Alias Resolution Based on ICMP Rate Limiting

Kevin Vermeulen, Burim Ljuma, Vamsi Addanki, Matthieu Gouel, Olivier Fourmaux, Timur Friedman, Reza Rejaie

Alias Resolution Based on ICMP Rate Limiting

Kevin Vermeulen[1], Burim Ljuma[1], Vamsi Addanki[1], Matthieu Gouel[1], Olivier Fourmaux[1], Timur Friedman[1], and Reza Rejaie[2]

[1] Sorbonne Université
[2] University of Oregon

Abstract. Alias resolution techniques (e.g., MIDAR) associate, mostly through active measurement, a set of IP addresses as belonging to a common router. These techniques rely on distinct router features that can serve as a signature. Their applicability is affected by router support of the features and the robustness of the signature. This paper presents a new alias resolution tool called Limited Ltd. that exploits ICMP rate limiting, a feature that is increasingly supported by modern routers that has not previously been used for alias resolution. It sends ICMP probes toward target interfaces in order to trigger rate limiting, extracting features from the probe reply loss traces. It uses a machine learning classifier to designate pairs of interfaces as aliases. We describe the details of the algorithm used by Limited Ltd. and illustrate its feasibility and accuracy. Limited Ltd. not only is the first tool that can perform alias resolution on IPv6 routers that do not generate monotonically increasing fragmentation IDs (e.g., Juniper routers) but it also complements the state-of-the-art techniques for IPv4 alias resolution. All of our code and the collected dataset are publicly available.

1 Introduction

Route traces obtained using `traceroute` and similar tools provide the basis for generating maps that reveal the inner structure of the Internet's many autonomously administered networks, but not necessarily at the right level of granularity for certain important tasks. Designing network protocols [42] and understanding fundamental properties of the Internet's topology [18] are best done with router-level maps. Rather than revealing routers, `traceroute` only provides the IP addresses of individual router interfaces. The process of grouping IP addresses into sets that each belong to a common router is called *alias resolution*, and this paper advances the state of the art in alias resolution.

A common approach to alias resolution is to send probe packets to IP addresses, eliciting reply packets that display a feature that is distinctive enough to constitute a signature, allowing replies coming from a common router to be matched. This paper describes a new type of signature based upon a functionality, *ICMP rate limiting*, in which an Internet-connected node (router or end-host) limits the ICMP traffic that it sends or receives within a certain window of time. This new signature enjoys much broader applicability than existing ones for IPv6

alias resolution, thanks to ICMP rate limiting being a required function for IPv6 nodes. The signature also complements IPv4 existing signatures.

Our contributions are: (1) The Limited Ltd. algorithm, a new signature-based alias resolution technique that improves alias resolution coverage by 68.4% on Internet2 for IPv6 and by 40.9% on SWITCH for IPv4 (2) a free, open source, and permissively licensed tool that implements the algorithm.

We evaluate Limited Ltd. by comparing its performance to two state-of-the-art alias resolution tools: Speedtrap [29] for IPv6, and MIDAR [26] for IPv4, using ground truth provided by the Internet2 and SWITCH networks.

The remainder of this paper is organized as follows: Sec. 2 provides technical background and related work for both alias resolution and ICMP rate limiting. Sec. 3 describes the Limited Ltd. technique in detail. Sec. 4 presents the evaluation. Sec. 5 discusses ethical considerations and Sec. 6 summarizes our conclusions and points to future work.

2 Background and Related Work

Limited Ltd. is the latest in a long line of alias resolution methods stretching back over twenty-plus years. An inventory of all previously known techniques (Table 1) shows that there are only four techniques known to work for IPv6. Of these, there is a publicly-available tool for only one: Speedtrap [29]. But Speedtrap has a known limitation of only working on routers that generate monotonically increasing IPv6 fragmentation IDs, whereas there is an entire class of routers, such as those from Juniper, that do not generate IDs this way. Relying upon monotonically increasing IP IDs for IPv4, as does state-of-the-art MIDAR [26], presents a different issue: fewer and fewer routers treat IPv4 IP IDs this way due to a potential vulnerability [15,2]. Limited Ltd. is a publicly available tool that does not rely upon monotonically increasing IDs, thereby enabling IPv6 alias resolution on Juniper routers for the first time and IPv4 alias resolution on a growing class of routers for which MIDAR will no longer work.

Regarding ICMP, the Internet Control Message Protocol: its IPv4 and IPv6 variants [34,13] allow routers or end-hosts to send error and informational messages. The RFC for ICMPv6 [13] cites the "bandwidth and forwarding costs" of originating ICMP messages to motivate the need to limit the rate at which a node originates ICMP messages. It also recommends the use of a token bucket mechanism for rate limiting. It explicitly calls for compatibility with `traceroute` by stating that "Rate-limiting mechanisms that cannot cope with bursty traffic (e.g., traceroute) are not recommended". Furthermore, it states that, in the case of "ICMP messages [being] used to attempt denial-of-service attacks by sending back to back erroneous IP packets", an implementation that correctly deploys the recommended token bucket mechanism "would be protected by the ICMP error rate limiting mechanism". The RFC makes ICMP rate limiting mandatory for all IPv6 nodes. ICMP rate limiting is a supported feature on all modern routers but its implementation may vary by vendor [14,12,11,9,23,20,24,22] based on ICMP message type and IP version. ICMP rate limiting can be performed on incoming

