# Ting: Measuring and Exploiting Latencies Between All Tor Nodes

Frank Cangialosi
University of Maryland
frank@cs.umd.edu

Dave Levin
University of Maryland
dml@cs.umd.edu

Neil Spring
University of Maryland
nspring@cs.umd.edu

## ABSTRACT

Tor is a peer-to-peer overlay routing network that achieves unlinkable communication between source and destination. Unlike traditional mix-nets, Tor seeks to balance anonymity and performance, particularly with respect to providing low-latency communication. As a result, understanding the latencies between peers in the Tor network could be an extremely powerful tool in understanding and improving Tor's performance and anonymity properties. Unfortunately, there are no practical techniques for inferring accurate latencies between two arbitrary hosts on the Internet, and Tor clients are not instrumented to collect and report on these measurements.

In this paper, we present Ting, a technique for measuring latencies between arbitrary Tor nodes from a single vantage point. Through a ground-truth validation, we show that Ting is accurate, even with few samples, and does not require modifications to existing clients. We also apply Ting to the live Tor network, and show that its measurements are stable over time. We demonstrate that the all-pairs latency datasets that Ting permits can be applied in disparate ways, including faster methods of deanonymizing Tor circuits and efficiently finding long circuits with low end-to-end latency.

## Categories and Subject Descriptors

C.4 [**Performance of Systems**]: Measurement Techniques; C.2.2 [**Computer-Communication Networks**]: Network Protocols; C.2.0 [**Computer-Communication Networks**]: Security and Protection

## Keywords

Latency measurement; Tor; Deanonymization

*IMC'15,* October 28–30, 2015, Tokyo, Japan.
Copyright is held by the owner/author(s). Publication rights licensed to ACM.
ACM 978-1-4503-3848-6/15/10 ...$15.00.
DOI: http://dx.doi.org/10.1145/2815675.2815701.

## 1. INTRODUCTION

Tor [7] is a popular peer-to-peer service for providing unlinkable communication and anonymous services. It operates by allowing a source node to create a *circuit* through multiple Tor *relays*, each of whom learns only about the node preceding and the node following it in the circuit. To keep the nodes on the path from linking the source and destination, Tor circuit creation mandates at least two relays.

Arguably, the primary feature of Tor that has led to its widespread success is Tor's balance between anonymity and good end-to-end performance, particularly with respect to network latency. One consequence of this trade-off is that Tor defaults to circuits of length three: an entry node, a middle node, and an exit node. Another consequence of this design choice is that Tor relays do not arbitrarily introduce delays or "mixing" like in other anonymity systems [22, 4, 29], and instead forward as quickly as is possible (and fair).

Tor's performance and anonymity are therefore highly dependent on the round-trip time latencies between the nodes within the Tor network. For example, there have been many proposals for improving how Tor selects its circuits [2, 28, 20, 8] that benefit from understanding the inter-Tor-node latencies, and as we demonstrate in Section 5.1, latency knowledge can speed up existing deanonymization techniques [16, 9, 12, 10].

However, to date, there are no techniques available within Tor or through other Internet measurement to accurately measure the round-trip times between two arbitrary Tor nodes. Researchers and practitioners have thus had to rely on approximations such as geographic distances, which simply cannot model network phenomena such as triangle inequality violations (TIVs) in Internet routing [15].

In this paper, we present *Ting*, a technique for accurately measuring the round-trip times between *any arbitrary pair of Tor nodes*. Ting operates strictly at Tor's "data plane": it carefully constructs circuits and directly measures latencies to and through Tor relays. Critically, Ting works without requiring any modifications to the Tor protocol, to Tor clients, or special permission from Tor users. Through a ground-truth validation on PlanetLab [21], we show that Ting is extremely accurate, imposes little communication or computational overhead on the Tor network, and can be run from a single host.

We also show three applications of such an unprecedentedly accurate and thorough latency dataset. Among these, we show for the first time the presence of TIVs in the Tor network, and the extent to which Ting's measurements can be used to improve end-to-end paths.

Ting fills a gap that has formed in network measurement tools: the ability to directly measure the latencies between two hosts, neither of which are under the control of the experimenter. The King technique [11], introduced in 2002, indirectly measured latency between two arbitrary hosts by cleverly constructing queries to publicly available recursive DNS servers. Unfortunately, since then, most publicly available DNS servers have disallowed recursive queries due to security concerns, rendering King narrowly applicable. Conversely, we show that Ting can be used to infer with *direct measurements* the end-to-end latencies among *any* pair of active Tor nodes. In other words, as Tor's user base increases, so too does Ting's applicability. We show that Tor's current user base span a diverse set of networks (including, in particular, residential hosts) from among ∼6000 unique /24 networks, making Ting a viable tool for wide-scale network measurement.[1]

This paper makes the following contributions:

- We present the design, implementation, and validation of Ting, a technique for measuring round-trip times between any arbitrary pair of Tor relays. Ting does not require modifications to or special participation from Tor clients. To the best of our knowledge, *Ting is the only practical tool today for measuring pairwise RTTs in the Tor network.*
- We thoroughly validate Ting's accuracy (80% of the time, its estimates are within 10% of ground-truth), stability (over a week, Ting's estimates vary by less than 5ms), and trade-offs between speed and accuracy.
- We present algorithms that use all-pairs latency measurements to drastically improve the time (a median 1.5× speedup) to deanonymize Tor circuits.
- Finally, we show that Ting's measurements can be used to improve path selection: We find an abundance of so-called triangle inequality violations in inter-Tor-node latencies, and show that circuits *longer* than three hops can be used to achieve *lower* end-to-end latencies.

The rest of this paper is structured as follows. In Section 2, we present related work on ascertaining and applying round-trip time estimations among arbitrary hosts. We present the design of Ting in Section 3, and an extensive validation of Ting in Section 4. In Section 5, we show several ways to apply Ting's measurements, including circuit deanonymization and improved path selection. We conclude in Section 6.

The Ting code and the latency datasets it generated are publicly available at

`https://www.cs.umd.edu/projects/ting`

## 2. BACKGROUND AND RELATED WORK

Our motivation derives from two sources: the desire to directly perform latency measurements between two hosts not under our control, and the desire to improve the performance or anonymity of Tor circuits by performance-informed path selection. With measured latencies between Tor nodes, we can determine whether the Internet paths follow geographic shortest paths, whether Tor performance is intentionally degraded, and whether there are opportunities for performance optimization. We can evaluate how often Tor's default random relay selection produces high latency paths and whether different path selection approaches might be more difficult to deanonymize. By taking advantage of Tor as a system that is representative of volunteer-administered overlay, we can also learn about the networks that host relay nodes (§5.3).

Prior work has observed that latency information would be beneficial, but have avoided attempting to incorporate it explicitly into the Tor protocol [20, 28]. Our approach is to use, rather than redesign, the protocol in order to measure inter-relay latency information. This permits immediate, incremental deployment of Ting-capable clients, since they can use existing Tor relays.

There are few workable alternatives for estimating the latency between Tor nodes. To make broad, immediate deployment possible, we cannot modify the Tor protocol, e.g., to ask relays to ping one another. There are large scale network measurement services that use hardware at clients to ping often: RIPE Atlas pings root DNS servers from specialized "probe" devices [24]. Such special purpose hardware [31, 25] could be applied to measure and construct a database of inter-node latencies, but they require users to deploy hardware in their networks. Conversely, Ting operates completely within the Tor peer-to-peer network, and does not require any additional user deployment. There has also been considerable work towards estimating inter-node latencies through the use of relatively few landmarks deployed throughout the network [6, 18, 33]. Such estimation systems offer considerably greater coverage than Ting—they can be applied to virtually any pair of nodes—but suffer from the fact that Internet latencies are inherently difficult to estimate accurately, e.g., due to triangle inequality violations [26, 15] (§5.2.1). Ting, on the other hand, achieves greater accuracy by performing *direct measurements*.

An approach that inspires us is that of Gummadi et al. [11], who aimed to estimate the latency between clients and servers by clever use of recursive DNS queries. Their "King" tool sent a recursive DNS request to a name server associated with the first host that could only be answered by a name server associated with the second. Although King required only that one of the two name servers support recursive queries, in recent years, DNS servers have stopped responding to recursive queries for concern over amplification. This means that using King directly is no longer practical.

Instead, in this paper, we attempt to apply the idea of King to Tor.

## 3. TING TECHNIQUE

In this section, we describe the design of *Ting*, our technique for determining the round-trip time (RTT) between two arbitrary Tor relays. We show that Ting is able to obtain theoretically accurate RTT measurements without requiring any modifications to Tor, without any explicit participation from other users, and while introducing only tiny amounts of work for other Tor nodes. Ting is able to do so by leveraging the fact that Tor allows users to select their own (almost arbitrary) end-to-end circuits. In Section 4, we show that these theoretical properties are upheld in practice.

[1] Ting's applicability is limited to where Tor can be deployed, and unfortunately, some countries censor Tor traffic to or from their residents [23].

### 3.1 Building Blocks

Ting has two major components. The first is the ability to construct nearly arbitrary end-to-end circuits. Fortunately, this can be done without requiring modifications to any Tor clients. In particular, we make use of Stem [30], a Tor controller that provides a clean programmatic interface for both constructing Tor circuits and attaching TCP connections to them. Even with this control, we are constrained to several natural policies that our local, unmodified Tor client enforces: (1) one-hop circuits are disallowed[2], and (2) a node cannot appear on a given circuit more than once. Both of these are logical policies for ensuring a user's anonymity, but we emphasize that we need not worry about anonymity with Ting—it performs its measurements explicitly, and does not, for instance, piggyback measurements on real user data. Nonetheless, we seek to be able to work within these constraints, so that we can operate without having to modify future versions of Tor and without requiring participation from other users.

