MERLIN: MEasure the Router Level of the INternet

Pascal Mérindol*, Benoit Donnet†, Jean-Jacques Pansiot*, Matthew Luckie‡, Young Hyun**
* LSIIT, Université de Strasbourg – France
† ICTEAM, Université catholique de Louvain – Belgium
‡ WAND Network Research Group, Waikato University – New Zealand
** CAIDA, University of California at San Diego – USA

***Abstract*— In this paper, we present a new router-level Internet mapping tool called MERLIN. MERLIN takes advantage of `mrinfo`, a multicast management tool that collects all IPv4 multicast enabled interfaces of a router and all its multicast links towards its neighbors. Our new probing tool fixes `mrinfo` technical limitations and eases the deployment of multicast probing campaign. We deploy and evaluate the performance of MERLIN at large scale. We investigate the completeness of MERLIN by providing a lower bound on the proportion of information that it may miss. We also demonstrate that the use of several vantage points is crucial to circumvent IGMP filtering in order to increase the amount of collected routers. MERLIN is a valuable tool for collecting the router-level Internet topology.**

I. INTRODUCTION

Internet topology discovery has been an intense research subject during the past decade [1], [2]. Most of the deployed tools are based on *traceroute* [3], [4], [5], [6], [7]. Traceroute discovers the Internet topology at the interface level, i.e., the IP interfaces of routers and end-hosts. All routers and some hosts have multiple interfaces, and each interface may appear as a separate entity in this topology. The resulting graph consists of the link-layer connections between those pseudo-nodes. However, inferring the Internet characteristics at the router level is an important concern for many reasons and, more specifically, for routing protocol design. It allows for understanding the structural and architectural design of IP networks at their core component granularity, the router.

Using IP level traces, if one wants to build a network map at the router level, it is necessary to gather all interfaces of a given router discovered with traceroute into a single identifier. This summary technique is called *alias resolution* [8], [9], [10], [11]. The accuracy of alias resolution has an important effect on the observed graph characteristics such as the node degree distribution [12]. However, alias resolution comes with several drawbacks. First, it is based on a preliminary traceroute campaign. Traceroute is known to be intrusive and redundant although improvements have been proposed to reduce its impact on the network [13]. It is also likely that traceroute will not discover all interfaces of a given router (in particular the ones used for backup paths). Second, alias resolution is either intrusive (it requires additional probing), or computationally expensive (it requires an intensive post-processing phase). Finally, alias resolution is not entirely accurate as it might generate false positives, i.e., two IP addresses are tagged as aliases while they are not.

Recently, `mrinfo`, a multicast management tool, has been used for topology discovery [14], [15]. `mrinfo` comes with the strong advantage of listing all multicast interfaces of a router and its multicast links towards others using a single probe. `mrinfo` offers, by design, a router-level view of the topology: it does not suffer from the same shortcomings resulting from combining traceroute and alias resolution techniques. However, its view is limited to multicast components and, in the same way that ICMP messages may be rate limited or filtered for traceroute probing, IGMP messages can be filtered by some ISPs [16].

In this paper, we start by pointing out several technical limitations of `mrinfo`: it suffers from an IP fragmentation issue and the lack of multiplexing support. In order to fix these limitations, we implement and evaluate a new tool, *MEasure the Router Level of the INternet* (MERLIN). MERLIN allows one to infer the multicast map of the Internet at the router level and is designed for large scale topology discovery campaigns.

Because `mrinfo`-like probing is only applicable to multicast-enabled routers, we investigate the notion of *completeness* and provide a lower bound on the quantity of topological data that a multicast probing campaign may miss compared to standard probing techniques. We then discuss the deployment of MERLIN on several geographically distributed vantage points and show that each vantage point is able to discover a significant portion of unique routers.

The remainder of this paper is organized as follows: Sec. II discusses `mrinfo` and its limitations; Sec. III presents our new tool MERLIN and discusses calibration procedures and limitations; Sec. IV evaluates the "correctness and completeness" of MERLIN; finally, Sec. V concludes this paper by summarizing its main achievements and discussing future research directions.

II. `MRINFO`

This section focuses on the original `mrinfo`. We first quickly describe the basics of this tool (Sec. II-A). We next explain our data collection methodology (Sec. II-B) and, finally, discuss and quantify the limitations of `mrinfo` (Sec. II-C).

A. Tool Description

In the late 1980s, the developers of IP multicast designed the MBone, an overlay network composed of tunnels that interconnected workstations running an implementation of DVMRP [17]. Several tools have been developed to monitor and debug the MBone [18]. Most of these tools have been deprecated with the replacement of DVMRP by the Protocol Independent Multicast (PIM) family of multicast routing protocols with one notable exception: `mrinfo`.

`mrinfo` uses the *Internet Group Management Protocol* (IGMP) [19]. DVMRP has defined two special types of IGMP messages that can be used to monitor routers [17]. Although current IPv4 multicast routers no longer use DVMRP anymore, they still support those special IGMP messages. Upon reception of an IGMP `ASK_NEIGHBORS` message, an IPv4 multicast router will reply with an IGMP `NEIGHBORS_REPLY` message that lists all its multicast adjacencies with some information about their state. Interested readers can find further details on `mrinfo` in [14], [15].

B. Data Collection

Previously, `mrinfo` measurements were conducted recursively with `mrinfo-rec` [14], [15], which would probe a target with `mrinfo` and then recursively invoke `mrinfo` on all IP addresses discovered in responses. This approach is designed to discover and study the largest multicast component reachable from a single starting target address, the *seed*, and from a single vantage point.

