paths exclusively by PP Traceroute belong to the same devices listed by TTL Traceroute. Ally classifies two addresses as part of the same router when these addresses provide replies whose IP identifier field proved to be set starting from a unique shared counter. Since this technique can generate false positives (i.e. addresses can be incorrectly classified as part of the same router if counters of different routers are casually temporarily synchronized), we tested each pair of addresses five times in different days to overcome such limitation[10]. The alias resolution technique identified 676 IPs as *in alias* with the devices discovered by TTL Traceroute: these routers used different source IPs when generating ICMP Time Exceeded and ICMP Parameter Problem messages. It is worth noticing that even when PP Traceroute reveals different interfaces of the routers encountered along the path, this additional information is particularly valuable especially when the final goal is to explore the topology of the network [8]: indeed, as described above, part of these interfaces are completely invisible or silent to both TTL Traceroute and traditional direct probing methods.



(a) IPs per path exclusively discovered by PP Traceroute.

(b) Comparison on discovered IPs.

Figure 3.21: PP and TTL Traceroute complementarity.

Finally, by aligning TTL and PP Traceroute traces, 115 of the non-aliased IPs ap-

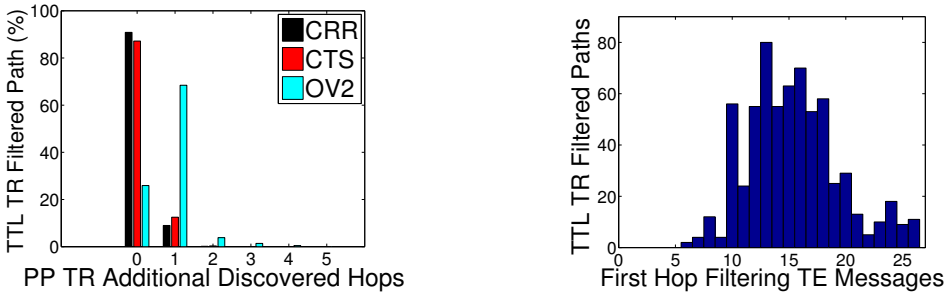
[10]This conservative process follows the basic idea behind other more advanced alias resolution techniques, such as Radargun [118] and Midar [119].

peared at the same position of anonymous routers in the corresponding TTL Traceroute trace. According to these experimental observations, PP Traceroute is able to assign an IP address to some devices unresponsive to traditional TTL-based Traceroute. Note how such a potentiality, though limited, appears useful especially when the final goal is to properly reconstruct the network topology from multiple IP traces, since this task proved to be extremely complex in the presence of anonymous routers [52, 53, 54].

Adopting a per-path point of view, Fig. 3.21a shows the percentage of paths (y-axis) in which PP Traceroute was able to discover a certain amount of IPs not reported by TTL Traceroute (x-axis): CRR and CTS discovered on all the traced paths at least one additional IP, while OV2 reached the same result in 63% of the traces. Jointly, the three methods (PP TR) always discovered at least one address not listed by TTL Traceroute, for a maximum of 12 IPs in a single trace. As reported in Fig. 3.21a, the average number of invisible addresses discovered by PP Traceroute per path is higher than the one discovered by each standalone method. Considering also the path length, Fig. 3.21b shows the average percentage of IPs discovered exclusively by TTL Traceroute, PP Traceroute, and by both of them. Such percentages are reported for paths aggregated per total number of discovered IPs. On average, PP Traceroute contributed from 10% to 20% of the total IPs discovered along the aggregated paths with a growing trend: while the percentage of IPs detected by TTL and PP Traceroute decreases, their relative coverage smoothly increases. The figure also shows how PP Traceroute did not report addresses invisible to TTL Traceroute in shorter paths: the main reason is that the devices located close to our vantage point are not compliant with PP Traceroute.

These experimental observations suggest that PP Traceroute has the potentiality to provide additional information on the vast majority of the traversed paths. This feature finds application in different fields. For instance, it can provide useful information to improve the accuracy and the coverage of Internet topology discovery and may help in performing network troubleshooting.

Circumventing filtering rules. In some scenarios, PP Traceroute proved to circumvent filtering policies. Globally, we observed 656 paths where TTL Traceroute was steadily filtered (reporting 5 consecutive unresponsive hops). Fig. 3.22a shows the number of paths where TTL Traceroute was filtered and the PP Traceroute methods were able to discover a specific amount of additional hops. CRR, CTS and OV2 discovered at least one additional hop beyond the filtering on 14.3%, 19% and 74.1% of these paths,



(a) Hops discovered by PP Traceroute beyond TTL Traceroute filtering.

(b) Position of the first unresponsive hop for TTL Traceroute.

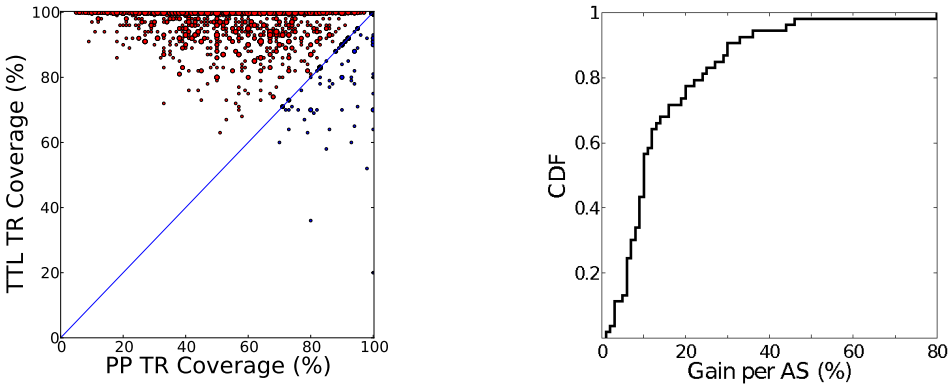
Figure 3.22: Overcoming ICMP Time Exceeded based filtering policies.

respectively. The lower penetration shown by the CRR and CTS methods is due to their limited exploring range. Indeed, as reported in Fig. 3.22b, on average, the first device appearing steadily unresponsive to TTL Traceroute was located at the 15^{th} hop, which is normally out of the exploring range of CRR and CTS. Accordingly, PP Traceroute seems potentially able to circumvent ICMP Time Exceeded based filtering in some paths.

Finally, Fig. 3.23a shows the AS permeability to PP and TTL Traceroute. Mapping each address discovered along the paths to the owner AS [99], we estimated the capacity of the two techniques to capture information about a specific AS: for 42 ASes, PP Traceroute discovered more IPs than TTL Traceroute.[11] The distribution of the gain in coverage assured by PP Traceroute for these ASes is reported in Fig. 3.23b: for 20% of the ASes, the gain was higher than 30% with a maximum of 80%. Most of these ASes are national Internet Service Providers. While such an experimental observation seems suggesting that PP Traceroute better fits some ASes, this analysis must be further investigated with additional experimental campaigns. We left this per-AS analysis as future work. The feature of circumventing filtering policies finds application in the fields of Internet topology discovery and path diagnosis, where it enables to respectively increase the exploring range and to infer the position of network devices filtering ICMP Time Exceeded replies along the path.

In conclusion, compared to the traditional TTL-based approach, a totally alternative tracing solution based on ICMP Parameter Problem provides additional information on the traversed paths. On the other hand, several factors prevented the proposed approach to reach its full potential as we explain in the following.

[11] We consider the set of ASes with more than 10 IPs jointly discovered by PP and TTL Traceroute.



(a) Coverage per AS.

(b) Gain in coverage per AS.

Figure 3.23: Discovered IPs per AS.

Detailed analysis of PP Traceroute performance

In this section, we experimentally deepen the main factors limiting the effectiveness of PP Traceroute.

Intrinsic limitations of the adopted approach. Some intrinsic limitations affect the proposed approach.

One of the factors affecting the performance of PP Traceroute is related to the reduced exploring range of our indirect probing methods. Indeed, since the maximum option size is just 40 bytes, the ICMP Parameter Problem-based indirect probing methods at the basis of PP Traceroute show a limited exploring range: CRR and CTS cannot discover more than 9 devices along each path, while OV2 can discover up to 25 devices.

Since CRR and CTS show a strongly limited exploring range compared to OV2, in our experimental campaign these two methods provided just limited additional information when compared to OV2. The three methods globally collected replies from 78 K IPs. Fig. 3.24 shows the intersection of IPs replying to each method: more than 16 K IPs replied to all the methods; most IPs which replied to both CCR and CTS also replied to OV2, though respectively 144 and 229 IPs did not answer to OV2. Although OV2 and CTS act on the same option, this result seems suggesting that the two events *some room but not enough room* and *overflow in overflow* not always induce the same reaction by a network layer device probably due to stack implementations only partially compliant with the standards. Furthermore, note how *some room but not enough room* may solicit an ICMP Parameter Problem message depending on the adopted IP option: for instance,

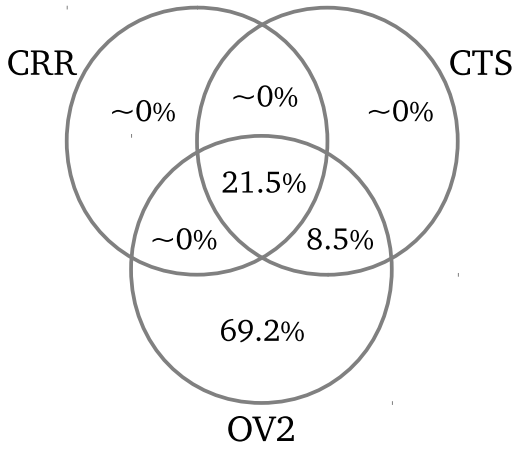


Figure 3.24: IPs responsive to the PP Traceroute methods.

6,675 IPs replied to CTS and not to CRR.

In conclusion, since no more than 25 devices can be traced in each path by PP Traceroute, a subset of the collected traces may be incomplete. At the same time, this limitation does not seem one of the main factors limiting the effectiveness of PP Traceroute. Indeed, paths longer than 25 hops represent less than 0.2% of all the paths contained in our dataset.

Limitations of the current networks. Besides the above mentioned intrinsic limitations, PP Traceroute effectiveness is also affected by external factors.

IP options may trigger filtering policies. IP options-equipped probe packets may suffer from in-transit filtering mainly located at the edge of the network [120]. In these cases, PP Traceroute would not be able to trace the entire path towards the destination. In order to quantify the impact of the filtering on the performance reached by PP Traceroute, in our experimental campaign we also targeted the destinations of the hitlist with two additional UDP probes, respectively equipped with the TS option (UDPTS) and the RR option (UDPRR), both crafted with 9 available slots. Being not malformed, these probes are purposely crafted to normally reach the targeted destination. Since the selected destinations reply to traditional UDP probes, if a destination does not reply with the expected ICMP Port Unreachable (PU) message, we consider the IP options as responsible for this behavior.

In Tab. 3.5, we report a breakdown of the destinations on the type of replies received when targeting them with UDPRR and UDPTS. Besides the expected ICMP Port Un-

Table 3.5: Hitlist breakdown on reply types.

UDPTS \ UDPRR		PU	PU*	PP	TE	HU	NO-R	
	TOT	91820	10674	2205	378	152	33686	
unfiltered	PU	84309	81342	11	1810	6	76	1064
unfiltered	PU*	10416	4	10305	17	-	3	87
unfiltered	PP	404	256	49	8	49	-	42
filtered	TE	322	3	-	-	318	-	1
filtered	HU	76	22	1	-	-	52	1
filtered	NO-R	43388	10193	308	370	5	21	32491
		unfiltered			filtered			

LEGEND

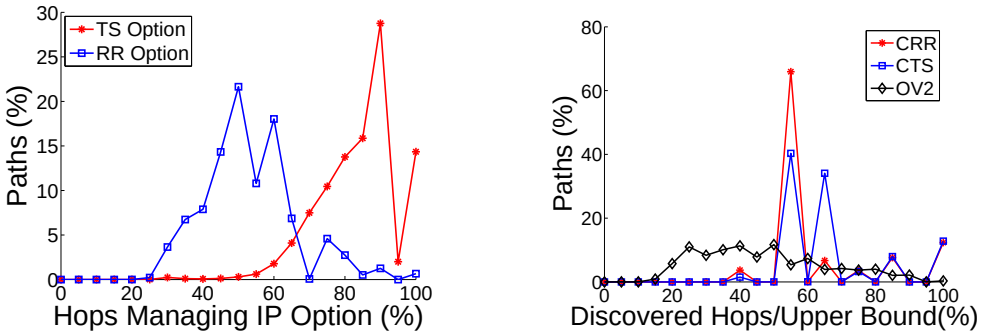
PU	= Port Unreach	TE	= Time Exceeded
PU*	= Port Unreach (Opt Removed)	PP	= Parameter Problem
NO-R	= No Reply	HU	= Host Unreach

reachable (PU) reply, we collected also ICMP Parameter Problem (PP), Time Exceeded (TE) and Host Unreachable (HU) replies. In addition, part of the addresses did not reply at all (NO-R) and some of them provided an ICMP PU message where the original probe sent to the destination and brought back in the payload of the ICMP error does not contain any IP option (PU*). The latter circumstance is a clear evidence that the IP option has been removed from the probe somewhere along the forward path, thus preventing PP Traceroute to entirely trace the path. The row and column labelled as TOT show the total number of addresses replying with a particular type of ICMP message to UDPRR and UDPTS, respectively: for instance, UDPRR solicited ICMP Parameter Problem messages from 2,205 IPs and UDPTS from 404 destinations. The remaining elements of the matrix in Tab. 3.5 reports the number of destinations which replied with an ICMP message type X when probed with UDPRR and type Y when probed with UDPTS, where X and Y are the corresponding row and column labels. For example, 76 addresses replied to UDPRR with ICMP HU and to UDPTS with ICMP PU, while 10, 193 addresses replied to UDPRR

with ICMP PU and did not reply to UDPTS probes. Typically, ICMP PU and ICMP Parameter Problem messages are directly sent by the probed destination, i.e. the probe was not filtered along the path (columns and rows in Tab. 3.5 labelled as *unfiltered*). ICMP Time Exceeded and ICMP HU imply that the probe has been explicitly dropped along the path due to an error condition while, if no reply is received (NO-R), the probe has experienced local or in transit filtering (columns and rows in Tab. 3.5 labelled as *filtered*).

Considering the overall number of probed destinations (139K), UDPRR (UDPTS) successfully reached the destination in 75% (68%) of cases although the RR (TS) option was removed in 11% (12%) of the ICMP PU replies and it was dropped in 25% (32%) of cases. Since all the targeted destinations were reachable without IP options, IP options represent the cause of such filtering. Comparing UDPRR and UDPTS along the same paths, they both reached the destination in 66% of cases, while the first (second) obtained replies where the second (first) failed on 25% (3%) of paths. Both probes were filtered when targeting 23% of destinations. Furthermore, note how some destinations act differently according to the type of option: for instance, 2,205 IPs provided ICMP Parameter Problem replies when probed with UDPRR, while 1,810 of them normally provided ICMP PU replies when probed with UDPTS.

Therefore, these results suggest that filtering at the edge networks represents one of the factors limiting the effectiveness of PP Traceroute.



(a) Hops managing IP options (percentage per path).

(b) Expressed potential of the PP Traceroute methods.

Figure 3.25: On the potential of PP Traceroute.

Not all the routers support IP options. A network device can be potentially discovered by PP Traceroute only if it supports the IP options. However, not all the routers in the Internet support the IP optional headers [92].

In order to estimate the number of devices supporting the IP options in the paths considered in our experimental campaign, we use the information collected by targeting the destinations with the UDPRR and UDPTS probes: since each hop along the path should insert at most one address or timestamp into the option data, it is possible to count the network devices which actually manipulate the IP option by wisely inspecting the original datagram carried by the ICMP Port Unreachable (ICMP PU) replies and counting the IPs inserted into the RR option (hereafter RRs) or the timestamps added into the TS option plus the overflow value (hereafter TSs[12]). Being the proposed ICMP Parameter Problem-based indirect probing methods able to discover only network devices manipulating IP options, this analysis allows to estimate the maximum number of devices along a path that can be potentially discovered by the methods at the basis of PP Traceroute. We refer to this number as the *method upper bound*.

Fig. 3.25a shows the percentage of paths in which we observed a specific percentage of hops supporting IP options related to the path length. On average, 86% of hops per path supported the TS option, while 54.9% supported the RR option. Moreover, in about 15% of paths, all the involved hops managed the TS option. It is worth noticing that this analysis is intrinsically limited for the RR option: indeed, once the option data is full, further network devices do not leave their mark when inspecting the option and cannot be counted. Accordingly, the real percentage of devices supporting the RR option in each considered path may be higher than the lower bound reported in Fig. 3.25a.

According to the results, not all the devices located along a path manipulate IP options, thus being invisible to PP Traceroute. At the same time, a significant portion of the hops in each path supports the IP options (at least the TS option) encouraging the application of our indirect probing approach.

Not all the routers supporting IP options provide ICMP Parameter Problem replies. We compare for each path the number of devices discovered by the PP Traceroute methods with their upper bound in order to evaluate how much the proposed methods reach their full potential. As shown in Fig. 3.25b, on average CRR, CTS and OV2 respectively collected replies from 51%, 57% and 42% of the devices representing their upper bound: roughly, half of the devices managing the options also replied to PP Traceroute. Accordingly, although a significant percentage of routers in each path support at least the TS option, not all the routers supporting the IP options provided their ICMP Parameter

[12]When TSs > 9, the overflow value is greater than 0.

Problem replies to the PP Traceroute methods.

We discovered that the main reason for this experimental evidence is related to routers not correctly implementing the RFC1812 (hereafter non RFC-compliant routers): some network devices manipulate the IP options but do not properly recognize the probe malformations. We consider these devices as non RFC-compliant and classify such an event as an anomaly since, even if the generation of an ICMP Parameter Problem message is not mandatory, the packet must be discarded in any case. The most common type of anomaly affects both the CRR and CTS methods and is generated by a router which ignores the condition *some room but not enough room* and just considers the option full, thus not causing the packet drop. In this case, the option malformation is recognized and reported by the next RFC-compliant router on the path. We named such anomaly *fake-full*. Another type of anomaly affects the OV2 method and is generated by a non RFC-compliant router which ignores the *overflow in overflow* condition, but rather just increments the current value of the overflow field causing it to roll back to zero. In this case, the packet may proceed along the path for further 16 hops before experiencing again the error condition and, normally, this bonus is enough to directly reach the destination. We refer to this anomaly as *ov-reset*. During our experiments, we observed that the first two hops in all the paths originated by our vantage point in Napoli caused both the anomalies: these routers are part of the campus network and belong to the CISCO 6500 series.

To better explain the impact of these anomalies on the trace collected with PP Traceroute, in the following we report and discuss the traces collected by using the three PP Traceroute methods towards google.com. In order to have a reference, we also report in

```
Traceroute from 143.225.x.x to 209.85.148.104,
protocol udp, algo hopbyhop, duration 1 s

 1  143.225.x.x       0.652 ms    0.492 ms    0.298 ms
 2  143.225.190.189   1.761 ms    1.749 ms    2.838 ms
 3  193.206.130.9     0.572 ms    0.630 ms    0.535 ms
 4  193.206.134.246   3.549 ms    3.521 ms    3.518 ms
 5  193.206.134.229  22.589 ms   12.971 ms   13.024 ms
 6  193.206.129.130  12.940 ms   12.925 ms   12.943 ms
 7  209.85.249.54    13.160 ms   13.020 ms   12.989 ms
 8  72.14.232.76     22.481 ms   22.791 ms   22.653 ms
    MPLS Label 749859 TTL=1
 9  72.14.236.21     21.930 ms   22.318 ms   21.963 ms
10  209.85.254.41    32.220 ms   22.686 ms   22.677 ms
11  209.85.148.104   22.782 ms   23.037 ms   22.866 ms
```

Figure 3.26: TTL Traceroute toward google.com.

Fig. 3.26 the same trace collected using TTL Traceroute: the destination appears located 11 hops away from the vantage point, where the 8^{th} hop is part of a MPLS tunnel. As shown in the trace reported in Fig. 3.27a, the CRR method solicited ICMP Parameter Problem messages (type 12, code 0) from the intermediate hops, while an ICMP PU message (type 3, code 3) from the destination. Note that the first probe (id = 1), crafted to solicit a reply from the 1^{st} hop, solicited a reply from the 3^{rd} hop (193.206.130.9). Since the first two hops added their IPs in the following probes, we deduce that they considered the option full without detecting the malformation, thus causing a *fake-full* anomaly. Accordingly, three replies were received from the 3^{rd} hop, which answered for both itself and the first two hops. The 4^{th} and 5^{th} hops were equal to those detected by TTL Traceroute, while the 6^{th} and 7^{th} ones were different: targeting them with an alias resolution tool [73], we found them to be interfaces of the same network devices discovered by TTL Traceroute. Since inside the MPLS tunnel the IP option is ignored, the 8^{th} probe solicited a reply from the 9^{th} hop. Similarly, the 10^{th} probe directly reached the destination, which replied with an ICMP PU message.