Ting's second component is an end-to-end echo client and server to allow us to collect RTT measurements through Tor circuits. While similar in spirit to `ping`—which uses ICMP messages to estimate RTTs—the key difference is that our application operates over TCP, and can thus be used over Tor. As we will see, we do not also require `ping` itself: all of our measurements occur strictly over Tor circuits.

Both of these components are easily attainable and available online, but we make our code and data public for the community.

### 3.2 Strawman Approach

Consider the task of determining $R(x, y)$: the RTT between two Tor nodes $x$ and $y$. Suppose further that we have our echo client running at a source node $s$ and echo server at destination node $d$, both under our control.

An initially tempting approach is to use a method similar in spirit to the technique used by King [11], as shown in Figure 1:

1. Create a circuit $C$ through nodes $x$ and $y$, attach a TCP connection from $s$ to $d$ to this circuit, and measure the end-to-end RTT $R_C(s, d) = R(s, x)+R(x, y)+R(y, d)$.
2. Ping $x$ from $s$ to obtain an estimate of the RTT between $s$ and $x$: $\widetilde{R}(s, x)$, and similarly obtain an estimate of the RTT between $y$ and $d$: $\widetilde{R}(y, d)$.
3. Subtract the values from (2): $R(x, y) \overset{?}{=} R_C(s, d) - \widetilde{R}(s, x) - \widetilde{R}(y, d)$.

Unfortunately, we have identified two sources of error that make this approach untenable. First, note that it relies heavily on `ping` traffic taking a sub-path of that taken by the Tor traffic. However, not all packets are treated equally; we have observed that some networks treat ICMP and TCP traffic differently, exhibiting significantly different latencies for each, and in ways that we did not find easy to predict. Moreover, we found that some networks exhibited differential treatment for Tor traffic in particular. We find this unsurprising; given the perceived sensitive nature of Tor traffic, we expected network operators to, e.g., apply additional firewall or monitoring rules. But again, we were unable to predictably determine how this special treatment affected latencies: sometimes we saw higher `ping` times than Tor, and vice versa. In summary, we find that any technique that mixes Tor and non-Tor traffic is subject to uncertainty in path differences, and thus that *only traffic over Tor should be used.*

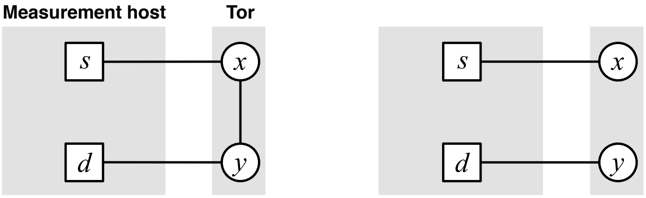


Figure 1: A strawman approach consisting of an end-to-end circuit (left) and direct RTT measurements (right). Unfortunately, because Tor does not permit one-hop circuits, direct RTT measurements would need to be performed using `ping`; because `ping` can experience noticeably different latencies, the estimate of $R(x, y)$ is likely to be inaccurate.

The second threat to validity with the above strawman is that it ignores the fact that, while processing Tor packets, each relay imposes a *forwarding delay*, which comprises (at least) the time to swap to and from user-space, the time the packet spends enqueued, and the time to decrypt and encrypt packets. Each of these time sinks is low in expectation, but because they can introduce additive errors, *forwarding delays must at be at least partially accounted for.* (We empirically evaluate Tor forwarding delays in Section 4.3.)

Guided by these two observations, we next present the design of Ting.

### 3.3 Ting Design

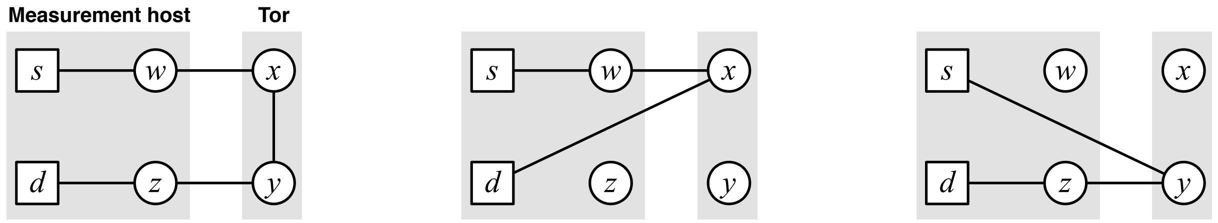
Ting determines the round-trip time between an arbitrary pair of Tor relays $(x, y)$. It extends the above strawman to operate strictly over Tor (instead of using incompatible `ping`s), and to account for Tor nodes' forwarding delays. With Ting, we run an echo client and server ($s$ and $d$), but run them on the same machine (or at least within the same subnet); the key distinction being that the RTT from $s$ to an arbitrary host is equal to $d$'s RTT to the same host.

As described in Section 3.2, we must avoid mixing Tor and `ping` measurements, but we also need the ability to isolate the RTT between the nodes we control and those we do not ($x$ and $y$). Ideally, we could create a one-hop circuit through $x$ or $y$, but recall that Tor disallows this. To this end, we locally run two Tor peers, $w$ and $z$, both also hosted within the same network as $s$ and $d$.

While this may seem like an extensive measurement infrastructure, in practice, we simply run all four processes on the same host $h$: the echo client and server ($s$ and $d$) and both of our Tor nodes ($w$ and $z$). However, our design extends to other configurations, as well.

Ting begins its measurement of $R(x, y)$ by first constructing a full circuit $C_{x,y} = (w, x, y, z)$, as in Figure 2(a). Ting then attaches to $C_{x,y}$ a TCP connection between $s$ and $d$, and measures the end-to-end RTT over this circuit. Let $F_i$ denote Tor node $i$'s forwarding delay. Then, because $s$, $w$, $z$, and $d$ are all running on the same host $h$, the overall RTT can be expressed as:

[2] In particular, by default, Tor relays refuse to be both the entry and exit node for a circuit.



(a) The RTT across the full circuit.

(b) Isolating the RTT to $x$.

(c) Isolating the RTT to $y$.

Figure 2: An overview of Ting's algorithm, capturing the steps necessary to isolate $R(x, y)$ by building multiple circuits through $x$ and $y$. Circles represent Tor clients, while squares represent our echo client and server.

$$\begin{aligned} R_{C_{x,y}}(s,d) \;=\; & R(h,h) + 2F_h + R(h,x) \\ & + 2F_x + R(x,y) + 2F_y \\ & + R(h,y) + 2F_h + R(h,h) \qquad (1) \end{aligned}$$

Each line of this equation represents a different part of the end-to-end path: the first line represents the RTT from $s$ to $w$ to $x$; the second line represents the RTT between $x$ and $y$; and the third line represents the RTT from $y$ to $z$ back to our server $d$.

The above equation contains our target measurement, $R(x, y)$, but it also contains additional latency measurements; in particular, we must subtract the RTT over Tor from $h$ to $x$ and from $h$ to $y$. To measure the RTT to $x$, Ting creates the circuit $C_x = (w, x)$, as depicted in Figure 2(b), and likewise for $y$, as in Figure 2(c). Attaching new TCP connections to them and measuring the end-to-end RTTs gives us:

$$R_{C_x}(s,d) = 2R(h,h) + 4F_h + 2R(h,x) + 2F_x \qquad (2)$$

$$R_{C_y}(s,d) = 2R(h,h) + 4F_h + 2R(h,y) + 2F_y \qquad (3)$$

Observe that all three equations account for two forwarding delays at their respective $x$ and $y$ nodes: this reflects the fact that there is both a ping and a pong message sent in opposite directions on the circuit.

All that remains is to calculate:

$$\begin{aligned} R_{C_{x,y}}(s,d) - \frac{1}{2}R_{C_x}(s,d) - \frac{1}{2}R_{C_y}(s,d) & \\ = R(x,y) + F_x + F_y & \qquad (4) \end{aligned}$$

We arrive at an estimate of the RTT between $x$ and $y$ with an expected error equal to the sum of the forwarding delays of the two nodes. The predominant factor in a node's forwarding delay is the number of other circuits and overall load at that node. For instance, if our measurement packet arrives at a node when our circuit is not first in the schedule, it will have to wait to be dequeued. To account for this, every time we measure an RTT, we do not limit it to a single sample; rather, we take multiple samples, and use the minimum value. We describe in Section 4 that, in practice, this results in minimum forwarding delays typically in the range of 0–3ms; for most pairs of $x$ and $y$, this is a negligible error within the typical variation of latencies.

## 3.4 Properties

Ting has several features that make it a feasible and effective measurement tool. Primarily, it operates strictly within Tor, meaning that we can be relatively certain that all measurement traffic will be treated equally: all packets will traverse the same paths, and through the same software stacks. As a result, the theoretical accuracy is high; although our final estimation (Eq. (4)) does not entirely eliminate forwarding delays, it accounts for them explicitly, and measures in such a way as to minimize their impact.

Additionally, Ting is trivial to deploy: it does not require any modifications to the Tor protocol, to existing nodes, or even to a local Tor client. Moreover, it can be run on a single host, as described above.

Finally, Ting can be applied to any pair of Tor nodes, regardless of whether they are exit nodes. In all of Ting's circuits, we use a node we control ($z$) as the circuit's exit node.