Year	Basis (**s**) = signature (**t**) = topology (**o**) = other	Algorithms and tools	Condition of applicability	IPv4	IPv6
				(***τ***) = tool (***δ***) = dataset	
1998 [32]	source IP address (**s**)	Pansiot and Grad [32]	respond with a common IP address in ICMP Destination Unreachable messages	yes	
		Mercator [16]		yes (***τ***) (***δ***)	
2002 [40]	IP ID (**s**)	*Ally* [40]	send replies with a shared IP ID counter that increases monotonically with each reply	yes (***τ***)	
		RadarGun [7]		yes (***τ***)	
		MIDAR [26]		yes (***τ***) (***δ***)	
2002 [40]	Reverse DNS (**o**)	*Rocketfuel* [40] AROMA [28]	IP address resolves to a name	yes	yes
2006 [17]	`traceroute` (**t**)	APAR [17]	respond with ICMP Time Exceeded messages	yes	
		kapar [25]		yes (***τ***) (***δ***)	
2010 [38]	IP Prespecified Timestamp option (**s**)	Sherry et al. [38] *Pythia* [30]	fill in timestamps as specified by the option	yes	
2010 [36]	IPv6 source routing (**s**)	Qian et al. [36,35]	source routing must be enabled		yes
2013 [29]	IPv6 fragmentation identifier (**s**)	*Speedtrap* [29]	IDs elicited from responses increase monotonically		yes (***τ***) (***δ***)
2013 [39]	IP Record Route option (**t**)	*DisCarte* [39]	fill in IP addresses as specified by the option	yes	
2015 [31]	IPv6 unused address (**s**)	Padmanabhan et al. [31]	126 prefixes on a point to point link		yes
2019	ICMP rate limiting (**s**)	*Limited Ltd.*	ICMP rate limiting shared by interfaces of the router	yes (***τ***) (***δ***)	yes (***τ***) (***δ***)

Table 1: Alias resolution methods

traffic or generated replies. Limited Ltd. makes no distinction between the two. It works whenever multiple interfaces of a router are subject to a common ICMP rate limiting mechanism, i.e., when there is a shared token bucket across multiple interfaces. Vendor documentation [23,20,24,11], indicates that `ping` packets are more likely to trigger shared ICMP rate limiting behavior. We validated this observation in a prior survey and in a lab environment. In particular on Juniper (model J4350, JunOS 8.0R2.8), we observed a shared ICMP rate limiting mechanism for Echo Reply, Destination Unreachable and Time Exceeded packets across all of its interfaces by default. But on Cisco (model 3825, IOS 12.3), we observed that the rates for Time Exceeded and Destination Unreachable packets are limited on individual interfaces by default, and only the rate for Echo Reply packets is shared across different interfaces [10]. Therefore, we adopted the `ping` Echo Request and Echo Reply mechanism in our tool to maximize the chances of encountering shared ICMP rate limits across router interfaces.

A few prior studies have examined ICMP rate limiting behavior in the Internet. Ravaioli et al. [37] identified two types of behavior when triggering ICMP rate limiting of Time Exceeded messages by an interface: on/off and non on/off. Alvarez et al. [4] demonstrated that ICMP Time Exceeded rate limiting is more widespread in IPv6 than in IPv4. Guo and Heidemann [19] later proposed an algorithm, FADER, to detect ICMP Echo Request/Reply rate limiting at very low probing rates, up to 1 packet per second. They found rate limiting at those rates for very few /24 prefixes. Our work is the first one that exploits the shared

nature of ICMP rate limiting across different interfaces of a router as a signature to relate these interfaces for alias resolution.

3 Algorithm

The main intuition behind our approach is that two interfaces of a router that implements shared ICMP rate limiting, should exhibit a similar loss pattern if they are both probed by ICMP packets at a cumulative rate that triggers rate limiting. The key challenges are to efficiently trigger rate limiting and reliably associate aliases based on the similarity of their loss patterns despite the noise due to independent losses of probes and replies.

Pseudo code 1 describes how Limited Ltd. divides a set of input IP addresses into subsets that should each be an alias set. It proceeds iteratively, taking the following steps in each iteration: First, a random IP address from the input set is selected as a *seed*, with all remaining members of the input set being *candidate* aliases for the seed. The seed is probed at incrementally higher rates until the rate r_s that induces ICMP rate limiting is identified (`find_rate`()). Then, the seed is probed at that rate of r_s while all of the candidates interfaces are simultaneously probed at low rates. All probing takes place from a single vantage point. Loss traces for reply packets from the seed and each of the candidate interfaces are gathered. It is very challenging to infer that two interfaces are aliases by directly correlating their loss traces. Instead, the algorithm extracts a set of features from each loss trace and collectively uses these as the signatures of the corresponding interfaces(`signatures`()). Using a classification technique (`classify`()), the algorithm examines whether the signatures of candidate and seed are sufficiently similar to classify them as aliases, in which case the candidate is added to an alias set (A_s). Each identified alias set is refined through further testing in order to reduce the chance of false positives (`refine`()). Finally, the alias set is removed from the input set, and iterations continue until the input set is empty. The remainder of this section further details these steps.

3.1 Triggering ICMP rate limiting

The goal of `find_rate`(s) is to efficiently determine r_s, the probing rate that triggers ICMP rate limiting at the router to which seed s belongs. It proceeds by probing the seed with ICMP Echo Request probes across multiple rounds, increasing the probing rate with each round until the loss rate of observed ICMP Echo Replies enters a target range. The target loss range should be sufficiently large to minimize the effect of random independent losses and also relatively small to minimize the load on the router. To satisfy these two opposing conditions, we empirically set the range at 5 to 10%. The probing rate remains constant during each round. The rate is low (64 pps) for the first round, and exponentially increases in consecutive rounds until the loss rate falls within (or

Algorithm 1 `Limited Ltd.`

Input
  S: a set of IP addresses
Output
  A: a set of alias sets
$K \leftarrow$ `controls`(S) : set of controls
$A \leftarrow \emptyset$
while $S \neq \emptyset$ **do**
  choose at random a seed $s \in S$
  $C_s \leftarrow S \setminus \{s\}$: candidate aliases for s
  $r_s \leftarrow$ `find_rate`(s) : rate limiting rate for s
  $\Sigma_s \leftarrow$ `signatures`(s, r_s, C_s, K) : set of signatures
  $A_s \leftarrow \{s\}$: alias set for s
  for $c \in C_s$ **do**: for each candidate c
    $\sigma_{s,c} \in \Sigma_s$ is the pairwise signature for s and c
    if `classify`$(\sigma_{s,c}) ==$ `true` **then**
      s and c are aliases
      $A_s \leftarrow A_s \cup \{c\}$: add c to the alias set
  $A_s \leftarrow$ `refine`(A_s) : try to reduce false positives
  $A \leftarrow A \cup \{A_s\}$: add the new alias set to A
  $S \leftarrow S \setminus A_s$: remove the aliases of s from S
return A