The trace collected with the CTS method, shown in Fig. 3.27b, discovered 8 IPs, thus with the same incompleteness of the CRR one. Such trace allows to better understand the behavior of the first two hops, which cause a *fake-full* anomaly, thus normally forwarding all the probes after having incremented the overflow value. This behavior can be easily detected analysing the 3 replies collected from the 3^{rd} hop: the first probe registered 2 overflow increments (1^{st} and 2^{nd} hops) before being recognized as malformed by the 3^{rd} network device; the second probe allowed the 1^{st} hop to insert a single timestamp, while the 2^{nd} hop incremented the overflow value; the third probe allowed both the 1^{st} and 2^{nd} hops to insert their timestamps, thus returning 0 in the overflow field. The same effect is caused by the 10^{th} hop on the 9^{th} probe. Finally, Fig. 3.27c shows the trace collected with OV2. Again, the first two hops appeared silent to our probes and simply incremented the overflow value without checking the *overflow in overflow* condition, thus causing the *ov-reset* anomaly. The corresponding replies coming from the targeted destination, however, allow to detect the last hop while resetting the overflow field: the first two probes directly reached the destination returning 8 and 7 as overflow value, being it already reset respectively at the 1^{st} and 2^{nd} hops. The probe addressed to the 8^{th} hop solicited a reply from the 9^{th} network device, while the 9^{th} and 10^{th} probes solicited replies directly from the destination, respectively reporting an overflow value of 15 and 14.

```
PP Traceroute from 143.225.x.x to 209.85.148.104
protocol udp, CRR method

hop  id              ip     icmp ttl      RR slot #1    RR slot #2      RR slot #3      RR slot #4    RR slot #5     RR slot #6    RR slot #7    RR slot #8 RR slot #9
  1                   *
  2                   *
  3   1    193.206.130.9  I(12,0)  98           0.0.0
      2    193.206.130.9  I(12,0)  98 143.225.190.190         0.0.0
      3    193.206.130.9  I(12,0)  98 143.225.190.190 193.206.130.6           0.0.0
  4   4  193.206.134.246  I(12,0)  97 143.225.190.190 193.206.130.6 193.206.130.252           0.0.0
  5   5  193.206.134.229  I(12,0)  96 143.225.190.190 193.206.130.6 193.206.130.252 193.206.131.249         0.0.0
  6   6  193.206.129.134  I(12,0)  95 143.225.190.190 193.206.130.6 193.206.130.252 193.206.131.249 193.206.129.4          0.0.0
  7   7   216.239.47.128  I(12,0)  94 143.225.190.190 193.206.130.6 193.206.130.252 193.206.131.249 193.206.129.4 216.239.47.216         0.0.0
  8
  9   8     72.14.236.21  I(12,0)  92 143.225.190.190 193.206.130.6 193.206.130.252 193.206.131.249 193.206.129.4 216.239.47.216 209.85.249.20         0.0.0
 10
 11   9   209.85.148.104   I(3,3)  90 143.225.190.190 193.206.130.6 193.206.130.252 193.206.131.249 193.206.129.4 216.239.47.216 209.85.249.20 72.14.238.116     0.0.0
```

(a) CRR method.

```
PP Traceroute from 143.225.x.x to 209.85.148.104
protocol udp, CTS method

hop  id              ip     icmp   ttl   TS overflow
  1                   *
  2                   *
  3   1    193.206.130.9   I(12,0)   98   OV:2
      2    193.206.130.9   I(12,0)   98   OV:1
      3    193.206.130.9   I(12,0)   98   OV:0
  4   4  193.206.134.246   I(12,0)   97   OV:0
  5   5  193.206.134.229   I(12,0)   96   OV:0
  6   6  193.206.129.130   I(12,0)   95   OV:0
  7   7    209.85.249.54   I(12,0)   94   OV:0
  8                   *
  9   8     72.14.236.21   I(12,0)   92   OV:0
 10                   *
 11   9   209.85.148.104    I(3,3)   90   OV:1
```

(b) CTS method.

```
PP Traceroute from 143.225.x.x to 209.85.148.104
protocol udp, OV2 method

hop  id              ip     icmp   ttl   TS overflow
  1                   *
  2                   *
  3   3    193.206.130.9   I(12,0)   98   OV:15
  4   4  193.206.134.246   I(12,0)   97   OV:15
  5   5  193.206.134.229   I(12,0)   96   OV:15
  6   6  193.206.129.134   I(12,0)   95   OV:15
  7   7   216.239.47.128   I(12,0)   94   OV:15
  8                   *
  9   8     72.14.239.63   I(12,0)   92   OV:15
 10                   *
 11   1   209.85.148.104    I(3,3)   90   OV:8
      2   209.85.148.104    I(3,3)   90   OV:7
      9   209.85.148.104    I(3,3)   90   OV:15
     10   209.85.148.104    I(3,3)   90   OV:14
```

(c) OV2 method.

Figure 3.27: PP Traceroute toward google.com.

In conclusion, the most important factor limiting the effectiveness of PP Traceroute is represented by routers not correctly implementing the standards. We believe that a generalized larger adoption of IP options in the Internet not only for measurement purposes will surely force vendors of network equipments to carefully take int account also these portions of the standards releasing the necessary upgrade of the deployed devices.

3.5.4 Toward a hybrid tracing solution

In this section, we describe how PP Traceroute could be improved as a stand-alone technique. Then, we discuss the potential and the challenges to take into account toward a hybrid path tracing solution.

Space for improvements.

Compared to the long history of TTL Traceroute, the study of PP Traceroute is just at the very beginning and there is large room for improvements.

First of all, most of the improvements proposed for the traditional Traceroute such as adapting the exploring direction to limit the intrusiveness [71, 72, 74], and exploiting other probe types, such as TCP and ICMP [76], can be profitably applied also to PP Traceroute. Furthermore, the OV2 limited exploring range of 25 hops can be extended. Indeed, by adopting the prespecified variant of the TS option (flag 3), it is possible to use any intermediate hop as pivot address to start counting the following hops on the path. Essentially, one can request a timestamp from a specific address along the path: if the insertion of this timestamp causes the option's data to be full, any additional routers inspecting the option on the remaining portion of the path will cause the increment of the overflow field, thus potentially allowing one to solicit ICMP Parameter Problem replies from routers located farther than 25 hops. We experimentally observed the effectiveness of a similar approach.

Finally, since the RFC1812 [49] expressly considers the generation of an ICMP Parameter Problem message *each time an incoming packet has to be discarded and no other ICMP message covers the detected problem*, several other ways to solicit the ICMP Parameter Problem error message in addition to the methods proposed in this section may exist.

Toward a hybrid tracing approach: potential and challenges

Our experimental campaign highlighted how PP Traceroute is able to provide additional information on the traversed paths when compared to TTL-based Traceroute, i.e. it can be used to complement Traceroute. This complementarity suggests the design of a hybrid Traceroute able to solicit both ICMP Time Exceeded and ICMP Parameter Problem messages at the same time by merging the two indirect probing mechanisms. In this section, we describe the potential and the challenges to take into account when developing such a hybrid tracing solution.

Potential. First of all, merging the two indirect probing approaches appears straightforward since a TTL-limited probe can be also easily equipped with a malformed IP option. The resulting hybrid Traceroute would be potentially able to (*i*) solicit replies from network devices/interfaces which are invisible to both traditional Traceroute and direct probing, (*ii*) assign an address to devices unresponsive for the traditional TTL-based Traceroute and (*iii*) circumvent ICMP Time Exceeded-based filtering policies. In particular, the hybrid Traceroute may potentially identify both hidden and anonymous routers along the path: indeed, such devices, respectively invisible and unresponsive to TTL-limited probes, may normally reply when solicited with the option equipped probes of our ICMP Parameter Problem-based approach. For instance, as we described in section 3.2, we observed more routers manipulating the TS option than those decrementing the TTL in about 6% of the analyzed paths: these devices invisible to the traditional TTL Traceroute could be precisely identified by such a hybrid tracing solution.

Challenges. The most challenging issue to take into account when merging the two approaches is related to those routers replicating the IP options in the external IP header of the reply. Indeed, the hybrid Traceroute injects into the network probes with both a limited TTL and a malformed IP option. When an intermediate router along the path recognizes an expiring TTL value in the probe, it sends an ICMP Time Exceeded reply back to the Traceroute originator. However, if the router replicates the malformed IP option in the external IP header of the ICMP Time Exceeded reply, the option malformation could be potentially recognized by another router located on the reverse path. In such a case, an ICMP Parameter Problem message would be sent back to the intermediate router to be simply discarded. As a consequence, since the Traceroute originator does not receive any reply, the intermediate router is erroneously labelled as anonymous. In

these cases, issuing additional TTL-limited packets without malformed option may represent a possible solution. Furthermore, properly rebuilding the IP path from the collected ICMP Parameter Problem and ICMP Time Exceeded replies also represents a challenging task, because both the TTL decreasing process and the integrity check of IP options are performed at the network layer of the TCP/IP stack [55], as we also described in section 3.2. Since the ordered sequence of these two operations depends on the particular stack implementation, it is complex to infer the right position in the IP path of a discovered address and non RFC-compliant routers causing *fake-full* and *ov-reset* anomalies further complicate this task.

3.5.5 Summary and discussion

Traceroute, the most adopted network diagnostic technique used to trace Internet paths, supports applications from both the industry (e.g. network troubleshooting) and research (e.g. inference of global aspects of the Internet such as the topology). However, several severe limitations affect this technique and despite the great effort of the research community and the improvements and workarounds proposed over the years, many of these limitations appear not definitively solved. On the other hand, none of the proposed improvements changed the basic TTL-based mechanism proposed in 1989 by Van Jacobson to trace the path: the only exception is the uncommon stand-alone adoption of the RR option already proposed in 1981.

The contribution documented in this section explore a simple idea: since many limitations are related to the basic TTL-based mechanism used in Traceroute (e.g. anonymous routers do not generate ICMP Time Exceeded messages, hidden routers do not decrement the TTL, etc.), totally alternative path tracing approaches not relying any more on the TTL may potentially allow one to mitigate or resolve the limitations of Traceroute.

As first step along this direction, we demonstrated in this section for the first time in literature that completely alternative path tracing approaches exist. We designed and evaluated PP Traceroute, an active probing technique that emulates the traditional TTL-based Traceroute by soliciting ICMP Parameter Problem messages from the routers located along the path instead of ICMP Time Exceeded replies. To reach this goal, PP Traceroute injects probe packets equipped with malformed IP options (TS and RR option): we proposed three different methods to solicit ICMP Parameter Problem messages. By targeting destinations in 19 K distinct ASes, we experimentally observed how this

totally alternative path tracing approach is often able to actually solicit ICMP Parameter Problem messages from the routers located along the path also revealing additional information on the traversed path when compared to the TTL-based Traceroute (e.g. additional interfaces/devices and filtering circumvention).

We also observed, however, that the proposed approach, though complementary to the TTL-based technique, did not express its full potential. We experimentally investigated the factors at the basis of this phenomenon discovering non RFC-compliant stack implementations as the major responsible of this result. Finally, in the light of the experimentally observed complementarity – being PP Traceroute a technique at a very early stage if compared to traditional Traceroute – we discussed (a.) how to potentially improve the technique, and (b.) how the PP- and TTL-based mechanisms could be profitably merged, by highlighting the potential and the challenges of the resulting hybrid approach.

3.6 Final remarks

In this chapter, we described methodologies and techniques designed to investigate, mitigate and possibly resolve important limitations affecting Internet path tracing techniques. Most of the proposed contributions were enabled by the adoption of an innovative measurement traffic enhanced by the optional headers of the IP protocol [47]. The idea behind these contributions is that IP options have been prematurely dismissed by the research community. Although not universally supported, probe packets equipped with IP options collect invaluable additional information about the traversed paths. Encouraged and inspired by the few pioneer works exploiting IP options in Internet measurements, we explored the adoption of this innovative traffic for augmenting Internet path tracing. More precisely, after having characterized how the routers typically manage IP options (with a particular focus on the TS option), we exploited IP options-equipped probe packets to (a.) detect and locate hidden routers – we provided a first quantification of the magnitude of this largely-ignored phenomenon discovering hidden routers in at least 6% of the analysed paths; (b.) identify third-party addresses – with the first active probing technique in literature able to detect third-party addresses, we discovered that this anomaly is very common affecting more than 90% of the investigated paths and about 17% of the AS-level links observed through Traceroute campaigns. In addition, we used the TS option to (c.) dissect the RTT when tracing Internet path overcoming the misleading information pro-

vided by Traceroute about the intermediate delays experienced along the path – thanks to the proposed approach we were able to isolate the contribution of an entire AS and home network to the overall RTT of end-to-end communications. Finally, since most of the limitations of Traceroute are related to the basic TTL-based mechanism, for the first time in literature after the introduction of Traceroute, we propose (d.) an innovative path tracing technique built on top of three different active probing methods that do not rely any more on the TTL field to solicit responses from the routers encountered along the path, thus being totally alternative to the classic Traceroute– these methods experimentally showed the ability to solicit actual replies from the network, to identify anonymous and hidden routers while also circumventing ICMP Time Exceeded-based filtering policies.

All these contributions aimed at augmenting the Internet path tracing techniques and more in general demonstrate the utility of IP options in Internet measurements encouraging a wider adopting of this particular type of traffic to measure the different aspects of the network.

Chapter 4

Assessing new limitations in Internet path tracing

In this chapter, we demonstrate that not all the limitations in Internet path tracing have been identified. More precisely, we discovered that the sequence of addresses reported by Traceroute may be a strongly biased representation of the router-level path followed by the traffic sent towards the destination: by using Traceroute, one may overestimate the number of equal cost router-level paths and infer false router-level path changes.

The analysis we conducted to identify and investigate these new limitations is enabled by alias resolution [10], i.e. the process of gathering under a unique identifier the addresses owned by the same network device. Indeed, thanks to alias resolution, we can transform the IP-level view of the path provided by Traceroute to a router-level view and study if and how the properties of the path change.

We first introduce the problem of alias resolution when tracing Internet paths. Then, we describe Pythia, a novel active probing technique we designed to address the alias resolution problem. Finally, by relying on Pythia and other alias resolution techniques, we discuss the experimental analysis we conducted to identify and investigate the new limitations affecting Internet path tracing.

4.1 On alias resolution and Internet paths

The new limitations we experimentally observed are related to the IP-level view of the path provided by Traceroute. Researchers have been using for decades Traceroute to investigate the topological properties of the Internet [8, 85, 73]: a common approach to this end is to perform large-scale experimental campaigns from multiple vantage points toward a

large number of network destinations. The collected IP-level paths are then manipulated to reconstruct a first IP-level representation of the topology obtained by intersecting the collected traces. However, researchers are well-aware that this representation of the topology may be a strongly biased representation of the real router-level infrastructure of the network since a single router may have dozens of interfaces potentially representing different nodes of the IP-level topology. For this reason, before trying to investigate and model relevant properties of the topology like the robustness or the reliability of the network, researchers typically apply alias resolution: the addresses owned by the same router are identified and merged in a unique node of the graph. In this way, the IP-level topology is transformed in a router-level topology.

While alias resolution is a well known problem in the field of Internet topology discovery, network operators and researchers commonly ignored this problem when investigating the properties of the path followed by the traffic sent toward a given network destination: in fact, the IP-level view provided by Traceroute is typically interpreted as an accurate view of the router-level path with each address identifying a different traversed router. There is a specific reason at the basis of this interpretation: users commonly assume that the addresses reported by Traceroute are associated to the incoming interfaces of the routers encountered along the path towards the destination, i.e. the source address of the ICMP Time Exceed messages solicited by Traceroute is interpreted as the address associated to the router interface on which the TTL-limited packet sent by Traceroute has arrived [121]. Hence, assuming the lack of concurrent routing change, each router should always expose the same incoming interface to Traceroute: as a consequence, users are typically induced to interpret different addresses in the output of Traceroute as different routers.

In this chapter, we demonstrate that this basic assumption does not always hold. Thanks to the adoption of alias resolution techniques, we transformed IP-level path provided by Traceroute to router-level path and observed that multiple addresses reported by Traceroute in the same path can be part of the same router: the IP-level view of the path reported by Traceroute may be a biased representation of the real router-level path followed by the traffic sent towards the destination. More precisely, by interpreting different addresses reported by Traceroute as different routers one may (i) overestimate the number of equal-cost paths towards the network destination and (ii) infer changes in the network routing even when the path in terms of traversed routers is perfectly unchanged.

4.2 A novel alias resolution technique

The analysis we conducted to uncover the new limitations affecting Internet path tracing is based on the adoption of alias resolution techniques. Among the adopted techniques, we also made use of Pythia, a novel active probing technique we developed to address the alias resolution problem for a specific category of routers, i.e. the any-interface stamping routers. In this section, we describe how this technique works in practice and the results of an experimental campaign we conducted to evaluate it.

4.2.1 Previous efforts

Several active probing techniques have been proposed over the years to solve the alias resolution problem [10].[1] In this section, we briefly provide an overview of the previous techniques.

Source address. One of the first alias resolution techniques is known as *common source address* [122]: the addresses A and B are classified as alias if a UDP probe packet sent toward A (B) elicits an ICMP Port Unreachable reply from B (A). A similar approach has been recently proposed in Palmtree [123] that induces the router owning the address A to generate an ICMP Time Exceeded message. Common source address and Palmtree infer addresses in alias (i.e. they belong to the same router) exclusively when they collect replies from addresses different from the targeted ones. As a consequence, these techniques cannot directly tell if two given IP addresses are in alias or not.

Shared IP ID counter. Since some routers maintain a single counter shared among different interfaces to set the IP-layer identifier (IPID) of the outgoing packets, other techniques perform alias resolution by monitoring the evolution of the IPID value over multiple solicited replies. This approach has been first proposed in *Ally* [73] and successively refined in *Radargun* [118] and *Midar* [119]. Recently, a similar approach has been applied also to IPv6 routers [124]. These techniques work exclusively on devices implementing an IPID counter shared among different interfaces and imply an adequate IPID sampling rate in order to infer the addresses in alias.

Timestamp option. The most related work and source of inspiration for our proposal is the technique introduced by Sherry *et al.* [93], one of the first works demonstrating the

[1] Active probing is not the only approach. For instance, oher techniques infer aliases by analyzing the graph of the topology.

potentialities of the IP prespecified TS option for Internet measurements (see Section 3.1). The TS option allows to prespecify in a single packet up to four IP addresses from which a timestamp is requested. By adopting the notation suggested by [93] also used in the previous chapter, hereafter X|ABCD refers to a generic IP probe packet equipped with the TS option, where X is the targeted destination and ABCD is the ordered list of prespecified IPs from which a timestamp is requested. The position of each prespecified address in the ordered list ABCD is essential since it implies that B cannot insert its own timestamp before A, C before B and A, and D before C, B and A. The basic mechanism proposed in [93] to determine if the addresses A and B are in alias or not is to send ICMP Echo Request probes having the format A|ABAB and B|BABA. The technique classifies the addresses as in alias when they provide ICMP Echo Reply messages with four timestamps recorded. In this case, A and B are necessarily part of the same router since the only other possible explanation is the presence of a persistent loop between the router owning A and the router owning B: indeed, four timestamps indicates that TS option has travelled across the router owning A, then the router owning B and then again through the router owning A and B. However, if there is a persistent loop between these two routers, the sender should not be able to receive the ICMP Echo Reply message. For those addresses providing only two timestamps, further investigations are performed: in particular, two addresses are declared as in alias only if (i) the provided timestamp values are consistent and (ii) the experimental observations are compliant with topological constraints.

Similarly to the other active probing techniques proposed in literature, also Pythia injects into the network synthetic traffic to solve the alias resolution problem. However, Pythia has been purposely designed to reconstruct a well-defined category of routers, the any-interface stamping routers. Compared to [93], Pythia uses (i) the prespecified TS option with a different rationale for arranging the addresses in the timestamp requests and (ii) UDP probe packets instead of ICMP Echo Request packets. Thanks to these design choices suggested by the knowledge gathered by experimenting with IP options, Pythia is able to potentially identify all the addresses owned by the same router, even if only one of these addresses is responsive, unlike all the other techniques.

4.2.2 The proposed solution

In this section, we first briefly recall how any-interface stamping routers manage the TS option and the basic principle exploited by Pythia to infer the alias relation including the

algorithm designed to this end. Finally, we discuss the advantages and limitations of the proposed approach.

Any-interface stamping routers

Pythia is purposely designed to solve the alias resolution problem for the any-interface stamping routers. As already described in Section 3.1, thanks to a large-scale measurement campaign targeting 1.7M addresses from multiple vantage points [95], we observed how routers managing the prespecified TS option may manipulate the option in different ways. In particular, any-interface stamping routers provide all the requested timestamps when owning the prespecified addresses independently from which specific interface is crossed by the probe packets. We observed several Juniper routers exposing a similar behavior. The routers part of this category represent about 10% of the devices in the Internet according to recent experimental campaigns [102].

Alias resolution with Pythia

The goal of Pythia is to identify among a set of potential candidates, *all* the addresses owned by the same any-interface stamping router.

The technique exploits UDP probe packets and the TS option as detailed in the following.

- *UDP probe packets.* UDP probe packets toward a high and presumably unused port allow to avoid ambiguities caused by the devices located along the reverse path. Indeed, these probes solicit ICMP Port Unreachable messages from the destination. Since the ICMP error messages typically returns the original packet triggering the error into the payload, it is possible to extract the TS option from the payload of the reply as affected exclusively by the forward path.

- *Prespecify the destination first.* While UDP probe packets allow to avoid ambiguities caused by the reverse path, the routers located along the forward path may still interfere by inserting their own timestamps in the option. To exclude such ambiguities, Pythia prespecifies the destination of the probe packet always as the first address into the TS option: in this way, none of the routers located along the forward path can insert its own timestamp before the destination.

The combination of these two mechanisms allows one to conclude that any timestamp observed into the TS option returned in the payload of the ICMP Port Unreachable reply has been surely inserted by the targeted device.

Given an initial set of addresses to alias, Pythia performs two phases: (i) preliminary test and (ii) alias resolution.

- *Preliminary test.* This phase aims to identify the subset of addresses owned by any-interface stamping routers and to exclude devices showing anomalous behaviors. To this end, for each address A of the initial set of addresses, Pythia sends two UDP probe packets A|AAAA and A|AZZZ, where Z refers to an address at the University of Napoli, known to be outside the traversed path. The first probe (A|AAAA) allows to split the set of candidate addresses in three main subsets: (a) *unresponsive* addresses, (b) *compliant* addresses − the ones providing 4 timestamps being owned by any-interface stamping routers; (c) *non-compliant* addresses − those providing less than 4 timestamps. As already proposed in [93], the second probe (A|AZZZ) allows to remove from the compliant address set the routers showing anomalous behaviors: since the address Z is surely not located on the traversed path, observing any timestamp associated to Z demonstrates that the targeted router inserts extra timestamps independently from the prespecified addresses. Since this behavior may strongly affect the accuracy of our results, when it is recognized, the corresponding address is considered non-compliant with the technique. We also consider as non-compliant addresses the destinations exposing the other non-RFC compliant behaviors described in Section 3.1.4. The sets of unresponsive and compliant addresses represent the input of the following phase.