`mrinfo-rec` experiments were run daily in order to understand the dynamics of the Internet graph. To maximize discovered topology, we used the set of responding routers of a given day as the seed for the next day's recursive run. This seeding procedure allowed us to take advantage of any changes in the routing system to discover new areas of the multicast-enabled Internet. Between May 1st, 2004 and December 31st, 2008, `mrinfo-rec` was able to discover 10,000 routers on average from a single vantage point in Strasbourg, France. However, we observed notable and sudden changes in data collection over this period. Such sudden and significant changes cannot be due to network dynamics: they are an artifact of `mrinfo-rec` launched from only a single vantage point, making data collection susceptible to filtering.

Moreover, `mrinfo-rec` is not scalable to large experiments and the initial implementation of the `mrinfo` client suffers from several drawbacks as explained in Sec. II-C. Our objective in this paper is to overcome those limitations to probe millions of IPs in a reasonable timescale.

In the remainder of the paper, all provided statistics and analysis rely on a global scale MERLIN probing campaign using a seed list with more than 1,5 million IP addresses. This campaign is based on six vantage points well distributed across the AS level graph and results in a router level map containing more than 50,000 routers. All details are given in Sec. IV.

C. Implementation Issues

We recently discovered that the initial `mrinfo` client implementation suffers from several issues and limitations. In this section, we investigate two critical problems: the lack of support for IGMP-fragmented `NEIGHBORS_REPLY` messages (Sec. II-C.1) and the difficulty to multiplex IGMP-based measurements (Sec. II-C.2).

While the first problem is simply a shortcoming of the initial `mrinfo` client, the second problem raises the question of a compromise between the efficiency and the correctness of a large-scale `mrinfo` campaign.

1) Fragmentation: Since they are encapsulated within IP headers, IGMP `NEIGHBORS_REPLY` messages may face fragmentation. The total size, in terms of bytes, of such a reply message can be computed as follows:

$$|\text{headers}| + \sum_{i=1}^{n} (8 + 4 \times m_i). \qquad (1)$$

where "header" refers to the sum of the IP and IGMP message headers (20 + 8 bytes), n is the number of local addresses belonging to the router and m_i refers to the number of distant addresses seen through the i^{th} local address. The description of a point-to-point link (i.e., a direct connection between two routers) takes up 12 bytes and consists of the two endpoint IP addresses and several attributes of the local address (the multicast metric, threshold, flags, and the number of distant addresses, which is $m_i = 1$ in this case). In contrast, a point-to-multipoint link (i.e., a broadcast oriented connection involving several routers connected through a layer-2 device) takes up $12 + (m_i - 1) \times 4$ bytes, which includes listing $m_i + 1$ IP addresses.

According to the DVMRP draft [17], the replying router should use path MTU discovery to determine whether a DVMRP message must be fragmented. When path MTU is unknown, the Requirements for Internet Hosts (RFC 1122) specifies a maximum packet size of 576 bytes. Note that a `NEIGHBORS_REPLY` message do not contain any port nor query numbers. Therefore, a large IGMP reply should be sent as several independent IGMP messages having only the source IP address in common.

Depending on their operating system (OS), we notice that routers manage differently the DVMRP fragmentation requirement. Indeed, one interesting feature of `mrinfo` is its ability to partially fingerprint the OS version of the responding routers and thus study their behavior differences. We are thus able to collect some statistics about the amount of deployed routers of a given brand and their OS respective behaviors. Obviously, the `mrinfo` view reflects the market: *Cisco* routers dominates ($\approx$ 78 %), directly followed by *Juniper* in a much smaller proportion ($\approx$ 7.5 %). The "other brands" (i.e., those we were not able to accurately classify) just represent a little bit more than 13% of the total amount of routers collected (see Sec. IV for details about our probing campaigns).

In response to `mrinfo` probes, Juniper routers with a large number of connections forge a single large IP packet that is "IP fragmented" by the sending interface. Although this behavior is incorrect according to the DVMRP draft, `mrinfo` easily handles these large responses since IP fragmentation is transparent to it. In contrast, Cisco routers with a large number of connections follow the draft recommendations and reply with multiple independent IP packets small enough to avoid IP fragmentation: we call this behavior "IGMP fragmentation". In this case, the initial `mrinfo` client does not deal with the multiple received packets: it only processes the first one because there is no continuation flag forcing to wait for the remaining fragments. Therefore, the initial version of `mrinfo` is unable to collect the entire interface list returned by large-degree *Cisco* routers.

Unfortunately, the router market being dominated by Cisco, this brand is the most common in our dataset ($\approx$ 78 %). To determine the impact of the IGMP fragmentation, we compute the number of concerned routers and the number of fragmented packets. Although, the proportion of routers generating fragmentation is quite low ($\approx$ 6 %), they may generate a great number of fragments (between 10 and 470 in 2% of the cases). Indeed, a small proportion of routers generate almost half of the returned traffic. Generally, routers generating dozens of IGMP fragments do not report interesting topological information: they mostly report non-publicly routable addresses. However, the IGMP fragmentation limitation may strongly reduce the efficiency of the recursion scheme: even a small amount of missed publicly routable IP may hide multicast neighbors potentially allowing to reach a large set of neighbors and so on.

