Fixing Ally's Growing Pains with Velocity Modeling

Adam Bender
Dept. of Computer Science
University of Maryland
College Park, MD, USA
bender@cs.umd.edu

Rob Sherwood
Dept. of Computer Science
University of Maryland
College Park, MD, USA
capveg@cs.umd.edu

Neil Spring
Dept. of Computer Science
University of Maryland
College Park, MD, USA
nspring@cs.umd.edu

ABSTRACT

Mapping the router topology is an important component of Internet measurement. Alias resolution, the process of mapping IP addresses to routers, is critical to accurate Internet mapping. Ally, a popular alias resolution tool, was developed to resolve aliases in individual ISPs, but its probabilistic accuracy and need to send $O(n^2)$ probes to infer aliases among n IP addresses make it unappealing for large-scale Internet mapping. In this paper, we present RadarGun, a tool that uses IP identifier velocity modeling to improve the accuracy and scalability of the Ally-based resolution technique. We provide analytical bounds on Ally's accuracy and validate our predicted aliases against Ally. Additionally, we show that velocity modeling requires only $O(n)$ probes and thus scales to Internet-sized mapping efforts.

Categories and Subject Descriptors

C.2.3 [**Computer-Communication Networks**]: Network Operations—*Network monitoring*; C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Network topology*

General Terms

Measurement, Experimentation

Keywords

Alias resolution, IP identifier, Velocity modeling, Ally

1. INTRODUCTION

Measured network topologies have proven useful for diagnosis [8, 11, 19], modeling [9, 10], and simulation of new protocols [20]. Yet their accurate construction, especially at Internet scale, remains difficult. We focus here on a specific problem within this context: alias resolution, the process of recognizing which of a set of IP addresses belong to interfaces on the same router.

IMC'08, October 20–22, 2008, Vouliagmeni, Greece.

Traceroute, including variants modified to aid Internet mapping [1, 3, 15], provides lists of IP addresses and adjacencies between them. However, more useful is the set of routers and their connections. As routers have many interfaces, each with a different IP address, alias resolution is required to construct the many-to-one mapping of addresses (aliases) to routers. Constructing an accurate mapping is vital [7]. False aliases cause disparate addresses to be grouped and entirely separate parts of a network to be connected. False negatives can inflate path diversity estimates: one path through each unresolved alias rather than a single path through a single router [19].

Both types of errors are possible. Ally, the Rocketfuel [17] tool for alias resolution, suffers from false positives and negatives; Rocketfuel notes that the lack of completeness of prior techniques led to even more false negatives. Analytical alias resolution [6], while able to resolve aliases for interface addresses that are unresponsive to probes, appears to have both types of error. Alias resolution by aligning the addresses from the record-route IP option can also be error-prone because of heterogeneous implementations [14].

Additionally, current probing methods do not scale to Internet-sized topologies. To find pairs of interface addresses that share an IP identifier (IP ID) counter (which we describe in more detail below), Ally probes each of $\binom{n}{2}$ possible combinations of n addresses, requiring $O(n^2)$ probes. In addition to being impractically slow for large values of n, rate limiting and non-static networks compound inaccuracies.

In this paper, we describe a scalable, accurate method for applying the IP ID alias resolution test en masse. To improve the accuracy of alias resolution, we developed a tool, RadarGun, that models the *rate* at which an interface's IP ID increases. By conducting an informed probing of addresses to recover IP identifiers, we can avoid several pitfalls. First, many routers rate limit ICMP responses, making them temporarily unresponsive when probed by Ally. RadarGun can control the rate at which probes are transmitted and collect enough probes from each address that a few missing packets do not affect the model. Second, different routers may advance through IP identifiers at different rates: some appear "busier" than others. Those with a fast counter may skip thresholds hard-coded into Ally, causing false negatives.

We evaluate our technique against reliable aliases and Ally-tested non-aliases to show correctness of inferences. We quantify the number of probes required and how closely they must be spaced and compute how much bandwidth would be required to resolve aliases among 500,000 addresses. Finally, we describe some interesting behaviors of the IP identifier counters, showing what appear to be periodic updates.

2. RELATED WORK

Pansiot and Grad [12] first noticed the importance of alias resolution in network mapping. Their technique sends probes to an IP address and discovers an alias if the response has a different source address than the destination of the probe: the source of the response is believed to be an alias for the destination of the probe. Govindan and Tangmunarunkit [5] extended this technique through source routing to increase completeness.