exceeds) the target range.[3] If the observed loss rate is within the target range, the probing is concluded and the last rate is reported as r_s. But if the loss rate is higher than the target range, up to eight additional rounds are launched in a binary search between the last two rates. If the loss rate still does not fall within the target range, the probing rate that generates the loss rate closest to the range is chosen. If the target loss range is not reached as the probing reaches a maximum rate (32,768 pps), the probing process ends without any conclusion. The duration of each round of probing should be sufficiently long to reliably capture the loss rate while it should also be limited to control the overhead of probing. We experimentally set the duration of each round of probing to 5 seconds, followed by a period, of equal length, of no probing. The right plot of Fig. 1 presents the CDF of the number of probing rounds to trigger the target loss rate for thousands of IPv4 and IPv6 interfaces (using our dataset from Sec. 3.3). We observe that for 90% of IPv4 or IPv6 interfaces, the ICMP rate limiting is triggered in less than 8 rounds of probing. The left plot of Fig. 1 shows the CDF of the probing rate that triggered the target loss rate (i.e., the inferred rate for triggering the ICMP rate limiting) across the same IPv4 and IPv6 interfaces. This figure indicates that for 70% (80%) of IPv6 (IPv4) interfaces, ICMP rate limiting is triggered at less than 2k pps. This result confirms that our selected min and max probing rate covers a proper probing range for

[3] We have explicitly verified that the actual probing rate is not limited by the network card or other factors.

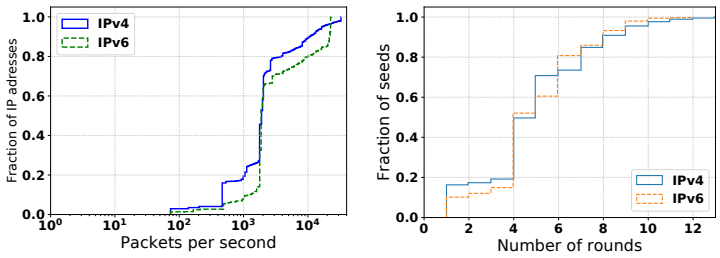


Figure 1: CDF of the probing rate r_s (left) and the number of probing rounds (right) to trigger ICMP rate limiting for 2,277 IPv4 and 1,099 IPv6 addresses.

more than 99% of interfaces. We note that the binary search process failed to reach the target loss rate for fewer than 1% of the interfaces. All the parameters of our probing strategy are empirically determined. Section 5 elaborates on the ethical considerations associated with the probing scheme.

3.2 Generating interface signatures

A signature based on the loss traces of individual interfaces is obtained by probing the seed interface at its target rate (r_s) while simultaneously probing each candidate interface at the low rate of R_c pps. Probing a large number of candidate interfaces in each round may lead to a better efficiency, but the aggregate probing rate should remain low so that it does not independently trigger ICMP rate limiting even if all those candidates are in fact aliases. To address these two constraints, we set the number of candidate interfaces that are considered in each round to 50 and R_c to 10 pps. In an unlikely scenario that all of these 50 candidate interfaces are aliases, this strategy leads to a 500 pps probing rate for the corresponding router that does not trigger ICMP rate limiting in 90% of routers, as we showed in the left plot of Fig. 1.[4]

Control Interface. In order to distinguish the observed losses in the loss traces for the target interfaces (i.e., seed s and individual candidate c) that are not related to ICMP rate limiting, we also consider another interface along the route to each target interface and concurrently probe them at a low rate (10 pps). These interfaces are called the *controls*, κ_s and κ_c. The control κ_i for target interface i is identified by conducting a Paris Traceroute [6] towards i and selecting the last responsive IP address prior to i.[5] The loss rate for κ_i also forms part of i's signature. In practice, the *controls* are identified at the beginning of the Limited

[4] The largest reported alias set by MIDAR and Speedtrap has 43 interfaces. Therefore, the likelihood of observing 50 candidate interfaces that are all aliases is low.

[5] Limited Ltd. maintains the flow ID necessary to reach κ_s in subsequent probing of s and κ_s.

	Seed s	Candidate c	Control κ_s	Control κ_c
Loss rate	x	x	x	x
Change point	x	x		
`gap` → `gap` transition probability	x	x		
`burst` → `burst` transition probability	x	x		
Pearson correlation coefficient	x			

Table 2: Selected features for a Signature.

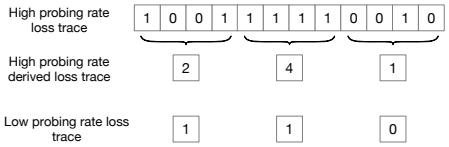


Figure 2: Mapping between loss traces with different length.

Ltd. procedure by conducting route traces to all IP addresses in the input set S. This corresponds to `controls`() and K is the resulting set of controls.

Inferring Alias Pairs. The above probing strategy produces a separate loss trace for each interface. We have found that when losses occur simultaneously at pairs of alias interfaces, they can do so in multiple ways, as the five examples in Fig. 4 illustrate. The black and white strokes in each trace correspond respectively to received and lost ICMP Echo Replies, and their varied patterns defy attempts to find simple correlations. We therefore use a machine learning classifier to identify pairs of aliases. It is based on the following features extracted from loss traces that, intuitively, we believe capture the temporal pattern of the losses in each trace. (See also Table 2.)

1. Loss rate: This is simply the number of losses in the trace divided by the total number of probes in the trace.
2. Change point detection: This is the point in a time series (such as our loss traces) when the probability distribution of a time series changes [5]. We adopt a method based on the variation of the mean and the variance [27].
3. Transition probabilities: These are obtained by using each loss trace to train a Gilbert-Elliot two-state Markov model, in which losses occur in the `burst` state and no losses occur in the `gap` state. The P(`gap` → `gap`) and P(`burst` → `burst`) transition probabilities are sufficient to fully describe the model since other two probabilities can be easily calculated from these. For example, P(`gap` → `burst`) = 1 - P(`gap` → `gap`).
4. Correlation coefficient: The Pearson correlation coefficient between the two loss traces is used as a measure of similarity between them. Calculating this coefficient requires both time series to have the same number of values but our loss traces do not meet this condition since we use a higher probing rate for the seed. To address this issue, we condition the seed's loss trace to align it with the loss trace of other interfaces as shown in Fig. 2. In this example, the length of the loss trace of the seed is four times longer than the ones from the other interfaces. We consider groups of four consecutive bits in the seed loss trace and convert it to the sum of the 1's. The resulting loss trace has a lower rate and can be directly correlated with other loss traces.