2) Multiplexing: This section focuses on performance issues related to large-scale `mrinfo` campaigns. The initial `mrinfo` client works as follow: first, it sends its IGMP query, then it waits for a possible reply during a given timer of t seconds. Possibly, it performs up to n retries if no response has been collected within the previous time frames. If we consider a set of targets consisting of m IP addresses, then the whole process may last $t \times n \times m$ seconds in the worst case. A large-scale run of one million targets ($m = 10^6$) with realistic parameters ($t = 2$ and $n = 2$) could last more than 46 days. It may seem like we only need to run multiple `mrinfo` instances on a single vantage point to reduce the running time. However, IGMP does not use ports or query numbers to multiplex incoming/outgoing connections. Therefore, a single computer having only one IP address should not simultaneously run multiple instances of `mrinfo`. Indeed, each parallel instance of `mrinfo` will treat received responses related to other instances as a reply to its last query leading so to confusion.

There is a seemingly obvious workaround that does not work in practice: running multiple parallel `mrinfo` instances, each instance may only treat the reply whose source IP is equal to its last query. However, in practice, a router can reply with an IP address different than the one queried (see Sec. III for more details). When the responding IP address Y does not match the probed IP address X, `mrinfo` reports a warning stating that Y has responded "instead of" X. This "instead of" behavior can be normal and is not rare. Roughly 10% of the replies fall in the "instead of" case, none of them involving Cisco routers. Hence, the workaround of checking the responding address will fail for "instead of" responses, since such a response will be ignored by all `mrinfo` instances (including the instance that elicited it). Thus, except using multihomed vantage points - having multiple IP addresses - or multiplying the number of vantage points, there does not exist a simple way to overcome this problem.

III. MERLIN

As detailed in Sec. II-C, one must modify the initial `mrinfo` client implementation to fix both issues without impacting the completeness of the probing campaign. In this section, we describe our new tool, MERLIN that stands for *MEasure the Router Level of the INternet*. We opt for an architecture not requiring the use of multiple IP addresses or multiple vantage points to ease its deployment. This new tool is easily configurable and provide an efficient and network-friendly probing approach: MERLIN minimizes the reprobing risk while it allows one to considerably improve the efficiency of a large-scale `mrinfo` based probing campaign. The basis of the MERLIN architecture is to decouple the sending and receiving processes in order to avoid the use of timers between queries and replies and improve the probing efficiency. With this new scheme, replies are indexed according to the source IP of the reply, so we do not rely on the targeted IP anymore. Furthermore, all replies having the same source IP address are considered as part of a larger reply in order to re-assemble IGMP fragmented packets reported by a given router.

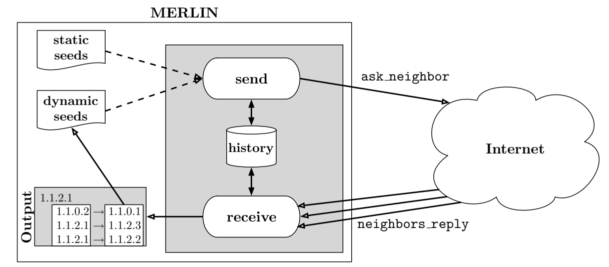


Fig. 1. MERLIN architecture

Sec. III-A describes the MERLIN architecture while Sec. III-B discuss its calibrations and limitations.

A. Architecture

Fig. 1 depicts the MERLIN architecture. The heart of MERLIN is made of two processes: *send*, in charge of sending probes to the network, and *receive*, in charge of processing the replies returned back by routers. These processes are totally decoupled and the recursion is embedded.

In order to minimize redundancy, the sending process never probes an IP address previously discovered: for efficiency, we use a hash table indexed on local IP addresses of each replying router (the "history" box in Fig. 1). Furthermore, to also minimize the memory consumption, we associate a linked list of IP header and data checksums to each source IP address: a packet is considered as new only if its checksum does not belong to the list of already recorded checksum. It allows one to reduce redundant replies while avoiding, at the same time, dozen of identical messages generated by some routers. Note that MERLIN keeps track of the actual binary reply format such that it is able to differentiate point-to-multipoint links from multiple point-to-point links using the same local address (see [15]). Moreover, to deal with the IGMP fragmentation issue and again to remain light in terms of memory consumption, MERLIN uses a timer s to determine when the information related to a given router can be flushed to the output file. If no new fragment associated to a router r has been collected during the previous time frame of s seconds, the data structure corresponding to r is freed and the output of r is definitively flushed.

The send process is fed by both a static IP address list (called *seeds* on Fig. 1) and a dynamic IP address list obtained from replies. This dynamic list is used for recursion. At the starting of MERLIN, the send process receives IP addresses from the static list. Once replies are collected from the receiving process, the dynamic list is built based on publicly routable IP addresses belonging to the neighbor address list and the recursion is engaged, i.e., the send process gives priority to targets from the dynamic list. Each time the dynamic list is empty (i.e., the current recursion is finished), the send process is again fed with IP addresses from the static list (the initial seeds). Recursion first is a design choice that has been made in order to minimize the probability of reprobing a given router. Moreover, this design choice also ensures to collect a connected part of the probed topology in a short timescale: it allows one to increase the topology consistency in case of topological changes. Indeed, the dynamics of the Internet graph may lead to false connectivity inferences when connected routers are probed in a timescale greater than the one of potential changes.