Rocketfuel [17] introduced the "Ally" technique, which attempts to detect whether two interface addresses share an IP identifier counter. The IP identifier is a 16-bit field in the IP header that allows fragmented packets to be reassembled; each fragment retains the unique identifier of the original packet. This identifier allows an endhost to collect the fragments that derive from the original packet and reconstruct them into the original datagram. A common technique for enforcing this uniqueness is to use a counter, which wraps to 0 when it reaches its maximum value ($2^{16}-1$). If the two addresses share a counter, the interfaces have the same host processor and IP stack, and must be aliases. We present details of how Ally determines if two addresses share a counter in Section 3.

Although Ally is able to resolve aliases that the source-address technique could not (because this implementation decision appeared more common than altering the source address), it has major shortcomings. Primarily, the number of probes required increases with the square of the number of addresses; in principle, every address must be compared to every other. Additionally, Ally is subject to false negatives with busy routers (Section 3.1).

Recent research has shown alternative methods for finding aliases. These methods are motivated by an inability to solicit responses from some router IP addresses. For example, addresses in the Abilene backbone are not responsive to UDP probes. Gunes and Sarac noted that incomplete or erroneous alias resolution can significantly alter the properties of the measured topologies [7] and proposed methods to infer aliases that use common addressing practice [6]. In prior work [14, 15], we noted the potential for using the record route IP option to find aliases during the execution of a traceroute. While the record route technique discovered 11% of the total aliases, Ally still contributed the bulk.

Each of these techniques has strengths and weaknesses. Although Ally might completely resolve those addresses that respond, it has potential to produce false positives (if counters happen to have similar values when probed) and false negatives (if a single counter increments quickly or probes are lost). Rocketfuel [17] advised verifying each putative alias at a later time with the expectation that such "accidental" aliases would be disproven. Even so, Teixeira et al. [19] reported significant errors in the alias resolution of Rocketfuel-measured maps that alter the appearance of path diversity when compared to real topologies. Feamster et al. [4] used an implementation of Ally in which each test was repeated 100 times and the majority opinion used to determine an alias. These issues and uses suggest rethinking this alias resolution method so that errors are avoided.

Bellovin [2] presented a similar problem: identifying distinct counters in packets that share a source address to count the number of hosts behind network address translators. His technique created a list of IP ID sequences; each observed packet would either be matched to an existing sequence based on time and ID value, or would be the start of a new sequence. The idea is that each sequence corresponds to a host behind the NAT.

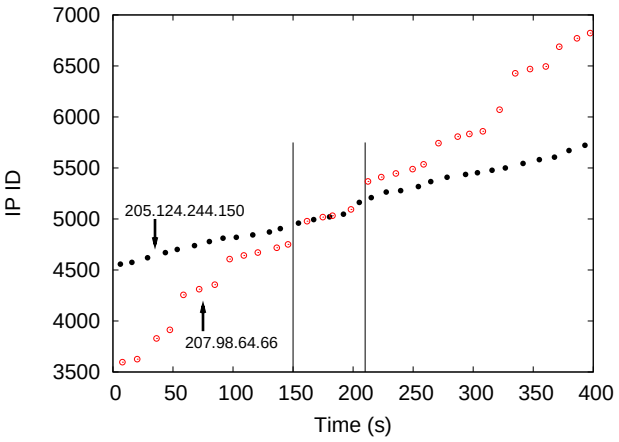

Figure 1: Two IP IDs that produce false positives between time 150 and 210

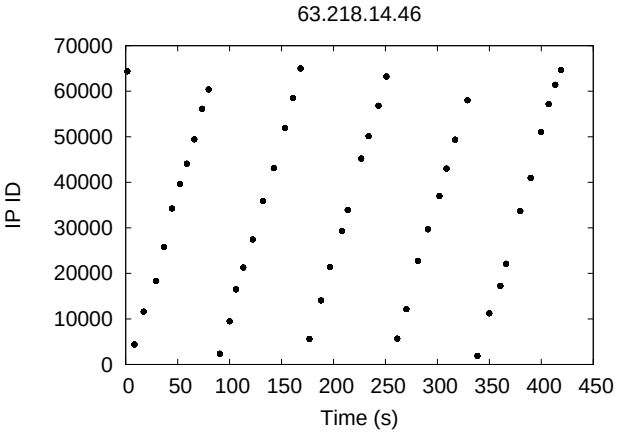

Figure 2: Router with quickly-increasing IP ID, causing false negatives when probed by Ally

