Traffic Analysis Against Low-Latency Anonymity Networks Using Available Bandwidth Estimation*

Sambuddho Chakravarty[1], Angelos Stavrou[2], and Angelos D. Keromytis[1]

[1] Columbia University, NY, USA
{sc2516,angelos}@cs.columbia.edu
[2] George Mason University, VA, USA
astavrou@gmu.edu

Abstract. We introduce a novel remotely-mounted attack that can expose the network identity of an anonymous client, hidden service, and anonymizing proxies. To achieve this, we employ *single-end controlled* available bandwidth estimation tools and a colluding network entity that can modulate the traffic destined for the victim. To expose the circuit including the source, we inject a number of short or one large burst of traffic. Although timing attacks have been successful against anonymity networks, they require either a *Global Adversary* or the compromise of substantial number of anonymity nodes. Our technique does not require compromise of, or collaboration with, **any** such entity.
To validate our attack, we performed a series of experiments using different network conditions and locations for the adversaries on both controlled and real-world *Tor* circuits. Our results demonstrate that our attack is successful in controlled environments. In real-world scenarios, even an under-provisioned adversary with only a few network vantage points can, under certain conditions, successfully identify the IP address of both Tor users and *Hidden Servers*. However, Tor's inherent circuit scheduling results in limited quality of service for its users. This at times leads to increased false negatives and it can degrade the performance of our circuit detection. We believe that as high speed anonymity networks become readily available, a well-provisioned adversary, with a partial or inferred network "map", will be able to partially or fully expose anonymous users.

1 Introduction

Low-latency anonymity systems strive to protect the network identity of users that are in need of services and applications that are latency sensitive or interactive in nature. Tor [27], JAP [22] and I2P [4] are examples of popular low-latency

* This work was partly supported by the Office for Naval Research through Grant N0014-09-1-0757 and the National Science Foundation through Grant CNS-07-14277. Opinions, findings, conclusions and recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the ONR or the NSF.

anonymity systems. The Achilles' heel of these systems lies in the fact that they do not modify the inter-packet delay to achieve low end-to-end latency. However, this makes them vulnerable to traffic pattern observation attacks [26]. Indeed, the adversary can correlate flow patterns in traffic flowing through one section of the network to that in another[1]. Timing attacks are feasible only by an adversary who can observe traffic in **all** the network links called a Global Adversary (GA). A key design assumption behind low-latency anonymizing systems it requires a tremendous amount of resources and effort to become a GA. Therefore, the goal is to provide adequate protection against all but a determined, and possibly targeted, attack by a GA.

Recently, however, a number of practical attacks against such anonymity systems have been proposed. These attacks typically leverage a small number of compromised network entities to partially or fully expose information about a user of these systems [7, 10, 20, 25, 32, 16]. Nonetheless, it is generally assumed that consistent tracking of the users of such anonymity systems is impractical for the large majority of users and organizations, because of the lack of many omnipresent adversaries. *We evaluate this assumption for present and future high-speed anonymity networks.*

Furthermore, popular low-latency anonymizing systems including Tor cannot guarantee consistent quality of service. This is due to a combination of the cryptographic computation, queuing delays, and traffic scheduling and conditioning [28]. Interestingly, this also degrades the adversaries ability to successfully track the anonymous users. Having low volume of traffic causes traffic analysis techniques to be less accurate [16]. *In this paper, we study the validity of this belief in a next-generation network and host infrastructure where the speed or throughput of the anonymizing network is not limited by the computational or communication capacity of the participating entities.*

To that end, we present a novel and effective method for performing source "trace-back" . Our approach can be employed to perform targeted traffic analysis against most low-latency anonymizing systems[2]. Contrary to the work by Murdoch *et al.* on traffic analysis for Tor relays [25], our technique can track not only the anonymizing relays, but also the victim's network location without the need of malicious Tor clients. We assume that the adversary controls, or can create a traffic fluctuation, in one end of the communication channel. The adversary modulates the traffic destined for the victim using a colluding end-point. Thereby, he or she attempts to detect the artificially induced traffic variations as they propagate over the candidate anonymizing proxies and routers, eventually leading to the anonymous client.