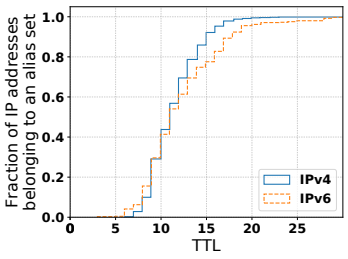


Figure 3: CDF of the TTL distance from the Limited Ltd. vantage point of the IP addresses belonging to an alias set in our training data.

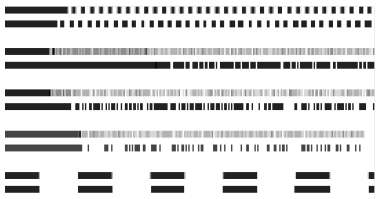


Figure 4: Raw times series of loss traces of pairs of aliases.

3.3 Classifying the signatures

We use the *random forest* classifier from the `scikit-learn` Python machine learning library [33]. If it identifies two interfaces as aliases based on their signatures, `classify`() returns `true`; otherwise, `false`. There are several challenges to building such a classifier: (1) it must learn from training data that represents the diversity of possible loss traces generated by pairs of aliases; (2) it should be able to distinguish between losses triggered by ICMP rate limiting and unrelated losses; (3) it should have a high precision, so that Limited Ltd. minimizes false positives; and (4) if the training data come from other alias resolution techniques, such as MIDAR and Speedtrap, it must be able to generalize to pairs that they cannot find. We tackled these challenges as follows.

Training and testing data. We have access to ground truth router-level topology for two networks, Internet2 and SWITCH, but these do not suffice to capture the richness of router behaviors in the Internet as a whole. We therefore randomly selected routable IPv4 and IPv6 prefixes from the RIPE registry [3], and conducted multipath Paris Traceroute [41] from PlanetLab Europe [1] nodes towards the first address in each prefix. This procedure yielded 25,172 IPv4 addresses in 1,671 autonomous systems (ASes) and 18,346 IPv6 addresses in 1,759 ASes from 6,246 and 4,185 route traces, respectively. We use MIDAR and Speedtrap to identify IPv4 and IPv6 alias sets, respectively, since both tools are known to have low false positive rates. Pairs of interfaces from these sets are used as labeled as `true`. For the `false` labels, we take the conservative approach of selecting pairs of IP addresses that are more than 6 hops from each other in a given route trace. The 6 hop value is empirically set, as 99.9% of the alias pairs identified by MIDAR and Speedtrap are fewer than 6 hops apart. This labeling process identified 70,992 unique IPv4 and 7,000 unique IPv6 addresses. 15,747 of IPv4 and 1,616 IPv6 addresses are labeled as aliases forming 2,277 IPv4 and 1,099 IPv6 alias sets, respectively. Fig. 3 shows the CDF of hop count

	IPv4			IPv6		
	Precision	Recall	F1 score	Precision	Recall	F1 score
Random forest	0.990	0.499	0.652	0.992	0.647	0.782
Multilayer perceptron	0.993	0.431	0.591	0.978	0.641	0.769
KNN	0.952	0.638	0.764	0.970	0.622	0.756
SVM	0.986	0.478	0.642	0.988	0.599	0.743

Table 3: Classifier performance on our test set averaged over ten training/testings.

	Gini index	
Feature	IPv4	IPv6
loss rate for the candidate c	0.169	0.192
`burst` $\rightarrow$ `burst` transition probability for the candidate c	0.113	0.125
`burst` $\rightarrow$ `burst` transition probability for the seed s	0.101	0.121
Pearson correlation coefficient	0.091	0.109
loss rate for κ_c, the control of the candidate c	0.077	0.104

Table 4: The five most important features of our random forest classifiers.

distance between our vantage point and selected IP addresses and indicates that these targets are 7-17 hops away from the vantage point. For each alias set, one address is chosen at random to play the role of the seed s, and the candidate set is composed of all of the other aliases in the set that are rounded up with some randomly selected non-aliases to make a C_s of size between 2 (minimum one alias and one non-alias) and 50 (our cap for the number of addresses to be simultaneously probed at a low rate). The high rate r_s at which to probe the seed is found through `find_rate`(s), and the signatures are generated through `signatures`(s, r_s, C_s, K).

Note that while our classifier is trained on alias sets identified by alias resolution techniques with known limitations, it is nonetheless able to identify new alias sets. We argue that this is because the training set is sufficiently rich due to its size and random selection of interfaces, providing considerable diversity and heterogeneity of loss traces across aliases. Our evaluation in Sec. 4 confirms this observation and confirms the ability of our technique to generalize patterns in the training dataset, i.e., the fourth aforementioned challenge.

Choice of classifier. We compared the performance of four classifiers that `scikit-learn` library offers, namely random forest, multilayer perceptron, k-nearest neighbors (KNN), and support vector machines (SVM). To this end, we evenly divided our dataset into a training and a test set, and compared these classifiers based on their precision, recall, and F1 score for both IPv4 and IPv6 datasets. Since `true` labels are only provided from aliases identified by MIDAR and Speedtrap, the recall values correspond to the portion of pairs of aliases in our training set that are detectable by both MIDAR and Limited Ltd. (IPv4) or by both Speedtrap and Limited Ltd. (IPv6). Table 3 presents the averaged result of this comparison after performing 10 randomized splits of the training and test sets. All classifiers exhibit relatively good performance. We have decided to use the random forest classifier, which is composed of 500 trees, as it has the highest precision for both IPv4 and IPv6, and the best F1 score for the IPv6 dataset.