A key feature of MERLIN is its friendly approach in probing, making it scalable as it avoids reprobing IP addresses previously discovered or already targeted. This is achieved by maintaining information about already processed IP addresses but, also, by slowing down the send process. Indeed, if the time between subsequent probes is too tight, it is very likely to probe the same router many times in case of discovering a highly connected portion of the network. For example, this happens when a pair of routers are connected through multiple logical/physical links or when several routers form a clique. In that case, two or more probes towards the same router can be sent before receiving its reply. Let us illustrate this situation with Fig. 2: router R_1 is able to see R_4 through two direct interfaces and is connected to routers R_2 and R_3. Now, let us imagine that after collecting the interfaces of R_1 the send process injects in the network four consecutive probes (within a tight timescale): two towards IP addresses belonging to R_4 (it cannot know yet that those addresses belong to the same router), and two respectively towards R_2 and R_3. At this step, R_4 is already probed two times. Moreover, if its reply is received after the ones of R_2 and R_3, the recursion will lead to sending two additional probes towards R_4 (the ones resulting from R_2 and R_3 IP neighbors list). This scenario can easily happen if router R_4 is slower than R_2 and R_3 to generate its IGMP response or if forwarding routes fluctuate among those routers. Thus, the only way to prevent routers from that redundant probing is to force waiting a reply using a timer before sending a new request. Sec. III-B describes how we calibrate MERLIN to achieve a good tradeoff between an efficient and network friendly probing scheme.

The send process of MERLIN considers two probing modes: the *dynamic* and the *static* modes. The dynamic mode occurs with the recursion based on the dynamic list. During this phase, the probe inter-departure time is fixed to a given value α. On the contrary, the static mode corresponds to probing based on the static list. In that case, the inter-departure parameter is fixed to a lower value β: $\beta \ll \alpha$. To minimize the reprobing risk, the sending process prioritizes its treatment tasks as follows: (1) if a new router has been discovered, it marks all its local addresses as already seen, (2) if there exist recursive IP addresses to probe, it elapses the probing with the timer α, (3) otherwise it uses the static list and elapses probes with the timer β. Those choices have been made regarding several considerations detailed in [20] in order to reduce the probability to reprobe the same router.

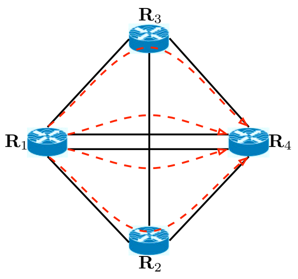


Fig. 2. Reprobing risk on R_4

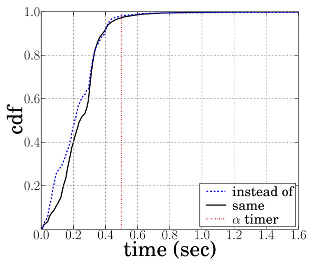


Fig. 3. Request/replies delays

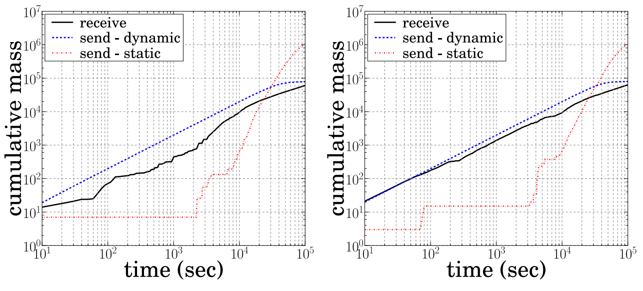
MERLIN is fully written in C and is freely available on request. It works on Linux and FreeBSD distributions and includes several compilation options to extend its capabilities. For instance, it is possible to force the use of a given IP address for multihomed hosts. It is also possible to forbid the probing and/or indexing of a set of given IP addresses in order to use MERLIN sequentially among a set of vantage points.

B. Calibrating MERLIN

This section experimentally explains our parameter calibration choices. Some routers generate hundreds of IGMP fragmented packets just to describe their own interfaces list (see Sec. II-C.1). To deal with those rare and extreme cases, we need to choose a timer s sufficiently large to ensure the complete response reassembling. In practice, even for routers generating more than a hundred replies (the maximum observed is 470 fragments for a given router), we measure that all responses arrived in the tight timeframe of 0.1 second. However, in order to perform a good tradeoff between CPU and memory usage, we decide to set a default timer of $s = 5$ seconds ($s \gg 0.1$ to ensure the correct reception of all fragments even with network troubles). Thus, the number of routers flushed in a given timeframe is limited while the number of CPU interruptions remain low and the memory is freed suffcently often.

In order to investigate the choice of the recursive inter-departure parameter (α), we perform an experimental analysis. Fig. 3 has been obtained thanks to our previous `mrinfo-rec` tool. Indeed, for such an analysis, we need to deterministically link the target IP and the source IP of the reply. Furthermore, to avoid confusion among replies, we use a very large timer value ($t = 10$ seconds) before sending the next probe. We can notice that in the vast majority of the cases, responses are returned back to the vantage point in less than 0.5 second: about 99% of replies are collected before the expiration of this timer. Thus, we decide to set $\alpha = 0.5$ second by default to avoid most of the reprobing risk using the recursion mode.

The choice of β is made differently because the probability of probing a given router twice or more using the static list is



(a) San Diego

(b) Hamilton

Fig. 4. Dynamic vs. static list - July, 9th 2010

low. We decide to set $\beta = 0.05$ second, i.e., at maximum, 20 probes are sent per second. This value offers a good compromise for limiting the rate of the send process while being able to probe more than 1.5M of IP addresses in less than one day. With this static calibration, the probability of reprobing the same router is insignificant. The success rate of the static list (i.e., the probability that an IP belonging to the static list respond to IGMP probes[1].) decreases over time due to the recursion and is, on average, under 2%. Finally, note that the static list is randomly sorted and the number of interfaces of a router is small compared to the number of IP belonging to the static list.