- *Alias resolution.* In this phase, Pythia performs all the operations required to identify the addresses contained in the initial set owned by the same any-interface stamping router. Let us denote with α and β the ordered lists of addresses respectively compliant and unresponsive. The technique iteratively performs three steps:

  1. An address A is popped from α, hereafter we refer to this address also as the *pivot*. During this iteration, the technique tries to infer all the addresses in alias with the pivot. To this end, a new ordered list, named γ, is created by

concatenating α and β: γ contains all the addresses potentially in alias with A. Let us assume that γ contains the addresses B,C,D,E and so forth.

2. Pythia sends a first UDP probe packet A|ABCD and counts the number of collected timestamps to (i) infer the addresses in alias with the pivot and (ii) determine the next probe to send as reported in Tab. 4.1. For instance, when the probe A|ABCD solicits an ICMP Port Unreachable message where the returned TS option contains two timestamps (i.e. the ones associated to A and B), this is a clear evidence that A and B are in alias: indeed, the probe is crafted such that only the router owning A is allowed to insert timestamps. Furthermore, the lack of a timestamp associated to C implies that the same router has not recognized this address as an owned one. At the same time, we do not have any clue about D because this address appears just after C in the ordered list of prespecified addresses. Accordingly, when the probe A|ABCD collects two timestamps, we conclude that A is in alias with B but not with C. Since B and C have been already tested, two new addresses are extracted from γ and prespecified in the next probe (A|ADEF). Note how Pythia is able to infer up to 4 addresses in alias within one probe when four timestamps are collected. These UDP probe packets are sent until all the addresses in γ have been tested against the pivot.

3. Once all the addresses contained in γ have been tested against the pivot, Pythia stores the pivot and all the addresses recognized as in alias with it. These addresses are also removed from α and β. As long as a new pivot is available, i.e. α is not empty, the technique performs a new iteration starting from the first step.

A retransmission mechanism is also adopted to deal with possible rate limiting policies employed by the router owning the pivot address. Pythia is made publicly available to the research community.[2]

[2]http://traffic.comics.unina.it/pythia/

Table 4.1: Pythia - Inferences and next probes to send according to the timestamps collected with UDP A|ABCD.

Collected Timestamps	Inference	Next Probe
Only A stamps	A,B not in alias	A|ACDE
Only A and B stamp	A,B in alias; A,C not in alias	A|ADEF
Only A, B and C stamp	A,B,C in alias; A,D not in alias	A|AEFG
A, B, C and D stamp	A,B,C,D in alias	A|AEFG

Advantages and drawbacks

Compared to the state-of-the-art alias resolution techniques, Pythia shows both advantages and drawbacks.

First of all, to the best of our knowledge, Pythia is the unique active probing technique in literature potentially able to identify up to four addresses in alias within a single probe packet whereas, to reach the same result, traditional pairwise techniques would require to test six different pairs of addresses[3]. Besides the linear probing complexity of the preliminary step, Pythia requires a single probe packet to infer if two addresses are in alias or not, whereas other pair-wise techniques such as Sherry *et al.* [93] and Ally [125] require at least two and three probes, respectively. Finally, differently from all the other techniques, Pythia is able to tell if a given address B is in alias or not with the pivot even if B does not reply at all to active probing.

On the other hand, Pythia is not free of limitations. The TS option has an impact on the router responsiveness [120, 95] reducing the set of addresses that could be used as pivot. Furthermore, once selected, a pivot is targeted with multiple probe packets and this may cause the targeted router to be silent to our probes due to the exceeding of specific ICMP rate limiting thresholds. Note that reordering the probes may strongly help in mitigating this limitation.

4.2.3 Experimental analysis

In this section, we describe (i) the experimental campaign we adopted to evaluate Pythia and to compare it with other techniques; (ii) the set of performance metrics we consider in the evaluation; (iii) the main findings for the evaluation phase.

[3]When transitivity closure is not applied.

Methodology

To experimentally evaluate Pythia, we used the information provided by the MERLIN platform [85] as a reference: MERLIN natively provides a router-level view of the network by exploiting IGMP probing [126]. Although affected by several limitations such as filtering [86] and the scope limited to the multicast enabled part of the network [85], the information provided by IGMP probing is typically considered highly accurate and has been already used as a reference in several previous works [93, 127].

During a preliminary experimental campaign based on MERLIN, we collected information about 777 Juniper routers[4] located in 12 distinct ASes of different size (tier-1, transit and stub networks). We tested Pythia on Juniper routers because empirical evidences suggest that these devices act as any-interface stamping routers [95, 102]. We compared Pythia to Palmtree [123] and Motu, a publicly available tool developed by CAIDA[5] that implements the technique proposed by Sherry *et al* [93].

From the routers of the reference dataset, we extracted $6,503$ addresses and applied the following methodology to deal with the quadratic probing and computational complexity of the employed techniques. Each tested technique was evaluated on 100 different chunks. To generate a chunk, we performed three steps: (1) we first randomly selected 10 routers of the reference dataset and extracted all their addresses; (2) from this set of addresses we randomly selected up to 50 IPs; finally (3) we generated all the possible combinations of two addresses starting from the IPs sampled during the previous step. Techniques requiring in input a list of addresses, such as Pythia and Palmtree, were fed with the lists obtained during the second step, while those requiring in input IP pairs (Motu) were fed with the lists obtained at the third step. This two-step sampling process allowed to (i) strongly reduce the time required to obtain the experimental results and (ii) preserve in each chunk a significant number of addresses actually in alias. Finally, since a well-known problem for active probing technique is the dependence of the obtained results on the used vantage point, we tested Pythia and the other techniques from 12 PlanetLab nodes [80].

[4]The *IGMP* probing provides also some indications about the brand of the router. Interested readers may refer to [126] for more details.

[5]http://www.caida.org/tools/measurement/motu/

Table 4.2: Alias resolution performance metrics.

Name	*Acronym*	*Formula*	*Description*
Applicability	APP	$\frac{\|D\|}{\|D\|+\|U\|}$	How applicable is the alias resolution technique?
Positive Hit Ratio	PHR	$\frac{\|TP\|}{\|TP\|+\|FN\|+\|UP\|}$	What is the fraction of pairs in alias that is properly aliased by the technique?
Negative Hit Ratio	NHR	$\frac{\|TN\|}{\|TN\|+\|FP\|+\|UN\|}$	What is the fraction of pairs not in alias that is properly dealiased by the technique?
Hit Ratio	HR	$\frac{\|TP\|+\|TN\|}{\|D\|+\|U\|}$	What is the fraction of pairs properly aliased or dealiased by the technique?
Mismatch Ratio	MR	$\frac{\|FP\|+\|FN\|}{\|D\|+\|U\|}$	What is the fraction of pairs wrongly aliased or dealiased by the technique?
Positive Predictive Value	PPV	$\frac{\|TP\|}{\|TP\|+\|FP\|}$	How much can we trust the technique when two addresses are declared as in alias?
Negative Predictive Value	NPV	$\frac{\|TN\|}{\|TN\|+\|FN\|}$	How much can we trust the technique when two addresses are declared as not in alias?

TP: True Positive	FN: False Negative	UP: Unknown Positive	P= TP ∪ FP	U= UP ∪ UP
FP: False Positive	TN: True Negative	UN: Unknown Negative	N= TN ∪ FN	D= P ∪ N

Performance Metrics

Properly evaluating and comparing alias resolution techniques is not straightforward. We adopt multiple performance metrics as explained in the following. Two given addresses can be classified by a generic alias resolution technique as (i) in alias, (ii) not in alias or (iii) unknown − i.e. they are not-classifiable for some reasons − independently on how the technique works. Accordingly, to compare different techniques tested over the same initial set of addresses, one possibility is to consider all the pairs extracted from this set.

By inspecting the results generated by a specific technique, the set of pairs can be split in three disjoint sets: $P-$ pairs classified as in alias; $N-$ pairs classified as not in alias; $U-$ not-classifiable pairs. By taking into account the ground truth, these three sets can be further exploded in True Positive (TP) and Negative (TN), False Positive (FN) and Negative (FN), Unkown Positive (UP) and Negative (UN). Obviously $P = TP \cup FP$, $N = TN \cup FN$ and $U = UP \cup UN$. Furthermore, we refer to the set containing all the pairs classified by the technique as the Decision set $D = P \cup N$. We used these sets to evaluate the tested techniques according to the performance metrics reported in Tab. 4.2.

These metrics allow us to estimate the level of applicability (*applicability*), accuracy (*hit and mismatch ratio*) and trustworthiness (*positive and negative predictive value*) for

each technique.

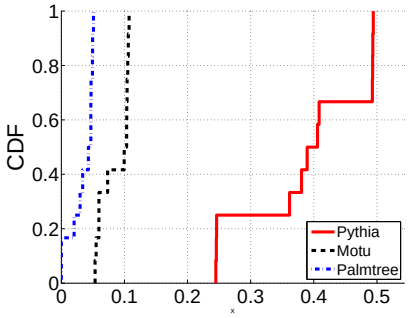
Experimental results

In this section, we present the results obtained with a measurement campaign conducted between the 1^{th} and 14^{th} of May 2013 from 12 PlanetLab nodes. For each vantage point, we created a unique file containing all the pairs probed in the chunks and the corresponding outcomes of the alias resolution technique tested. Since Palmtree cannot directly infer if two given IP addresses are in alias or not, we considered transitivity closure on its results. Fig. 4.1 reports the distributions of the performance metrics over the vantage points (1a-g) and aggregated statistics (4.1b).

Applicability. Pythia is able to classify many more pairs than the other tested techniques (Fig. 4.1a): on average, Pythia, Motu and Palmtree classified one pair for every 2.6, 11.6, 28.6 pairs. Thus, Pythia was 4.5 and 11 times more applicable than Motu and Palmtree, respectively. This result can be explained by considering that, unlike the other techniques, Pythia is able to classify a pair even if only one of the two addresses replies.

Hit and Mismatch Ratio. Pythia showed a higher hit ratio but also a higher mismatch ratio when compared to the other techniques (Fig. 4.1c and 4.1d). However, we registered an absolute gain in hit ratio that is much more significant than the loss we observed in terms of mismatch ratio: by comparing Pythia to Motu (Palmtree), the hit ratio grew on average from 8.5% (3.4%) to 37.8% whereas the mismatch ratio from 0.1% (< 0.1) to 1%.

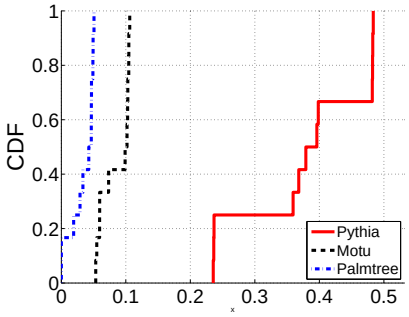
Positive and Negative Hit Ratio. Considering that the vast majority of the pairs in the dataset consists of addresses not in alias (about 80% of all the pairs), one could imagine that the higher hit ratio of Pythia is determined exclusively by pairs correctly identified as not in alias: this intuition is only partially true. Indeed, both the positive and negative hit ratio for Pythia resulted higher than the other techniques (Fig. 4.1e and 4.1f), although the gain was much more significant over the pairs actually not in alias. Experimental results showed that Pythia, Motu and Palmtree were able to correctly identify respectively 57.3%, 47.3% and 19.8% of the pairs actually in alias.



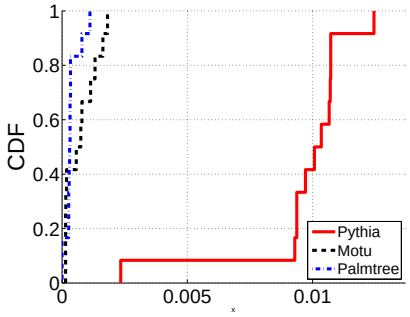
(a) Applicability.

Metric	Pythia	Motu	Palmtree
APP	38.8 (9)	8.6 (2)	3.5 (2)
HR	37.8 (9)	8.5 (2)	3.4 (2)
MR	1.0 (0)	0.1 (0)	0.0 (0)
PHR	57.3 (6)	47.3 (12)	19.2 (10)
NHR	33.6 (10)	0.1 (0)	0.0 (0)
PPV	99.6 (0)	99.8 (0)	98.60 (2)
NPV	96.5 (1)	33.3 (33)	-

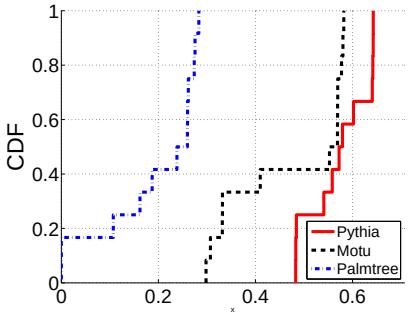
(b) Statistics in%: Format Mean(St Dev).



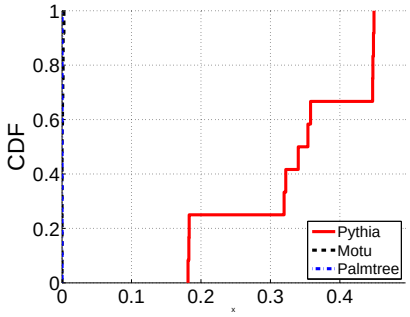
(c) Hit Ratio.



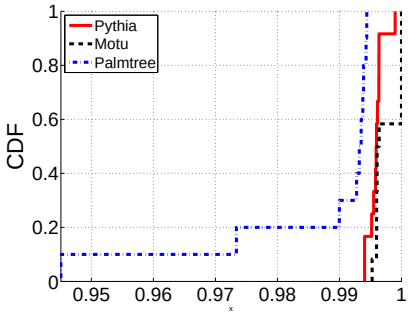
(d) Mismatch Ratio.



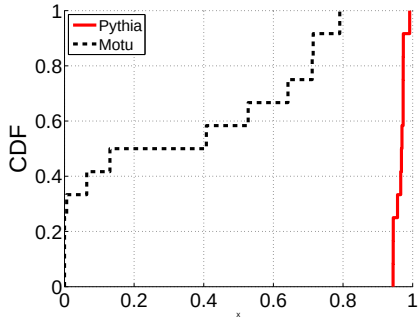
(e) Positive Hit Ratio.



(f) Negative Hit Ratio.



(g) Positive Predictive Value.



(h) Negative Predictive Value.

Figure 4.1: Performance metric distributions over the vantage points.

Positive and Negative Predictive Value. Compared to the other techniques, Pythia showed a similar positive predictive value and a much higher negative predictive value. Accordingly, when Pythia declares a pair of addresses as in alias, its level of trustworthiness is comparable to the other techniques. At the same time, when it declares a pair of IPs as not in alias, its level of trustworthiness is three times higher than Motu[6].

To deepen the comparative evaluation of Motu and Pythia, we also performed a per-pair analysis. To this end, we first aggregated the data collected by all the vantage points. In this process, we did not observe conflicts among the decisions taken by the same technique from different vantage points.

Tab. 4.3 reports the breakdown of the pairs on the decisions taken by Motu and Pythia. Thanks to the information provided by the reference dataset, we split the decision set of each technique in correct and wrong decisions. No decision refers to the pairs declared as not-classifiable by the technique. Globally, correct, wrong and no decisions account for 10.6%, 0.2% and 89.2% for Motu and 48.5%, 1.0% and 50.5% for Pythia, respectively. The subset of pairs classified by both the techniques represent 10.8% of the total in the dataset: when the techniques judged the same pair, they always took the same (mostly correct) decision with very few exceptions. On the other hand, both the techniques were not able to classify more than a half of the total pairs (50.5%). Interestingly, while Motu is not able to provide any additional information about the pairs not classified by Pythia, the latter showed a significant marginal utility when compared to Motu. Indeed, Pythia was able to classify 46.7% of all the pairs not classified by Motu, taking the correct decision in 97.9% of these cases.

This result suggests that for the subset of pairs classifiable by both the techniques, Motu and Pythia are essentially equivalent. However, Pythia was able to take correct decisions on a wide set of pairs not classifiable by Motu.

4.2.4 Summary and discussion

Alias resolution techniques represent an invaluable tool for measuring the network. They are commonly exploited to reconstruct the topology of the Internet but as we will shortly describe in the next sections, they also proved extraordinary helpful to uncover new limitations in Internet path tracing. In this section, we contribute to the state of the art

[6]Palmtree does not perform dealiasing.

Table 4.3: Pythia versus Motu: pair breakdown on classification (%)

		MOTU			
		Correct Decision	Wrong Decision	No Decision	
PYTHIA	Correct Decision	10.6	0	37.9	→ **48.5**
	Wrong Decision	0.001	0.2	0.8	→ **1.0**
	No Decision	0.007	0	50.5	→ **50.5**
		↓	↓	↓	
		10.6	**0.2**	**89.2**	

in the alias resolution field by designing, implementing and evaluating Pythia, an innovative active probing technique based on probe packets equipped with the prespecified TS option. The idea behind Pythia is diluting the great heterogeneity of devices characterizing the Internet in well-defined and easy to recognize categories of routers: for each category, researchers should identify the most accurate and reliable technique (or set of techniques) in order to address the alias resolution problem at the Internet scale. As first step according to this new research direction, we identified a first category of routers, i.e. any-interface stamping routers. Then, we designed our technique to purposely solve alias resolution for this specific category: the experimental evaluation demonstrates how Pythia is able to reach over this category of routers performance higher than state of the art techniques being able to classify with similar or higher accuracy many more addresses by also injecting into the network a lower amount of measurement traffic.

4.3 New limitations in Internet paths tracing

In this section, we describe the new experimentally observed limitations affecting Internet path tracing. Our analysis is supported by the adoption of Pythia and other state of the art alias resolution techniques.

4.3.1 Overestimation of equal-cost paths

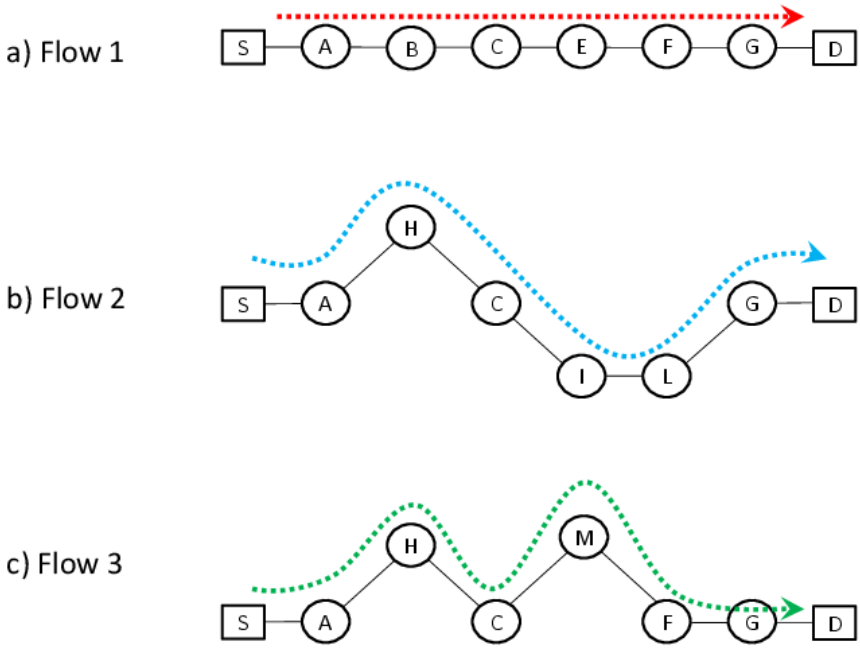
In this section, we describe how Traceroute may induce one to overestimate the number of equal cost paths towards the destination when each address is interpreted as a different

router.

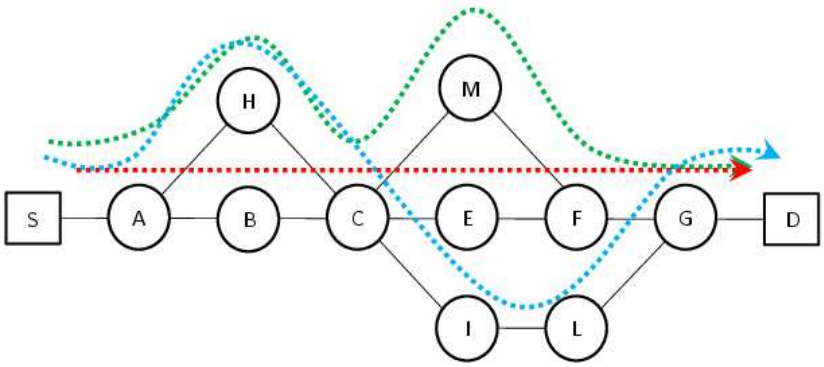
As explained in Chapter 2, recent achievements in Internet path tracing allow users to trace all the equal-cost paths towards the network destination [65]. This goal is achieved by injecting into the network TTL-limited probe packets part of different traffic flows. The ratio behind this approach is that routers performing load balancing on a per-flow basis should forward packets of different flows along different paths. Note that routers may perform load balancing also on a per-packet basis: each packet is forwarded along a different path possibly according to a round-robin strategy. In [65], authors observed that about 39% of the investigated source-destination pairs in the Internet traverse a load balancer: multiple equal cost paths toward the destination seems a very common phenomenon.

The presence of a load balancers along the path is normally recognized when probe packets crafted with the same initial TTL value solicit ICMP Time Exceeded messages from different IP addresses. For example, Fig. 4.2 reports the three traces discovered with the probe packets part of three different traffic flows: packets part of flow 1 discover the sequence of addresses A B C E F G D; probes part of the flow 2 discovers A H C I L G D; probes part of flow 3 discovers A H C M F G D. In this stylised Traceroute's output, we observe that there are two addresses at the second hop (B and H) and at the fifth hop (L and F), and three addresses at four hop (I E M). Commonly this output is interpreted as follows: the routers (addresses) followed by multiple addresses perform load balancing (per-flow load balancing in this sample scenario) across multiple equal-cost paths towards the destination, i.e. the routers owning A and C are load balancers.