Finally, Table 4 shows the five most important features of our random forest classifiers based on the Gini index [8] that describes the weight of individual features in the classifier's decision. This table reveals a few important points.

First, no single feature dominates the classifier's decision, particularly for IPv6. This confirms the complexity of the patterns for relating loss traces of aliases, as they cannot be accurately learned by a small number of features or simple threshold-based rules. Second, this table also illustrates that most of our engineered features are indeed very important in distinguishing loss traces of aliases. Third, the use of κ_c as one of the main features suggests that the classifier distinguishes losses related to rate limiting from other losses.

3.4 Refining the alias set

Independent network loss could accidentally result in classifying unrelated interfaces as aliases, i.e., generating false positives. To reduce the chance of this, Limited Ltd. incorporates a refinement step, `refine(`A_s`)`, that involves repeating `signature()` and `classify()` on the previously-identified alias set A_s. If a candidate c fails to be (re)classified as an alias of the seed s, it is removed from the alias set. This step is repeated until the alias set remains unchanged over two iterations. Sec. 4 evaluates the resulting reduction of false positives.

4 Evaluation

We evaluate Limited Ltd. with regards to its ability (i) to identify alias pairs that state-of-the-art techniques, namely MIDAR and Speedtrap, are unable to identify, and (ii) to maintain a low rate of false positives.

Dataset. We evaluate Limited Ltd. on ground truth data from the Internet2 and SWITCH networks. For Internet2, router configuration files were obtained on 10 April, with measurements conducted on 11 and 12 April 2019. There were 44 files, each corresponding to a single router. All are Juniper routers. The files concern 985 IPv4 and 803 IPv6 addresses/interfaces, from which we removed 436 IPv4 addresses and 435 IPv6 addresses that did not respond to any probes sent by either MIDAR, Speedtrap, or Limited Ltd.. The resulting dataset consists of 6,577 IPv4 and 2,556 IPv6 alias pairs. For SWITCH, a single file was obtained on 3 May, with measurements conducted 3-5 May 2019. The file identified 173 Cisco routers running either IOS or IOS-XR. From the 1,073 IPv4 and 706 IPv6 addresses listed in the file, we removed 121 IPv4 and 29 IPv6 unresponsive addresses. The resulting dataset consists of 4,912 IPv4 and 2,641 IPv6 alias pairs.

Reducing false positives. We computed the distribution of number of rounds for `refine`() to finalize the alias set for each seed in our dataset: For 79% (98%) of all seeds, `refine`() takes 2 (3) more rounds. Note that the minimum of two rounds is required by design (Sec. 3.4) This basically implies that `refine`() only changed the alias set for 20% of the seeds in a single round.

		IPv4			IPv6		
		MIDAR	ltd ltd	MIDAR ∪ ltd ltd	Speedtrap	ltd ltd	Speedtrap ∪ ltd ltd
Internet2	Precision	1.000	1.000	1.000	N/A	1.000	1.000
	Recall	0.673	0.800	0.868	N/A	0.684	0.684
SWITCH	Precision	1.000	1.000	1.000	1.000	1.000	1.000
	Recall	0.090	0.499	0.599	0.384	0.385	0.772

Table 5: Evaluation on ground truth networks.

Results. Table 5 presents the precision and recall of MIDAR, Speedtrap, Limited Ltd., and the union of both tools on IPv4 and IPv6 ground truth data from the Internet2 and SWITCH networks. Note that it is possible for recall from the union of both tools to be greater than the sum of recall values for individual tools, as we observe in the SWITCH results. This arises from the transitive closure of alias sets identified from the two tools that leads to the detection of additional alias pairs. The main findings of Table 5 can be summarized as follows:

1. Limited Ltd. exhibits a high precision in identifying both IPv4 and IPv6 alias pairs from both networks with zero false positives.
2. Limited Ltd. can effectively discover IPv6 aliases that state-of-the-art Speedtrap is unable to find. In the Internet2 network that uses Juniper routers, Limited Ltd. was able to identify 68.4% of the IPv6 alias pairs while Speedtrap was unable to identify any. In the SWITCH network that deploys Cisco routers, Limited Ltd. and Speedtrap show comparable performance by identifying 38.5% and 38.4% of the IPv6 alias pairs, respectively. The results were complementary, with the two tools together identifying 77.2% of the IPv6 alias pairs, a small boost beyond simple addition of the two results coming from the transitive closure of the alias sets found by each tool.
3. Limited Ltd. can discover IPv4 aliases that state-of-the-art MIDAR is unable to find. In the Internet2 network, Limited Ltd. identifies 80.0% while MIDAR detects 67.3% of aliases. In the SWITCH networks, Limited Ltd. identified 49.9% while MIDAR detects only 9.0% of all aliases.

A couple of detailed observations follow. We conducted follow up analysis on the behavior of Speedtrap and MIDAR to ensure proper assessment of these tools. First, we examined Speedtrap's logs to diagnose Speedtrap's inability to detect any IPv6 aliases for Internet2. We noticed that every fragmentation identifier time series that Speedtrap seeks to use as a signature, was either labeled as random or unresponsive. This was not surprising, as prior work on Speedtrap [29] also reported that this technique does not apply to the Juniper routers that primarily comprise Internet2. Second, we explored MIDAR's logs to investigate the cause of its low recall for SWITCH. We learned that only one third of the IPv4 addresses in this network have monotonically increasing IP IDs.

Limitations and future work. Because ICMP rate limiting could be triggered at thousands of packets per second, Limited Ltd. requires the sending

of many more packets than other state-of-the-art alias resolution techniques. The maximum observed probing rate during the experiments for this paper was 34,000 pps from a single vantage point during a 5-second round. On Internet2 (SWITCH), MIDAR and Speedtrap sent 164.5k (106k) and 4k (12.7k) probe packets while Limited Ltd. sent about 4,8M (12.7M) packets. In future work, we plan to explore ways to reduce the overhead of probing and make Limited Ltd. more scalable.

5 Ethical Considerations

Limited Ltd. works by triggering limits in routers that are there for protective reasons. This raises ethical concerns, which we discuss below. To evaluate the impact of Limited Ltd., we have taken two steps: experiments in a lab environment (Sec. 5.1 and Appendix A), and feedback from operators (Sec. 5.2).