Fig. 4 plots the evolution over time of main MERLIN actions: the number of probes sent (from dynamic and static list) and the number of received replies according to the vantage point. Due to space constraints, we only show plots for two vantage points: San Diego and Hamilton (two of the most useful vantage points, see Sec. IV for details). The horizontal axis gives the time (in seconds) from the starting of the probing until the end. The probing lasted roughly 31 hours and we consider the probing campaign launched on July, 9th 2010. The vertical axis provides the cumulative mass of probes sent and replies received. Finally, it is worth to notice that Fig. 4 uses a log-log scale, as it highlights more easily the first probing periods.

During the early moments of a MERLIN measurement campaign, the recursion (i.e., probes sent via the dynamic list) "does the job": MERLIN has a recursion-first nature. Indeed, during the first hour of probing, we notice that a very low number of static probes are used while the number of received replies is close to the number of recursive probes sent (especially during the first minutes), meaning that we are able to collect large multicast components. This also means that the success rate of the recursion mode (e.g., the probability that multicast neighbors respond to IGMP probes) is relatively high during this first phase: we are able to collect large multicast connected components. However, this success rate rapidly decreases over time and the use of static probes becomes more and more required. After the first hours, the situation completely changes: now, static seeds are often solicited and recursion phases become shorter.

[1]Note that such an analysis does not reflect the global probing coverage of MERLIN. Indeed, the static list is updated each time MERLIN collects a new router such that all collected interfaces are removed from the static list. Sec. IV-A.1 provides an approximation of the actual multicast coverage.

Thus, after having consumed the largest multicast components retrievable thanks to the target list, MERLIN finds small sets of isolated routers. Keeping in mind that MERLIN "removes" already discovered IP addresses from the static list, this phenomenon is quite logical: seeds are just used as a new point of departure for recursion but relevant and independent seeds (i.e., those allowing to discover new large connected components) are quickly consumed.

In practice, to illustrate this degradation, in less than a third of the probing time, half of routers are discovered on all vantage points although the static list is differently sorted on each vantage point. However, In addition, we also observed that during the first hours of the probing period (mainly dependent on the recursion mode), it is likely that each vantage point discovers a common part of the global topology whereas the last hours of the campaign allow them to find isolated and more specific multicast component.

IV. PERFORMANCE ANALYSIS

This section provides statistics and discussions about the MERLIN deployment and the collected data set. We conduct our study using six vantage points distributed across the Internet: Strasbourg (France), Louvain-la-Neuve (Belgium), Napoli (Italy), San Diego (USA), Redwood City (USA), and Hamilton (New Zealand). The main advantage of using these six vantage points is the ability to circumvent IGMP filtering applied on some border routers that limit the scope of `mrinfo` probes. We discuss their utility in a dedicated section (Sec. IV-B). In addition, in contrast to the previous approach, we use a large list of IP addresses as seeds. This list is made of 1,643,005 IP addresses selected as follows:

- 1.2 million addresses from CAIDA's Archipelago traceroute measurements [4],
- 3,580 addresses from known topologies provided by research and educational networks,
- 24,429 addresses from a Tier-1 ISP,
- 155,674 addresses from traceroute, record route, and IP timestamps measurements issued from the Reverse Traceroute system [21] and
- 224,762 addresses that initially responded to `mrinfo-rec` probes using the four previous datasets.

Data has been collected with several runs, between July 9th 2010 and July 29th 2010. All data collected has been merged into a single super dataset focusing on relevant and unique information. This dataset gives us 480,000 IP addresses aggregated into almost 50,000 routers scattered in more than 3,000 ASes. The raw data collected is available online at `http://inl.info.ucl.ac.be/content/mrinfo`.

In this section, we first report our efforts to cross-validate the data contained in responses to MERLIN probes. At issue is the frequency of responses that contain addresses that belong to other routers; these addresses might be stale, owing to interfaces being configured with an address that is later shifted to another router, or be anycast addresses. We test the interface addresses returned with *Ally* [9] and *Mercator* [22] probes. Ally infers aliases if a sequence of probes sent to alternating IP addresses yields responses with incrementing, interleaved

IP-ID values. Mercator infers aliases when a router responds with a different source address than that probed. More recent tools for alias resolution [10], [23] are more appropriate for constructing a complete router-level graph; Ally lets us carefully probe addresses with a high probability of being aliases without inducing rate limiting.

We tested 41,224 routers; the set consists of routers that reported at least two addresses not in RFC 1918 prefixes. We were unable to obtain information with Ally or Mercator for 6,135 (14.9%) routers that would allow us to judge the MERLIN response. Of the 35,089 MERLIN routers that we did test, 28,003 (79.8%) were in complete agreement with Ally and/or Mercator techniques. A further 6,747 (16.4%) routers did not have conflicting alias resolution data, but we did not obtain a response for all interfaces. In total, 913 (2.6%) of MERLIN routers had some conflicting alias resolution data.

This cross-validation analysis shows us that data collected with MERLIN is highly consistent with results coming from Ally or Mercator. The cases of disagreement comes from a combination of Ally's limitations (assuming a shared counter when the counter could be scoped to individual line cards), and assumptions about addresses mapped to a single router: most of those conflicts seem to be due to stale configurations generating pseudo-anycast addresses.

In the following, we first evaluate the completeness of the collected dataset (Sec. IV-A). Then we investigate the importance of each vantage point (Sec. IV-B).

A. Completeness

The "completeness" of MERLIN is quantifiable based on two axes: (i) the *proportion of multicast routers* (Sec. IV-A.1), in which we estimate a lower bound for the proportion of the Internet that is MERLIN compliant, and, (ii), the *proportion of multicast interfaces* (Sec. IV-A.2), in which we examine the ability of MERLIN to return a complete set of interfaces for a given router.