As extensively described later in this chapter, there are several reasons for which the actual path in terms of sequence of routers traversed by the traffic might have completely different properties. More precisely, we experimentally observed that different addresses appearing at the same hop may be part of the same router. In the sample scenario of Fig. 4.2, the addresses B and H, L and F, and I, E and M may represent different interfaces of the same routers: in this case, (i) differently from what Traceroute suggests, the real router-level path is unique and (ii) there are no load balancers at all along the path towards the destination. By applying alias resolution, we can resolve IPs to routers: this allows us to differentiate if addresses belong or not to the same router.



(a) IP-level paths reported by Traceroute with probe packets part of different flows.



(b) Multiple equal cost paths inferred towards the destination.

Figure 4.2: Traceroute reports multiple equal cost paths towards the destination. We discovered that addresses reported at the same hop often belong to the same router: H and B, M E and I as well as F and L may belong to the same router: in this case, despite what Traceroute suggests, there is only one router-level path towards the destination.

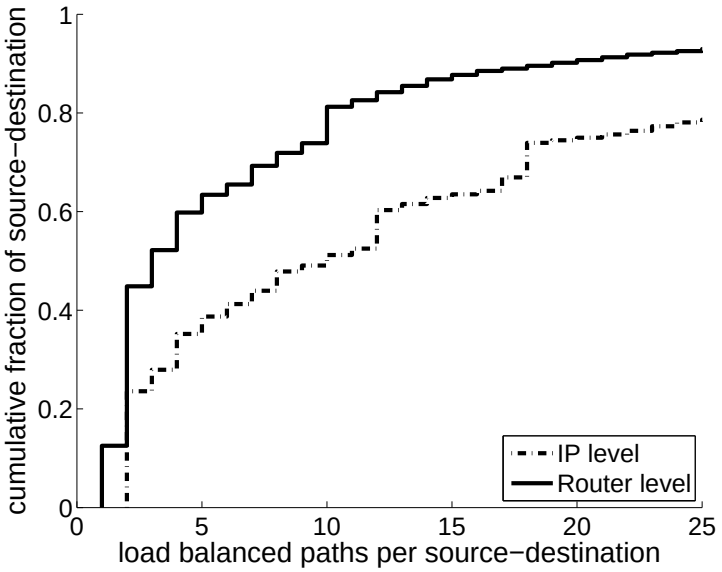


Figure 4.3: Equal-cost paths per source-destination before and after the alias resolution process.

Experimental analysis

In order to assess the existence of the phenomenon, we considered Internet routes traced with Traceroute and exposing multiple equal-cost paths: we launched Paris Traceroute with the Multi-path Discovery Algorithm injecting probe packets as part of different flows [65] from 14 PlanetLab vantage points toward about $2.3K$ destinations in distinct /12 prefixes randomly selected among those addresses responsive to ICMP Echo Request probe packets according to the PREDICT project[7]. We focused on Internet routes exposing multiple equal-cost paths $(8,066)$ and applied Pythia and other state of the art alias resolution techniques [128, 125, 10, 123] on the addresses appearing in the same trace at the same hop. Our goal is to discover if multiple equal cost paths still persist at the router level. We consider two addresses as in alias only when declared as such by the majority of the adopted techniques.

Our results suggest that Paris Traceroute– a Traceroute variant specifically designed to accurately capture equal-cost paths [64] – in fact drastically overestimates the prevalence of load balanced paths. Fig. 4.3 reports the distribution of equal-cost paths at the IP level as suggested by Traceroute, and at the router-level after having applied the alias resolution: the figure shows that the number of paths between a source-destination pair decreased by 45% on average as we went from Paris Traceroute's IP level paths to router

[7]IP Address Hitlist, PREDICT USC-LANDER http://www.isi.edu/ant/traces/dataset_list.html

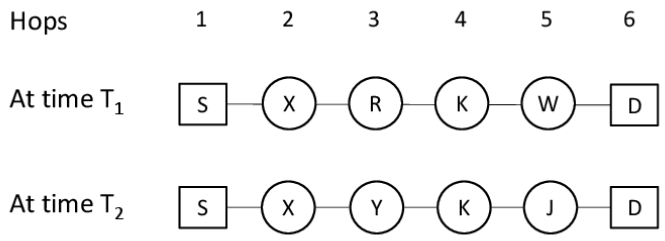


Figure 4.4: Two Traceroute traces collected towards the same destination at two different points in time reveals different addresses at the 3rd and 5th hops suggesting that the path has changed. We discovered that often differing addresses belong to the same router: if R and Y, and W and J belong to the same routers, despite what Traceroute suggests, the path towards the destination is perfectly unchanged.

level paths. In fact, 14% of traces identified by Traceroute as having multiple equal-cost paths turned out to actually be a unique router-level path. In all these cases, Traceroute provides misleading information and impact the properties of the traced paths. Our results suggest that the phenomenon of router-level load balancing is potentially overestimated: multiple equal cost paths towards the destination might often be just an artefact of the adopted Internet path tracing technique. In our data, we observed a reduction of the inferred per-flow load balancers by 25% (from 4,500 to 3,376 units) and by 10% of per-packet load balancers (from 486 to 436 units).

Later in this chapter, we describe the possible causes explaining this result.

4.3.2 Inference of false path changes

In this section, we describe how using Traceroute to monitor an Internet path over time may also suggest a false path change. According to Traceroute, the traffic sent to the destination is forwarded along a new different path compared to what previously observed while in fact the path is perfectly unchanged: the traffic sent towards the destination traverses exactly the same unchanged sequence of routers.

Let us imagine that we are interested in monitoring over time the path between a source S and a destination D. Fig. 4.4 reports two sample traces obtained by launching

Traceroute from S to D in two different instant of time. We want to understand if the path has changed or not. Similar studies are performed for instance to investigate the stability of the routes in the Internet [18, 19] or to troubleshoot the network [100] or build and maintain an overlay network [34]. By comparing the two collected traces, we discover two different addresses at the 3rd (R and Y) and 5th (W and J) hops. Due to these *differing hops*, the common interpretation of a similar outcome is that the path has changed since each address in a trace reported by Traceroute is interpreted as a different router. However, even if Traceroute suggests a routing change, there are several reasons for which the actual router-level path towards the destination may be perfectly unchanged as we describe later in this chapter: the IP-level view provided by Traceroute may change even if the actual router-level path has not changed at all. One possibility to assess similar circumstances is to exploit alias resolution: by applying alias resolution on the differing hops (R and Y, and W and J in Fig. 4.4) it is possible to check if these addresses are part or not of the same router and thus if the path has really changed or not.

Experimental analysis

In order to investigate this phenomenon, we analyzed the Traceroute traces made publicly available by the iPlane project [15]. We inspected the paths collected in two consecutive days related to the same source-destination pairs (about 721,780 pairs).

We adopted a conservative approach by considering only those source-destinations for which the paths collected in the consecutive days (i) differ for at least one hop (a differing hop), (ii) are unchanged in terms of number hops and (iii) do not contain unresponsive routers. This process generates a final set of 38,844 source-destination pairs. Finally, we adopted again multiple alias resolution technique [125, 128, 10, 123] to check if the addresses appearing as differing hops belong or not to the same router. Globally, we investigated 18,943 pairs of addresses but only 14,225 have been successfully judged as in alias or not due to unresponsive addresses: 37.1% of these pairs of addresses were classified as part of the same router . The impact of the alias resolution on the observed routing changes is depicted in Fig. 4.5 reporting the number of differing hops per source-destination pair at the IP-level as suggested by Traceroute and at the router-level after having applied the alias resolution. Experimental results demonstrate that the phenomenon exists and it is not uncommon: surprisingly, 32.1% of the paths changed at the IP-level turned out to be unchanged at the router-level (due to unresponsive addresses,

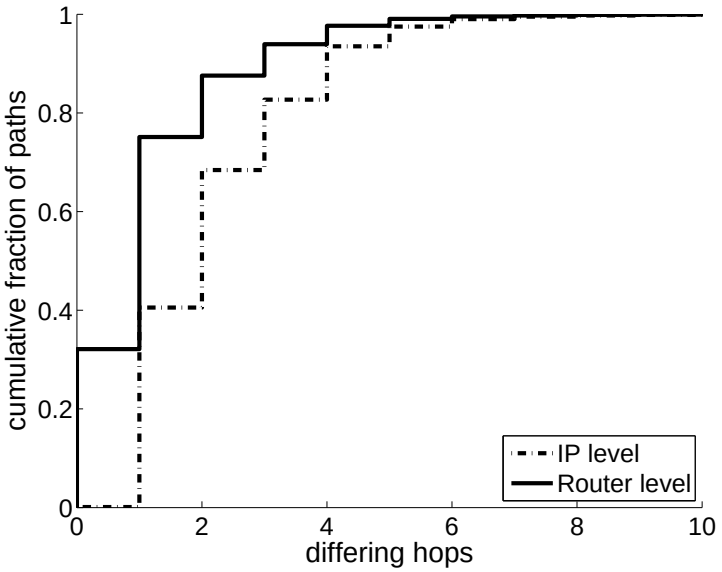


Figure 4.5: Differing hops per path before and after the alias resolution process - conversely to what Traceroute suggests, 32.4% of the paths are actually unchanged.

we estimated the lower bound of the actual magnitude of the phenomenon). We observed unchanged paths containing up to 6 differing hops, but, in most of the cases, unchanged paths differed at a single IP hop: globally, we observed that about 54% of the paths with a single differing hop are actually unchanged. Our results suggest that when a routing change is determined by a unique differing hop there is a significant probability that the path is actually unchanged. Note that in our analysis, we focused on the Internet routes where the IP-level path has changed but the number of hops has not. Authors in [19] discovered that this phenomenon is very common: for 62% of the path changes observed in the range of a large-scale experimental campaign, the number of hops towards the destination remained perfectly the same.

In conclusion, when Traceroute is used to monitor Internet paths over time, researchers and network operators may wrongly conclude that the path towards the destination has changed.

4.3.3 Causes overview

In this section, we discuss several causes explaining the experimentally observed phenomena described above.

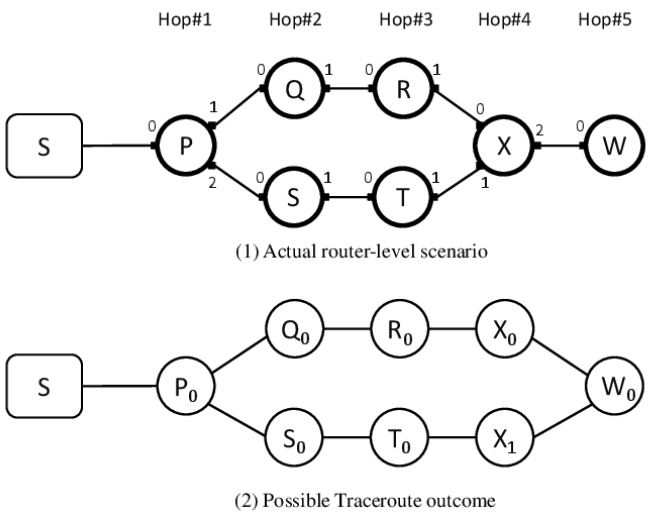


Figure 4.6: Multiple equal cost paths merging at a given hop cause Traceroute to report at this hop different addresses being actually part of the same router. The router X in this scenario exposes two different addresses to the Traceroute originator.

Merging of multiple equal-cost paths

Two different IP addresses reported by Traceroute at the same hop and part of the same router might be explained in case of multiple equal-cost paths merging at the same router. Fig. 4.6 depicts a sample scenario also acknowledged in [65]: the router P is balancing the traffic on two different equal cost router-level paths. On the route towards the network destination, these two paths reach the same router (X) through two different interfaces. When solicited multiple times by Traceroute, the router X exposes different interfaces according to which equal-cost path is followed by the TTL-limited probe packet.

This scenario does not explain the overestimation of equal-cost paths because it does imply the presence of multiple paths. Furthermore, it does not provide useful information to explain the inference of false routing changes since different IP addresses in different traces should still identify different routers.

Multiple links between routers

Researchers in [65] also marginally described another scenario potentially explaining at least part of the observed inaccuracy. Indeed, an approach used by network operators to increase the link capacity between two routers is to connect multiple cables between them, i.e. the traffic sent by the first router to the second router may arrive at different interfaces

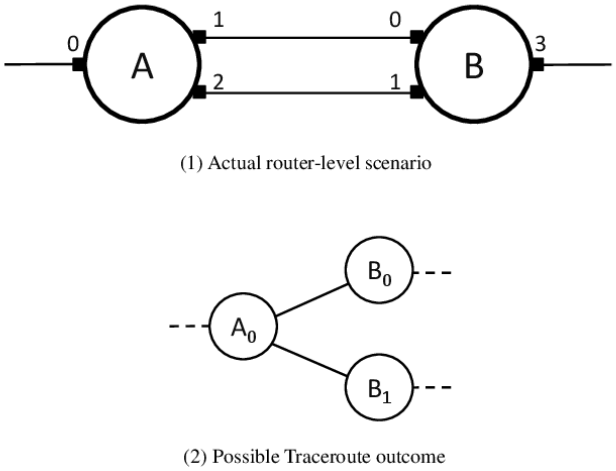


Figure 4.7: Multiple links between consecutive routers may cause Traceroute to report multiple addresses of the same router. In this scenario, the router B exposes different addresses to the Traceroute originator.

of the second router (see Fig. 4.7). If there are multiple cables connecting the router A to the router B and A performs load balancing, different probe packets with the same initial TTL value may reach the router B crossing each time a different link among the ones connecting A to B and, each time, a different incoming interface of B is reported in the Traceroute's output. This scenario allows us to explain the overestimation of equal-cost paths: more precisely, although Traceroute correctly discovers that packets flow across different links and reach different interfaces, the number of equal cost router-level paths towards the destination is overestimated with an impact on the inferred properties of the path such as the robustness of the communication to network failures involving routers.

On the other hand, this scenario does not explain the inference of false path changes. Indeed, according to this explanation, Traceroute reports all the incoming interfaces of the routers encountered by the traffic along the path towards the destination. If at a certain point in time Traceroute reports different addresses, the only possible explanation according to this scenario is that the path has changed. However, thanks to the alias resolution we observed that, although the IP-level view provided by Traceroute has changed, the sequence of traversed routers towards the destination may be perfectly unchanged (this happened for about 32.1% of the monitored paths). Hence, not all the observed cases can be explained by relying exclusively on this scenario.

RFC-compliant routers

Differently from the cases reported above, we describe in this section a scenario able to explain *all* the experimentally observed inaccuracy.

A possible source of inaccuracy causing both the overestimation of equal-cost paths and the inference of false path changes is represented by those routers implementing the RFC1812 [49]: we refer to these routers as *RFC-compliant*. These routers expose to Traceroute the outgoing interface (i.e. the interface selected by the router to send the ICMP Time Exceeded packet back to the Traceroute originator) rather than the incoming interface (i.e. the interface on which the Traceroute probe packet has arrived). Although generally considered not very common, RFC-compliant routers exist: the Cisco 3660 routers running IOS 12.0(7)XK1 are RFC-compliant routers [61]. The presence of these routers in the network may easily explain both the observed phenomena as we discuss in the following.

Overestimation of equal-cost paths. The overestimation of equal-cost paths can be easily explained in case of RFC-compliant routers performing load balancing on the reverse path. Indeed, RFC-compliant routers may expose different interfaces when solicited multiple times with the TTL-limited probe packets issued by Traceroute because there are multiple equal cost paths between the RFC-compliant router and the Traceroute originator: according to which specific reverse path is selected each time by the router, a different IP address is reported by Traceroute.

Discovering the presence of multiple equal-cost paths between an intermediate router and the Traceroute originator may be interesting but it also seems totally not related to the goal of this type of measurements: researchers and network operators use Traceroute to infer the properties of the forward path followed by the traffic sent towards the destination but in this way they may observe how the intermediate routers decide to forward the traffic to reach the Traceroute originator. Also, note that due to routing asymmetry prevalent in the Internet [21], discovering how an intermediate router decides to foward the traffic to the Traceroute originator may not reveal any useful information about the reverse path from the destination: indeed, the intermediate router may be not part of the reverse path from the destination.

False routing change inference. RFC-compliant routers explain also the inference of false routing changes. Traceroute traces collected at the time T_1 and T_2 expose different

addresses at a given hop. A possible explanation of this outcome is the presence at the differring hop of an RFC-compliant router: at the time T_1, this router selects a first given interface to issue the ICMP Time Exceeded reply to the Traceroute originator. However, at the time T_2, probably due to a routing change, the same router uses a different interface to reply to the Traceroute originator. Essentially, the traffic issued from the Traceroute originator to the destination is always forwarded along the same forward path (links, interfaces and routers). However, the IP-level view provided by Traceroute may still differ if an intermediate RFC-compliant router uses different interfaces to reach the Traceroute originator.

Again, in this case, Traceroute reports information about how the intermediate routers forward the traffic to reach the Traceroute originator and not about the actual forward router-level path followed by the traffic sent towards the destination.

4.4 Final remarks

In this chapter, by using alias resolution, we experimentally demonstrated the existence of two additional limitations of the most widely adopted path tracing technique: Traceroute may induce one to overestimate the number of equal cost paths towards the destination and may also suggest non-existing changes of the network routing. Essentially, Traceroute reports interfaces not routers and we observed that this IP-level view of the path can be a poor representation of the real router-level path followed by the traffic sent to the destination.

Our analysis is enabled by alias resolution, the set of techniques aiming at identifying those addresses owned by the same network device. To advance the state of the art in this research field, we proposed Pythia, a novel alias resolution technique able to reach performance higher than other state of the art techniques when solving the alias resolution problem for a specific category of routers, i.e. the any-interface stamping routers. Thanks to the adoption of Pythia and other alias resolution techniques, we observed how over the set of considered paths, about 14% of the Internet routes that expose multiple equal-cost paths at the IP-level turned out be a unique router-level path. In addition, we observed that 32.1% of the paths that has changed at the IP-level according to Traceroute turned out be perfectly unchanged at the router-level. We discussed the potential causes of such an inaccuracy identifying RFC-compliant routers exposing the outgoing interface to

Traceroute as a possible explanation for both the observed phenomena.

As a general lesson, independently from which specific cause or combination of causes determine the inaccuracy described above, we learned that the common assumption that each address in the Traceroute's output identifies a different router does not hold any more. A first immediate impact of our analysis is that Traceroute cannot be considered any more as a reliable stand-alone approach to investigate properties of the Internet such as the path stability (e.g. the overal likelihood that a path remains unchanged over a long period of time [18]). Indeed, by relying on Traceroute, researchers observed that Internet paths are stable 96% of the time, but experience short-lived instability periods [19, 18]. These short-lived instabilities may represent just an artefact of Traceroute: the paths may only have changed at the IP-level but may be perfectly unchanged at the router-level. Accordingly, Internet paths may be much more stable than what currently assessed in literature: the limitations of the adopted measurement tools must be very carefully considered.

Chapter 5

An architecture for Internet path tracing on demand

Researchers trace Internet paths for several reasons: to infer the topology of the Internet [8, 85], to detect, investigate and resolve issues like outages, reachability problems and network failures [100, 101, 25, 24], to geolocate network resources and services [40, 41, 42], to monitor and predict over time the Internet paths and their performance [29, 19, 18, 15], and so on.

Researchers, network operators and systems often exploits path tracing techniques to explore specific Internet routes. However, two open challenges must be faced when reaching this goal as already discussed in Section 2.3: (i) the lack of a large number of well-distributed vantage points from which issuing path tracing measurements and (ii) the inability of current methodologies and techniques to identify the vantage point to use and the destination to target in order to explore an Internet routes with a priori known characteristics (e.g. the sequence of ASes to traverse). These two challenges cause a suboptimal access to the routes of interest and potentially impact the ability of researchers to collect all the information required for a deep understanding of the phenomenon under investigation.

In this chapter, we propose our contribution toward the solution of these challenges. More precisely, after having highlighted the previous efforts related to the proposed approach, we describe a general architecture of a system designed to (i) integrate multiple experimental testbeds such that researchers and network operators can have access to their vantage points through a unique common interface that masks the heterogeneity of the interfaces and constraints characterizing each testbed and (ii) satisfy complex user

queries describing the characteristics of the paths of interest rather than the vantage points to use or the destinations to target. Finally, we present PANDA (PAth oN-DemAnd), a first working implementation of the proposed architecture, and provide the results of the experimental analysis we conducted to highlight its performance.

5.1 Related work

In this section, we describe the works and projects related to the system detailed in this chapter.

5.1.1 Experimental testbeds

Over time, several experimental testbeds have been developed and deployed to allow researchers, network operators and systems to perform measurements over the Internet. We report here a brief description of some of these testbeds.

Planetlab [80]. PlanetLab is a global overlay network for developing and accessing broad-coverage network services. It currently consists of 1,182 nodes at 580 sites located at academic institutions and industrial research labs mostly covering Europe and North America but with vantage points located all over the world.[1] Thanks to the virtualization, researchers share the computational resources of the same vantage point and can perform in an isolated environment the network experiments. Users usually access the vantage points through *ssh* connections. Compared to other testbeds, very few constraints are imposed by Planetlab: for instance, not all the probe packets are allowed (e.g. IGMP packet probing is not permitted [85]).

RIPE ATLAS [6]. This project comprises a well-distributed set of USB-powered embedded devices that allow users to issue low-volume non-intrusive measurements such as Ping, Traceroute and DNS lookup. The project includes a credits-based system: users hosting probes gain credits over time. Once acquired, these credits can be spent to perform measurements. Each measurement has a specific cost: for instance, tracing an Internet path with Traceroute has a cost of 60 credits. Since a user gains once per day 21,600 credits for each hosted probe, a user hosting a single probe can trace no more than

[1] March 2014.

360 Internet paths per day. Once the credits are depleted no more measurements can be issued. Measurements can be orchestrated by using either a web-based interfaces or a Restful API. Currently, the project comprises about 4,987 active probes primarily located in the RIPE NCC service region (Europe).