3. ALLY: ALIAS RESOLUTION WITH IP ID

Ally tests if two addresses, A and B, are interfaces on the same router by sending a UDP packet to each address. The destination(s) respond to each probe with an ICMP "port unreachable" error. Ally records the IP ID of each response packet. Let the IP IDs be $id_{A,1}$ and $id_{B,1}$. If the IP IDs of the first are "close enough," specifically, if $id_{A,1} - 10 < id_{B,1} < id_{A,1} + 200$, Ally sends another pair of probes to the same addresses, though in reverse order. The "close enough" test is repeated for the second probe pair, $id_{A,2}$ and $id_{B,2}$. If both tests pass, and both $id_{A,1} < id_{A,2}$ and $id_{B,1} < id_{B,2}$, then A and B are marked as aliases. If the IP IDs observed from either pair of probes fail the test, then A and B are classified as non-aliases.

3.1 Ally's Shortcomings

While Rocketfuel [17] notes that Ally is subject to false positives, we show that it is susceptible to false negatives as well. False positives occur when two different routers happen to have similar IP ID values when they are probed.

Figure 1 shows such a case, extracted from our dataset of IP IDs sampled over time. For any two probes sent in the range of time demarcated by the vertical lines, Ally will infer a false alias.

On the other hand, if the IP ID counter of a router increases rapidly, then the observed IP IDs from that router may not fall within the range that Ally expects. Thus two interfaces on the same router may be classified as non-aliases. Figure 2 shows a router whose IP ID was observed to increase by almost 800 every second, implying it sources nearly 800 packets per second. If any probe packet is delayed by more than 250 milliseconds, the router's IP ID counter may have incremented beyond Ally's threshold. In addition, we observe some routers who do not use a counter for the IP ID values. IP ID-based techniques cannot be used to infer aliases among these routers.

Ally was created with the intention of mapping only individual ISPs. Problems arise when extending Ally to Internet-scale topologies. Namely, whenever Ally attempts to infer an alias between interfaces A and B, Ally requires fresh values of $id_{A,1}$ and $id_{B,1}$. Thus, for every possible alias pair among n addresses, between two and four probes are sent, meaning up to $4\binom{n}{2}$ total probes are required.

In addition, rate-limiting by routers presents a significant problem. If a router limits the rate at which it issues ICMP packets, Ally resorts to a weaker test, because it cannot receive paired responses. We measure the responsiveness of routers in Section 4.1.

4. VELOCITY MODELING

In this section, we present our main contribution, the technique of *velocity modeling*. This is an attempt to determine if a given interface uses a counter to derive the value of the IP ID in outgoing ICMP or TCP packets, and if so, to model the rate at which the counter increases. This gives us a model of the IP ID over time, which we can use to infer aliases. While this technique has many possible uses, which we discuss later in Section 6, we primarily focus on using velocity modeling for alias resolution.

We developed a tool, RadarGun, to perform velocity-based alias resolution. RadarGun is built on top of Scriptroute [18]. Instead of probing an address directly whenever the value of the IP ID is needed, as Ally does, RadarGun creates a model of the IP ID over time and predicts the expected IP ID at various times. To do this, RadarGun estimates the rate, or velocity, at which the IP ID increases. Our insight is that probes sent to addresses that are aliases for the same router will have similar velocities, while probes sent to two non-aliases will (with high probability) show different velocities. For example, Figure 3 shows two (non-alias) addresses with disparate velocities.

4.1 Collecting Data

To model the velocity of an IP address, RadarGun sends probe packets (either UDP, eliciting an ICMP error message, or TCP ACK, garnering a TCP RST) to the address and records the IP ID of each response and the time it was received. The set of responses forms a time series [16]. RadarGun then fits the responses from each address to a line, using least squares linear regression. Thus each address that returns more than three responses provides us with a slope (velocity, in units of packets per second) and offset (the y-intercept at an arbitrary time—we choose the time that the first probe was sent).