In sum, Ting is a novel, easy to use tool that provides unprecedented insight into the RTTs within the Tor network. In the next section, we validate Ting empirically and demonstrate that it can determine the latency of a pair of Tor nodes in less than 15 seconds with high accuracy.

# 4. TING VALIDATION

In this section, we validate Ting along several dimensions. First, using a set of globally distributed PlanetLab [21] nodes as ground-truth, we measure Ting's accuracy and evaluate how many samples are necessary to minimize errors caused by forwarding delays. Then, we turn to measuring Ting on the *real Tor network* and demonstrate that it achieves measurements that are consistent over time.

## 4.1 Ground-truth Testbed

In order to prove that Ting measurements are representative of the actual latency between two Tor servers, we require a ground-truth to compare to. To this end, we ran Tor relays on 31 PlanetLab [21] hosts. Further, to ensure that these hosts realistically reflect the geographically diverse Tor network, we guided our selection of random hosts such that:

- They covered a wide geographic area: 6 countries throughout Europe, 9 states throughout the U.S., and at least one relay in each of the following regions were represented: Asia, South America, Australia, and the Middle East.

- Their geographic distribution resembled that of the current Tor network, which contains a concentration

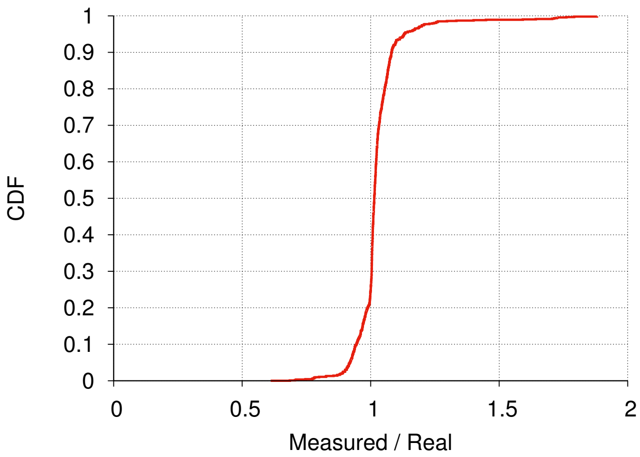


Figure 3: CDF of Ting's estimation for a given pair on PlanetLab relative to "real" latency values reported by `ping`.

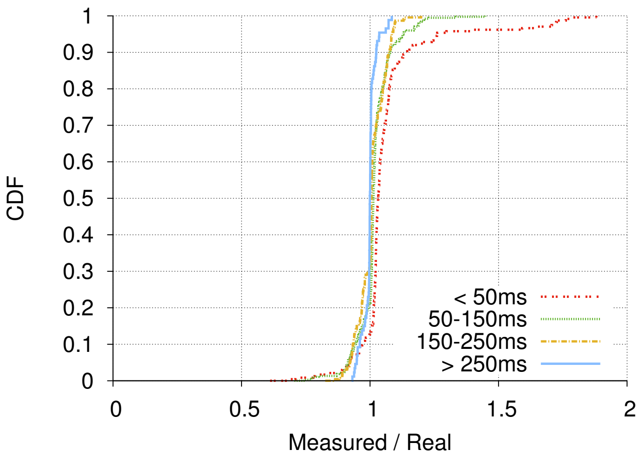


Figure 4: CDF of Ting performance separated into different intervals based on real latency.

of relays in the U.S. and Europe, and only a few nodes sparsely distributed throughout other countries.

- The latencies between all the pairs were unique and ranged from very close (~0ms) to nearly antipodal (~500ms).

We ran an unmodified version of Tor-0.2.4.22, with a restrictive exit policy that only allowed exiting to two specific IP addresses under our control (to avoid take-down notices to or from the institutions hosting the PlanetLab nodes). In addition, we maintained the relays for over a month before conducting our measurements in order to ensure that they would be receiving standard usage and traffic patterns in addition to our probe traffic, which we assume will be the situation for any arbitrary relays we attempt to measure.

For all of our experiments, we performed our measurements using two machines. The first machine ran both our client and server ($s$ and $d$ from Section 3). The client program was written in Python and controlled a Tor onion proxy (running a patched version of Tor-0.2.3.25) using the Stem controller library [30]. The server program, also written in Python, was an extremely minimal TCP-based echo server.

The second machine ran two instances of an unmodified version of Tor-0.2.4.22 to act as our relays $w$ and $z$. To demonstrate that Ting operates with minimal setup or modification, we allowed these two nodes to publish their descriptors to the Tor directory authorities, but this is not necessary; one can prevent Tor from publishing the descriptors (by supplying "`PublishDescriptors 0`" in the torrc configuration file) and simply hard-code the descriptors (mainly, the public keys) into the client's descriptor list.

## 4.2 Ting Accuracy

To begin our validation of Ting, we measure the RTT between all 930 pairs of nodes from our 31-node testbed, and compare these against direct, all-pairs `ping` measurements. We probe each pair in a randomized order, taking 1000 Ting samples[3] followed immediately by 100 `ping`s. As described in Section 3, we use the minimum sample from both techniques as their final estimate of the pairs' RTTs.

Figure 3 shows the distribution of the ratio between Ting's estimate and the ground-truth RTT measurement. In this plot, an $x$-value of 1 represents perfect accuracy. For 91% of the pairs, Ting's estimates are within 10% of the true value.[4] Fewer than 2% of all estimates had an error greater than 30%. The lack of an obvious skew to either side of $x = 1$ indicates that Ting is effective at ruling out the vast majority of errors caused by forwarding delays.

Ting is able to obtain such high rates of accuracy in large part because it is able to perform its measurements *directly* through the nodes whose RTTs it seeks to ascertain. In comparison, King [11] used DNS resolvers near the nodes it sought to measure, but could not completely account for the latencies between resolvers and their respective clients; as a result, King exhibits a distribution skewed to the left of $x = 1$ [11, Fig. 5].

To understand if Ting's accuracy is dependent on how large the true end-to-end RTT is, we break up the accuracy data into four latency regimes in Figure 4, based on the ground-truth RTTs. These results show that Ting is more accurate for pairs with greater end-to-end latency, as the CDF for each successive range becomes increasingly vertical and centered around $x = 1$, with the final range of 250–500ms having a nearly vertical curve at $x = 1$. We can also observe that a majority of outliers (identified as tails on the curve) come from pairs measured in the "$<$ 50ms" group, indicating that what appears to be a large relative error is in fact attributable to a small absolute error.

For some applications, it suffices to know only the rank order of latencies [11]. The Spearman's rank-order correlation between Ting's estimates and our ground-truth data is 0.997 (a value of 1.0 denotes perfect agreement in rank order).

We find these results extremely encouraging, particularly given that they were collected on PlanetLab nodes, which

[3] We show in Section 4.4 that Ting can achieve comparable accuracy with far fewer samples.

[4] For the remaining pairs, we observed errors distributed mostly uniformly across the hosts we measured (as opposed to a few nodes being highly erroneous).

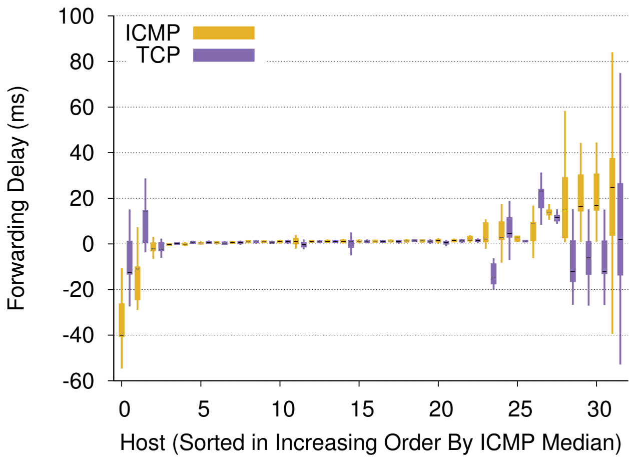


Figure 5: Forwarding delays measured across 31 Tor relays. Abnormal values indicate networks that treat ICMP, TCP, and Tor traffic differently. For all other networks, Ting can consistently find low forwarding delays.

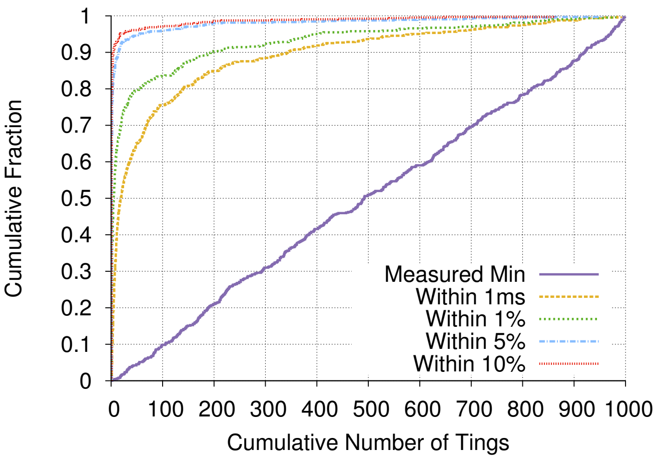


Figure 6: Number of samples required to reach different approximations of the minimum of 1000 samples.

are known to exhibit high variance in their computation and communication speeds.

To summarize, these ground-truth results show that Ting provides an accurate method of estimating the round-trip times between hosts on the Tor network without requiring modifications to Tor or explicit participation from other Tor users. Moreover, our experimental setup included nodes that were geographically distributed and not dedicated to our use—PlanetLab is a shared infrastructure. Thus, these experiments additionally show that Ting's accuracy is not highly dependent on factors about the host which are out of our control. This is an important implication moving forward in assuming that Ting can be used to measure all relays on the network.