To observe the induced traffic fluctuation, the attacker requires novel tools that can quickly and accurately characterize the available bandwidth of individual links and entire network paths. We previously introduced LinkWidth [14],

[1] High latency anonymity systems such as *Mixminion* [17] modify the inter-packet latencies to curtail traffic analysis.

[2] In our experiments, we used Tor but our approach is applicable to all proxy-based anonymity systems

a *single-end controlled* available-bandwidth estimation tool that is based on the algorithm behind TCP Westwood [18]. LinkWidth does not require a TCP listening server. Furthermore, it offers bandwidth estimation of relays and routers connecting the victim clients (and Hidden Servers) to the anonymizing network. We focus on using such available bandwidth estimation for sensing deliberately induced TCP traffic fluctuations and confirming the probable identity of anonymous end points.

To verify the validity of our approach, we performed a series of experiments. We evaluated the predictive accuracy of our technique on controlled environments using different network topologies. We also tested our technique on Tor circuits created using Tor nodes that are part of the public Tor network. In our experiments, we achieved varying success in exposing the identity of the victims. On an average, we detected 49.3% of the Tor relays participating in our circuits. In addition, we were also successful several times in identifying the intermediate network routers connecting Tor clients and Hidden Services to their Entry Nodes.

We posit that a real adversary equipped with many appropriately located high-bandwidth hosts and a partial map of the network, could effectively attack timing-preserving anonymizing systems. Some prior efforts in generating such maps are the *Internet Mapping Project* [6], *AS Peering Analysis* [1], *iplane* [23], and similar efforts by CAIDA [2]. Searching a map depicting the ingress and egress routers of an Autonomous System (AS) involves little complexity. An adversary equipped with such information would probe for the induced available bandwidth variation, **only on** the inter-domain routers in search for the AS that hosts the victim anonymous client[3].

The objective of this paper is not to demonstrate this search process but to answer the following question: *Can a well-provisioned (Semi-)Global Adversary equipped with probing nodes scattered close to most inter-AS routers, and "sender-only" controlled probing tools like LinkWidth, measure bandwidth fluctuation on network routers and anonymizing relays associated to a communication session?*

Having a large number of distributed measurement nodes ("vantage points") allows for better coverage of the network links. However, there is no requirement for network affinity of the vantage points to the victim or the Tor relays used in the circuit. Moreover, we do not assume that the attacker has access to routers, network infrastructure nodes (*e.g.,* DNS or DHCP servers), or anonymizing relays; nor do we exploit software vulnerabilities that inadvertently expose the true network identity of the user. *The novelty of our technique lies in aiding an adversary with bandwidth resources to perform traffic analysis for determining anonymizing network's entry-points, anonymous clients and location hidden services in a communication session.*

[3] Further, resolving down to the end hosts might require ISP router maps through services such as as *Rocketfuel* [5]

2 Related Work

The majority of the attacks against low-latency anonymization systems, employ some form of traffic analysis. Many of the successful methods apply *active* techniques, employing practical *predecessor attack* [34] derivatives like [30, 9, 35, 16], which involve inside network elements. Other suggest *passive* attacks that simply count the number of packets at ingress and egress links of the anonymizing network [8].

There are yet some other attacks which try to track changes in non-anonymous system parameters (*e.g.* CPU and memory usage) in network nodes due to change in anonymous traffic flowing through them. Such attacks are called *side channel* attacks. Some examples of *side channel* attacks are presented in references [36, 24]. Here the adversary observes changes in CPU usage by observing skews in TCP timestamps, that are brought about by changes in CPU temperature. Other such attacks use network latency as a side-channel to reveal identity of anonymous network system parameters [20, 25]. The authors of those papers demonstrate the use of inter-packet latency and round-trip times (RTT) to reveal identity of anonymous peers and anonymizing network elements.