Looking glasses. Many ASes expose web-based form that allow users to issue Traceroute from machines deployed inside their network. In order to trace Internet paths from this set of vantage points one must face a strong heterogeneity in terms of (a.) web technologies – these pages may strongly differ as well as the mechanism required to request to trace a given path (HTTP post, HTTP get, name of required parameters, etc.); (b.) the mechanism adopted to report the obtained results – they may also strongly in terms of underlying tools used to perform the measurements. The latter factor requires the ability to correctly identify and parser the relevant information generated by different looking glasses after the measurement is performed. Since this type of websites strongly suffer from network attacks, they are typically very carefully monitored: any minimally suspicious user behavior causes the user IP to be banned. Hence, while there are no credits when using these systems, these vantage points are constrained in that Traceroute measurements must be issued with a very conservative rate.

Other testbeds. Several other testbeds regularly performing Traceroute measurements from multiple vantage points exist. MobiPerf [83] is an open source application for measuring network performance on mobile platforms. Tracing Internet paths is part of the measurements allowed by this application. Users can download and install the application on their smartphone to regularly monitor the performance of the network they are connected to. The data are anonymously collected and made available for scientific purpose. Archipelago proposed by CAIDA [2] is an active measurement infrastructure regularly tracing Internet paths. The collected data is used to reconstruct the topology of the Internet made publicly available to the research community. DIMES [16] relies on an altruistic approach in which users join the platform for the betterment of science. Researchers managing the platform can issue Traceroute measurements from the client downloaded by the volunteers. Dasu [129] is built on top of a BitTorrent client and exploits the success of this application to reach a large amount of users. Despite the previously cited projects, Dasu provides visibility on broadband performance from residential

networks as well as the opportunity to trace Internet paths originated from these networks. Similar projects are Hobbit [82] and Bismark [81] that characterize the broadband performance by respectively exploiting a client- and a home router-based approach. The previously cited projects issue Traceroute measurements to investigate Internet routes for disparate purposes. However, to the best of our knowledge, while the collected datasets are made publicly available to the research community, most of these projects do not allow researchers or network operators to autonomously plan and perform experimental campaigns by using the vantage points managed by these projects.

The list reported above is far from being exhaustive but it provides a clear idea that several experimental testbeds exist with multiple vantage points located in completely different locations (universities, residential networks, ISP backbones, industry labs, smartphones) providing complementary point of views on the Internet routes. Unfortunately, due to the highly heterogeneous interfaces adopted by these projects, we noticed that researchers tend to use always the same testbed, thus not taking advantage of a more complete visibility of the Internet routes that one could reach by jointly using the vantage points made available by multiple experimental testbeds. Aggregating the vantage points of these projects and making them available to the research community through a unique simple interface is one of the goal of the architecture we propose.

5.1.2 Path prediction techniques

Several scientific works proposed techniques and methodologies to predict Internet paths. These works try to answer the question: *what is the path followed by the traffic sent by a given node A toward a given node B?* These works try to predict the path followed by the traffic originated by an arbitrary node toward an arbitrary destination. One approach to this end is to exploit already collected paths: in this case, by assuming no routing changes, the predicted paths can be approximated with the results obtained the last time the path has been monitored. If no previous results are available, other works stitched trace segments extracted from previously traced paths. Such an approach has been proposed in [28], slightly modified in [30] and currently exploited in the iPlane project [15]. Other approaches, like iPlane Nano [130], models the inter-domain routing and exploits BGP-related information to predict the paths with a coarse-grained granularity. Finally, recently proposed approaches exploited the likelihood of path changes to improve the path

prediction accuracy [29].

Compared to these approaches, we aim at answering a completely different question: *given a set of vantage points, what are the (vantage point, destination) pairs to consider such that the traced paths satisfy specific a priori known characteristics?* The input and output of the proposed system are therefore completely different: the input of our system is not a pair of network nodes but a partially defined Internet path. In addition, while the cited works aim at providing in output predicted yet accurate paths, the users of the proposed system expect to receive real traced paths.

5.2 An architecture for tracing Internet paths

In this section, we highlight the general architecture of a system designed to integrate multiple testbeds and to satisfy complex queries.

5.2.1 Desirable features

The following features are desirable for the system under investigation.

- *Monitoring on-demand:* the system should be able to trace Internet paths nearly in real-time, i.e. the system should trace the path as soon as possible after the user queries is received such that the user can investigate the current routing in the Internet.

- *Highly expressive queries:* the system should accept user queries specified in a highly expressive and intuitive language. A user query describes the characteristics of the paths of interest: starting from these characteristics, the system identifies the vantage points to use and the destinations to target. As for the characteristics of the paths of interest, we consider in this thesis the traversed ASes. We left as future work queries at a finer-granularity specifying the PoPs, routers or geographic locations to traverse.

- *Campaign-oriented:* the system should be designed to support measurement campaigns providing an interface through which the user is allowed to submit a query and monitor its evolution over time.

- *Public access to the collected measurements:* the system should store all the performed measurements and make the collected dataset periodically publicly available to the research community as done by similar projects [15, 2].

- *Comprehensive:* the system should be able to trace Internet paths from as many experimental testbeds providing public access to vantage points able to issue Traceroute measurement as possible.

- *Completeness:* if there exists at least one path in the Internet with the characteristics required by the user, the system should be able to satisfy the user query. However, since the system forcedly manages a limited amount of vantage points, the system should be able to satisfy any user query that can be satisfied by at least one path originated by the managed vantage points according to the current routing in the Internet.

- *Scalability:* the time required to manage a user query should be independent of the current load of the system.

A system designed with these goals in mind allows users to (i) asynchronously submit complex queries to the system; (ii) monitor the evolution of the query in the system; (iii) retrieve the results when ready; and (iv) observe Internet paths as dictated by the current routing in the Internet. This path monitoring on demand service combined with the ability to satisfy complex user queries and the integration of multiple experimental testbeds make the system a promising tool for researchers and network operators investigating complex phenomena on the Internet.

5.2.2 Architecture overview

In this section, we describe a general architecture designed to solve the issues described in the previous sections. Fig. 5.1 reports the general architecture consisting of several functional blocks that interact to satisfy a user query. The main modules of the architecture are in charge of (i) interacting with the user (System Front End); (ii) permanently storing the collected results (Data Manager); interacting with the experimental testbeds (Testbed Drivers); and (iv) performing all the steps required to satisfy the user query (Query Manager and Prediction Engines).

We detail in the following the functionality of each block of the proposed architecture.

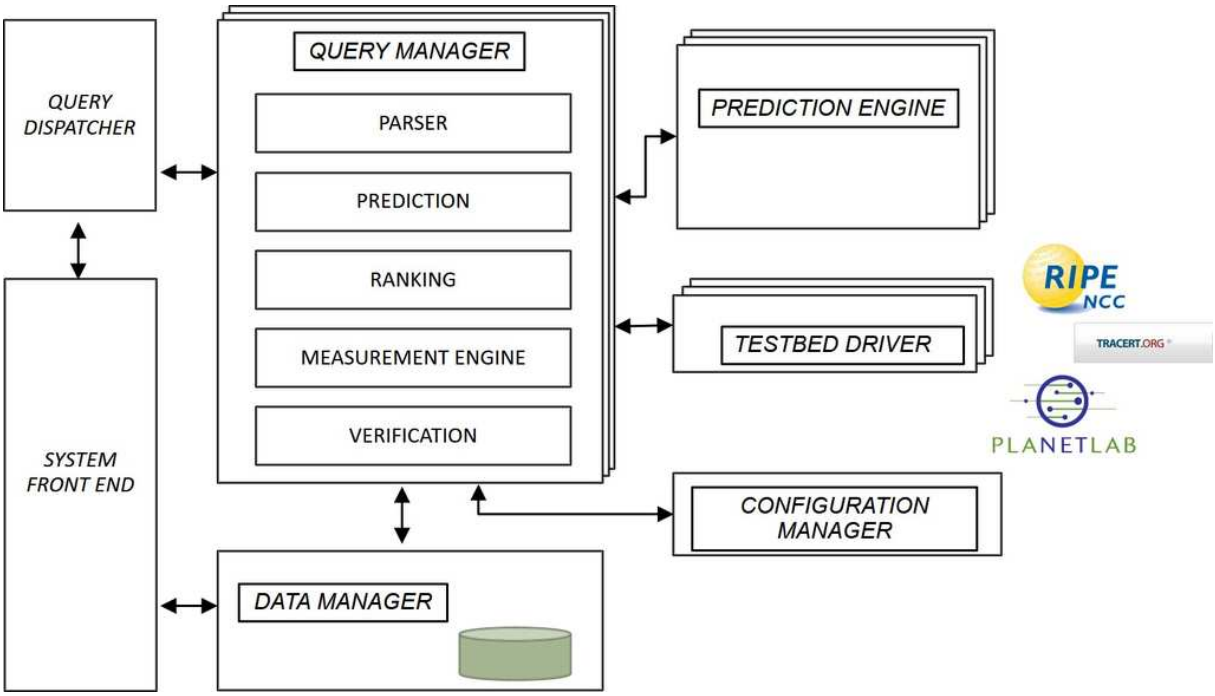


Figure 5.1: General path tracing system architecture.

System Front End

The System Front End is the module of the system in charge of managing the interaction with the user. The users interact with the System Front End in order to (i) submit their queries; (ii) monitor the status of the submitted queries and (iii) retrieve the collected results.

By also learning from similar projects [6], the interaction between the system and the user should follow the following steps:

1. The user makes use of a generic text editor to create a query according to a specific highly expressive and intuitive path-query language.

2. The user exploits the mechanism provided by the System Front End to submit the query. The user is not requested to wait for the completion of the path tracing operations since one of the desired features of the system is working in asynchronous mode according to a measurement-oriented paradigm: the system immediately returns to the user a numeric identifier that uniquely identifies the submitted query. Thanks to this identifier, the user is allowed to monitor the evolution of the query inside the system and retrieve the collected results when ready.

3. The user monitors the status of the query through the mechanisms provided by the System Front End. Essentially, by polling the system, the user can easily understand if the system has collected the required results or if any problem occurred during the execution of the query.

4. Once the results are available, the user can retrieve the results by interacting with the System Front End.

To reach this goal, the System Front End needs to interact with the other modules of the architecture.

Query Dispatcher

The Query Dispatcher is in charge of scheduling the incoming queries to the currently allocated Query Managers, the modules of the architecture taking care of all the steps required to satisfy a given user query. The Query Dispatcher can also allocate new Query Managers to manage the high volume of the incoming requests considering also the available resources of the system. Different Query Manager allocation strategies and query scheduling policies can be implemented in the Query Dispatcher: different approaches may have an impact on the scalability of the system defined as the ability of the system to satisfy a given user query as independently of the current load of the system as possible.

Testbed Drivers

The Testbed Drivers allow the system to exploit the vantage points made available by different experimental testbeds (see Sec. 5.1). A driver is in charge of interacting with a specific testbed. These modules mask the heterogeneity of the interfaces of different testbeds and operate a first homogenization when collecting the results. More precisely, these modules should implement the following functionalities:

- Provide the identifiers of those ASes where the vantage points of the experimental testbed are located.

- Provide the identifiers of the vantage points located within a given AS.

- Issue a Traceroute measurement from a given vantage point toward a given network destination.

- Retrieve and convert the traced Internet path.

The interaction with the Testbed Drivers can potentially follow exactly the same paradigm imagined for the interaction between the user and the System Front End. Furthermore, not necessarily these modules need to be part of the system: they can also be imagined as external services exploited by the system to perform the required measurements.

Prediction Engines

The Prediction Engine receives as input the queries defined by the user. The output of the Prediction Engine is a set of measurements potentially able to satisfy the user query. More specifically, for each query, the Prediction Engine returns multiple records including (i) the experimental testbed to use; (ii) the identifier of the vantage point from which the measurement must be issued; (iii) the range of addresses in which the destination must be selected, (iv) the predicted AS-level path, (v) the originating and destination AS, and (vi) a confidence score indicating the likelihood that the predicted path corresponds to the actual path.

The system is designed to interact with different Prediction Engines and combine the results such that only the most likely-to-match measurements are performed by the system.

Data Manager

This module of the system is in charge of permanently storing all the relevant information related to each query submitted to the system. This information includes but it is not limited to: (i) the original user query, (ii) its identifier, (iii) the submission and completion time, (iv) any errors encountered in the system while serving the query, (v) the current status of the query, (vi) aggregated information about the performed measurements. Finally, the Data Manager stores the information about all the results collected by the system to satisfy the user request comprising (vii) the exploited testbeds, vantage points and targeted destinations, (viii) the collected IP level traces and their AS-level conversion, (ix) the AS-level path predicted by the adopted prediction engine, (x) the originating and targeted ASes, (xi) some indications related to if the collected paths satisfy the user query. Periodically, the Data Manager dumps, compresses and stores all the collected results to share the measurements with the research community.

All the data managed by the Data Manager are made available to the other modules of the system. More specifically, the Query Manager interacts with the Data Manager to update the information related to the managed query (like its status); the System Front End interacts with the Data Manager when the user is requesting an update about the current status of a given query or to retrieve the results collected by the system.

Configuration Manager

The Configuration Manager is in charge of managing the current configuration of the system since many modules rely on a set of global and local parameters. For instance, a sample parameter of the system is the length of time the system needs to consider before making the collected data publicly available or the file system directory where storing this information. Since the system is designed to support large-scale measurement campaigns, it is reasonable to imagine that the system must be always on-line: a static configuration approach would forcedly imply the rebooting of the system potentially affecting the ongoing measurements. Conversely, by acting through the Configuration Manager, the administrator can easily modify the parameters of the system without impacting the current measurements.

Query Manager

The Query Manager is the core of the system since it manages all the steps required to satisfy a given user query. Each Query Manager is spawned by the Query Dispatcher and sequentially performs the steps described as follows.

1. *Parser:* during this phase, the user query is parsed and syntactic and semantic checks are applied to assess the correctness of the query according to the rules of the path-query language. The relevant information are extracted and stored to be easily accessed in the following steps by also interacting with the Data Manager.

2. *Prediction:* the user query manipulated during the previous step is provided as input to the available Prediction Engines. In this way, the Query Manager collects a set of measurements likely-to-match the user query. Aggregated statistics related to the available measurements are stored by interacting with the Data Manager. In addition, since the Prediction Engines typically predict AS-level paths, they do not indicate a specific destination to target but usually an entire range of destination.

Hence, the Query Manager also identifies during this phase for each likely-to-match measurement which specific destination must be targeted.

3. *Ranking:* at this point, the Query Manager may potentially have multiple measurements to issue. The system must decides the subset of measurements to actually perform in order to satisfy the user query. During this phase, different strategies could be applied.

4. *Measuring:* once the subset of measurements to issue have been selected, the Query Manager interacts with the Testbed Drivers in order to trace the identified Internet paths. In order to reduce as much as possible the serving time of each query, these measurements should be issued in parallel whenever the exploited experimental testbeds allow a similar workload. The Internet paths traced during this step are also enriched with additional information. For instance, the traced paths are usually sequences of IP addresses: these addresses must be associated to the owning AS in order to verify if the user query has been satisfied or not. All the traced paths and their details are permanently stored by interacting with the Data Manager.

5. *Verification:* the final step is to verify if the collected paths satisfy the user query.

Any severe error occurred during the phases described above causes the interruption of the execution of the query and all the relevant information about the encountered errors is permanently stored for debugging purposes. After the Verification step, the final results are stored by the Data Manager and can be accessed by the user by interacting with the System Front End.

Challenges

The general architecture described above implies the resolution of the following challenges.

Selecting responsive destinations. As described above, the Prediction Engine typically predicts paths at the AS-level. These engines may not provide a clear indication about which specific network destination must be targeted but report an entire range of addresses (a network prefix). In order to monitor an Internet path, however, the system must decide which specific destination to target. This decision is non-trivial: if the selected destination is non-active, i.e. it is not associated to any active network interface,

the traced path reported by Traceroute will appear as filtered showing multiple anonymous routers at the end of the trace. As already described in Chapter 2, tracing Internet paths toward unresponsive destinations is largely time-consuming: since it is not possible to distinguish between few routers along the path discarding the Traceroute probe packets from an unresponsive destination, Traceroute further explores the path with additional probe packets and increased TTL values. In case of unresponsive destination, these additional steps won't lead to the discovery of any additional router along the path causing the expiration of the internal timeout considered by Traceroute to receive a response from a router. Furthermore, tracing a complete path is of the utmost importance to verify if the user query has been satisfied. For these reasons, it is important that the system identifies a responsive destination in the range of addresses suggested by the Prediction Engine.

Note that devices in the Internet may reply or not depending on the type of adopted probe packets: packets equipped with different transport protocols proved to have different responsiveness levels [95, 131]. Hence, changing the transport protocol of the probe packets issued by Traceroute may potentially help to solicit a reply from the targeted destination [76]. Unfortunately, not all the experimental testbeds allow the users to select the transport protocol to adopt.

Selecting sources and destinations. Methodologies and techniques proposed in literature allow to predict the path from a given source toward a given destination [15, 29, 30]. However, the Prediction Engines part of the system must perform a completely different task: they must identify the vantage point and the destination connected by a network path that satisfies the characteristics requested by the user as dictated by the current routing. Accordingly, there is not a direct way to adopt the methodologies proposed in literature to solve the query submitted by the user.

Ranking the measurements. The system may identify multiple likely-to-match measurements to perform in order to satisfy the user query. When only a subset of paths are requested by the user, the system must identify which measurements to issue. In this process, several factors might be considered:

- *Convenience of the vantage point:* this metric allows the system to consider the cost of issuing a new measurements from the specific experimental testbed. For example,

for the Planetlab testbed, a measurement is costless so the convenience is very high. For a RIPE ATLAS probe, instead, a new measurement has a great cost due to the very limited number of measurements that can be issued every day from this testbed. Looking glasses are also very convenient but no as convenient as Planetlab nodes since looking glasses can be used with a very limited rate.

- *Confidence score of the prediction:* it is the confidence provided by the Prediction Engine representing the probability that the user query would be satisfied by issuing this measurement.

- *Vantage point availability:* it captures the instability of some vantage points that are not always available over time. For example, it may happen that Planetlab nodes are temporarily not available being not reachable through the Internet. Similarly, all the RIPE ATLAS probes become unavailable when the credits are depleted.

- *Prediction benefit:* it provides an indication of the benefit in terms of future improved accuracy the Prediction Engines would gain if the measurement would be performed.

By properly combining the factors above, one can compute an overall score for each measurement, rank them and issue the subset of measurements with the highest score. Modelling and combining these factors is non-trivial.

5.3 Internet path tracing with PANDA

As first implementation of the general architecture described in the previous section, we developed PANDA– PAth tracing oN DemAnd. The goal of PANDA is to demonstrate the potential of the proposed path tracing on-demand paradigm.

5.3.1 The path-query language

One of the desirable feature of the system is to provide to the user a simple and effective way to express complex queries. By analysing the literature, we noticed that several scientific works are interested in tracing Internet paths originated by, traversing or reaching one or multiple ASes. Accordingly, PANDA comprises a path-query language that allows the users to easily specify the ASes that must be contained by the paths to trace.

```
SELECT {PATH-QUERY-REGEX}
WITHIN {DEADLINE}
LIMIT  {N}
WHERE  {DISCRIMINATOR}
```

Figure 5.2: Basic structure of a query according to the proposed path-query language.

Any query submitted to PANDA must be compliant with the scheme reported in Fig. 5.2. A user query is a SQL-like statement where only the first line is mandatory.

The keyword *select* is followed by a regular expression[2] through which the user can easily specify the AS numbers the paths of interest must traverse or simply contain. The keyword *within* in the second (optional) line allows the user to specify a deadline for the system: any result obtained after this deadline is meaningless for the user. The deadline is a length of time relative to the query submission time: accepted values must be compliant with a format comprising a number followed by *s* - seconds, *m* - minutes, *h* - hours or *d* - days. By default, the deadline is infinite. The third line starting with the keyword *limit* is optional and allows the user to specify the number of paths the system must returns. This is an upper bound such that no more than these number of paths are traced by the system. By the default, the system is requested to return a single traced path. Finally, the last optional line starting with the keyword *where* is intended to provide a way for the user to influence the decision related to which subset of likely-to-match measurements the system should perform. Accepted values are max-different-(source or destination)-(pref24, pref16 or pref8).

```
SELECT [" "]20965[" "]1239[" "]
WITHIN 5m
LIMIT 10
WHERE max-different-destination-pref24
```

Figure 5.3: PANDA: A first example of query.

A sample query is reported in Fig. 5.3: the user is requesting the system to trace no more than 10 different paths (LIMIT 10), crossing the link between the GEANT (AS20965) and SPRINT (AS1293) networks (SELECT [" "]20965[" "]1239[" "]); results must be made available within the next 5 minutes (WITHIN 5m) and the traced paths must reach network destinations contained in different /24 subnets (WHERE max-different-destination-pref24). Other sample queries are reported and explained in Fig. 5.4.

[2]Regular expressions represent a special text string for describing a search pattern.

5.3.2 PANDA's modules

PANDA includes all the modules of the general architecture implementing the functionalities already described as well as new modules. In the following, we briefly provide few implementation details.

System Front End. The user interacts with PANDA through an XMLRPC interface. Three methods are made available by the System Front End: (i) *submit* invoked to submit a query; (ii) *status* invoked to check the status of a previously submitted query and (iii) *results* invoked to retrieve the results collected by PANDA to satisfy the user query. The submit method provides an identifier to the user, this identifier is a mandatory input for the other methods. The identifier is obtained by the System Front End by interacting with the Data Manager. Each new user query is inserted by the System Front End in a queue shared with the Query Dispatcher module.

Query Dispatcher. When resources are available, the Query Dispatcher extracts a query from the queue shared with the System Front End and spawns a new Query Manager thread that autonomously performs all the steps required to satisfy the user query.