## 4.3 Forwarding Delays

In Section 3.2, we made two claims about forwarding delays that we now seek to validate: (1) that forwarding delays are typically negligible, but (2) that they must be at least partially accounted for in order to minimize the error introduced in the event that they are non-negligible.

In order to measure the forwarding delay for a given node $x$, we apply the following approach, which resembles our method of measuring latency:

1. Set up an echo client $s$ and echo server $d$ on one host, and two instances of Tor, $w$ and $z$, on a second host.

2. Create a circuit $C_1 = (w, z)$ from $s$ and probe $d$ to measure end-to-end RTT $R_{C_1}(s, d) = R(s, w) + F_w + R(w, z) + F_z + R(z, d)$.

3. Use `ping` or `tcptraceroute` from $s$ to $w$ to obtain $\widetilde{R}(s, w)$ and $\widetilde{R}(z, d)$, the *estimated* RTTs between the hosts, which should be equivalent as $w$ and $z$ are running on the same host.

4. Subtract these latencies to calculate the forwarding delay of $w$ and $z$, $F_w = F_z = \frac{R_{C_1}(s,d) - \widetilde{R}(s,w) - \widetilde{R}(z,d)}{2}$ (here we leverage the fact that $\widetilde{R}(w, z) \approx 0$).

5. Create a circuit $C_2 = (w, x, z)$ from $s$ and probe $d$ to measure the end-to-end RTT $R_{C_2}(s, d) = R(s, w) + F_w + R(w, x) + F_x + R(x, z) + F_z + R(z, d)$.

6. Use `ping` or `tcptraceroute` from $s$ to $w$ and from $w$ to $x$ to obtain $\widetilde{R}(s, w) = \widetilde{R}(z, d)$ and $\widetilde{R}(w, x) = \widetilde{R}(x, z)$, estimates of the respective RTTs.

7. Subtract the values from (4) and (6): $F_x = R_{C_2} - F_w - F_z - 2\widetilde{R}(w, x) - 2\widetilde{R}(s, w)$

We use both `ping` (which uses ICMP) and `tcptraceroute` (which uses TCP) to investigate whether it is necessary or sufficient to measure RTTs using the same transport layer protocol as Tor's.

We applied this technique to compute the forwarding delay for our set of 31 Tor relays running on PlanetLab once an hour over a 48-hour period. Figure 5 shows the distributions of our measurements over time, with individual nodes sorted by median `ping`-measured forwarding delay. The box-plots capture the median, interquartile ranges, and minimum and maximum values within the interquartiles. Nearly 65% of all nodes exhibit a distribution tightly closed around the range of 0–2ms with very little variance. These nodes match our expectations of a low minimum forwarding delay: if a packet arrives at a Tor relay very shortly before when it would be scheduled to be dequeued, then the forwarding delay should consist only of the time to process the packet, which mostly consists of symmetric key cryptography, and is thus fast.

However, the remaining 35% of nodes show extremely odd behavior, with forwarding delays often *negative*. These abnormalities are due to our observation from Section 3 that not all packets are treated equal: negative forwarding delays indicate that it took less time to communicate with a node over Tor than to directly ping that node, sometimes on the order of tens of milliseconds. This is simply not possible unless packets follow different paths. Using both ICMP- and TCP-based measurements makes this even more stark; we see for these outliers consistent disparity between how ICMP and TCP packets are treated. We therefore place very little confidence in forwarding delays measured from networks that exhibit such disparate behavior; of all remaining networks, we see only near-zero estimates of forwarding delay.

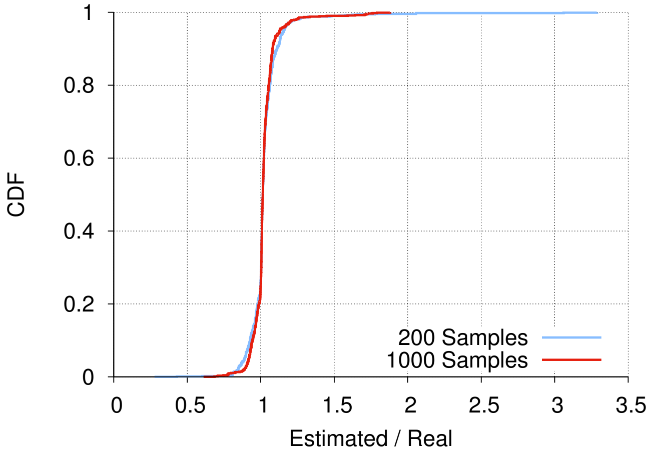


Figure 7: Comparison of taking 200 samples for each pair vs. 1000 samples for each pair.

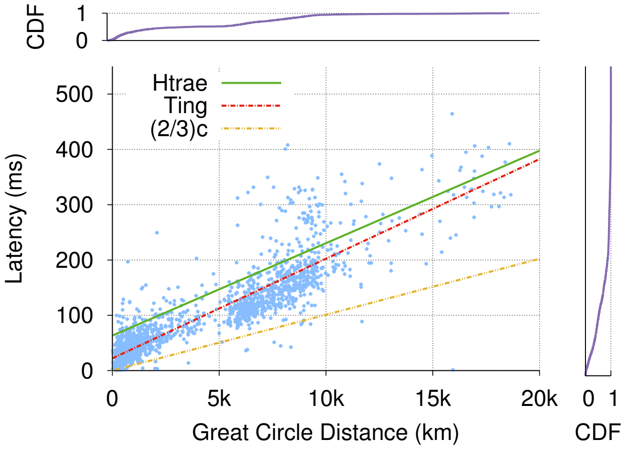


Figure 8: Relationship between geographic distance and RTTs measured by Ting, using GPS coordinates from Neustar IP Geolocation [17].

We draw two important conclusions from this experiment. First, for networks that do not appear to differentiate how they treat packets of different protocols, Ting observes forwarding delays to be relatively negligible, at ∼2ms. This explains why our theoretical estimate in Eq. (4) is able to perform so well in practice. Second, this experiment concretely demonstrates the importance of avoiding using `ping` or even TCP-based RTT estimation techniques in combination with Tor-based measurements. Collectively, these results validate Ting's approach and accuracy.

## 4.4 Sample Sizes

The Ting algorithm takes as a parameter the number of times to sample each circuit, which allows one to adjust the balance between speed of measurement and accuracy. Since we only end up using the minimum value of all RTTs measured for each circuit, the question is: how many samples does it take to reach a true minimum, or at least how many does it take to reach within an acceptable interval of this minimum?

Jansen et al. [13] observed that it can sometimes take an incredible number of latency samples through Tor before obtaining a true minimum. We recreated this experiment by using Ting to measure 100 random pairs of *live* Tor nodes, taking 1000 samples from each circuit. In Figure 6, we present how many iterations were necessary across all pairs to obtain the minimum estimate, and approximations thereof. This confirms the prior result of Jansen et al.: it does indeed take a considerable number of iterations to reach the actual minimum. However, we also present the number of iterations necessary to get *close to* the minimum. Even modest departures from minimum can be found far more quickly; for instance, to get within 1ms of the minimum requires roughly 25× fewer probes at the median.

To further solidify this fact, we re-measured the latency between the 930 pairs of Planet Lab nodes from our experiment in 4.2, this time taking 200 samples rather than 1000. In Figure 7, we plot the CDF of the ratio between measured and real values (again using `ping` as a ground-truth for RTTs) and compare this to the CDF of taking the minimum of 1000 samples. The fact that the CDFs are almost identical reinforces our claim that there is very little benefit to taking a large number of samples.

Because the number of samples is a tunable parameter, one can turn it down even further to measure more quickly. For instance, in our experiments, Ting took an average of 2.5 minutes to measure a pair using 200 samples in order to ensure very high accuracy. However, if one were willing to accept 5% error, then Ting could measure a pair in less than 15 seconds on average. For the remainder of the experiments in this paper, we continue using 200 samples.

## 4.5 Ting on the Live Tor Network

We now investigate Ting's effectiveness in measuring pairs in the wild, outside of our controlled environment, and without the limitations imposed by PlanetLab. We used Ting to measure 10,000 pairs of Tor relays, chosen uniformly at random from the pool of all currently running relays. Since we cannot directly compare the accuracy of specific measurements in this scenario, we now look at general trends.

Figure 8 shows the relationship between the Ting-measured RTTs and the geographic distance for each of the 10,000 pairs of Tor nodes. We used the Neustar IP Geolocation service [17] to obtain an estimate of the GPS coordinates for each of the relays, and calculated the great circle distance between each pair. Additionally, in the margins, we plot CDFs of the two axes to show the relative distribution of latencies and distances covered. We annotate the plot with three lines. One line represents the generally accepted *maximum speed* that packets can traverse a given distance in the Internet: 2/3 the speed of light. This serves as a sanity check; indeed, we see only a handful of points below this line. Manually inspecting these nodes, we see that they are almost all likely errors in the underlying geolocation database.