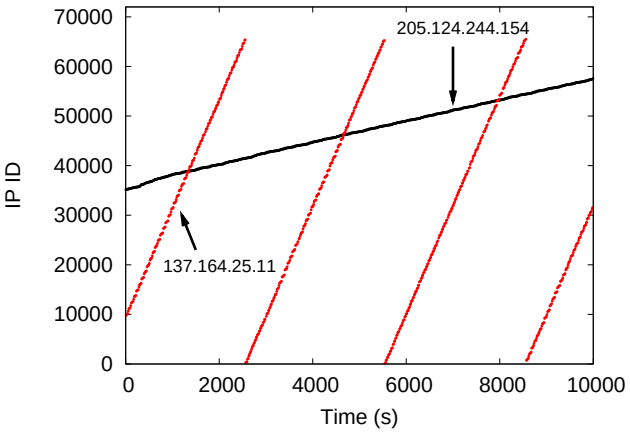

Figure 3: Examples of routers with different velocities. Both suggest that they can be modeled linearly.

TCP and UDP	TCP only	UDP only	Neither
3942 (43.5%)	1121 (12.4%)	658 (7.3%)	3335 (36.8%)

Table 1: Number of addresses that responded to probing with different protocols.

The analysis is complicated by the fact that counter-based IP IDs wrap when they reach their maximum value ($2^{16}-1$). To account for this, RadarGun keeps a counter of the number of times the IP ID has wrapped, $nWrap$, in an attempt to model the IP ID as a monotonically-increasing counter over an arbitrarily long time. RadarGun estimates the time to wrap from the first few samples observed from an address. Wrapping needs to be taken into account in two situations: when the IP ID of a probe is less than the IP ID of the previous probe, and when RadarGun has not seen a response from an address for longer than the expected time to wrap. In the first case, RadarGun assumes the counter has wrapped and increments $nWrap$. Because probes are spaced several seconds apart, the chance that IP IDs arrive out of order due to delay is minimal. In the second case, RadarGun increments $nWrap$ by the expected number of times that the counter has wrapped, based on the initial estimate.

Another issue that frustrated data collection was the unresponsiveness of routers. Tools like RadarGun and Ally require addresses to respond to direct probes. To evaluate the rate at which routers will respond to probes, we obtained a set of 9,056 addresses that DisCarte [14] had discovered by running `traceroute` between pairs of PlanetLab [13] nodes.

We probed each address with 200 TCP ACK packets sent 34 seconds apart and 200 UDP packets sent 35 seconds apart (the probing rate was a function of Scriptroute's internal bandwidth limiting). Table 1 shows how many addresses were responsive to each combination of protocols. Figure 4 shows how many responses were received for each protocol from each address (note the addresses are sorted by number of probes returned for each protocol, and that the two y-values associated with a point on the x-axis do not necessarily correspond to the same address). As more addresses responded to TCP probes, we used TCP probes to derive all further results in this paper. However, our technique would clearly benefit from being able to use both TCP and UDP.

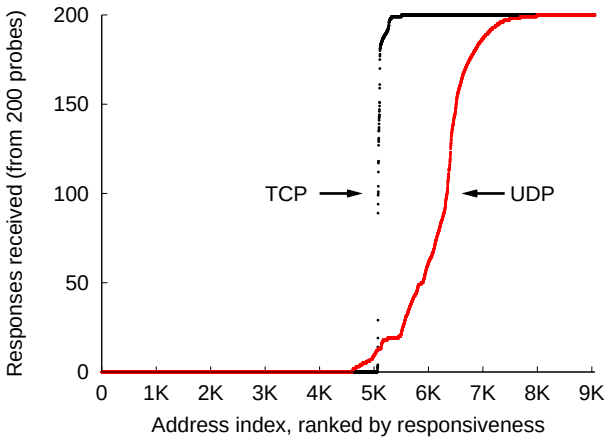

Figure 4: Probing 9,056 IP addresses at a constant rate shows that interfaces are more responsive to TCP.