5.1 Lab experiments

We have run experiments in a lab environment on conservatively chosen hardware (over 10 years old) to show that Limited Ltd. has a controlled impact. Our findings are that: (1) routers being probed with Echo Requests by the tool remain reachable to others via `ping` with a high probability; and (2) Router CPUs show a manageable overhead at the highest probing rate, leading us to believe that our measurements are unlikely to impact the control and data planes. (3) Both Limited Ltd. and existing measurement techniques impact troubleshooting efforts (e.g., `ping`, `traceroute`). Limited Ltd. does not stand out in terms of impact compared with other accepted techniques. Appendix A details the experiments which support these conclusions.

5.2 Real-world operator feedback

In addition to lab experiments, we conducted joint experiments with SURFNET and SWITCH to evaluate the potential impact of Limited Ltd.. The experiment consisted in running Limited Ltd. on their routers while they were monitoring the CPU usage. Each run lasted about 1 minute. For SURFNET, we ran Limited Ltd. on two Juniper routers: an MX240 and an MX204. The operator observed a 4% and 2% CPU overhead. The operator also told us that the CPU overhead was observed on the MPC (line modules) CPU and not the central routing engine CPU. For SWITCH, we ran Limited Ltd. on three Cisco routers: an NCS 55A1, an ASR 9001, and an ASR-920-24SZ-M. On the two first routers, the operator told us that there was no observable change in CPU utilization. On the third router, which has a lower CPU capacity than the two others, the operator observed a CPU overhead up to 29%. These results confirm our belief that Limited Ltd. is unlikely to impact the control and data planes.

6 Conclusion

This paper presents Limited Ltd., a new, high-precision alias resolution technique for both IPv4 and IPv6 networks that leverages the ICMP rate limiting feature of individual routers. We have shown that ICMP rate limiting can generate loss traces that can be used to reliably identify aliases from other interfaces. Limited Ltd. enables IPv6 alias resolution on networks composed of Juniper routers that the state-of-the-art Speedtrap technique is not able to identify. As a part of our future work, we plan to enhance the efficiency of Limited Ltd. and explore the use of ICMP rate limiting for fingerprinting individual routers. Both the source code for Limited Ltd. and our dataset are publicly available[6].

Acknowledgments

We thank Niels den Otter from SURFnet and Simon Leinen from Switch network for their time in conducting joint experiments of Limited Ltd.. We thank people from Internet2 and Switch for providing the ground truth of their network. We thank the anonymous reviewers from both the PAM TPC and our shepherd, for their careful reading of this paper and suggestions for its improvement. Kevin Vermeulen, Olivier Fourmaux, and Timur Friedman are associated with Sorbonne Université, CNRS, Laboratoire d'informatique de Paris 6, LIP6, F-75005 Paris, France. Kevin Vermeulen and Timur Friedman are associated with the Laboratory of Information, Networking and Communication Sciences, LINCS, F-75013 Paris, France. A research grant from the French Ministry of Defense has made this work possible.

A Ethical considerations

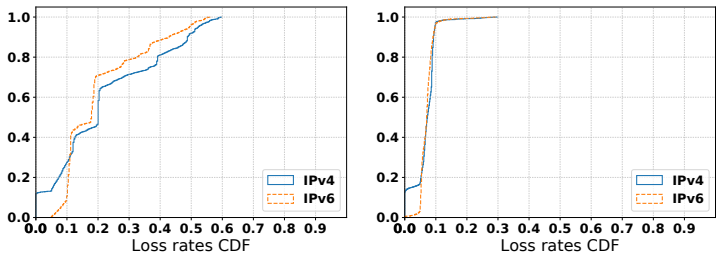
A.1 Precautions taken.

We take two precautions, that we understand to be community best practice: We sent all probing traffic from IP addresses that were clearly associated via WhoIs with their host locations, either at our institution or others hosting PlanetLab Europe nodes. We have also set up a web server on the probing machines with a contact email, so that any network operators could opt out from our experiment. We received no notice whatsoever from network operators expressing concern about our measurements. Though this is a positive sign, it could be that there are impacts that were not noticed, or that the concerns did not reach us. We therefore pushed our examination further, as detailed in the following sections.

A.2 Impact on other measurements.

Limited Ltd.'s `find_rate`() aims to find an ICMP Echo Request probing rate that produces an Echo Reply trace with a loss rate in the $[0.05, 0.10]$ range.

[6] https://gitlab.planet-lab.eu/cartography



(a) For `find_rate`() (b) For `signatures`()

Figure 5: Maximum loss rates

While it is searching for this rate, it can induce a loss rate above 0.10. If it does so, it proceeds to a binary search to find a lower probing rate for which traces falls within the desired range. Fig. 5 shows that loss rates can go as high as 0.60.

The impact on reachability for the IP addresses of that node is that there is a worst case 0.60 probability that a single `ping` packet to such an address will not receive a response if it arrives at the node during the five seconds of highest rate probing time. Most pings occur in series of packets, so the worst case probabilities are 0.36 for two `ping` packets being lost, 0.22 for three, 0.13 for four, 0.08 for five, and 0.05 for six. These are worst case probabilities for the five seconds at highest loss rate. Average reachability failure probabilities are 0.22 for one `ping` packet, 0.05 for two, 0.01 for three, and so on, while a node is being probed at its highest rate. To judge whether such a level of interference with other measurements is exceptional, we compare it to the impact of the state-of-the-art Midar tool. Midar has a phase during which it elicits three series of 30 responses each, using different methods for each series: TCP SYN packets, to elicit TCP RST or TCP SYN-ACK responses; UDP packets to a high port number, to elicit ICMP Destination Unreachable responses; and ICMP Echo Request packets, to elicit ICMP Echo Reply responses [26]. The probing rate is very low compared to Limited Ltd.: a mere 100 packets per second across multiple addresses. This is not a concern for the TCP and ICMP probing. However, the UDP probing taps into an ICMP rate limiting mechanism that tends to be much less robust than the typical ICMP Echo Reply mechanism on some routers. ICMP Destination Unreachable messages are often rate limited at 2 packets per second, which is $1/500^{\text{th}}$ the typical rate at which ICMP Echo Reply messages are rate limited. (For example, the default rate at which Cisco routers limit ICMP Destination Unreachable messages is 1 every 500 ms.)