Query Manager. Each Query Manager in the system is a thread performing all the steps described in the previous section. More precisely, the user query inside the system is a data structure that contains all the intermediate results achieved by the Query Manager while performing the steps. After each step, results as well as possible errors are permanently stored by the Data Manager. Severe errors cause the execution of the query to be interrupted.

```
a) SELECT ^137[" "]
b) SELECT [" "]137$
c) SELECT [" "]137[" "]
d) SELECT (^|[" "])137[" "]29065($|[" "])
r) SELECT (^|[" "])137([" "]|.+[" "])29065($|[" "])
f) SELECT (^|[" "])137([" "]|.+[" "])29065$
```

Figure 5.4: Sample queries – (a) Any path originated from the GARR network (AS137); (b) Any path targeting the GARR network; (c) Any path traversing the GARR network; (d) Any path traversing the direct link between the GARR and GEANT networks; (e) Any path traversing the GARR and, at a certain point, also the GEANT networks; (f) Any path traversing the GARR network when targeting destinations inside the GEANT network.

Data Manager. The Data Manager module exposes to the other modules of the system a set of interfaces that allows to permanently store all the relevant information about the submitted queries. Data are stored by interacting with a MySQL database.

Testbed Drivers. PANDA interacts with external Testbed Drivers developed and maintained by the Networked System Laboratory of the University of Southern California (USC) headed by professor Ramesh Govindan. These Testbed Drivers provide an API to issue Traceroute measurements from (i) the RIPE ATLAS testbed comprising almost 5 thousands vantage points and (ii) about 500 different web-based looking glasses. Together these two testbeds cover 1,912 distinct ASes: this coverage may appear limited if compared with the total number of existing active ASes in the Internet (more than 40,000). Note, however, that Planetlab, one of the most used testbeds in literature, provides vantage points located in no more than 600 different sites.

Configuration Manager. The Configuration Manager is invoked by most of the modules of the system to retrieve the current value of the configuration parameters of interest. This module periodically reads a configuration file to load the values for the global and local parameters. The administrator can easily modify the current configuration of the system by overwriting the values of the parameters contained in the configuration file. This approach does not require the system to be rebooted when the configuration has changed.

Prediction Engines. PANDA exploits the inter-domain routing information to identify the (vantage point, destination) pairs likely to satisfy the user query. We provide more details about the process in the next section.

Beside the modules originally contained in the general architecture, PANDA comprises the following two additional modules:

Responsive destination lookup module. One of the challenge for the system is to trace Internet paths toward responsive destinations. To face this problem, PANDA exploits the hitlist of responsive addresses made available by the PREDICT project [98]: this project periodically probes with ICMP Echo Request packets a significant portion

of addresses in the Internet and reports for each subnet /24 the most responsive address along with a responsiveness score. By using this information, PANDA can select responsive destinations to target when available. Since by default Traceroute issues UDP probe packets, we recomputed the responsiveness score of the addresses by also considering UDP probe packets: whenever it is possible, PANDA selects destinations responsive to both UDP and ICMP Echo Request probe packets. When no responsive destinations are available in the range of addresses returned by the Prediction Engine, the destination is randomly selected. Within PANDA, this module interacts exclusively with the Query Manager. However, its functionality may be helpful in any measurement system requiring addresses responsive to UDP or ICMP Echo Request probe packets and could be easily extended toward a stand-alone service.

IP-to-AS mapping module. Since the user query is related to the set of ASes to traverse, the IP-level view provided by Traceroute must be converted to the AS-level to verify if the query has been satisfied. To this end, the IP-to-AS mapping module exposes API to (i) map a given IP address to the owning AS or (ii) map an entire Traceroute trace. Internally, this module invokes an online public service made available by the Team Cymru [99]. The module also exploits an internal cache such that only the not already mapped addresses are submitted to the online service in order to moderate as much as possible the probing overhead.

BGP data crawling module. As described later in this section, PANDA exploits the information related to the inter-domain routing to identify the (vantage point, destination) pairs likely to satisfy the user query. For this reason, PANDA comprises a module in charge of downloading from the Internet all the available data related to the current inter-domain routing. Several international projects like RouteViews from the University of Oregon [5] and RIPE RIS [6] have routers (BGP feed) located in specific locations of the Internet that receives BGP routes advertised by peering ASes (BGP feeders) as their best routes toward network prefixes. A snapshot of all the active received BGP routes is made periodically available by the two cited projects (data from 23 BGP feeds made available by RouteViews every 4 hours, and from 17 BGP feeds made available by RIPE RIS every 8 hours). This module automatically downloads and converts this compressed data to be then easily managed by the Prediction Engine.

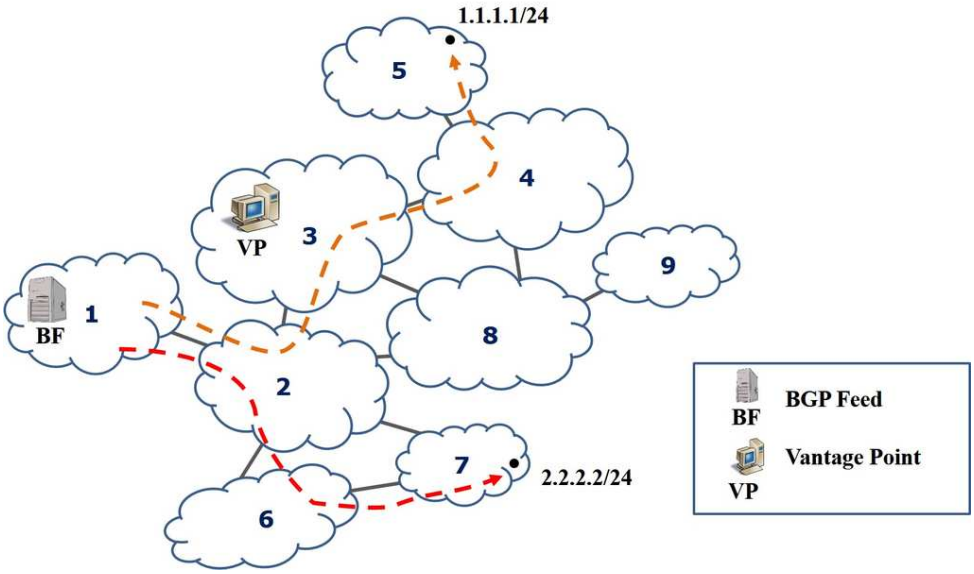


Figure 5.5: BGP-based prediction engine: An inter-domain routing sample scenario.

5.3.3 A BGP-based prediction engine

In this section, we describe the Prediction Engine exploited by PANDA to satisfy the user query. As already described above, this process is based on the information related to the inter-domain routing.

Currently, BGP is the standard inter-domain routing protocol adopted in the Internet. Through this protocol, ASes exchange their best routes to reach network prefixes [132]. Each BGP route contains several information including (i) the destination network prefix and (ii) an AS path indicating the sequence of ASes that the traffic would traverse to reach that network prefix if the traffic would be forwarded to the advertising AS.

Let us consider the Fig. 5.5 as a reference: the BGP feed receives the best routes advertised by AS1. Let us suppose that AS1 advertises the route towards the network prefix 1.1.1.1/24 containing the AS path AS1 AS2 AS3 AS4 AS5. This information can be interpreted as follows: the 1.1.1.1/24 is part of the AS5 (this is also the basic scheme adopted to map IP addresses to the owning AS [99]); the best route to reach these addresses for the AS4 is to directly forward the traffic to the AS5, for AS3 is to forward the traffic to AS4, for AS2 is to forward the traffic to AS3, finally, AS1 has selected as best route toward 1.1.1.1/24 to forward the traffic to AS2. Hence, according to the information owned by the BGP feed, the traffic sent by AS1 toward addresses in 1.1.1.1/24 should flow across the AS2 AS3 AS4 before reaching AS5. Let us imagine that the BGP feed

is also able to issue Traceroute measurements. If the user is looking for paths crossing the AS2 or the link between the AS3 and AS4, thanks to the information provided by the BGP feed, we can select as vantage point the BGP feed itself and as destination one address in 1.1.1.1/24 in order to satisfy the user query.

Unfortunately, the BGP feeds from RouteViews and RIPE RIS cannot be used to issue Traceroute measurements and the vantage points made available by the experimental testbeds are not always co-located with the BGP feeds. In this case, not all the information provided by the BGP feeds are helpful. Let us consider again Fig. 5.5 and let us imagine to have only one vantage point located in the AS3: this AS does not directly announce its best routes to the BGP feed. However, by inspecting the best routes advertised by the AS1 to the BGP feed, we can still infer at least part of the best routes of the AS3. For example, AS1 advertises as best route toward 1.1.1.1/24 the AS path AS1 AS2 AS3 AS4 AS5: we can easily state that the best route of AS3 toward 1.1.1.1/24 is AS3 AS4 AS5. Note that we might not be able to infer *all* the best routes of AS3: for instance, if the best routes of AS1 toward 2.2.2.2/24 is AS1 AS2 AS6 AS7, this does not provide us any information about the best route of AS3 toward 2.2.2.2/24.

Essentially, the BGP feeds provide useful information to predict the AS-level path taken by the traffic originated by the vantage points toward network prefixes. However, when the BGP feeds and the vantage points are not co-located, only a subset of best routes originated by the ASes hosting vantage points can be inferred. On the other hand, the lack of information might be mitigated by combining the routes captured by multiple BGP feeds.

To exploit the mechanism described above, the data periodically downloaded and uncompressed by the BGP data crawling module is manipulated in order to extract all the best routes for the ASes hosting the vantage points of the system. More precisely, each best route containing the AS-level path AS_1, .. ,AS_i, .., AS_n is inspected: if no ASes hosting vantage points are contained, the path is discarded. Otherwise, for each AS_j hosting vantage points, the subpath AS_j, AS_{j+1}, .. AS_n is extracted and stored as best route for the AS_j. At the end of this process, PANDA has a repository containing all the available best routes for the ASes hosting vantage points.

In order to satisfy the user query, the Prediction Engine must simply apply the regular expression on the best routes extracted from the BGP feeds: if a path AS_1, .. AS_i, .. AS_n towards the network prefix *a.b.c.d/xx* satisfy the regular expression, the Prediction

Engine extracts all the vantage points contained in AS_1 and returns to the Query Manager all the pairs (vantage point in AS_1, *a.b.c.d/xx*). This operation is made for any matching routes contained in the route repository. Successively, the Query Manager interacts with the Responsive destination lookup module to select one responsive IP for each network prefix returned by the Prediction Engine.

5.4 Experimental analysis

In this section, we reports the results of the evaluation of PANDA, a first implementation of the general path monitoring architecture reported in Sec. 5.2.

Several scientific works would have benefited from using a system like PANDA. By analysing the literature, we have extracted the following three categories of user queries of interest for the research community:

- Traverse the AS_x [85, 86, 73, 44, 87].
- Traverse the link between AS_x and AS_y [13, 11, 12, 22].
- Traverse the AS_x on the path to reach the AS_y [77].

In order to evaluate PANDA, we downloaded the best routes made available by Packet Clearing House (PCH) [7] and collected by BGP feeds located in ASes different from those hosting the BGP feeds exploited by PANDA[3]. From the extracted AS-level paths, we randomly generated 1,000 queries for each category. We used this test set of queries to evaluate PANDA and to compare it with other projects as described in the following. Note that the process described above allows us to generate queries related to *existing* Internet paths whereas totally randomly generated queries might correspond to paths not allowed by the current routing in the Internet.

5.4.1 BGP-derived best routes

In order to obtain a complete view of the AS-level paths followed by the traffic originated by the ASes hosting the vantage points, PANDA exploits BGP-derived information ob-

[3]Note that PANDA does not exploit the information provided by PCH since most of the BGP feeds of this project are located at IXP and, treated as peers, provide information about a very limited amount of Internet routes compared to the other projects [14].

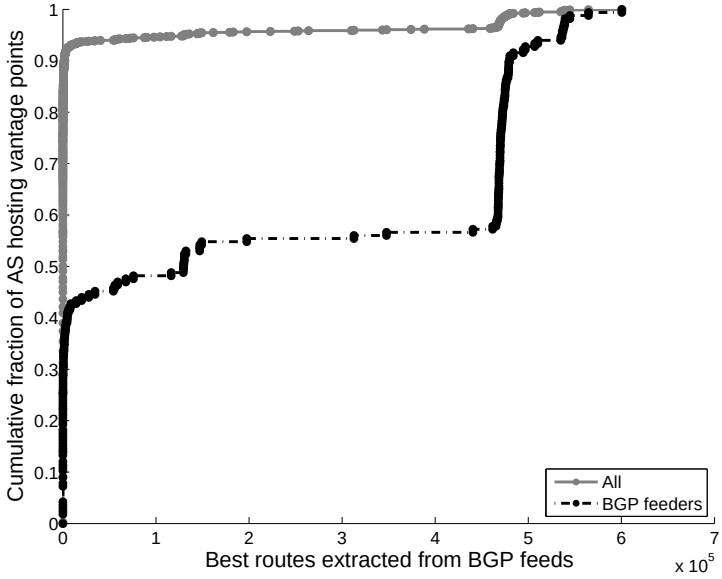


Figure 5.6: Best routes extracted from RouteViews and RIPE RIS for the ASes hosting vantage points made available by RIPE ATLAS and through looking glasses.

tained by periodically crawling the dataset made publicly available by RouteViews and RIPE RIS.

Fig. 5.6 reports the distribution of the number of best routes extracted from the BGP feeds related to the ASes hosting vantage points. A complete view of the AS-level paths originated by a given AS normally implies best routes toward more than 400,000 network prefixes (the exact number might be hard to estimate due to route aggregations [132]). The figure shows how BGP feeds provide an exhaustive view of the best routes for about 5% of the ASes hosting vantage points (94 distinct ASes). For most of the other ASes, BGP feeds provide less than 10,000 routes. The figure also shows the distribution of best routes observed for ASes hosting vantage points but also directly feeding BGP feeds. Over this set, the fraction of ASes providing an exhaustive view of their best routes represent about 45%: even when the BGP feed and the vantage point are co-located in the same AS, the BGP feed may provide only a partial view of the best routes of the AS. Several factors may explain this result. BGP feeds are known to be an incomplete source of information [14]: being not well distributed in the Internet, BGP feeds provide much more information about large transit ASes than ASes in the lower part of the AS hierarchy. In addition, some BGP feeds are treated as *peers* and not as *customers* as expected: since according to the valley-free policy [132] an AS does not forward routes learned from non-customers to

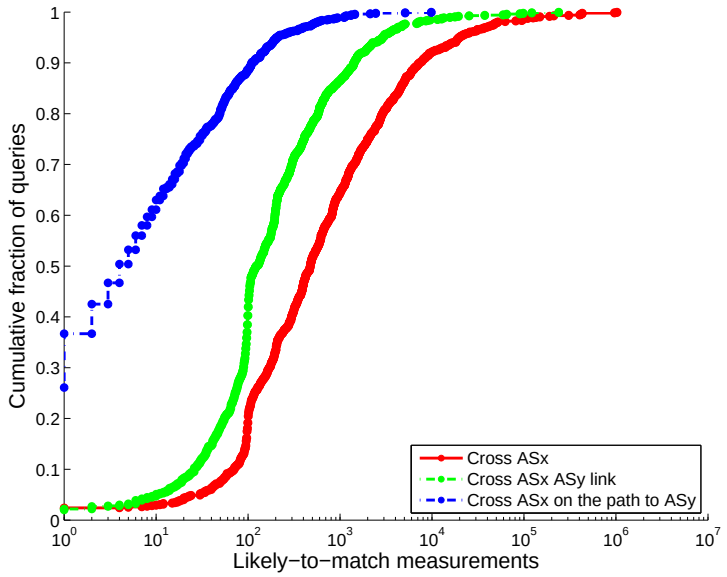


Figure 5.7: Likely-to-match measurements per query grouped by category.

non-customers, the feeders provide to the BGP feeds treated as peer information related only to the internal routes and the routes learned from their customers.

In conclusion, by using the information related to the inter-domain routing passively collected by international research projects, we can predict part of the paths followed by the traffic originated by the ASes hosting our vantage points (on average, about 42M of paths): although partial, gathering this knowledge does not actually require any active measurements on the network.

5.4.2 Query coverage

For a given regular expression, the Prediction Engine may identify several (vantage point, network prefix) pairs likely to satisfy the user query. Fig. 5.7 reports the distribution of the number of likely-to-match measurements for each query grouped by category. Queries like traversing a given AS or two consecutive ASes can be satisfied by a large number of likely-to-match measurements: for both categories, more than 99.98% of the queries have at least one likely-to-match measurement. The situation is slightly different for those queries requesting to traverse an AS on the path toward another AS. As a much more complex type of query to satisfy, the location of the used vantage points is critical. For about 26.1% of the queries in this category, the Prediction Engine was not able to provide any likely-to-match measurement.

On average, the Prediction Engine identifies respectively 8K, 1K and 66 likely-to-match measurements for the queries requesting to traverse an AS, two consecutive ASes or an AS on the path toward another AS.

These results show how the system typically has a large number of measurements to potentially satisfy a user query. Assuming no constraints in terms of resources and time, all these likely-to-match measurements could be issued in order to maximize the possibility to satisfy the user query. However, the used experimental testbeds impose constraints forcing the adoption of conservative strategies.

5.4.3 Query hit rate

In order to evaluate PANDA, we submitted to the system the queries of each category and computed the hit rate as the fraction of query satisfied. In this analysis, we adopted for the system a minimal and naive configuration: only one measurement is performed to satisfy a query. The measurement to perform is randomly selected among the set of likely-to-match measurements returned by the Prediction Engine.

To have a reference, we compared PANDA with the bruteforce approach implemented by the iPlane project [15]. iPlane exploits every day all the active Planetlab vantage points to trace millions of Internet paths (on average about 24 millions Traceroute traces are collected per day). The collected dataset is then made publicly available to the research community[4]. We compare PANDA to iPlane since a researcher might decide to exploit this daily collected dataset to investigate the phenomenon of interest. For this reason, we applied the same set of regular expressions also on the traces collected by iPlane to compute the corresponding hit rate.

A first important difference between PANDA and iPlane is that the iPlane dataset is refreshed daily. This implies the risk that the paths provided by iPlane are obsolete, i.e. not valid any more according to the current routing in the network. PANDA, instead, provides a monitoring on-demand service: any collected Traceroute trace reflects the current routing in the Internet. Furthermore, iPlane is agnostic with the respect to the goals and objectives of researchers and network operators. As consequence, while invaluable for longitudinal studies on the routing and Internet topology, a bruteforce approach like the one implemented in iPlane may be largely inefficient: indeed, this approach is strongly time- and network resources-consuming and it is also potentially

[4]iPlane project. http://iplane.cs.washington.edu/

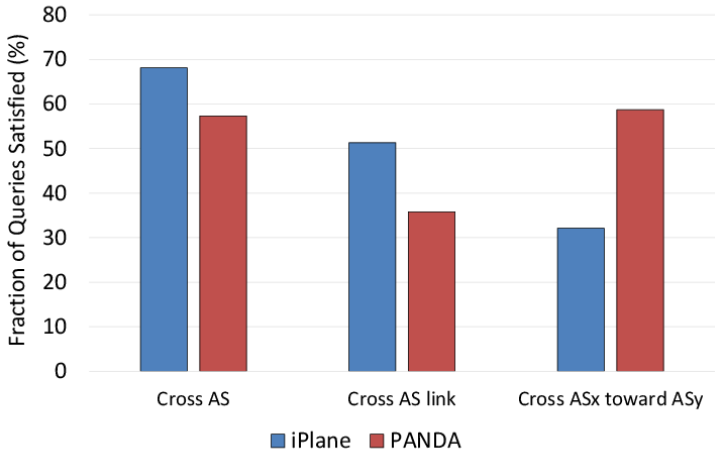


Figure 5.8: Fraction of queries satisfied by (i) PANDA and (ii) iPlane project (a bruteforce approach).

unable to provide the actual information researchers and network operators are looking for because, for instance, part of the requested paths simply cannot be explored by using the set of managed vantage points.

Fig. 5.8 reports the fraction of satisfied queries grouped by category. Surprisingly, even if PANDA performs only about 3K measurements against the 24M measurements performed by iPlane, the fraction of satisfied queries is similar or higher than the one reached by the bruteforce approach. By performing only one measurement for each submitted query, PANDA reaches a hit rate of 57.3%, 35.8% and 58.7% respectively for the queries requesting to traverse a given AS, a given AS-level link or an AS on the path toward another AS whereas iPlane reaches an hit rate of 68%, 51% and 32%.

Note that this result is not determined by vantage points used by PANDA and co-located with the PCH BGP feeds from whose information the regular expressions are generated: none of the vantage points used by PANDA and iPlane are co-located with these BGP feeds. The reason at the basis of this result, instead, seems related to the amount and location of the used vantage points. iPlane uninterruptedly exploits all the available active vantage points (171 on average) coming from a unique experimental testbed (i.e. Planetlab) and, even if a bruteforce approach is applied, not all the user queries have been satisfied. PANDA, instead, makes parsimoniously use of almost 5K vantage points coming from multiple testbeds (RIPE ATLAS and looking glasses) reaching a similar or a higher hit rate than iPlane. We can conclude that the network friendly

combination of a large amount of well-distributed vantage points and a prediction-based mechanism to identify which vantage point to use and destination to target provides good performance comparable to the one reached by employing a bruteforce approach from the vantage points of a single experimental testbed.

Finally, part of the likely-to-match measurements issued by PANDA failed, i.e. they provided Internet paths not satisfying the characteristics defined requested by the user. Several factors may explain this result: (a.) Traceroute may be filtered along the path, for instance, before traversing the requested AS (b.) control plane (BGP) and the data plane (Traceroute) may diverge [62]; (c.) the IP-to-AS mapping might not be perfectly accurate due to third-party addresses [102], anonymous routers [52], unmapped hops, Internet exchange points [13], multi-origin AS prefixes [133] or sibling ASes [134]; also (d.) traffic engineering policies like hot- and cold-potato routing cause ASes to possibly have multiple AS-level paths towards the same network prefix [135]. Issuing more measurements for each user query may represent a first possibility to increase the hit rate of the system.