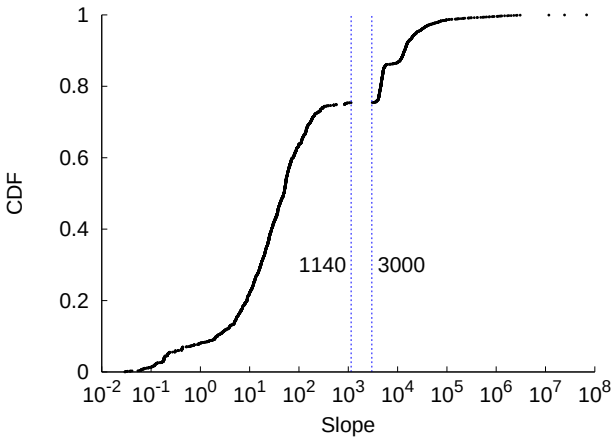

Figure 5: CDF of interpolated IP ID slopes

We leave to future work a study of whether routers use the same counter for TCP and UDP packets and any effect on accuracy the use of multi-protocol probes may have.

By reducing the number of probes sent and responses required for accurate results, RadarGun does not suffer as much from rate limiting and is thus more complete than Ally. As Ally requires that an address be responsive every time that it is probed, whereas RadarGun does not, Ally claims more pairs as unresponsive.

4.2 Modeling Velocities

RadarGun (and Ally) can accurately infer aliases only for routers whose IP ID is implemented as a counter. Some operating systems, such as versions of BSD, insert psuedo-random values in the IP ID field [2]. To determine the distribution of implementations among routers, we examined the slopes inferred from the "unwrapped" data points both for accuracy and "sanity." Figure 5 shows a CDF of calculated slopes. This figure suggests that addresses can be partitioned in to two sets: those with a slope below 1140 and those with a slope above 3000 (there are no intermediate values of slope). Lower slopes suggest a linear model is appropriate; higher slopes might not be modeled accurately.

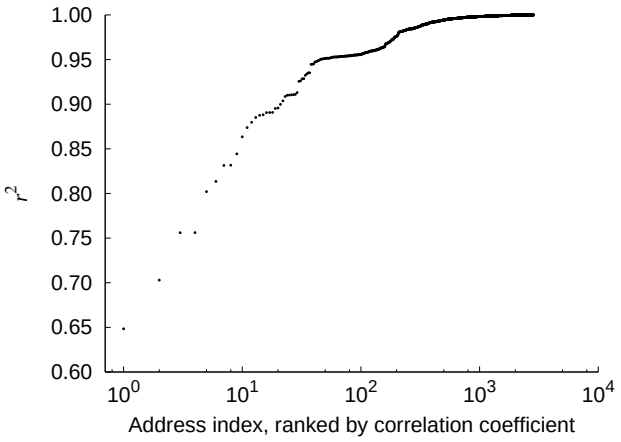

Figure 6: The square of correlation coefficient (r^2) for addresses classified as "linear"

Unresponsive	4,240	(46.8%)
Linear	2,841	(31.4%)
Non-linear	968	(10.7%)
ICMP "Destination unreachable"	698	(7.7%)
IP ID always 0	208	(2.3%)
Reflects the IP ID of probe	101	(1.1%)

Table 2: Classification of 9,056 intra-PlanetLab addresses

When unwrapping samples from routers that use random IP IDs (or who source packets so frequently that the IP ID wraps often enough to appear as random), every IP ID sample that is less than the previous sample suggests that the "counter" has wrapped. This adds 2^{16} to the unwrapped data and leads to a large inferred slope.

We visually inspected the samples that we obtained and found that the largest calculated slope that we felt could be correctly modeled as linear was 881.8 packets per second. Samples with slopes larger than this appeared random. We re-probed the addresses of these samples at a higher probe rate, and inspecting the results still did not suggest that they could be distinguished from psuedorandom values.

Using this cut-off, we can classify our set of 9,056 addresses into various categories, as shown in Table 2. Addresses were classified as "unresponsive" if they responded to fewer than 25% of probes. Figure 6 shows the correlation coefficients (r^2 values) of the unwrapped samples that we classified as linear. As the correlation coefficients rapidly converge to 1, this shows that the series of samples can accurately be modeled by a linear approximation.

4.3 Inferring Aliases

We now describe the test we use to determine if two IP addresses, A and B, are aliases for the same router. Let S_A be the set of ($time$, $IP\ ID$) samples collected from A, and S_B be the samples collected from B. Assume for ease of exposition that several samples from S_A were collected before the earliest sample in S_B and that several pairs in S_B were collected after the latest point in S_A.