The other two lines represent fits to more accurate latency datasets. At the top is the estimate derived from a large study of latencies among Halo gamers as part of the Htrae system [1], and below that is our linear fit to our own data. The gap between these two lines is due to the fact that Htrae measured *median* latencies, while we seek to detect the minimum latency between a pair of nodes. Their

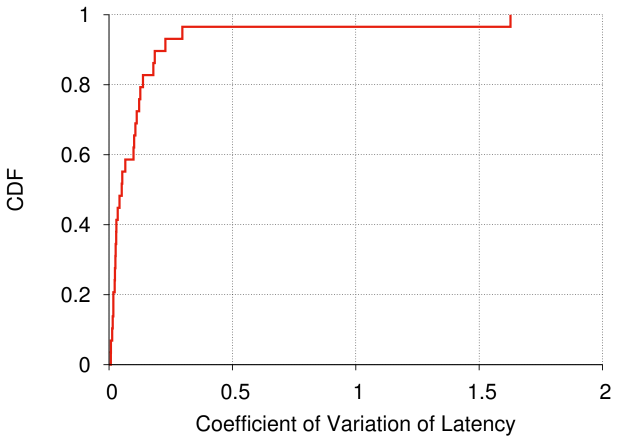


Figure 9: Measuring the latency between a pair of relays using Ting produces relatively consistent results over time.

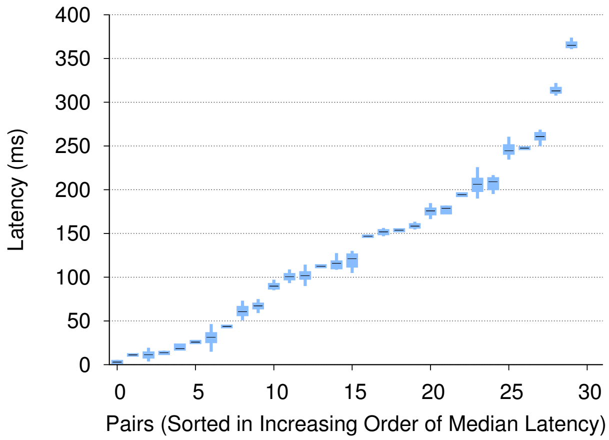


Figure 10: Larger relative variance is revealed as small absolute errors when the mean is low.

similar values are encouraging validation. However, one interesting feature of our linear fit is that it has a greater slope. We believe this is due to the surge of latencies between 5000 and 10,000 km: these are likely international links that are traversing at least one country's border. Note, for instance, that the CDF at the top of the plot does not have the same shape as the one on the right; if the relationship was strictly linear, they should have the same shape. We speculate that this is evidence that, at least for international circuits, Tor traffic is being treated differently than more traditional traffic, such as gaming. More investigation into these differences is an interesting area of future work.

## 4.6 Stability of Ting Measurements over Time

In the following section, we demonstrate the impact that an all-pairs RTT matrix can have on various applications. However, an all-pairs matrix can be time-consuming to calculate. Here, we measure whether Ting captures snapshots that are representative over time—that is, how *stable* are Ting's measurements?

In order to evaluate the stability of Ting measurements over time, we picked a set of 30 pairs of Tor relays and measured the RTT between them once an hour over the course of a week. The pairs were chosen such that: (1) Both relays were running for over a month, in order to mitigate the chance that they would go down over the course of the experiment, (2) The distribution of the RTTs of the pairs would match the distribution shown in Figure 8, which displays a relatively uniform distribution from low to high latencies, in order to observe the effects of RTT on variance.

In Figure 9 we plot the CDF of the coefficient of variance $c_v$ (the standard deviation normalized to the mean) which can be used to compare the degree of variation between a series of data sets with different means. 96.7% of all pairs (all but one pair) have $c_v < 0.5$, indicating high stability; further, over 50% of pairs have $c_v \approx 0$.

Although the $c_v$ measures variance well in a majority of cases, it is very sensitive to changes when the mean is low. Figure 10 provides another view of the data, displaying Ting's distribution of measurements for each pair. Again, these boxes show the median, interquartiles, and the maximum and minimum within the interquartile ranges. The outlier in Figure 9 corresponds to the first pair in Figure 10, indicating that the high $c_v$ was in fact biased towards having a low mean (∼3ms). 67% of the pairs do not show a single outlier, and have third and first quartiles that differ by less than 5ms. Even in the remaining cases where the latencies span a wider range, such as pair 15, it is worth noting that the outliers are still relatively close to the mean, and do not represent a very large error. This suggests that taking measurements with Ting infrequently and caching them is sufficient, and thus permits obtaining a large dataset of RTTs between Tor nodes.

## Summary

This section has shown that Ting is incredibly accurate because it can measure nodes *directly* rather than through a nearby proxy, as with prior approaches [11]. Moreover, we demonstrated that Ting's measurements are resilient over time, and do not require many samples to obtain reasonable accuracy. Ting can thus be used to generate all-pairs RTT measurements of the Tor network. In the next section, we show how we can leverage such an all-pairs matrix to benefit various applications.

# 5. APPLICATIONS

In this section, we discuss three disparate applications that benefit from the highly accurate RTT measurements Ting provides. We provide what we believe to be the first deanonymization technique that precludes certain circuits through application of latency measurements. We also show how Ting can be used to find longer circuits that results in *lower* end-to-end latency. Finally, we show Ting's value as a network measurement platform by evaluating the diversity of Tor relays.

## 5.1 Deanonymization of Tor

Tor seeks to balance anonymity with low-latency communication, and as a result, various techniques have been introduced to *deanonymize* users by introducing small but noticeable fluctuations in their latencies [16, 9, 12, 10].

A common form of deanonymization assumes that the attacker is somewhere on the path, from source to destination: either the attacker is on the three-node Tor circuit (consisting of an entry node, a middle node, and an exit node), or it is the destination itself (e.g., in the case of a malicious server). The attacker's goal is to determine all of the nodes on the circuit, as this has been shown to assist in determining the source and destination [12].

Active probing attacks have been shown to make deanonymization with an on-path attacker possible. The attacker can determine if Tor relay $t$ is on the victim source-destination path by (1) creating many circuits through $t$ and sending traffic through them, and (2) seeing if this induces extra delay on the victim's packet inter-arrival times. This is a somewhat heavy-handed approach to deanonymization—it is expensive for an attacker to launch, as it requires creating multiple circuits simply to rule out a single Tor relay. For such attacks to be feasible in practice, it is important that the number of active probes performed remain small.

### 5.1.1 Speeding up deanonymization with Ting

Here, we consider how knowledge of RTTs between all pairs of Tor nodes can speed up existing deanonymization algorithms. The specific setting we consider is when the attacker is the destination: he already knows the exit node, and wishes to determine the entry and middle nodes. Our insight is, broadly speaking, that because the attacker knows the end-to-end RTT $R_{\mathsf{e2e}}$, then we can rule out any circuit whose hops' RTTs add up to greater than $R_{\mathsf{e2e}}$.

More concretely, consider the standard, RTT-unaware deanonymization process. Initially, the attacker knows: its RTT $r$ to the exit node, the exit node $x$ itself, and the end-to-end RTT $R_{\mathsf{e2e}}$. Suppose during the deanonymization algorithm, the attacker learns that Tor node $c$ is on the circuit. Let $R(a, b)$ denote the RTT between Tor nodes $a$ and $b$. Then the attacker learns the following by "ignoring too-large RTTs":

- If there exists no potential entry node $e$ such that $R(e, c) + R(c, x) + r \leq R_{\mathsf{e2e}}$, then $c$ cannot be the middle node in the circuit, and therefore must be the entry node.
- Alternatively, if there exists no potential middle node $m$ such that $R(c, m)+R(m, x)+r \leq R_{\mathsf{e2e}}$, then $c$ cannot be the entry node, and must be the middle node.
- If $c$ has been identified as the entry node, then any node $m$ for which $R(c, m) + R(m, x) + r > R_{\mathsf{e2e}}$ cannot be in the circuit, and therefore $m$ need not be tested.
- Similarly, if $c$ has been identified as the middle node, then any node $e$ for which $R(e, c) + R(c, x) + r > R_{\mathsf{e2e}}$ cannot be in the circuit and need not be tested.

Each of these rules is somewhat conservative; the inequalities do not take the RTT between the source and the entry node into account, and thus the above criteria will likely remove only extreme outliers. The RTT information that Ting provides can further speed up deanonymization by preferentially testing nodes who are more likely to be on the circuit. Our insight is as follows: Nodes choose circuits at random, and therefore, if for node $i$ to be on the path, the source would have to be improbably close to or far from an entry node, then $i$ is probably not on the path. Of course, the source being very close to or far from $i$ is not definitive proof that $i$ is not in the circuit, so we do not rule $i$ out. Instead, we assign a "score" to each node, and preferentially test nodes with the lowest score. More concretely, we apply Algorithm 1 at every iteration of the deanonymization process.

**Algorithm 1** Informed target selection for fast deanonymization.

```
1. For each node $i$ who has not yet been ruled out:
    (a) Enumerate all possible circuits $C$ involving $i$, after
        applying the above criteria for ignoring too-large RTTs.
    (b) Let $R(c)$ denote the sum of RTTs of circuit $c$, and
        let $\mu$ denote the average RTT among all pairs of
        Tor nodes; then $i$'s score is $\min_{c \in C}\{|R_{\mathsf{e2e}} - (R(c) + r + \mu)|\}$.
2. Probe the node with the lowest score, and then apply
   the criteria for ignoring too-large RTTs.
```

This algorithm uses $\mu$—the average RTT across the entire all-pairs data supplied by Ting—to approximate the expected (average) RTT between the source and its entry node. Thus, this algorithm chooses the node whose expected end-to-end latency, $R(c) + r + \mu$, most closely approaches the measured end-to-end latency $R_{\mathsf{e2e}}$.