In the following, we assume that a multicast router r reports the same list of interfaces whatever the choice of the targeted IP address as long as it belongs to r. Second, a given list of multicast interfaces belonging to the same router may appear several times. Obviously, we do not consider more than one instance of this list, unless the responding source IP of the reply is not contained in the returned list of multicast interfaces. Indeed, in this case, the source IP address can be added to the router as a "purely unicast interface". Those cases serve as a basis to quantify the completeness of MERLIN as described in Sec. IV-A.2.

1) Proportion of Multicast Routers: Without having a complete knowledge of the Internet topology, it is difficult to estimate which proportion of the network responds to MERLIN, e.g., the probing coverage of MERLIN. In this section, we try to provide a lower bound of this proportion according to our list of seeds and our set of vantage points.

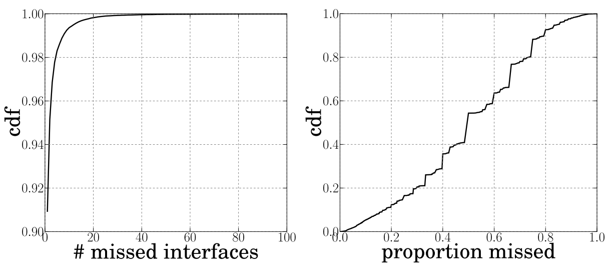
Our global static list for seeding MERLIN is made of 1,643,005 IP addresses. Among these targets, 1,223,715 IP addresses come from the Archipelago dataset [4]. Assuming that this "hitlist" is representative of the active Internet space (e.g., they are well distributed across the Internet), we can establish a rough approximation of the multicast coverage in the Internet. Note that this hitlist results from an active traceroute measurement phase filtered to mainly focus on active backbone IP addresses (belonging to routers): there does not exist any reasons that such a hitlist favors or disfavors the presence of multicast enabled interfaces.

Looking at the intersection between the previous `mrinfo-rec` campaign and the 1.2M IP addresses coming from the Archipelago dataset, we retrieve 61,988 IP addresses in common. Thus, reported to the Archipelago dataset, both lists share a common subset greater than 5% of the hitlist. If one considers that the Archipelago hitlist is representative of the whole Internet backbone, one could say that, at least, 5% of the active Internet address space supports multicast. This value is a lower bound for two reasons: (i) some multicast routers may not respond to IGMP probes, and (ii), the use of a limited number of vantage points is not sufficient to avoid all IGMP filtering (see Sec. IV-B). Among those 5%, it is worth noticing that the coverage of an IGMP probing campaign is not uniformly distributed across the AS level graph. Indeed, some stub AS supporting multicast may be entirely discovered whereas others will remain completely hidden. This "black or white effect" can also appear on top Tier AS due to IGMP filtering.

Moreover, we can notice that 162,774 IP addresses (224,762 - 61,988), i.e., almost 12% of both datasets ($\frac{162{,}774}{162{,}774+1{,}223{,}715}$), is reported only by `mrinfo-rec` while the Archipelago dataset contains a specific subset of 83%. Finally, the use of MERLIN with the whole seeding list allows us to additionally collect 255,238 IPs (480,000 - 224,762) not discovered by `mrinfo-rec`. Those two observations emphasize the efficiency of MERLIN and its complementarity with traceroute based campaigns.

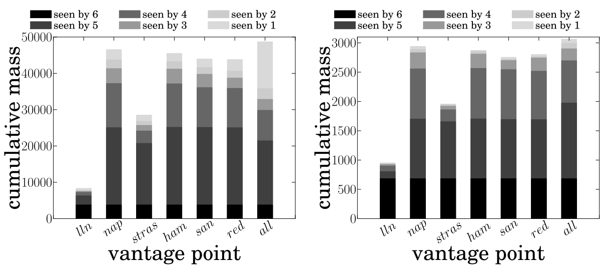
2) Proportion of multicast interfaces: Here we evaluate the ability of MERLIN to return a complete set of interfaces for a given multicast router: MERLIN being a multicast tool, by definition, it will only report its multicast interfaces and adjacencies. In practice, a multicast router can be configured at the interface granularity: each interface can independently support multicast. Furthermore, an ISP may decide to enable multicast only on a given portion of its network. Nevertheless, an ISP supporting IP multicast should enable multicast everywhere in its network to ensure the correct PIM tree establishment. Only two exceptions may arise: inter-area border routers and AS border routers. An area border router does not need to support multicast adjacencies with routers belonging to non multicast areas. Between AS, the BGP routing protocol can use specific multicast forwarding entries to disseminate PIM messages. Thus, it is likely that a multicast border router will not enable multicast on all its interfaces.

In this section, we try to quantify those missed unicast interfaces. Although MERLIN does not report purely unicast interfaces of the probed router (they do not appear in the interface list), a router can answer via an unicast interface: this IP address is then contained in the source IP field of the response. Thus, we are able to provide a lower bound on the quantity of missing interfaces by using source IP addresses



(a) Raw number (b) Proportion

Fig. 5. Interfaces missed for responding routers



(a) routers discovered (b) ASes discovered

Fig. 6. Vantage point utility

(the index of replies) not belonging to the reported multicast alias. The static list containing both multicast and unicast interfaces, and keeping in mind that MERLIN does not avoid the reprobing of a given router indexed on a non-reported IP address, we are able to estimate a lower bound on the number of occurrences of such cases.