Although significant, this class of attacks can be ineffective when the bottlenecks of the network paths connecting the anonymously communicating peers and the adversary to the candidate networking elements, do not coincide. In such situations, the adversary's measurement capabilities, constrained by the bottleneck link speed, might not accurately detect network link contentions due to the anonymous traffic. Such contentions are essential to perceive changes in the measured RTT. The poor end-to-end QoS of popular systems such as Tor, further degrades the attacker's ability to launch such attacks. Low volume traffic leads to low network congestion; thereby causing little variations in measured RTT between the adversary and the victim. Results by Evans *et al.* indicate that this intuition is likely correct [16].

Our approach stems out of RTT based traffic analysis [25, 20], and partially from traffic pattern injection techniques [31, 21]. Unlike predecessor attack derivatives, our attack assumes nominal control of networking elements. The adversary induces end-to-end network traffic fluctuation by colluding with a malicious server with which anonymous clients communicate. Using Linkwidth, the adversary tries to observe the induced bandwidth fluctuations on candidate anonymizing network elements and routers leading to the anonymous client.

Our approach seems similar to targeted denial of service based technique, presented by Burch and Cheswick to "trace-back" to the source of a DoS source that used IP spoofing [11]. Their aim was to cause interference in the remote routers and notice fluctuations in the attack-DoS traffic. Using a network map they would iteratively trace-back to eventually identify the source of a DoS, or at least its approximate location (*e.g.,* hosting ISP). Our technique is similar to theirs. We, however, aim to "trace" available bandwidth fluctuations on network routers and anonymizing proxies that carry traffic between the anonymous parties.

3 Attack Methodology

Before delving into the attack details, we discuss the threat model for which we claim that our attacks are effective.

Threat Model

Our primary focus is an adversary who can induce "traffic fluctuations" into a targeted anonymity preserving channel and observe it "trickle" towards the victim. The adversary may have hosts under his control on many ASes or could be at a network vantage point with respect to routers in various subnets. Each of these hosts may be running a copy of bandwidth estimation tools such as LinkWidth. They would also be at a network vantage point with respect to all candidate victim relays and network routers. This is feasible in toady's' Internet: the inter-router media connecting the routers of major tier-3 ISPs can support over 10 Gbps (let alone the tier-1 and tier-2 ISPs). Most have 30–40% under-utilized spare capacity. Therefore, while we are not in a position of having a large set of vantage points, this does not prevent others from being able to do so. Often, adversaries like us, might be sensing "under-utilized high-bandwidth" links. Popular anonymity preserving systems, like Tor relays, often dedicate a considerable fraction of the traffic to forwarding client traffic. However, such small fluctuations in high capacity links, are less than what most state-of-the-art available link bandwidth estimation techniques can detect accurately. Nevertheless, we demonstrate that for some common network topology configurations and parameters, an adversary can harness bandwidth estimation to trace the "unknown" path a client uses to connect to a server.

Traffic Analysis Methodology

We used available bandwidth estimation tools to detect induced traffic fluctuations on candidate anonymizing relays. We identify probable candidate network routers that could reach this relay through a single network hop. Since most anonymity preserving relays are at the network edges, the network routers that could directly reach the candidate relays would be their default gateways. This intuition was re-applied on default gateways to determine which network interface, and hence the network routers within one network hop, exhibited similar fluctuations. We repeated the tracking process recursively until the fluctuations were tracked down to the source network (and possibly the source itself).

To quantify our detection capabilities, we performed extensive experiments initially in a controlled emulation environments, viz. DETER [3]. Some of the experiments were further validated in controlled lab-environment. We also uncovered real-world Tor clients, and hidden servers which communicated using public Tor relays, with some success. Our approach requires a "map" presenting with information of inter-domain routers for the Tor Entry Node and the victim. In our experiments, we did not use elaborate maps. We only considered result obtained from running *traceroute* between the targeted victim and its Tor Entry