**Weighted Node Selection.** As stated, our informed target selection algorithm (Alg. 1) assumes that each node in a Tor circuit is chosen uniformly at random from the set of all active nodes. In practice, Tor no longer operates this way, but rather assigns a *weight* to each node, reflecting how its measured bandwidth compares to the overall population's. The benefit of preferentially choosing higher-capacity nodes is that it increases the throughput of the overall circuit, but the downside is the circuits become more predictable. These weights can be incorporated into our algorithm by simply dividing each node's score by the node's weight. However, in the remainder of this section, we evaluate Ting in its worst case scenario—when all weights are equal (traditional Tor).

### 5.1.2 Evaluation

To evaluate how well RTT values can speed up deanonymization efforts, we used Ting to generate an all-pairs RTT dataset among a set of 50 randomly chosen Tor nodes. We present the distribution of RTTs in Figure 11, and note that the shape of the distribution is consistent with that from Figure 8. Using this all-pairs dataset, we simulate three deanonymization techniques, each of which assumes the existence of a technique such as that described by Murdoch and Danzeis [16] to brute-force probe whether a given Tor node is on a circuit. For all of our results, we use 1000 runs from our simulator, with the source chosen uniformly at random from the set of Tor nodes. In addition to the two techniques described above (that which ignores too-large RTTs, and our informed target selection), we also include as a baseline an RTT-unaware technique that simply brute-force tests nodes until it has discovered the entire circuit.

The critical evaluation metric is how many probes it takes to deanonymize a circuit. Recall that, because such brute-force probes are more bandwidth-intensive and time-consuming in practice, techniques which require fewer probes will finish more quickly and with less impact on the network as a whole.

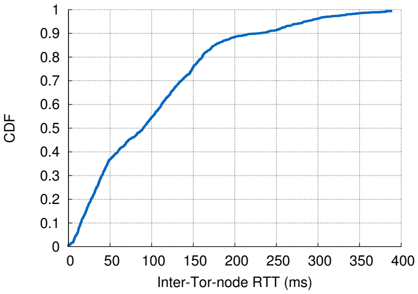


Figure 11: Distribution of RTTs from running Ting on all pairs of a random 50-node set of Tor nodes.

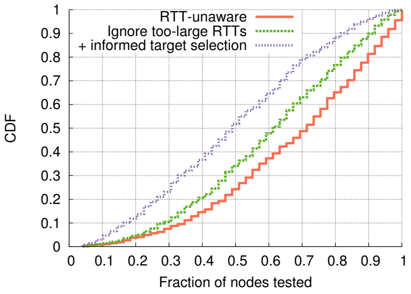


Figure 12: Knowledge of RTT between all Tor nodes speeds up deanonymization.

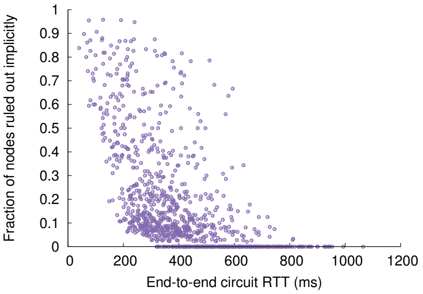


Figure 13: RTT knowledge is particularly useful when deanonymizing circuits with lower end-to-end RTT.

Figure 12 shows the distribution of the fraction of nodes each technique needed to probe in order to deanonymize the circuit. Without any RTT knowledge, determining the entry and middle nodes of the circuit requires probing a median of 72% of the network. Simply ignoring too-large RTT values improves this noticeably, requiring probes to a median of 62% of the network. Incorporating the more intelligent target node selection described in Algorithm 1, we see an even more drastic decline, requiring probes to a median of 48% of the network. In other words, through the application of Ting's RTT information, deanonymization efforts can experience a 1.5× speedup.[5]

Next, we seek to better understand for what circuits Ting's information helps deanonymize. Intuitively, our deanonymization algorithms are best suited for circuits with low end-to-end RTTs. As a simple example: if the end-to-end RTT were 50ms, then any node with an RTT greater than 50ms to the exit node could not be the middle node in the circuit. On the other hand, if the end-to-end RTT were over one second (larger than any single RTT we measured), then any node could ostensibly be in the circuit.

Using the same 1000 runs from our simulator, we plot in Figure 13 the fraction of nodes that we could ignore due to too-large RTTs, and compare that to the end-to-end RTT—from the source, through the circuit, to the destination. This shows a strong correlation: the lower the end-to-end RTT, the more nodes Ting helps us to rule out, while for the absolute highest RTTs, Ting's information was not helpful. However, interestingly, Ting successfully speeds up deanonymization efforts of circuits with moderate to high RTTs. We conclude that, along with the algorithms from this section, Ting can considerably speed up the deanonymization for most circuits.

### *5.1.3 Defenses*

Unfortunately, defending against this kind of attack is difficult; as long as there is a pair of nodes that provide lower latency than other pairs, this deanonymization attack could apply. One countermeasure would be to artificially inflate latencies within a circuit, but the Tor designers do not appear willing to accept this cost [7]. Another approach that would slow down, but not completely eliminate, this deanonymization attack would be to randomize the *length* of circuits. The primary concern with this approach is that longer circuits typically result in greater latencies, but as we will show in Section 5.2.2, latency measurements from Ting can guide the creation of longer circuits without higher latency.

In summary, Ting's ability to measure all-pairs RTT among Tor nodes can considerably speed up existing deanonymization techniques, particularly (though not exclusively) with smaller end-to-end RTTs. We believe that these techniques can be improved further, and that it is an interesting area of future work.

## 5.2 Improving Path Selection in Tor

Recall that a typical Tor circuit consists of three relays: one guard, one middle, and one exit. By default, a Tor client selects these relays at random according to the bandwidth capacity of each router, as reported by the set of trusted Tor bandwidth authorities.[6] The rationale behind using three hops is that it is the minimum required to provide unlinkability between source and destination, and has thus been expected to avoid extra end-to-end latency.

There have been considerable efforts toward reducing end-to-end RTTs [2, 28, 20, 8], but lacking the ability to efficiently predict a circuit's RTT complicates these efforts. For instance, LASTor [2] relies on geographic distances as a proxy for latencies; while we have shown a strong correlation between distance and RTT (Section 4), we demonstrate here that there are many instances where latency can be reduced in ways that geographic distance *cannot* predict.

Here, we investigate whether explicit measurements can guide path selection toward paths that have lower latency or longer length at modest performance cost.

### *5.2.1 Triangle Inequality Violations*

A *triangle inequality violation* (TIV) in routing occurs when the lowest-latency path between two nodes $s$ and $d$ is not the direct path between them ($s \leftrightarrow d$), but rather through an intermediate ($s \leftrightarrow r \leftrightarrow d$). In other words, when there is a TIV, the shortest distance between two points is not a straight line, and a relay can provide access to this shorter path [26, 15].

[5] We have also evaluated the weighted version of our deanonymization algorithms, and find that, compared to a scheme that tests nodes in decreasing order of weight, the Ting-based approach speeds up deanonymization by a median of 2×; an even greater improvement than with unweighted node selection.

[6] Tor applies several other filters, such as requiring that relays come from distinct /16s; our results extend to such criteria, but we do not enumerate them here for ease of presentation.

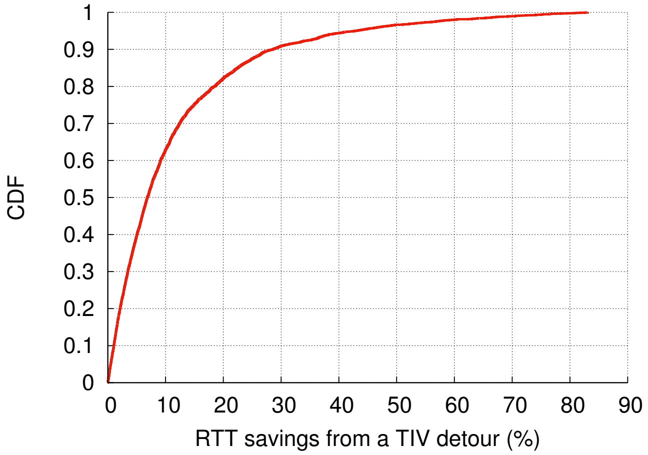


Figure 14: Savings from using a TIV relay instead of direct paths between two Tor nodes.

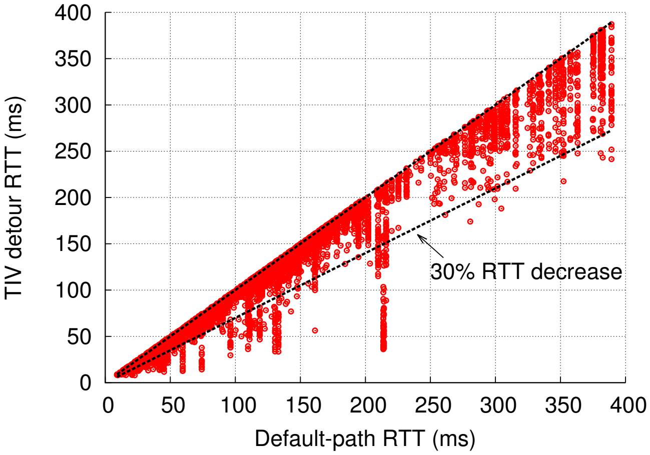


Figure 15: TIV-capable pairs are not relegated to any particular range of RTTs.