We split the samples into three sets: the samples of S_A that were received before any samples from S_B (termed the *head*), the samples of S_B that were received after any samples from S_A (the *tail*), and the remaining samples that roughly overlap in time (the *middle*). The middle may be

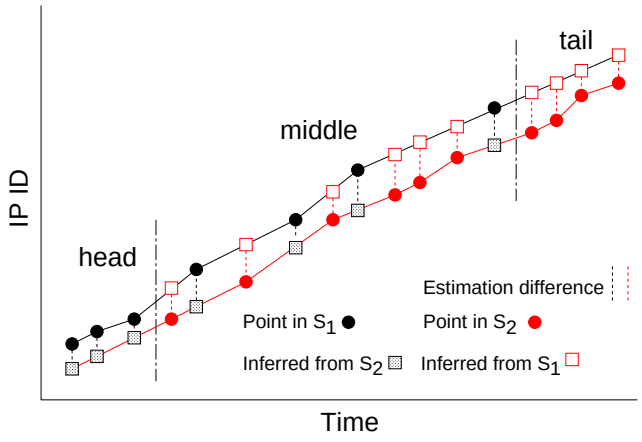

Figure 7: Partitioning points into sets

empty; the head or tail may have just one element; points in the head and tail may be from the same address. Figure 7 shows a diagram depicting how samples are partitioned.

For each sample (t, id) in $S_A \cup S_B$, we compute the *distance* between id and the expected value of the other IP ID at time t interpolated from the corresponding set of samples. The distances are summed across all samples, and divided by the number of samples to yield an average distance per sample. First, RadarGun sets a variable sum to 0. To calculate the distance of a sample (t_H, id_H) in the head, RadarGun estimates the value of B's IP ID at time t_H using the linear approximation of S_B to get an estimate id'_H, and adds $|id'_H - id_H|$ to sum. RadarGun executes a similar process to compute the distances between samples in the tail.

For samples in the middle, RadarGun is able to make a more accurate estimation. Let $(t_{A,1}, id_{A,1})$ and $(t_{A,2}, id_{A,2})$ be samples in S_A and (t_B, id_B) be a point in S_B such that $t_{A,1} \leq t_B < t_{A,2}$. The estimated value of id_A at time t_B is interpolated based on the two points in S_A:

$$id_A^{est} = (id_{A,2} - id_{A,1}) \frac{t_B - t_{A,1}}{t_{A,2} - t_{A,1}} + id_{A,1}$$

$$sum \mathrel{+}= |id_B - id_A^{est}|$$

Let $\Delta_{A,B} = \dfrac{sum}{|S_A \cup S_B|}$ be the average distance between observed and expected IP ID per probe. If two IP addresses have a small $\Delta_{A,B}$, they are likely to be aliases, whereas a large $\Delta_{A,B}$ indicates that the addresses are not aliases. In the next section, we give possible values for the threshold between aliases and non-aliases, and show how well our classification algorithm works on known aliases and non-aliases.

5. VALIDATION

To validate the accuracy of velocity modeling as a means of alias detection, we computed the average distance of pairs of IP addresses drawn from two sets. One set (aliases) contained 932 known alias pairs, as determined by a common source address [5, 12]. This technique is not probabilistic, so these address pairs are very likely to be actual aliases.

Neither RadarGun nor Ally can correctly claim two addresses as non-aliases when both addresses do not derive IP ID values from a counter. Thus, to obtain a set of likely non-aliases, we ran Ally on all 4,034,220 pairs of 2,841 addresses that RadarGun reported as linear. The 3,055,241 pairs that Ally reports as non-aliases comprise the dataset non-aliases.

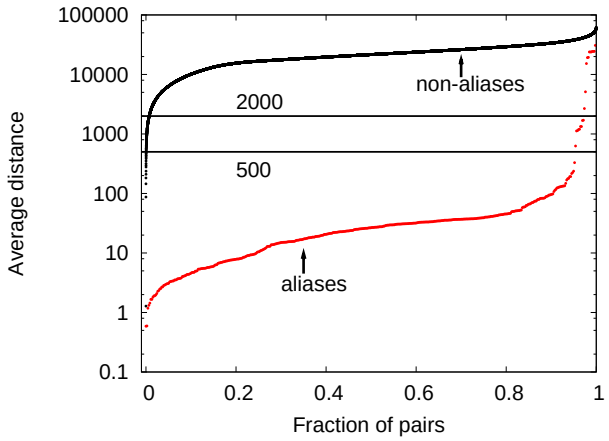

Figure 8: Average distance between alias and non-alias address pairs

	aliases		non-aliases
	RadarGun	Ally	RadarGun
Aliases	436	418	1,605
Non-aliases	12	293	3,033,204
Undetermined	9	-	20,432
Non-linear	469	-	0
Unresponsive	6	215	0

Table 3: The accuracy of RadarGun and Ally.