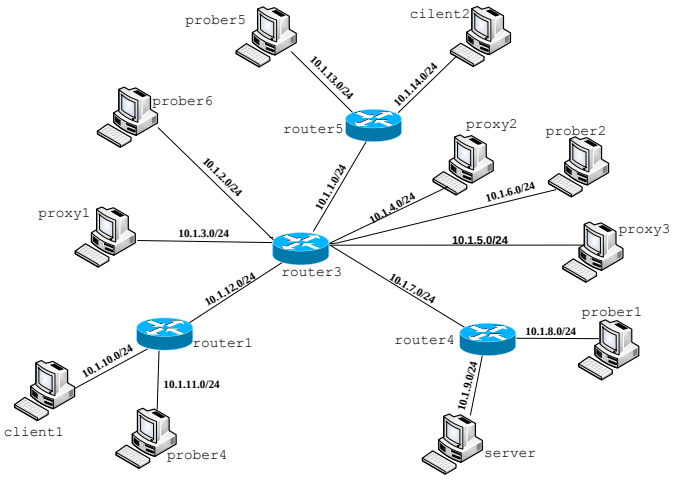


Fig. 1. DETER testbed network topology.

Node. Moderate success rate is primarily due to a combination of our inadequate network vantage points and low end-to-end throughput offered by the Tor relays as compared to the available link capacities of routers and relays. Lastly, our accuracy was also affected by the presence of background cross traffic on regular network routers, resulting in higher false negatives when compared to the in-lab or DETERLAB experiments' results.

4 Experimental Evaluation

Our attack technique can be applied to low-latency anonymity systems that are based on *Onion Routing* [19]. Tor is a good example and chief among onion routing anonymizing based systems. In such systems, there seems to be a trade-off between anonymity guarantees and performance [28]. We show that in both controlled and real-world experiments, a well-provisioned adversary can expose the anonymity of Tor relays and end-points. This is performed by determining the available bandwidth fluctuations on a large number of relays and network routers.

4.1 Experiments Using the DETER Testbed

To determine the effectiveness of our attack, we used DETER [3] to emulate various network topologies. DETER is a network emulation system that offers commodity machines and switches connected with user specified topologies. Users can specify the operating systems, network connections, link speeds, routing protocols and other parameters.

Figure 1 depicts one of the topologies we used to validate our approach. In these experiments, we used Iperf [29] and LinkWidth to detect the fluctuation of the available link bandwidth on network routers along the path connecting the server to the actual client[5].

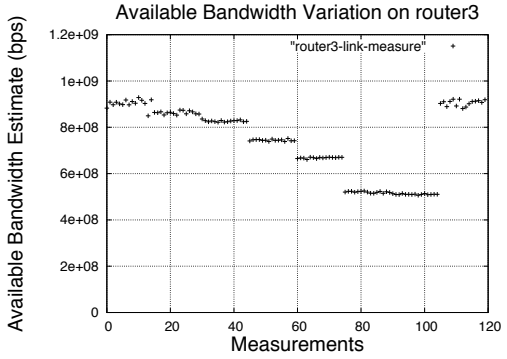


Fig. 2. The detected available bandwidth on the router connected to the victim `router3` drops uniformly as client traffic increases.

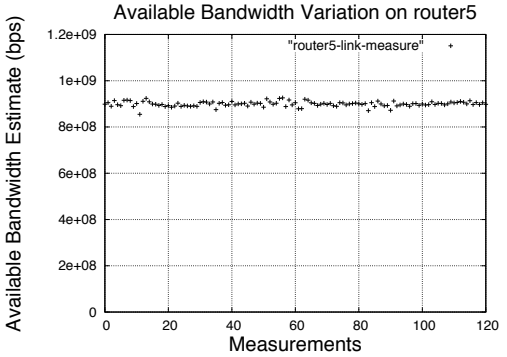


Fig. 3. We correctly measured the absence of consistent available bandwidth fluctuation on `router5` (not in the victim's path).

The host, marked as `server` in the topology figure (Figure 1), was the colluding web server. In addition, there were two client hosts, `client1` and `client2`. `client1` was the victim downloaded a large file from `server`, using HTTP; `client2` was idle.