Here, we investigate to what extent TIVs exist in Tor, and whether they permit lower-latency paths to a wide range of circuits. To investigate this question, we make use of our 50-node all-pairs Ting dataset, and simply identify all pairs of nodes $(s, d)$ such that there exists a node $r$ for which $R(s, d) > R(s, r) + R(r, d)$. Surprisingly, we find that, for 69% of all pairs of Tor nodes in our data, there exists at least one circuit that results in a TIV.[7]

Figure 14 shows the distribution of the percentage improvement in RTT from using a TIV relay. Here, the $x$-axis reflects the ratio between $R(s, r) + R(r, d)$ to $R(s, d)$. The median decrease in latency is 7.5%; this is somewhat modest, but an all-pairs RTT dataset allows us to choose relays that offer greater RTT savings: 10% of TIVs reduce RTTs by 28% or more.

To show that TIVs in Tor are not specific to high or low latency paths, we plot in Figure 15, for every TIV we find, the correlation between the default-path RTT and that through the TIV relay. The number of available TIVs and overall savings are mostly consistent, regardless of the default path's RTT. Substantial drops below $x = y$ typically indicate performance-insensitive Internet routing; smaller drops may indicate congestion. Greater detail about such paths is provided by Detour [26] or PeerWise [15].

Using Ting, we are able to demonstrate the prevalence of TIVs in the Tor network. This has two important ramifications: First, *geographic distance is an imperfect proxy for RTTs in Tor.* Distances do not violate the triangle inequality, while Tor often does. Direct measurements of node-to-node latencies are necessary to find these paths. Second, the traditional assumption that longer circuits yield greater end-to-end latency is not necessarily true: *longer circuits can reduce latencies*, if chosen in a way that favors TIVs. Encouraged by this result, we now investigate: do circuits that are long but quick compromise user anonymity?

### *5.2.2 Longer Circuits*

Tor's default of three-hop circuits balances between anonymity and end-to-end latencies, but are there longer circuits that actually *reduce* latencies? Longer circuits may increase anonymity by frustrating tracing, but of course increases the resource use at relays. To answer this question, we again make use of the 50-node, all-pairs RTT dataset provided by Ting, and calculate circuits with lengths from 3 to 10. We sample 10,000 random circuits for each length $\ell$ and scale our results to the maximum number of circuits: $\binom{50}{\ell}$.

**Round-Trip Times.** Figure 16 shows, for varying circuit lengths, how many circuits are available for a given RTT. Clearly, circuits with more hops are able to obtain higher maximum RTTs: for instance, we identified no 3-hop circuits in our dataset with RTT over 1 second, but we identified over 1M 10-hop circuits with RTT over 2 seconds. More interestingly, observe the range of 200–300ms; in our dataset, we find roughly 10,000 3-hop circuits able to achieve RTTs in this range. But we are able to find an order of magnitude more 4-hop circuits able to achieve the same RTTs—for 10-hop circuits, there are *four orders of magnitude* more circuits capable of achieving the same RTT. This scales up so drastically because of how quickly $\binom{50}{\ell}$ grows: though the probability of any given $\ell$-hop circuit having a given RTT is low, there are many circuits for even moderate sizes of $\ell$.

These results show that, if chosen with knowledge of inter-Tor-node RTTs, *longer paths can be used in lieu of default 3-hop paths without imposing greater RTTs.*

**Circuit Diversity.** One potential concern, however, is that these long circuits with low latency reduce anonymity by relying on a few well-connected nodes. To evaluate this concern, we plot in Figure 17, for each circuit length, the median probability of a node being on a path with RTT = $x$. Intuitively, this metric captures how "entropic" the set of circuits are for a given RTT and path length. For instance, in our dataset, the set of 10-hop circuits capable of achieving an RTT of 1.9 seconds is very small: this is reflected by the small value at $x$ =1.9. Figure 17 shows that, for many values of circuit length, low-latency circuits do not rely on a small

[7] Since Ting estimates *minimum* RTTs, this indicates that 69% of pairs may exhibit at least one TIV, but does not directly allow us to reason about the average reduction in latency.

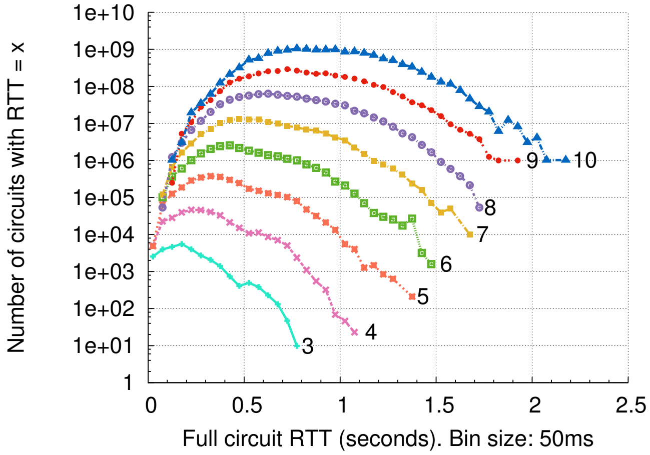


Figure 16: Longer Tor circuits have more options for lower-latency paths, as well as higher-latency paths. Each line is annotated with its corresponding circuit lengths.

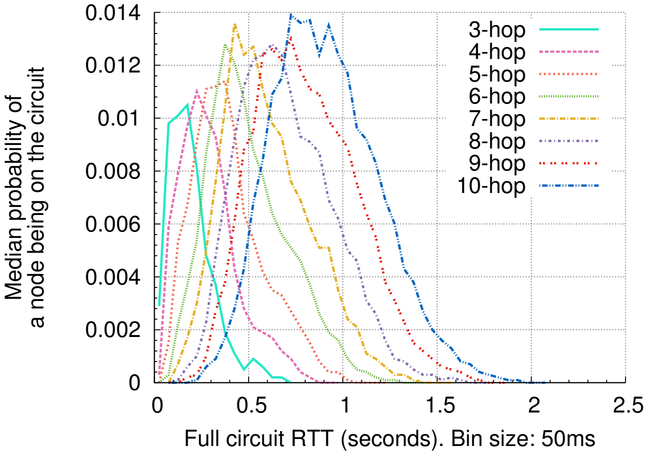


Figure 17: Choosing shorter or longer paths induces skew in the probability of a given node being selected to be on a circuit.

fraction of peers; it is only 10-hop circuits which significantly sacrifice entropy for RTTs less than 500ms.

Using measurements like those in Figure 17, if an attacker knows a victim's circuit length and end-to-end RTT, then he may be able to quickly pare down the set of potential circuits in use. The most entropic region for any given circuit length is naturally in the intermediate RTT values; choosing circuits in this range of RTT values thwarts such attacks.

A user can increase the number of potential circuits for a given RTT by consulting these data that Ting provides. For instance, suppose the user seeks a circuit in the range of 200–300 ms. Within this range of RTTs, there are many 3-hop, 4-hop, and 5-hop circuits. As a result, were an attacker to learn the end-to-end RTT (e.g., if the attacker were running a web server and sought to identify which users were connecting via Tor), there would be over two orders of magnitude more circuits that the user could have constructed than if he had restricted himself only to 3-hop nodes.

**Ramifications of Longer Circuits.** The use of longer circuits represents a tragedy of the commons: the end-users may benefit, but longer circuits consume more resources from the Tor network as a whole. However, because circuits are determined strictly by the source, there is little the system can do to prevent a selfish but rational user from adopting longer circuits. We believe that approaches to provide incentive-compatibility in Tor [19, 3, 5] are of increasing importance as longer circuits become feasible.

Although longer circuits can help defend against attackers external to Tor, it is less clear how they fare in the presence of active adversaries within the Tor network. Assuming a fixed fraction of active, bad Tor nodes, as circuits grow longer, the probability increases that an adversary is one of the chosen hops—this, too, makes deanonymization by traffic analysis possible [14]. Whether this increased probability outweighs the obfuscation of longer circuits merits future study.

The results in this section suggest that there is potential for a larger design space than Tor's three-hop default: longer hops need not induce greater latency. How this should be leveraged to achieve different balances between security and performance is an interesting area of future work.

## 5.3 Ting as a Measurement Platform

The final application we consider is arguably Ting's most natural: as a platform for directly measuring latencies between hosts that researchers may not otherwise have access to.

King [11] measured the latency between DNS servers thought to be near clients or servers in the Internet to estimate the latency between those hosts. In 2002, Gummadi et al. found that 72–79% of authoritative name servers supported remote recursive queries to support King; we find that only 3% continue to today, presumably due to concerns over amplification attacks. This change inspired us to apply Ting to the same problem: to estimate latency between hosts thought to be near Tor relays. Relative to King, Ting has an advantage in accuracy in that the Tor node representing a prefix is a member of that prefix, rather than an authoritative name server that may be much better connected or remote. However, Ting has the disadvantage in that not every residential subnet has a Tor node. In this section, we evaluate this coverage and show that Tor nodes are spread between residential networks and data centers.

**Coverage.** We can evaluate Ting's coverage in three dimensions: geography, network, and host type. The geographical coverage of Tor is well-known: On one hand, many countries are represented: Tor Metrics [32] reported 77 countries with relays in November 2014. On the other hand, there are exceptional countries in which Tor is blocked.