Fig. 5(a) provides the cumulative distribution of the number of missed interfaces per router: they correspond to purely unicast interfaces present in the static list (they are not reported in the multicast alias but we can gather them to it if the router is able to respond through them). In at least 9% of the cases, it seems that MERLIN is not able to collect the entire alias. The largest number of missed interfaces for a single router we faced during our measurements is 88. All interfaces falling in those 9% are unicast interfaces not reported by MERLIN replies in the set of multicast interfaces of a router. If those interfaces were not present in the static list, they would have been missed. Looking at Fig. 5(a), we observe that for most of these cases, less than ten interfaces are missing and can be reported as purely unicast.

To better understand the situation, we also plot in Fig. 5(b) the relative proportion of missing interfaces, i.e., the number of purely unicast interfaces compared to the total number of IP addresses (both virtually added unicast and reported multicast). We note that this relative lack is uniformly distributed across the 9% of impacted routers: whatever the level of loss, the occurrence probability remains roughly equal.

B. Importance of Vantage Point

The goal of this section is to emphasize the importance of using several vantage points to avoid IGMP filtering by intermediate networks. A MERLIN probe may be dropped on the forward path, and a IGMP response may also be filtered on the reverse path if the return path differs. Note that there exist two kinds of IGMP filtering behaviors: a multicast router may drop a MERLIN query addressed to it (*local filtering*) or it may drop any MERLIN queries going through it (*transit filtering*). While the local filtering concerns individual routers, transit filtering is more challenging: all requests following a path containing such a filtering router are dropped. In practice, we can distinguish three cases: either a router does not apply IGMP filtering at all, or it only applies local filtering, or both local and transit filtering (we assume that cases where routers only apply transit filtering make no sense). Hence, the use of multiple independent vantage points may allow us to increase the MERLIN coverage. Indeed, some non-filtering routers unreachable via a given vantage point (due to the transit filtering of others) may become reachable through another independent vantage point. More precisely, the term "independent" is related to the AS level graph location of the vantage points: considering a given target r, the more the forwarding paths between the vantage points and r differ, the more likely it is to reduce the impact of IGMP filtering and to increase MERLIN coverage.

Fig. 6(a) shows the utility of each vantage point. For each vantage point, we plot the absolute quantity of routers it discovers, and how many vantage points observe each router. The individual stacks reflect the utility of each vantage point, and the stack labeled "all" refers to the global utility of the union of routers discovered via all vantage points. Interestingly, each vantage point is able to discover between 1,000 and 3,000 unique routers (i.e., they cannot be seen by other vantage points). For the complete set of routers discovered, 30% are discovered by individual vantage points. This proportion is higher than the 15% of routers that belong to the total intersection ("seen by 6"). This first result highlights the importance of each vantage point: their individual utility cannot be considered as marginal.

Moreover, we can also understand the importance of each vantage point independently. From Fig. 6(a), we notice that the Napoli vantage point is the most efficient, directly followed by the ones in New Zealand, San Diego, and Redwood City. On the contrary, Louvain-la-Neuve and Strasbourg are clearly more subject to IGMP filtering. In all cases, the relative proportion of "seen by n" is roughly uniform among the set of vantage points. It seems that the total number of routers seen through a given vantage point is a sufficient information to understand the importance of a vantage point: each vantage point brings an almost constant number of unique routers while the robustness it provides (routers seen through n points, with $1 < n < 6$) mostly depends on the total of routers it discovers.

We can interpret those results as follows: generally, all vantage points are able to discover routers belonging to Tier-1 ("seen by 6 and 5", because there exist non-IGMP filtered paths between the vantage points and Tier-1 ASes). Their success in probing the global Internet depends on the inter-domain forwarding and filtering policies induced by their providers connectivity. Generally, the further the target, the more likely a filter: each vantage point improves the global view due to its ability to better discover the AS graph portion around it.

However, Fig. 6(b) mitigates this first observation. The proportion of AS discovered through only one vantage point is quite low compared to the respective proportion using a per router perspective (Fig. 6(a)). Indeed, the ASes "seen by 1" correspond to small stub or Transit ASes not containing many IPs. If one considers the multicast part of an AS as a connected graph, the recursion should allow us to discover this entire graph. However, if some multicast routers do not respond to MERLIN (their OS does not activate IGMP capabilities for public users), the recursion may stop facing this "wall". Even if the static list contains an IP address belonging to the multicast component located on the other side of this wall, the forwarding path used to reach it may be subject to filtering policies. Each vantage point is subject to different filtering policies according to the prefix containing the target.

To conclude, the utility of multiple vantage points using MERLIN is completely different from the one of a tool such as `traceroute` [24]. The utility of using multiple vantage points decreases according to the number of used locations. However, it does not quickly become marginal as each vantage point continues to provide a constant and unique capacity to probe its close environment (Stub and Transit AS within a low number of hops). Further, each vantage point is able to reach prefix subsets of larger AS (Tier-2 and Tier-1 ASes) thanks to specific paths allowing it to circumvent IGMP filtering of its other providers. Each vantage point can take benefit of its unique situation in the AS level graph to reveal a specific area. In practice, MERLIN should be deployed on several locations well spread around the global AS level graph, and piloted in a way that favors the discovery of new responding routers.

V. CONCLUSION

In this paper, we discussed the implementation, the deployment and the validation of MERLIN, a new tool for discovering the Internet topology at the router level. MERLIN, based on a multicast management tool called `mrinfo`, comes with the strong advantage of listing all IPv4 multicast interfaces of a router and its links towards its neighbors. On the one hand, MERLIN fixes bugs and limitations inherent to `mrinfo`. On the other hand, MERLIN is designed to offer a configurable tradeoff between efficiency and network friendliness. The data collected with MERLIN can be used for performing typical topology studies [14], [15].