We found that, when an IP address is a `traceroute` destination, Midar can completely block ICMP Destination Unreachable messages coming from that destination. Fig. 6 illustrates the impact. The figure shows two `traceroute` results, the top one from before or after Midar being run, and the bottom one during Midar probing. During the Midar run, we see that `traceroute`

```
traceroute to 83.99.240.90 (83.99.240.90), 30 hops max, 60 byte packets
 1  132.227.123.1  0.350 ms  0.308 ms  0.363 ms
 2  134.157.167.125  0.342 ms  0.435 ms  0.415 ms
 3  134.157.254.124  0.495 ms  0.468 ms  0.444 ms
 4  195.221.127.181  0.359 ms  0.350 ms  0.327 ms
 5  193.51.181.102  1.326 ms  1.325 ms  1.303 ms
 6  193.51.180.108  1.395 ms  1.290 ms 193.51.177.117  1.239 ms
 7  83.97.89.9  1.029 ms  1.011 ms  0.989 ms
 8  * * *
 9  62.40.98.36  8.879 ms  8.866 ms  8.951 ms
10  * * *
11  87.245.233.222  38.814 ms  38.846 ms  38.902 ms
12  87.245.242.26  39.142 ms 87.245.242.29  38.878 ms 87.245.242.26  39.012 ms
13  85.254.1.16  39.105 ms 85.254.1.128  39.029 ms 85.254.1.16  39.236 ms
14  82.193.72.232  39.287 ms  39.548 ms 82.193.72.130  39.519 ms
15  83.99.240.90  45.451 ms * *
```

(a) Before and after MIDAR run

```
traceroute to 83.99.240.90 (83.99.240.90), 30 hops max, 60 byte packets
 1  132.227.123.1  0.436 ms  0.341 ms  0.343 ms
 2  * * *
 3  134.157.254.124  0.421 ms  0.457 ms  0.434 ms
 4  195.221.127.181  0.358 ms  0.349 ms  0.319 ms
 5  193.51.181.102  1.209 ms  1.196 ms  1.200 ms
 6  193.51.177.117  0.860 ms 193.51.180.108  0.948 ms 193.51.177.117  0.896 ms
 7  83.97.89.9  0.975 ms  0.973 ms  0.950 ms
 8  * * *
 9  62.40.98.36  8.565 ms  8.607 ms  8.544 ms
10  * * *
11  87.245.233.222  38.801 ms  38.768 ms  38.748 ms
12  87.245.242.29  38.945 ms 87.245.242.26  39.009 ms  38.988 ms
13  85.254.1.16  39.028 ms  39.065 ms  39.070 ms
14  82.193.72.232  39.238 ms  39.426 ms 82.193.72.130  39.400 ms
15  * * *
16  * * *
17  * * *
18  * * *
19  * * *
20  * 83.99.240.90  45.314 ms *
```

(b) While MIDAR is running

Figure 6: Example erroneous `traceroute` result

receives no responses while it is probing hop 15, where the destination is in fact to be found. The normal functioning of `traceroute` is to continue probing at higher and higher hop counts. Only a few seconds later, when `traceroute` is sending probes to hop 20, does it start to receive ICMP Destination Unreachable messages from the destination. The result is an erroneous `traceroute`, indicating that the destination is five hops further away than it actually is. We observed this erroneous `traceroute` effect on 2,196 IP addresses out of a dataset of 10,000 IPv4 addresses collected from across the Internet. For both Limited Ltd. and MIDAR, transient interference with other measurements can be observed for the few seconds during which an IP address is being probed. Our conclusion is not that the diminution in `ping` reachability induced by Limited Ltd. is necessarily anodyne. Care should be taken to circumscribe this effect. But we observe that it does not stand out in terms of its impact on other measurements.

CPU usage. We now examine the CPU overhead generated by Limited Ltd., and its potential impact on the forwarding plane and other features involving the CPU. We have run an experiment in a local network with our own Cisco (model 3825, IOS 12.3) and Juniper (model J4350, JunOS 8.0R2.8) routers. The experiment consists in measuring three metrics while `find_rate`() routine of Limited Ltd., which has the highest probing rate, is running. We measured: (1) The CPU usage of the router, (2) the throughput of a TCP connection between the two end hosts, and (3) the rate of BGP updates. ICMP rate limiting is configured on both our Juniper and Cisco routers with an access list [21,10], limiting the ICMP input bandwidth destined to the router to 1,000 packets per second, which is the default configuration on Juniper routers.

TCP throughput was unaffected, at an average of 537 Mbps and BGP updates remained constant at 10 per second. CPU usage was at 5% for Cisco and 15% for Juniper when Limited Ltd. was not probing. During the probing, the maximum overhead was triggered for both at a maximum probing rate of 2,048 packets per second, with a peak at 10% for Cisco and 40% for Juniper during 5 seconds. Our conclusion is that there is an impact of high probing rates on CPU, but we do

not witness a disruptive impact on either the data plane (TCP throughput) or the control plane (BGP update rate).