In terms of network diversity, we show the number of unique /24 IP address prefixes and the total number of relays in the Tor network from February 28, 2015 to April 28, 2015 in Figure 18. We consider /24 prefixes because they represent a network allocation likely to be geographically clustered. At any point in this two-month window, we observed between 5426 and 6044 unique /24s. Compared with past data collected by Tor Metrics [32], the total number of relays has grown by roughly 30% since one year prior. If the

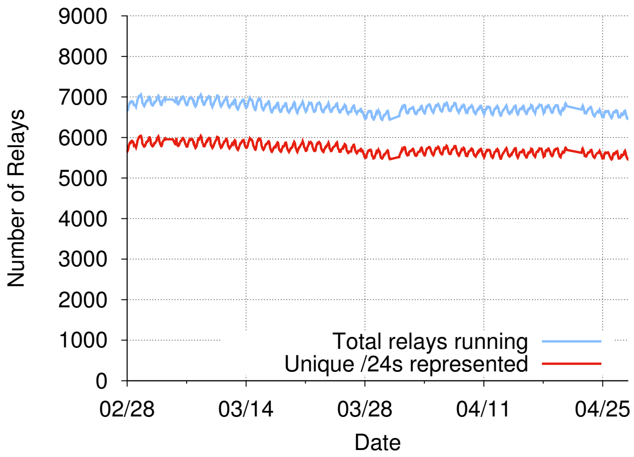


Figure 18: Number of /24s represented in the Tor network from February to April 2015.

Tor network continues to grow, it will become increasingly useful for medium-scale measurement across the network.

Finally, host types are diverse. The pool of relays is predominantly volunteer-run, and thus many relays are run from within homes. To quantify the number of residential hosts, we extended the residential detection technique described by Schulman et al. [27], which involves classifying hosts based on their reverse DNS name, including suffix and presence of numbers—the original technique is only intended for hosts within the U.S., while our extension also considers hosts in Europe. Using this technique, we found that, of the 5484 currently running Tor relays with a reverse DNS name, at least 3355, or roughly 61%, are residential. This underestimates the fraction of residential hosts both because there are a number of Tor relays outside of the U.S. and Europe which are not accounted for, and because the remaining 1150 of the 6634 currently running relay addresses have no reverse DNS name. Data centers are also represented: 361 are at hosting sites identified by reverse DNS name (`linode.com`, `amazonaws.com`, `ovh.com`, `cloudatcost.com`, `your-server.de`, and `leaseweb.com`), and another 345 are within Digital Ocean's IP address range.

### Summary

This section demonstrated a wide swath of applications that benefit from Ting's accurate RTT measurements. We believe the set of applications to which Ting is beneficial is both broader and deeper than what is covered here. But even with these few examples, we can conclude that Ting's accurate, all-pairs RTT estimation can be an extremely powerful tool in securing and improving Tor.

## 6. CONCLUSION

In this paper, we presented Ting, a novel and practical technique for measuring round-trip times between any arbitrary pair of Tor relays with accuracy and at scale. Ting sets itself apart from previous attempts to compute latency information in that: (1) it does not require any modification of the Tor client or protocol, (2) it does not require any additional infrastructure deployment, and (3) it calculates latencies by sending packets along the entire network path and software stack, ensuring an accurate reflection of Tor relays' view of the network. By conducting experiments on both our own controlled relays on PlanetLab and live relays on the Tor network, we validated Ting's ability to estimate true latencies.

We suggest and evaluate three applications wherein data acquired by Ting could be used to improve Tor performance. On the other hand, we provide evidence that it could also be used to improve the latency of Tor while maintaining, and even improving, the level of anonymity it provides, by greatly increasing the set of acceptable circuits for a given RTT, though we leave specific algorithms to future work. Finally, Tor's geographic diversity, both in terms of countries and number of /24s covered, makes it a viable platform for measuring a wide swath of the Internet. We believe that Ting provides unique insight into measurements within residential networks, and hope that it will complement existing techniques. Furthermore, given the current rate of growth of the Tor network, we suspect that Ting's usefulness will scale with Tor's increased adoption. We believe these applications to be representative of a much larger set of improvements to the Tor network that explicit RTT measurements could permit.

### Acknowledgments

We thank our shepherd, Phillipa Gill, and the anonymous IMC reviewers for their helpful comments. This research was supported in part by NSF grant CNS-1409191, the National Security Agency as part of a Science of Security lablet, and an Amazon Web Services in Education grant.

## 7. REFERENCES

[1] S. Agarwal and J. R. Lorch. Matchmaking for online games and other latency-sensitive P2P systems. In *ACM SIGCOMM*, 2009.

[2] M. Akhoondi, C. Yu, and H. V. Madhyastha. LASTor: A low-latency AS-aware Tor client. In *IEEE Symposium on Security and Privacy*, 2013.

[3] E. Androulaki, M. Raykova, S. Srivatsan, A. Stavrou, and S. M. Bellovin. PAR: Payment for anonymous routing. In *Symposium on Privacy Enhancing Technologies (PETS)*, 2008.

[4] D. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 24(2):84–88, Nov. 1981.

[5] Y. Chen, R. Sion, and B. Carbunar. XPay: Practical anonymous payments for Tor routing and other networked services. In *Workshop on Privacy in the Electronic Society (WPES)*, 2009.

[6] R. Cox, F. Dabek, F. Kaashoek, J. Li, and R. Morris. Practical, distributed network coordinates. In *Workshop on Hot Topics in Networks (HotNets)*, 2003.

[7] R. Dingledine, N. Mathewson, and P. Syverson. Tor: The second-generation onion router. In *USENIX Security Symposium*, 2004.

[8] R. Dingledine and S. J. Murdoch. Performance improvements on Tor or, why Tor is slow and what we're going to do about it. Online: `https://www.torproject.org/press/presskit/2009-03-11-performance.pdf`, 2009.

[9] N. S. Evans, R. Dingledine, and C. Grothoff. A practical congestion attack on Tor using long paths. In *USENIX Security Symposium*, 2009.

[10] Y. Gilad and A. Herzberg. Spying in the dark: TCP and Tor traffic analysis. In *Privacy Enhancing Technologies*, pages 100–119. Springer, 2012.

[11] K. P. Gummadi, S. Saroiu, and S. D. Gribble. King: Estimating latency between arbitrary Internet end hosts. In *ACM Internet Measurement Workshop (IMW)*, 2002.

[12] N. Hopper, E. Y. Vasserman, and E. Chan-Tin. How much anonymity does network latency leak? *ACM Transactions on Information and System Security (TISSEC)*, 13(2):13, 2010.

[13] R. Jansen, J. Geddes, C. Wacek, M. Sherr, and P. Syverson. Never been KIST: Tor's congestion management blossoms with kernel-informed socket transport. In *USENIX Security Symposium*, 2014.

[14] A. Johnson, C. Wacek, R. Jansen, M. Sherr, and P. Syverson. Users get routed: Traffic correlation on Tor by realistic adversaries. 2013.

[15] C. Lumezanu, R. Baden, D. Levin, N. Spring, and B. Bhattacharjee. Symbiotic relationships in Internet routing overlays. In *Symposium on Networked Systems Design and Implementation (NSDI)*, 2009.

[16] S. J. Murdoch and G. Danezis. Low-cost traffic analysis of Tor. In *USENIX Security Symposium*, 2005.

[17] Neustar IP Geolocation. `https://www.neustar.biz/services/ip-intelligence`.

[18] T. E. Ng and H. Zhang. Towards global network positioning. In *ACM Internet Measurement Workshop (IMW)*, 2001.

[19] T.-W. Ngan, R. Dingledine, and D. S. Wallach. Building incentives into Tor. In *Financial Cryptography (FC)*, 2010.

[20] A. Panchenko and J. Renner. Path selection metrics for performance-improved onion routing. In *Symposium on Applications and the Internet (SAINT)*, 2009.

[21] L. Peterson, T. Anderson, D. Culler, and T. Roscoe. A blueprint for introducing disruptive technology into the Internet. In *Workshop on Hot Topics in Networks (HotNets)*, 2002.

[22] M. K. Reiter and A. D. Rubin. Crowds: Anonymity for web transactions. *ACM TISSEC*, 1(1):66–92, Nov. 1998.

[23] Reporters Without Borders. Enemies of the Internet 2013 Report. `https://surveillance.rsf.org/en/wp-content/uploads/sites/2/2013/03/enemies-of-the-internet_2013.pdf`, Mar. 2013.

[24] RIPE NCC. RIPE Atlas. `https://atlas.ripe.net`.

[25] SamKnows. `https://www.samknows.com`.

[26] S. Savage, A. Collins, E. Hoffman, J. Snell, and T. Anderson. The end-to-end effects of Internet path selection. In *ACM SIGCOMM*, 1999.

[27] A. Schulman and N. Spring. Pingin' in the rain. In *ACM Internet Measurement Conference (IMC)*, 2011.

[28] M. Sherr, M. Blaze, and B. T. Loo. Scalable link-based relay selection for anonymous routing. In *Privacy Enhancing Technologies Symposium (PETS)*, 2009.

[29] R. Sherwood, B. Bhattacharjee, and A. Srinivasan. $\mathcal{P}5$: A protocol for scalable anonymous communication. *Journal of Computer Security*, 13(6):839–876, 2005.

[30] Stem Controller Library. `https://stem.torproject.org`.

[31] S. Sundaresan, S. Burnett, N. Feamster, and W. De Donato. BISmark: A testbed for deploying measurements and applications in broadband access networks. In *USENIX Annual Technical Conference*, 2014.

[32] Tor Metrics. `https://metrics.torproject.org`.

[33] B. Wong, I. Stoyanov, and E. G. Sirer. Octant: A comprehensive framework for the geolocalization of Internet hosts. In *Symposium on Networked Systems Design and Implementation (NSDI)*, 2007.