More in general, the proposed campaign-oriented system performing path monitoring on-demand where measurements are orchestrated starting from the real objectives of researchers and network operators may also represent an effective and efficient alternative solution to the credit-based systems adopted to save and control the managed network resources, like the one implemented in the RIPE ATLAS system.

5.4.4 Query service time

PANDA offers a monitoring on-demand service. While such a paradigm appears particularly effective to investigate ongoing phenomena in the Internet, the system should provide results to the users in a reduced amount of time. Fig. 5.9 reports the distribution of the *query service time* defined as the length of time between the submission of the user query and the instant of time in which the results are made available to the user. First of all, the query service time is independent of the category of queries. On average, the system requires 110 seconds to collect and made available the collected results, a length of time reasonable for the investigation of most of the aspects of interest except for transient events that represent, however, a category of phenomena well-known to be very hard to assess.

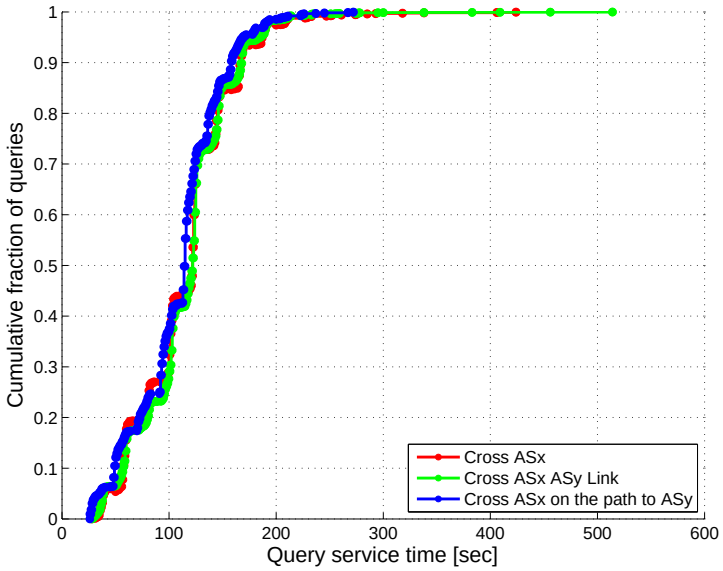


Figure 5.9: Time required by PANDA to satisfy a user query.

5.5 Final remarks

In this chapter, we proposed a general architecture able to perform path tracing on demand and designed to support researchers and network operators aiming at tracing Internet paths charactered by a priori known properties. By analysing the literature, we observed how several scientific works would have benefited from a similar system by submitting queries requesting the system to trace path traversing a given AS [85, 86, 73, 44, 87], a link between two ASes [13, 11, 12, 22] or an AS on the path toward another AS [77]. To this end, the architecture solves two important challenges such as (i) aggregating under a unique interface the vantage points made available by different experimental testbeds, thus masking the great heterogeneity of the interfaces exposed by these testbeds and (ii) identifying the vantage point to use and the destination to target in order to the trace the path of interest for the user.

As a first implementation of the proposed architecture, we designed, implemented and evaluated PANDA. PANDA aggregates the vantage points made available by RIPE ATLAS and about 500 looking glasses. Users interact with PANDA through an XMLRCP interface to submit queries to the system and monitor their status over time. Queries are defined according to a specifically designed path query language. PANDA exploits BGP-derived information to identify vantage points and destinations likely to satisfy the user query. The experimental evaluation showed that PANDA is able to satisfy a fraction

of user queries that is similar or higher than the one reached by a bruteforce approach that use all the available vantage points of a given experimental testbed to trace millions of paths: PANDA reaches this result by tracing an amount of paths that represent less than 0.017% of the paths traced by the bruteforce approach.

As a first measurement project that systematically benefits from PANDA, we are currently modifying the MERLIN platform [126, 85, 86] to exploit the path tracing on-demand service offered by PANDA: MERLIN aims at reverse engineering the router-level topology of a given AS by using an IGMP recursive probing approach triggered by a preliminary Traceroute-based experimental campaign. This preliminary campaign is now entirely delegated to PANDA.

5.6 Future directions

We describe in the following the directions we plan to investigate in future to extend PANDA.

Security. By providing public API, the security is one of the most important aspect to take into account. PANDA must employ advanced mechanisms to (i) defend the system itself from network attacks (ii) defend experimental testbeds from abuse (iii) defend the networks and devices in the Internet from being attacked. These three goals are equally important. Advanced authorization mechanisms are required to identify any user of the system (currently, PANDA adopts a basic HTTP authorization headers). Rate limiting constraints should be applied to limit the query submission rate by also employing admission control mechanisms: if no resources are available no new queries should be accepted. Similarly, rate limiting constraints should be applied when (i) using the same vantage points of a given experimental testbeds and (ii) when targeting the same network device or subnet. Finally, the system should apply specific strategies to ban users in case of misbehaviours.

Aggregating private and public testbeds. The proposed system offers path monitoring on-demand and it is able to satisfy user queries by taking advantage of a large number of vantage points to perform only very limited number of focused measurements. This approach allows users to save both time and network resources. One future direction of this project is to aggregate the vantage points made available by multiple experimental testbeds encouraging also the participation of those research projects releasing public

Traceroute-based dataset but not providing access to the managed vantage points.

Additional Prediction Engines. An important open challenge for the system is to obtain information about all the paths followed by the traffic originated by an AS hosting a vantage point. Results showed that BGP data provides an exhaustive view of the best routes only for a limited number of ASes. Accordingly, an important future direction is to design additional prediction engines to improve the performance of the system. For instance, one possibility is to take into account previously collected traces and the stability of the Internet routes. If an Internet route proved to be very stable and satisfies the user query, we might decide to trace again this route since there is a good chance that the user query will be actually satisfied. Another possibility is also to explore and adapt the path stitching approach proposed in [28].

Toward historical queries. Finally, another future direction is to extend the system to accept also historical queries containing regular expressions to apply on traces collected in specific time range: essentially, users ask the system to return paths satisfying the regular expression that have been measured at certain point in the past. A similar mechanism would make large Traceroute-derived datasets searchable potentially enabling advanced a posteriori investigation of the impact on the network routing of large-scale events such as network failures and power outages [101].

Chapter 6

Conclusions

The Internet is the largest distributed system ever built by the human kind supporting applications used by more than 2 billions of users worldwide [1]. Monitoring the Internet is essential to guarantee high operational standards and a positive evolution of this infrastructure. However, this operation is particularly challenging mainly due to the radically distributed ownership of its constituent parts: indeed, the Internet consists of independent networks, the Autonomous Systems (AS), that exchange traffic based on agreements dictated by local economic and technological constraints. The competition and collaboration among these independent networks are at the basis of the evolution of the Internet and the general lack of visibility and centralized control over the infrastructure makes the Internet a critical yet largely opaque communication system. In this scenario, the research community has developed measurement methodologies and techniques to gather the fundamental knowledge required for a deep understanding of how this highly complex and dynamic ecosystem actually operates.

Among the aspects of interest, tracing Internet paths (i.e. routers and links traversed by the network traffic on the path towards a network destination) has attracted large interest from the research community with several applications. For instance, tracing Internet paths proved helpful when managing and troubleshooting the network [22, 23, 100, 101, 25, 24], monitoring or predicting Internet paths and their performance [29, 15], building efficient overlay networks [34], reverse engineering the network infrastructure [8, 9, 85], etc. Unfortunately, despite the numerous applications, several severe limitations affect path tracing techniques causing the information obtained about the paths under investigation to be potentially incomplete or inaccurate with an impact on the applications relying on Internet path tracing. For instance, state of the art implementations of path

tracing techniques may provide inaccurate topological information about the network causing a distortion of the inferred network topology.

Furthermore, researchers, network operators and systems relying on path tracing techniques are often interested in exploring specific Internet routes about which they know at least some characteristics such as part of the traversed ASes. Unfortunately, the ability to explore these routes critically depends on the number and locations of the vantage points used to perform the path tracing measurements and on the ability to identify the vantage point to use and the destination to target in order to explore a path with a priori known characteristics: both these factors represent open challenges.

In this thesis, we developed solutions (i.e. methodologies, techniques and a system) for advancing the state of the art in the research field of Internet path tracing. We started presenting in Chapter 1, an overview of the importance of tracing Internet paths. Then, we discussed in Chapter 2 the path tracing techniques proposed in literature as well as the known limitations affecting them. We also detailed the open challenges we identified when using Internet path tracing to explore paths with a priori known characteristics. Starting from the next chapter, we detailed the contributions of the thesis: in Chapter 3, we discussed the results of our research activities conducted to investigate, mitigate and resolve important largely-ignored limitations in Internet path tracing. In Chapter 4, we discussed two additional experimentally-observed limitations of the state of the art path tracing techniques. Finally, in Chapter 5, we proposed a general architecture and a first implementation able to aggregate the vantage points of different experimental testbeds and to identify the measurement to issue in order to explore paths with a priori known characteristics. We briefly describe in the following the main contributions of the thesis.

As a first contribution, we proposed a set of innovative active probing techniques augmenting state of the art path tracing techniques with the specific goal to address unresolved limitations in Internet path tracing (Chapter 3). We developed several novel measurement techniques built on top of a particular measurement traffic composed by IP-options equipped probe packets: these probe packets have the potential of gathering additional invaluable information on the traversed paths.

More precisely, we designed and implemented a measurement multistep technique in order to identify and locate hidden routers in the traced paths (Section 3.2). Hidden routers represent a largely-ignored limitation of path tracing techniques having a great

impact on the inferred topological properties of the network. By using our technique, we were able to quantify the magnitude of this phenomenon: we detected hidden routers in at least 6% of the observed paths with the great majority of them located in the proximity of the targeted destination network supporting the idea that many of these devices are middleboxes (e.g. NATs, Firewalls, etc.). We also presented an active probing technique to identify third-party addresses in Traceroute traces (Section 3.3). Third-party addresses may induce one to conclude that the traffic sent towards the destination is flowing across an AS not actually traversed: very few researchers have investigated third-party addresses with conflicting conclusions by relying on limited heuristics methods or not easily to collect pre-acquired knowledge about the network topology. To the best of our knowledge, we are the first ones proposing an active probing technique able to classify the addresses reported by Traceroute as third-party addresses or not: according to our results, third-party addresses are very common affecting more than 90% of the analyzed paths. We also observed that third-party addresses affect about 17% of the AS-level links extracted from large scale Traceroute-based experimental campaign, thus representing an important source of potential distortion when path tracing is used to infer the AS-level connectivity as proposed in several scientific works [13, 11, 12, 62]. Furthermore, since the information provided by path tracing techniques about the intermediate delays experienced by the traffic sent towards the destination is potentially misleading (e.g. due to asymmetric routing), we developed an active probing technique able to accurately dissect the overall RTT of a traced path in different chunks related to specific portions of the network path (Section 3.4). We demonstrated the practical utility of this approach by isolating the contribution of (i) an home network and (ii) an entire Internet Service Provider to the overall RTT of an end-to-end communication. Finally, we demonstrated that path tracing solutions totally alternative to the ones universally adopted actually exist (Section 3.5). More precisely, we designed, implemented and evaluated three active probing methods able to solicit ICMP Parameter Problem messages instead of ICMP Time Exceeded messages from the routers located along the path: these methods inject into the network probe packets equipped with malformed IP options. The experimental evaluation demonstrated that routers in the Internet actually reply to this innovative type of probe packets: this alternative path tracing solution proved to be complementary to the classic TTL-based tracing solution providing additional information on the paths under investigation, although the proposed approach is not able to express its full poten-

tial mainly due to TCP/IP stack implementations not fully compliant with the Internet standard.

As a second contribution, we identified for the first time in literature two new additional limitations of the state of art path tracing technique (Chapter 4): we discovered that Traceroute may induce one to (i) overestimate the number of equal cost router-level paths towards the destination and to (ii) infer non-existing changes in the network routing. Surprisingly, about 32% of the analyzed paths that has changed according to Traceroute proved to be actually unchanged. In addition, about 14% of the paths exposing multiple equal cost paths towards the destination according to Traceroute turned out be a unique router-level path. Essentially, we demonstrated that the IP-level view of the path obtained by using state of the art implementations of Traceroute may be a biased representation of the actual router-level path followed by the traffic sent towards the destination. For this analysis, we used alias resolution techniques to identify those addresses owned by the same network device: among these techniques, we also employed Pythia, a novel active probing technique we presented in this thesis to solve the alias resolution problem for a specific category of routers on which we obtained promising results when compared to other techniques.

Finally, as third and last contribution of the thesis, we designed, implemented and evaluated PANDA, a first implementation of a general architecture we presented in Chapter 5, providing a path tracing on demand service that aggregates under a unique simple interface the vantage points made available by different experimental testbeds very rarely jointly used due to the great heterogeneity of the exposed interfaces. The proposed system is also able to resolve complex user queries requesting path tracing measurements by describing the characteristics of the Internet paths of interest: PANDA exploits a BGP-based prediction engine to identify the vantage point to use and the destination to target in order to satisfy the user request. The experimental evaluation demonstrated how PANDA is able to achieve similar or higher performance when satisfying queries suggested by the analysis of the scientific literature when compared to a bruteforce approach. This promising result is reached by monitoring few thousands of paths compared to the millions of paths traced by the bruteforce approach.

Bibliography

[1] Internet World Stats. http://www.internetworldstats.com/dsl.htm.

[2] Kimberly Claffy, Young Hyun, Ken Keys, Marina Fomenkov, and Dmitri Krioukov. Internet mapping: From art to science. In *Proceedings of the 2009 Cybersecurity Applications & Technology Conference for Homeland Security*, CATCH '09, pages 205–211, Washington, DC, USA, 2009. IEEE Computer Society.

[3] S. Hares and D. Katz. Administrative Domains and Routing Domains: A model for routing in the Internet. RFC 1136 (Informational), December 1989.

[4] Giuseppe Aceto, Alessio Botta, Walter de Donato, Pietro Marchetta, Antonio Pescapè, and Giorgio Ventre. Open source platforms for Internet monitoring and measurement. In *SITIS*, pages 563–570, 2012.

[5] Route Views. University of oregon route views project. http://www.routeviews.org/, 2000.

[6] NCC RIPE. Ripe atlas. https://atlas.ripe.net/, 2010.

[7] Packet Clearing House. BGP Feeds and IXP Directory. *https://prefix.pch.net/*.

[8] Benoit Donnet and Timur Friedman. Internet topology discovery: A survey. *IEEE Communications Surveys and Tutorials*, 9(1-4):56–69, 2007.

[9] Walter Willinger and Matthew Roughan. Internet topology research redux. *ACM SIGCOMM eBook: Recent Advances in Networking*, 2013.

[10] Ken Keys. Internet-scale IP alias resolution techniques. *SIGCOMM Comput. Commun. Rev.*, 40(1):50–55, January 2010.

[11] Yihua He, Georgos Siganos, Michalis Faloutsos, and Srikanth Krishnamurthy. Lord of the links: A framework for discovering missing links in the Internet topology. *IEEE/ACM Transactions on Networking*, 17(2):391–404, April 2009.

[12] Kai Chen, David R. Choffnes, Rahul Potharaju, Yan Chen, Fabian E. Bustamante, Dan Pei, and Yao Zhao. Where the sidewalk ends: Extending the Internet As graph using traceroutes from P2P users. In *Proceedings of the 5th International*

Conference on Emerging Networking Experiments and Technologies, CoNEXT '09, pages 217–228, New York, NY, USA, 2009. ACM.

[13] Brice Augustin, Balachander Krishnamurthy, and Walter Willinger. IXPs: Mapped? In *Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement Conference*, IMC '09, pages 336–349, New York, NY, USA, 2009. ACM.

[14] Enrico Gregori, Alessandro Improta, Luciano Lenzini, Lorenzo Rossi, and Luca Sani. On the incompleteness of the AS-level graph: A novel methodology for bgp route collector placement. In *Proceedings of the 2012 ACM Conference on Internet Measurement Conference*, IMC '12, pages 253–264, New York, NY, USA, 2012. ACM.

[15] Harsha V. Madhyastha, Tomas Isdal, Michael Piatek, Colin Dixon, Thomas Anderson, Arvind Krishnamurthy, and Arun Venkataramani. iplane: An information plane for distributed services. In *Proceedings of the 7th Symposium on Operating Systems Design and Implementation*, OSDI '06, pages 367–380, Berkeley, CA, USA, 2006. USENIX Association.

[16] Yuval Shavitt and Eran Shir. Dimes: Let the Internet measure itself. *SIGCOMM Comput. Commun. Rev.*, 35(5):71–74, October 2005.

[17] H. Kardes, M. Gunes, and T. Oz. Cheleby: A subnet-level Internet topology mapping system. In *Communication Systems and Networks (COMSNETS), 2012 Fourth International Conference on*, pages 1–10, Jan 2012.

[18] Vern Paxson. End-to-end routing behavior in the internet. *SIGCOMM Comput. Commun. Rev.*, 36(5):41–56, October 2006.

[19] Ítalo Cunha, Renata Teixeira, and Christophe Diot. Measuring and characterizing end-to-end route dynamics in the presence of load balancing. In *Passive and Active Measurement*, PAM'11, pages 235–244, Berlin, Heidelberg, 2011. Springer-Verlag.

[20] Yihua He, Michalis Faloutsos, Srikanth Krishnamurthy, and Bradley Huffaker. On routing asymmetry in the Internet. In *Global Telecommunications Conference, 2005. GLOBECOM'05. IEEE*, volume 2, pages 6–pp. IEEE, 2005.

[21] Yaron Schwartz, Yuval Shavitt, and Udi Weinsberg. On the diversity, stability and symmetry of end-to-end Internet routes. In *INFOCOM IEEE Conference on Computer Communications Workshops, 2010*, pages 1–6. IEEE, 2010.

[22] Neil Spring, Ratul Mahajan, and Thomas Anderson. The causes of path inflation. In *Proceedings of the 2003 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, SIGCOMM '03, pages 113–124, New York, NY, USA, 2003. ACM.

[23] Arpit Gupta, Matt Calder, Nick Feamster, Marshini Chetty, Enrico Calandro, and Ethan Katz-Bassett. Peering at the Internet's frontier: A first look at ISP interconnectivity in africa. In *Passive and Active Measurement Conference 2014*.

[24] Zheng Zhang, Ying Zhang, Y. Charlie Hu, Z. Morley Mao, and Randy Bush. iSpy: detecting IP prefix hijacking on my own. In *SIGCOMM '08: Proceedings of the ACM SIGCOMM 2008 conference on Data communication*, pages 327–338, New York, NY, USA, 2008. ACM.

[25] Ethan K. Bassett, Harsha V. Madhyastha, John P. John, Arvind Krishnamurthy, David Wetherall, and Thomas Anderson. Studying black holes in the Internet with Hubble. In *Proceedings of the 5th USENIX Symposium on Networked Systems Design and Implementation*, NSDI'08, pages 247–262, Berkeley, CA, USA, 2008. USENIX Association.

[26] Nick Feamster, David G. Andersen, Hari Balakrishnan, and M. Frans Kaashoek. Measuring the effects of Internet path faults on reactive routing. *SIGMETRICS Perform. Eval. Rev.*, 31(1):126–137, June 2003.

[27] Ethan Katz-Bassett, Colin Scott, David R. Choffnes, Ítalo Cunha, Vytautas Valancius, Nick Feamster, Harsha V. Madhyastha, Thomas Anderson, and Arvind Krishnamurthy. Lifeguard: Practical repair of persistent route failures. In *Proceedings of the ACM SIGCOMM 2012 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, SIGCOMM '12, pages 395–406, New York, NY, USA, 2012. ACM.

[28] Harsha V. Madhyastha, Thomas Anderson, Arvind Krishnamurthy, Neil Spring, and Arun Venkataramani. A structural approach to latency prediction. In *Proceedings of the 6th ACM SIGCOMM Conference on Internet Measurement*, IMC '06, pages 99–104, New York, NY, USA, 2006. ACM.

[29] Italo Cunha, Renata Teixeira, Darryl Veitch, and Christophe Diot. Predicting and tracking Internet path changes. In *Proceedings of the ACM SIGCOMM 2011 Conference*, SIGCOMM '11, pages 122–133, New York, NY, USA, 2011. ACM.

[30] D. K. Lee, Keon Jang, Changhyun Lee, Gianluca Iannaccone, and Sue B. Moon. Path stitching: Internet-wide path and delay estimation from existing measurements. In *INFOCOM*, pages 201–205, 2010.

[31] Xing Jin, Wanqing Tu, and S.-H.G. Chan. Traceroute-based topology inference without network coordinate estimation. In *Communications, 2008. ICC '08. IEEE International Conference on*, pages 1615–1619, May 2008.

[32] Minseok Kwon and Sonia Fahmy. Topology-aware overlay networks for group communication. In *Proceedings of the 12th International Workshop on Network and*

Operating Systems Support for Digital Audio and Video, NOSSDAV '02, pages 127–136, New York, NY, USA, 2002. ACM.

[33] Xing Jin, Yajun Wang, and S.-H.G. Chan. Fast overlay tree based on efficient end-to-end measurements. In *Communications, 2005. ICC 2005. 2005 IEEE International Conference on*, volume 2, pages 1319–1323 Vol. 2, May 2005.

[34] Junghee Han, D. Watson, and F. Jahanian. Topology aware overlay networks. In *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, volume 4, pages 2554–2565 vol. 4, March 2005.

[35] John Dilley, Bruce Maggs, Jay Parikh, Harald Prokop, Ramesh Sitaraman, and Bill Weihl. Globally distributed content delivery. *IEEE Internet Computing*, 6(5):50–58, September 2002.

[36] Josh Karlin, Stephanie Forrest, and Jennifer Rexford. Nation-State Routing: Censorship, Wiretapping, and BGP. *CoRR*, abs/0903.3218, 2009.

[37] Xueyang Xu, Zhuoqing Morley Mao, and J. Alex Halderman. Internet censorship in china: Where does the filtering occur? In *PAM*, pages 133–142, 2011.