We resolved each pair by sending 30 probes to each of the IP addresses in aliases and non-aliases. This required $30 \times 2,841 = 85,230$ total probes, whereas Ally sent 8,092,038 probes to resolve all pairs. Figure 8 shows the average distance between each address pair in aliases and non-aliases.

When classifying address pairs as aliases or not, we set two thresholds for the average distance between samples. For pairs whose average distance is below the lower *alias* threshold, we classified them as aliases. We chose the value of 500 for the alias threshold, based on the examination of Figure 8. Pairs whose average distance is above the larger *non-alias* threshold are classified as non-aliases. We used 2,000 as the non-alias threshold. We classify pairs between these thresholds as "Undetermined."

When using these thresholds, RadarGun produces the results shown in Table 3. The row entitled "Non-linear" counts the number of pairs we did not classify because at least one of the addresses could not be modeled as linear. The row entitled "Unresponsive" reflects the number of pairs we did not classify because at least one of the addresses did not return more than 25% of the probes.

By observing Table 3, it appears that RadarGun is still incomplete, yet more accurate than Ally. When RadarGun cannot model some addresses as linear, it refrains from making a conclusion about pairs involving those addresses; Ally reports those pairs as non-aliases with high probability. RadarGun finds more responsive pairs as it expects some probes to go unanswered; Ally requires that an address respond to all probes.

6. DISCUSSION

In this section, we discuss our intended work to improve alias resolution with velocity modeling, as well as other potential uses for velocity modeling.

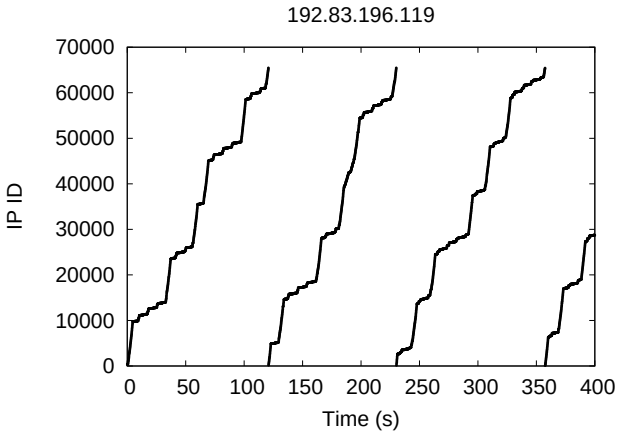

Figure 9: Router that exhibits possible routing updates

Obtaining an accurate model of the errors incurred by using velocity modeling is an important step in continuing this line of research. While our initial results show that velocity modeling can easily distinguish aliases from non-aliases, our data set was relatively small. We will first run RadarGun on a larger data set in an attempt to obtain not just an accurate value of the threshold between aliases and non-aliases, but an understanding of why this value is appropriate as well. We intend to compare the results of RadarGun against the results of other alias resolution tools and techniques, to get a better understanding of the error rates and causes of all such resolvers.

Velocity modeling may be useful for other applications. We have shown that for most routers, the IP ID counter increases at a steady rate. Therefore, observed changes to this expected rate can be indicative of anomalies. For instance, routing updates may be visible by observing changes in velocity. During normal operation, routers send relatively few packets. Sudden spikes in activity, evidenced by a rapidly increasing IP ID, might indicate a routing update. Figure 9 shows what we believe to be a router that issues periodic routing update messages. However, we have not corroborated this hypothesis, nor do we have other possible explanations for the observed behavior.

The slopes of some routers, such as that in Figure 9, are not constant. To obtain greater accuracy when resolving aliases for these routers, a resolver should intersperse probes to all addresses, i.e., make the *middle* set (Section 4.3) as large as possible. RadarGun is able to make a more accurate estimate of the expected IP ID of an address for series of probes that overlap in time. This estimate is not based on the inferred slope of the address, which in the case of such routers, may not accurately predict the actual IP ID value at all times.