We deployed MERLIN on six machines spread around the world and evaluated its performance. We highlighted the importance of using multiple vantage points in order to circumvent IGMP filtering. In addition, we validated and evaluated the completeness of MERLIN: we first perform a cross-validation on reported alias, and we investigate the proportion of multicast enabled interfaces and routers in the Internet. Future work should reveal how we can guide MERLIN vantage points from a coordinating entity in order to improve its coverage while, at the same time, limiting the probing redundancy.

ACKNOWLEDGEMENTS

This work has been partially supported by the European funded 216372 Trilogy project. Mr. Donnet's work is supported by the FNRS (Fonds National de la Recherche Scientifique, rue d'Egmont 5 – 1000 Bruxelles, Belgium.).

REFERENCES

[1] B. Donnet and T. Friedman, "Internet topology discovery: a survey," *IEEE Communications Surveys and Tutorials*, vol. 9, no. 4, pp. 2–15, December 2007.

[2] H. Haddadi, G. Iannaccone, A. Moore, R. Mortier, and M. Rio, "Network topologies: Inference, modeling and generation," *IEEE Communications Surveys and Tutorials*, vol. 10, no. 2, pp. 48–69, April 2008.

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

[4] k. claffy, Y. Hyun, K. Keys, M. Fomenkov, and D. Krioukov, "Internet mapping: from art to science," in *Proc. IEEE Cybersecurity Applications and Technologies Conference for Homeland Security (CATCH)*, March 2009.

[5] Y. Shavitt and E. Shir, "DIMES: Let the internet measure itself," *ACM SIGCOMM Computer Communication Review*, vol. 35, no. 5, October 2005.

[6] H. V. Madhyastha, T. Isdal, M. Piatek, C. Dixon, T. Anderson, A. Krishnamurthy, and A. Venkataramani, "iPlane: An information plane for distributed services," in *Proc. USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, November 2006.

[7] M. Luckie, "Scamper: a scalable and extensible packet probet for active measurement of the Internet," in *Proc. USENIX/ACM Internet Measurement Conference (IMC)*, November 2010.

[8] R. Braden, "Requirements for Internet hosts – communication layers," Internet Engineering Task Force, RFC 1122, October 1989.

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

[10] A. Bender, R. Sherwood, and N. Spring, "Fixing Ally's growing pains with velocity modeling," in *Proc. ACM/USENIX IMC*, October 2008.

[11] M. H. Gunes and K. Sarac, "Resolving IP aliases in building traceroute-based internet maps," *IEEE/ACM Transactions on Networking (ToN)*, vol. 17, pp. 1738–1751, december 2009.

[12] ——, "Importance of IP alias resolution in sampling Internet topologies," in *Proc. IEEE Global Internet Symposium*, May 2007.

[13] B. Donnet, P. Raoult, T. Friedman, and M. Crovella, "Deployment of an algorithm for large-scale topology discovery," *IEEE Journal on Selected Areas in Communications (JSAC), Sampling the Internet: Techniques and Applications*, vol. 24, no. 12, pp. 2210–2220, Dec. 2006.

[14] J.-J. Pansiot, P. Mérindol, B. Donnet, and O. Bonaventure, "Extracting intra-domain topology from `mrinfo` probing," in *Proc. Passive and Active Measurement Conference (PAM)*, April 2010.

[15] P. Mérindol, B. Donnet, O. Bonaventure, and J.-J. Pansiot, "On the impact of layer-2 on node degree distribution," in *Proc. ACM/USENIX Internet Measurement Conference (IMC)*, November 2010.

[16] D. Dugal, C. Pignatoro, and R. Dunn, "Protecting the router control plane," Internet Engineering Task Force, Internet Draft (Work in Progress) draft-ietf-opsec-protect-control-plane-03, August 2010.

[17] T. Pusateri, "Distance vector multicast routing protocol version 3 (DVMRP)," Internet Engineering Task Force, Internet Draft (Work in Progress) draft-ietf-idmr-dvmrp-v3-11, October 2003.

[18] P. Sharma, E. Perry, and R. Malpani, "IP multicast operational network management: Design, challenges, and experiences," *IEEE Network*, vol. 17, no. 2, pp. 49–55, March 2003.

[19] S. Deering, "Host extensions for IP multicasting," Internet Engineering Task Force, RFC 1112, August 1989.

[20] P. Mérindol, B. Donnet, J.-J. Pansiot, M. Luckie, and Y. Hyun, "MERLIN: MEasure the Router Level of the INternet," Université catholique de Louvain, Technical Report 2010-3, September 2010.

[21] E. Katz-Bassett, H. V. Madhyastha, V. K. Adhikari, C. Scott, J. Sherry, P. van Wesep, T. Anderson, and A. Krishnamurthy, "Reverse traceroute," in *Proc. USENIX NSDI*, June 2010.

[22] R. Govindan and H. Tangmunarunkit, "Heuristics for internet map discovery," in *Proc. IEEE INFOCOM*, March 2000.

[23] K. Keys, Y. Hyun, and M. Luckie, "Internet-scale alias resolution with MIDAR," February 2010, ISMA Workshop on Active Internet Measurements (AIMS).

[24] P. Barford, A. Bestavros, J. Byers, and M. Crovella, "On the marginal utility of network topology measurements," in *Proc. ACM SIGCOMM IMW*, Nov. 2001.