[38] Min Suk Kang, Soo Bum Lee, and Virgil D Gligor. The crossfire attack. In *Security and Privacy (SP), 2013 IEEE Symposium on*, pages 127–141. IEEE, 2013.

[39] Max Schuchard, Abedelaziz Mohaisen, Denis Foo Kune, Nicholas Hopper, Yongdae Kim, and Eugene Y. Vasserman. Losing control of the Internet: Using the data plane to attack the control plane. In *Proceedings of the 17th ACM Conference on Computer and Communications Security*, CCS '10, pages 726–728, New York, NY, USA, 2010. ACM.

[40] Bernard Wong, Ivan Stoyanov, and Emin Gün Sirer. Octant: A comprehensive framework for the geolocalization of Internet hosts. In *Proceedings of the 4th USENIX Conference on Networked Systems Design & Implementation*, NSDI'07, pages 23–23, Berkeley, CA, USA, 2007. USENIX Association.

[41] B. Gueye, A. Ziviani, M. Crovella, and S. Fdida. Constraint-Based Geolocation of Internet Hosts. *IEEE/ACM Transactions on Networking*, 14(6):1219–1232, 2006.

[42] Ethan Katz-Bassett, John P. John, Arvind Krishnamurthy, David Wetherall, Thomas Anderson, and Yatin Chawathe. Towards IP geolocation using delay and topology measurements. In *Proceedings of the 6th ACM SIGCOMM Conference on Internet Measurement*, IMC '06, pages 71–84, New York, NY, USA, 2006. ACM.

[43] Tobias Flach, Ethan Katz-Bassett, and Ramesh Govindan. Quantifying violations of destination-based forwarding on the Internet. In *Proceedings of the 2012 ACM*

Conference on Internet Measurement Conference, IMC '12, pages 265–272, New York, NY, USA, 2012. ACM.

[44] Ratul Mahajan, Neil Spring, David Wetherall, and Tom Anderson. Inferring link weights using end-to-end measurements. In *Proceedings of the 2Nd ACM SIGCOMM Workshop on Internet Measurment*, IMW '02, pages 231–236, New York, NY, USA, 2002. ACM.

[45] Benoit Donnet, Matthew Luckie, Pascal Mérindol, and Jean-Jacques Pansiot. Revealing MPLS tunnels obscured from Traceroute. *SIGCOMM Comput. Commun. Rev.*, 42(2):87–93, March 2012.

[46] Umar Javed, Italo Cunha, David Choffnes, Ethan Katz-Bassett, Thomas Anderson, and Arvind Krishnamurthy. Poiroot: Investigating the root cause of interdomain path changes. *SIGCOMM Comput. Commun. Rev.*, 43(4):183–194, August 2013.

[47] J. Postel. Internet Protocol. RFC 791 (INTERNET STANDARD), September 1981. Updated by RFCs 1349, 2474, 6864.

[48] Van Jacobson and S Deering. Traceroute tool, 1989.

[49] F. Baker. Requirements for IP Version 4 Routers. RFC 1812 (Proposed Standard), June 1995. Updated by RFCs 2644, 6633.

[50] Hongyi Zeng, Peyman Kazemian, George Varghese, and Nick McKeown. A survey on network troubleshooting. Technical report, Technical Report Stanford/TR12-HPNG-061012, Stanford University, 2012.

[51] RA Steenbergen. A practical guide to (correctly) troubleshooting with traceroute. *North American Network Operators Group*, pages 1–49, 2009.

[52] Mehmet Hadi Gunes and Kamil Saraç. Resolving anonymous routers in Internet topology measurement studies. In *INFOCOM*, pages 1076–1084, 2008.

[53] B. Yao, R. Viswanathan, F. Chang, and D. Waddington. Topology inference in the presence of anonymous routers. In *INFOCOM 2003*, volume 1, pages 353–363. IEEE, 2003.

[54] X. Jin, W.P.K. Yiu, S.H.G. Chan, and Y. Wang. Network topology inference based on end-to-end measurements. *JSAC*, 24(12):2182–2195, 2006.

[55] Rob Sherwood and Neil Spring. Touring the Internet in a TCP sidecar. In *Proceedings of the 6th ACM SIGCOMM Conference on Internet Measurement*, IMC '06, pages 339–344, New York, NY, USA, 2006. ACM.

[56] Rob Sherwood, Adam Bender, and Neil Spring. Discarte: A disjunctive Internet cartographer. In *Proceedings of the ACM SIGCOMM 2008 Conference on Data Communication*, SIGCOMM '08, pages 303–314, New York, NY, USA, 2008. ACM.

[57] B. Carpenter and S. Brim. Middleboxes: Taxonomy and issues, 2002.

[58] CISCO Systems. Asa/pix/fwsm: Handling ICMP pings and traceroute. http://www.cisco.com/image/gif/paws/15246/31.pdf.

[59] S. Zander, G.J. Armitage, and P. Branch. Dynamics of the IP Time To Live field in Internet traffic flows. *CAIA Tech. Rep. 070529A*.

[60] Gregory Detal, Benjamin Hesmans, Olivier Bonaventure, Yves Vanaubel, and Benoit Donnet. Revealing middlebox interference with tracebox. In *Proceedings of the 2013 Conference on Internet Measurement Conference*, IMC '13, pages 1–8, New York, NY, USA, 2013. ACM.

[61] Zhuoqing Morley Mao, Jennifer Rexford, Jia Wang, and Randy H. Katz. Towards an accurate AS-level traceroute tool. In *Proceedings of the 2003 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, SIGCOMM '03, pages 365–378, New York, NY, USA, 2003. ACM.

[62] Yu Zhang, Ricardo Oliveira, Yangyang Wang, Shen Su, Baobao Zhang, Jun Bi, Hongli Zhang, and Lixia Zhang. A framework to quantify the pitfalls of using traceroute in AS-level topology measurement. *IEEE Journal on Selected Areas in Communications*, 29(9):1822–1836, October 2011.

[63] Y. Hyun, A. Broido, and K.C. Claffy. On third-party addresses in traceroute paths. In *Passive and Active Measurement Conference 2003*.

[64] Brice Augustin, Xavier Cuvellier, Benjamin Orgogozo, Fabien Viger, Timur Friedman, Matthieu Latapy, Clémence Magnien, and Renata Teixeira. Avoiding traceroute anomalies with paris traceroute. In *Proceedings of the 6th ACM SIGCOMM Conference on Internet Measurement*, IMC '06, pages 153–158, New York, NY, USA, 2006. ACM.

[65] Brice Augustin, Timur Friedman, and Renata Teixeira. Measuring load-balanced paths in the Internet. In *Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement*, IMC '07, pages 149–160, New York, NY, USA, 2007. ACM.

[66] CISCO. How does load balancing work. http://www.cisco.com/c/en/us/support/docs/ip/border-gateway-protocol-bgp/5212-46.htm.

[67] Juniper. Configuring load-balance per-packet action. http://www.juniper.net/techpubs/en_US/junos12.2/topics/usage-guidelines/policy-configuring-per-packet-load-balancing.html.

[68] Brice Augustin, Timur Friedman, and Renata Teixeira. Measuring multipath routing in the Internet. *IEEE/ACM Transactions on Networking*, 19(3):830–840, June 2011.

[69] Matthew Luckie, Amogh Dhamdhere, kc claffy, and David Murrell. Measured impact of crooked traceroute. *SIGCOMM Comput. Commun. Rev.*, 41(1):14–21, January 2011.

[70] D. Veitch, B. Augustin, R. Teixeira, and T. Friedman. Failure control in multipath route tracing. In *INFOCOM 2009, IEEE*, pages 1395–1403, April 2009.

[71] T. Moors. Streamlining traceroute by estimating path lengths. In *IP Operations and Management, 2004. Proceedings IEEE Workshop on*, pages 123–128, Oct 2004.

[72] A. Botta, W. de Donato, A. Pescape, and Giorgio Ventre. Discovering topologies at router level: Part II. In *Global Telecommunications Conference, 2007. GLOBECOM '07. IEEE*, pages 2696–2701, Nov 2007.

[73] Neil Spring, Ratul Mahajan, David Wetherall, and Thomas Anderson. Measuring ISP topologies with rocketfuel. *IEEE/ACM Transactions on Networking*, 12(1):2–16, February 2004.

[74] Benoit Donnet, Philippe Raoult, Timur Friedman, and Mark Crovella. Efficient algorithms for large-scale topology discovery. In *SIGMETRICS*, pages 327–338, 2005.

[75] Burton H. Bloom. Space/time trade-offs in hash coding with allowable errors. *Commun. ACM*, 13(7):422–426, July 1970.

[76] Matthew Luckie, Young Hyun, and Bradley Huffaker. Traceroute probe method and forward IP path inference. In *Proceedings of the 8th ACM SIGCOMM conference on Internet measurement*, IMC '08, pages 311–324, New York, NY, USA, 2008. ACM.

[77] Ethan K. Bassett, Harsha V. Madhyastha, Vijay K. Adhikari, Colin Scott, Justine Sherry, Peter Van Wesep, Thomas Anderson, and Arvind Krishnamurthy. Reverse traceroute. In *Proceedings of the 7th USENIX conference on Networked systems design and implementation*, NSDI'10, page 15, Berkeley, CA, USA, 2010. USENIX Association.

[78] Justin Cappos, Ivan Beschastnikh, Arvind Krishnamurthy, and Tom Anderson. Seattle: A platform for educational cloud computing. In *Proceedings of the 40th ACM Technical Symposium on Computer Science Education*, SIGCSE '09, pages 111–115, New York, NY, USA, 2009. ACM.

[79] CAIDA. Archipelago measurement infrastructure. http://www.caida.org/projects/ark/, 2008.

[80] Andy, Brent, Larry, and Mike Wawrzoniak. Operating System Support for Planetary-Scale Network Services. pages 253–266.

[81] Srikanth Sundaresan, Walter de Donato, Nick Feamster, Renata Teixeira, Sam Crawford, and Antonio Pescapè. Broadband Internet performance: a view from the gateway. *SIGCOMM Comput. Commun. Rev.*, 41(4):134–145, August 2011.

[82] Walter de Donato, Alessio Botta, and Antonio Pescapé. HoBBIT: a Platform for Monitoring Broadband Performance from the User Network. In *Proc. of Sixth International Workshop on Traffic Monitoring and Analysis (TMA)*, London, UK, April 2014.

[83] Qiang Xu, Junxian Huang, Zhaoguang Wang, Feng Qian, Alexandre Gerber, and Zhuoqing Morley Mao. Cellular data network infrastructure characterization and implication on mobile content placement. *SIGMETRICS Perform. Eval. Rev.*, 39(1):277–288, June 2011.

[84] Sachit Muckaden. Myspeedtest: active and passive measurements of cellular data networks. 2013.

[85] Pietro Marchetta, Pascal Mérindol, Benoit Donnet, Antonio Pescapè, and Jean-Jacques Pansiot. Topology discovery at the router level: A new hybrid tool targeting ISP networks. *IEEE Journal on Selected Areas in Communications*, 29(9):1776–1787, 2011.

[86] Pietro Marchetta, Pascal Mérindol, Benoit Donnet, Antonio Pescapè, and Jean-Jacques Pansiot. Quantifying and mitigating IGMP filtering in topology discovery. In *GLOBECOM*, pages 1871–1876, 2012.

[87] Ratul Mahajan, Ming Zhang, Lindsey Poole, and Vivek Pai. Uncovering performance differences among backbone isps with netdiff. In *Proceedings of the 5th USENIX Symposium on Networked Systems Design and Implementation*, NSDI'08, pages 205–218, Berkeley, CA, USA, 2008. USENIX Association.

[88] J. Postel. User Datagram Protocol. RFC 768 (INTERNET STANDARD), August 1980.

[89] J. Postel. Internet Control Message Protocol. RFC 792 (INTERNET STANDARD), September 1981. Updated by RFCs 950, 4884, 6633, 6918.

[90] J. Postel. Transmission Control Protocol. RFC 793 (INTERNET STANDARD), September 1981. Updated by RFCs 1122, 3168, 6093, 6528.

[91] Pierre Fransson and Andreas Jonsson. End-to-end measurements on performance penalties of ipv4 options. In *Global Telecommunications Conference, 2004. GLOBECOM'04. IEEE*, volume 3, pages 1441–1447. IEEE, 2004.

[92] Rodrigo Fonseca, George Manning Porter, Randy H. Katz, Scott Shenker, and Ion Stoica. Ip options are not an option. Technical Report UCB/EECS-2005-24, EECS Department, University of California, Berkeley, Dec 2005.

[93] Justine Sherry, Ethan K. Bassett, Mary Pimenova, Harsha V. Madhyastha, Thomas Anderson, and Arvind Krishnamurthy. Resolving IP aliases with prespecified timestamps. In *Proceedings of the 10th annual conference on Internet measurement*, IMC '10, pages 172–178, New York, NY, USA, 2010. ACM.

[94] Andrew D Ferguson and Rodrigo Fonseca. Inferring router statistics with IP timestamps. In *Proceedings of the ACM CoNEXT Student Workshop*, page 11. ACM, 2010.

[95] Walter de Donato, Pietro Marchetta, and Antonio Pescapé. A hands-on look at active probing using the IP prespecified timestamp option. In *Proceedings of the 13th International Conference on Passive and Active Measurement*, PAM'12, pages 189–199, Berlin, Heidelberg, 2012. Springer-Verlag.

[96] Alistair K. TSPS investigations, CAIDA report. http://www.caida.org/~alistair/projects/tsps-investigations.html.

[97] Pietro Marchetta and Antonio Pescapè. Drago: Detecting, quantifying and locating hidden routers in traceroute ip paths. In *INFOCOM*, pages 3237–3242, 2013.

[98] PREDICT ID USC-LANDER. IP Address Hitlist, internet- address- hitlist- it47w-20120427, 2010-03-29 to 2012-05-30.

[99] T. Cymru. IP to ASN mapping. *http://www.team-cymru.org/Services/ip-to-asn.html*, 2012.

[100] Chi Zhang, Ming an1d Zhang, Vivek Pai, Larry Peterson, and Randy Wang. PlanetSeer: Internet path failure monitoring and characterization in wide-area services. In *Proceedings of the 6th Conference on Symposium on Opearting Systems Design & Implementation - Volume 6*, OSDI'04, pages 12–12, Berkeley, CA, USA, 2004. USENIX Association.

[101] Ying Zhang, Z. Morley Mao, and Ming Zhang. Effective diagnosis of routing disruptions from end systems. In *Proceedings of the 5th USENIX Symposium on Networked Systems Design and Implementation*, NSDI'08, pages 219–232, Berkeley, CA, USA, 2008. USENIX Association.

[102] Pietro Marchetta, Walter de Donato, and Antonio Pescapé. Detecting third-party addresses in traceroute traces with IP timestamp option. In *Proceedings of the 14th International Conference on Passive and Active Measurement*, PAM'13, pages 21–30, Berlin, Heidelberg, 2013. Springer-Verlag.

[103] Pietro Marchetta, Walter de Donato, and Antonio Pescapè. Detecting third-party addresses in traceroute IP paths. In *SIGCOMM*, pages 109–110, 2012.

[104] Y. Rekhter, B. Moskowitz, D. Karrenberg, G. J. de Groot, and E. Lear. Address Allocation for Private Internets. RFC 1918 (Best Current Practice), February 1996. Updated by RFC 6761.

[105] PeeringDB. Exchange points list. https://www.peeringdb.com/.

[106] The CAIDA AS Relationships Dataset, June 2012. http://www.caida.org/data/active/as-relationships/.

[107] Isolario project. http://www.isolario.it/, 2013.

[108] G. Almes, S. Kalidindi, and M. Zekauskas. A Round-trip Delay Metric for IPPM. RFC 2681 (Proposed Standard), September 1999.

[109] Cristel Pelsser, Luca Cittadini, Stefano Vissicchio, and Randy Bush. From paris to tokyo: On the suitability of ping to measure latency. In *Proceedings of the 2013 Conference on Internet Measurement Conference*, IMC '13, pages 427–432, New York, NY, USA, 2013. ACM.

[110] Pietro Marchetta, Alessio Botta, Ethan Katz-Bassett, and Antonio Pescapè. Dissecting round trip time on the slow path with a single packet. In *PAM*, pages 88–97, 2014.

[111] P. Fransson and A. Jonsson. End-to-end measurements on performance penalties of IPv4 options. volume 3, pages 1441–1447 Vol.3, 2004.

[112] Pietro Marchetta, Valerio Persico, Antonio Pescapè, and Ethan Katz-Bassett. Don't trust traceroute (completely). In *Proceedings of the 2013 Workshop on Student Workhop*, CoNEXT Student Workhop '13, pages 5–8, New York, NY, USA, 2013. ACM.

[113] Lucas DiCioccio, Renata Teixeira, Martin May, and Christian Kreibich. Probe and pray: Using upnp for home network measurements. In *Passive and Active Measurement*, pages 96–105. Springer, 2012.

[114] Matthew Luckie. Scamper: A scalable and extensible packet prober for active measurement of the Internet. In *Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement*, IMC '10, pages 239–245, New York, NY, USA, 2010. ACM.

[115] P. Almquist. Type of Service in the Internet Protocol Suite. RFC 1349 (Proposed Standard), July 1992. Obsoleted by RFC 2474.

[116] R. Braden. Requirements for Internet Hosts - Communication Layers. RFC 1122 (INTERNET STANDARD), October 1989. Updated by RFCs 1349, 4379, 5884, 6093, 6298, 6633, 6864.

[117] Maxmind geoip. http://www.maxmind.com/app/ip-location.

[118] Adam Bender, Rob Sherwood, and Neil Spring. Fixing Ally's growing pains with velocity modeling. In *Proceedings of the 8th ACM SIGCOMM Conference on Internet Measurement*, IMC '08, pages 337–342, New York, NY, USA, 2008. ACM.

[119] Ken Keys, Young Hyun, Matthew Luckie, and Kim Claffy. Internet-scale IPv4 alias resolution with MIDAR. *IEEE/ACM Transactions on Networking*, 21(2):383–399, April 2013.

[120] R. Fonseca, G. Porter, R. Katz, S. Shenker, and I. Stoica. IP options are not an option. *Tech. Rep. Univ. of California*, 2005.

[121] RA Steenbergen. A practical guide to (correctly) troubleshooting with traceroute. *North American Network Operators Group*, pages 1–49.

[122] Jean-Jacques Pansiot and Dominique Grad. On routes and multicast trees in the Internet. *SIGCOMM Comput. Commun. Rev.*, 28(1):41–50, January 1998.

[123] M. Engin Tozal and Kamil Sarac. Palmtree: An IP alias resolution algorithm with linear probing complexity. *Comput. Commun.*, 34(5):658–669, April 2011.

[124] Matthew Luckie, Robert Beverly, William Brinkmeyer, and kc claffy. Speedtrap: Internet-scale IPv6 alias resolution. In *Proceedings of the 2013 Conference on Internet Measurement Conference*, IMC '13, pages 119–126, New York, NY, USA, 2013. ACM.

[125] Neil Spring, Ratul Mahajan, David Wetherall, and Thomas Anderson. Measuring ISP topologies with rocketfuel. *IEEE/ACM Transactions on Networking*, 12(1):2–16, February 2004.

[126] Pascal Mérindol, Benoit Donnet, J-J Pansiot, Matthew Luckie, and Young Hyun. MERLIN: MEasure the router level of the INternet. In *Next Generation Internet (NGI), 2011 7th EURO-NGI Conference on*, pages 1–8. IEEE, 2011.

[127] Larissa Spinelli, Mark Crovella, and Brian Eriksson. Aliascluster: A lightweight approach to interface disambiguation. In *Computer Communications Workshops (INFOCOM WKSHPS), 2013 IEEE Conference on*, pages 127–132. IEEE, 2013.

[128] Pietro Marchetta, Valerio Persico, and Antonio Pescapè. Pythia: Yet another active probing technique for alias resolution. In *Proceedings of the Ninth ACM Conference on Emerging Networking Experiments and Technologies*, CoNEXT '13, pages 229–234, New York, NY, USA, 2013. ACM.

[129] Mario A. Sánchez, John S. Otto, Zachary S. Bischof, David R. Choffnes, Fabián E. Bustamante, Balachander Krishnamurthy, and Walter Willinger. Dasu: Pushing experiments to the Internet's edge. In *Proceedings of the 10th USENIX Conference on*

Networked Systems Design and Implementation, nsdi'13, pages 487–500, Berkeley, CA, USA, 2013. USENIX Association.

[130] Harsha V. Madhyastha, Ethan Katz-Bassett, Thomas Anderson, Arvind Krishnamurthy, and Arun Venkataramani. iPlane Nano: Path prediction for peer-to-peer applications. In *Proceedings of the 6th USENIX Symposium on Networked Systems Design and Implementation*, NSDI'09, pages 137–152, Berkeley, CA, USA, 2009. USENIX Association.

[131] Mehmet H. Gunes and Kamil Sarac. Analyzing router responsiveness to active measurement probes. In *Proceedings of the 10th International Conference on Passive and Active Network Measurement*, PAM '09, pages 23–32, Berlin, Heidelberg, 2009. Springer-Verlag.

[132] Matthew Caesar and Jennifer Rexford. BGP routing policies in ISP networks. *IEEE Network*, 19(6):5–11, 2005.

[133] Xiaoliang Zhao, Dan Pei, Lan Wang, Dan Massey, Allison Mankin, S. Felix Wu, and Lixia Zhang. An analysis of BGP multiple origin AS (MOAS) conflicts. In *Proceedings of the 1st ACM SIGCOMM Workshop on Internet Measurement*, IMW '01, pages 31–35, New York, NY, USA, 2001. ACM.

[134] Lixin Gao. On inferring autonomous system relationships in the Internet. *IEEE/ACM Transactions on Networking*, 9(6):733–745, December 2001.

[135] Wolfgang Mühlbauer, Anja Feldmann, Olaf Maennel, Matthew Roughan, and Steve Uhlig. Building an AS-topology model that captures route diversity. *SIGCOMM Comput. Commun. Rev.*, 36(4):195–206, August 2006.