In regards to the scalability of RadarGun, we analyze the probe overhead that velocity modeling incurs when resolving aliases among 500,000 addresses. A single host with a 1 Mb/s line can send 3276 TCP probes per second. Probing each destination once every 34 seconds (the average probe time of our experiments), this host can probe 111,384 addresses. Five such hosts with tight clock synchronization, or a single host with a 10 Mb/s connection, can send 30 probes to each of the 500,000 addresses in just 17 minutes.

7. CONCLUSION

We have presented velocity modeling as a technique to resolve aliases. One of the advantages of velocity modeling is that the resolution process is separate from the probing process. Probes can be collected over time, and the resolver run off-line after collection is complete. The probes sent to a single address need not closely follow one another; our experimental results are accurate when each interface is probed approximately once every 34 seconds. As a result, when compared to Ally, our technique does not suffer as much from routers that rate-limit ICMP responses. Velocity modeling also requires far fewer probes per address when resolving aliases among a large number of addresses. This allows velocity modeling to scale to Internet-sized inputs.

Acknowledgments

We thank our shepherd Walter Willinger, Ethan Katz-Bassett, and the anonymous reviewers for helpful comments. This work was supported by NSF Awards ANI 0092806, CNS 0626629, and CNS 0435065.

8. REFERENCES

[1] B. Augustin, *et al.* Avoiding traceroute anomalies with Paris traceroute. In *IMC*, 2006.
[2] S. M. Bellovin. A method for counting NATted hosts. In *Internet Measurement Workshop*, 2002.
[3] B. Donnet, P. Raoult, T. Friedman, and M. Crovella. Efficient algorithms for large-scale topology discovery. In *SIGMETRICS*, 2005.
[4] N. Feamster, D. G. Andersen, H. Balakrishnan, and M. F. Kaashoek. Measuring the effects of Internet path faults on reactive routing. In *SIGMETRICS*, 2003.
[5] R. Govindan and H. Tangmunarunkit. Heuristics for Internet map discovery. In *Infocom*, 2000.
[6] M. H. Gunes and K. Sarac. Analytical IP alias resolution. In *International Conference on Communications*, 2006.
[7] M. H. Gunes and K. Sarac. Importance of IP alias resolution in sampling Internet topologies. In *Global Internet Symposium*, 2007.
[8] N. Hu, O. Spatscheck, J. Wang, and P. Steenkiste. Locating Internet bottlenecks: Algorithms, measurements, and implications. In *SIGCOMM*, 2004.
[9] L. Li, D. Alderson, W. Willinger, and J. Doyle. A first-principles approach to understanding the Internet's router-level topology. In *SIGCOMM*, 2004.
[10] H. V. Madhyastha, *et al.* A structural approach to latency prediction. In *IMC*, 2006.
[11] R. Mahajan, N. Spring, D. Wetherall, and T. Anderson. User-level Internet path diagnosis. In *SOSP*, 2003.
[12] J.-J. Pansiot and D. Grad. On routes and multicast trees in the Internet. *ACM CCR*, 28(1):41–50, 1998.
[13] L. Peterson, T. Anderson, D. Culler, and T. Roscoe. A blueprint for introducing disruptive technology into the Internet. In *HotNets*, pp. 59–64, 2002.
[14] R. Sherwood, A. Bender, and N. Spring. Discarte: A disjunctive internet cartographer. In *SIGCOMM*, 2008.
[15] R. Sherwood and N. Spring. Touring the Internet in a TCP sidecar. In *IMC*, 2006.
[16] R. H. Shumway and D. S. Stoffer. *Time Series Analysis and Its Applications: With R Examples*. Springer, 2 edn., 2000.
[17] N. Spring, R. Mahajan, and D. Wetherall. Measuring ISP topologies with Rocketfuel. In *SIGCOMM*, 2004.
[18] N. Spring, D. Wetherall, and T. Anderson. Scriptroute: A public Internet measurement facility. In *USITS*, 2003.
[19] R. Teixeira, K. Marzullo, S. Savage, and G. Voelker. In search of path diversity in ISP networks. In *IMC*, 2003.
[20] G. R. Yaun, *et al.* Large-scale network simulation techniques: examples of TCP and OSPF models. *ACM CCR*, 33(3):27–41, 2003.