References

1. PlanetLab Europe, https://www.planet-lab.eu
2. Private communication with CAIDA
3. RIPE Registry, https://www.ripe.net/publications/docs/ripe-508
4. Alvarez, P., Oprea, F., Rule, J.: Rate-limiting of IPv6 traceroutes is widespread: Measurements and mitigations. In: Proc. IETF 99 (2017)
5. Aminikhanghahi, S., Cook, D.J.: A survey of methods for time series change point detection. Knowledge and Information Systems **51**(2), 339–367 (2017)
6. Augustin, B., Cuvellier, X., Orgogozo, B., Viger, F., Friedman, T., Latapy, M., Magnien, C., Teixeira, R.: Avoiding traceroute anomalies with Paris Traceroute. In: Proc. IMC (2006)
7. Bender, A., Sherwood, R., Spring, N.: Fixing Ally's growing pains with velocity modeling. In: Proc. IMC (2008)
8. Breiman, L., Friedman, J., Olshen, R., Stone, C.: Classification and Regression Trees. Wadsworth and Brooks, Monterey, CA (1984)
9. Cisco: Cisco IOS quality of service solutions configuration guide, release 12.2SR; chapter: Policing and shaping overview, https://www.cisco.com/c/en/us/td/docs/ios/qos/configuration/guide/12_2sr/qos_12_2sr_book/polcing_shping_oview.html
10. Cisco: Configure commonly used IP ACLs, https://www.cisco.com/c/en/us/support/docs/ip/access-lists/26448-ACLsamples.html
11. Cisco: Control plane policing implementation best practices, https://www.cisco.com/c/en/us/about/security-center/copp-best-practices.html#7
12. Cisco: IPv6 ICMP rate limiting, https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/ipv6_basic/configuration/xe-3s/ip6b-xe-3s-book/ip6-icmp-rate-lmt-xe.pdf
13. Conta, A., Gupta, M.: RFC 4443, Internet Control Message Protocol (ICMPv6) for the Internet Protocol version 6 (IPv6) specification, IETF (2006)
14. Deal., R.A.: Cisco router firewall security: DoS protection, http://www.ciscopress.com/articles/article.asp?p=345618&seqNum=5
15. Ensafi, R., Knockel, J., Alexander, G., Crandall, J.R.: Detecting intentional packet drops on the Internet via TCP/IP side channels. In: Proc. PAM (2014)
16. Govindan, R., Tangmunarunkit, H.: Heuristics for Internet map discovery. In: Proc. INFOCOM (2000)
17. Gunes, M.H., Sarac, K.: Resolving IP aliases in building traceroute-based Internet maps. IEEE/ACM Transactions on Networking **17**(6), 1738–1751 (2009)
18. Gunes, M.H., Sarac, K.: Importance of IP alias resolution in sampling Internet topologies. In: Proc. GI (2007)
19. Guo, H., Heidemann, J.: Detecting ICMP rate limiting in the Internet. In: Proc. PAM (2018)
20. Juniper: Default ICMP rate limit on the system for host inbound connections, https://kb.juniper.net/InfoCenter/index?page=content&id=KB28184&cat=SRX_SERIES&actp=LIST
21. Juniper: IPv6 multicast routing on E series broadband services routers, release 15.1; access-list, https://www.juniper.net/documentation/en_US/junose15.1/topics/reference/command-summary/access-list.html

22. Juniper: Policer implementation overview, https://www.juniper.net/documentation/en_US/junos/topics/concept/policer-mx-m120-m320-implementation-overview.html
23. Juniper: System management and monitoring feature guide for switches; internet-options (ICMPv4), https://www.juniper.net/documentation/en_US/junos/topics/reference/configuration-statement/icmpv4-rate-limit-edit-system.html
24. Juniper: System management and monitoring feature guide for switches; internet-options (ICMPv6), https://www.juniper.net/documentation/en_US/junos/topics/reference/configuration-statement/icmpv6-rate-limit-edit-system.html
25. Keys, K.: Internet-Scale IP alias resolution techniques. ACM SIGCOMM Computer Communication Review **40**(1), 50–55 (2010)
26. Keys, K., Hyun, Y., Luckie, M., Claffy, K.: Internet-scale IPv4 alias resolution with MIDAR. IEEE/ACM Transactions on Networking **21**(2), 383–399 (2013)
27. Killick, R., Eckley, I.A.: changepoint: An R Package for Changepoint Analysis. Journal of Statistical Software **58**(3), 1–19 (2014), http://www.jstatsoft.org/v58/i03/
28. Kim, S., Harfoush, K.: Efficient estimation of more detailed Internet IP maps. In: Proc. ICC (2007)
29. Luckie, M., Beverly, R., Brinkmeyer, W., et al.: Speedtrap: Internet-scale IPv6 alias resolution. In: Proc. IMC (2013)
30. Marchetta, P., Persico, V., Pescapè, A.: Pythia: Yet another active probing technique for alias resolution. In: Proc. CoNEXT (2013)
31. Padmanabhan, R., Li, Z., Levin, D., Spring, N.: UAv6: Alias resolution in IPv6 using unused addresses. In: Proc. PAM (2015)
32. Pansiot, J.J., Grad, D.: On routes and multicast trees in the Internet. ACM SIGCOMM Computer Communication Review **28**(1), 41–50 (1998)
33. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in Python. Journal of Machine Learning Research **12**, 2825–2830 (2011)
34. Postel, J.: RFC 792, Internet Control Message Protocol, IETF (1981)
35. Qian, S., Wang, Y., Xu, K.: Utilizing destination options header to resolve IPv6 alias resolution. In: Proc. GLOBECOM (2010)
36. Qian, S., Xu, M., Qiao, Z., Xu, K.: Route positional method for IPv6 alias resolution. In: Proc. ICCCN (2010)
37. Ravaioli, R., Urvoy-Keller, G., Barakat, C.: Characterizing ICMP rate limitation on routers. In: Proc. ICC (2015)
38. Sherry, J., Katz-Bassett, E., Pimenova, M., Madhyastha, H.V., Anderson, T., Krishnamurthy, A.: Resolving IP aliases with prespecified timestamps. In: Proc. IMC (2010)
39. Sherwood, R., Bender, A., Spring, N.: Discarte: a disjunctive Internet cartographer. ACM SIGCOMM Computer Communication Review **38**(4), 303–314 (2008)
40. Spring, N., Mahajan, R., Wetherall, D.: Measuring ISP topologies with Rocketfuel. ACM SIGCOMM Computer Communication Review **32**(4), 133–145 (2002)
41. Vermeulen, K., Strowes, S.D., Fourmaux, O., Friedman, T.: Multilevel MDA-lite Paris Traceroute. In: Proc. IMC (2018)
42. Willinger, W., Alderson, D., Doyle, J.C.: Mathematics and the Internet: A source of enormous confusion and great potential. Notices of the American Mathematical Society **56**(5), 586–599 (2009)