With two clients on two separate branches, the available traffic fluctuation induced by `server` was observable only by one of the branches (the one leading to `client1`). The probing hosts on each of the subnets probe the intermediate links. To forward packets through `proxy1`, we used *squid* [33]. `client1` connected to `server` via `proxy1`. The `server` modulated the transmission rate to induce the necessary bandwidth fluctuation. The adversary probed for the bandwidth variation on the network elements in both branches using its probing hosts.

The reason we avoided installing an anonymizing system in this particular experiment is due to the poor QoS resulting from computational and network transport costs in systems like Tor. The motivation behind choosing *squid* was to demonstrate the effectiveness of our hypothesis; we assume that the relaying architecture of Onion Routing based systems would soon provide higher throughput rate like unencrypted *squid* proxy.

Therefore, the goal of the experiment was to demonstrate that an adversary can observe these induced traffic fluctuations, provided the anonymizing service does not degrade client–server traffic throughput.

[5] Due to the way DETER emulates the requested link capacities and the use of traffic shaping, Iperf was more accurate than LinkWidth in estimating the bandwidth variations

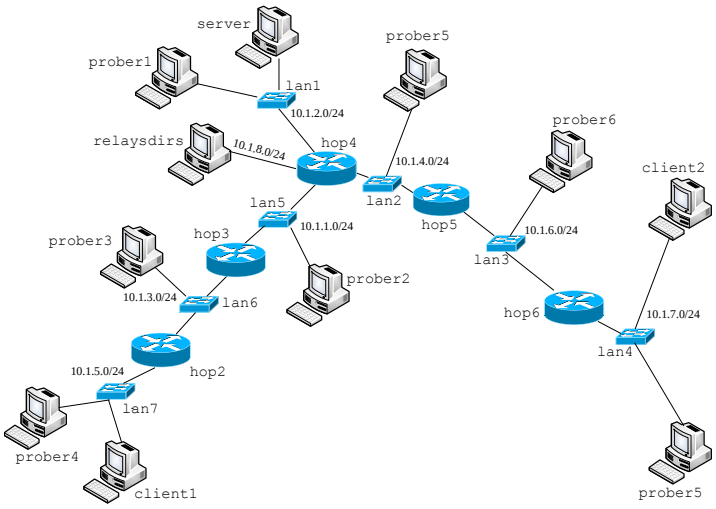


Fig. 4. In-Lab testbed network topology.

While the download was in progress, the server increased the transmission speed gradually by 50 Mbps, 100 Mbps, 200 Mbps, 300 Mbps and 500 Mbps, for every iteration of the experiment. The probing nodes (`probers`) measured the available bandwidth variation on both branches. The available link bandwidth fell steadily on all routers along the path carrying `client1`'s traffic. The probing of `router5` and `client2`, along the idle network branch, resulted in very high true negatives. For brevity, we only present the results obtained by probing `router3` and `router5`. The graphs representing these are presented in Figures 2 and 3. The graphs that show the fluctuations in available bandwidth for the rest of the hosts are presented in Appendix A.

The available bandwidth drops as the server increases its rate to the victim, and thus occupies greater share of the available bandwidth along the path via `router3`. Probing `router5` along the path connecting `client2` to `server` shows no significant fluctuation in available bandwidth. Overall, our experiments indicate that we can indeed detect small fluctuations by utilizing about 5–10% of the link bandwidth. Although this is a large value (approximately 50–100 Mbps), it is only a small fraction of the link bandwidth. This encouraged us to believe, that our technique will work well even in situations where only a small portion of the network link is being utilized by anonymous traffic.

We used a large file for our experiments. But we could have achieved the same fluctuation through multiple downloads of many small files. Through coordinated probing of the candidate links, momentary burst (due to small files being downloaded) can be easily detected. This is clearly evident from our results presented in Appendix A. The sudden fall in available link bandwidth from approximately 100% (900 Mbps) to 90% (800 Mbps) within a short interval (few seconds) and and sudden rise later to 100%, in tandem to the induced traffic fluctuations, proves this. Further evidence of LinkWidth's effectiveness to detect small bandwidth fluctuations is presented in in our technical report [14].

4.2 In-Lab Experiments

To further support the DETER results, we performed the same experiments in an in-lab environment using commodity machines connected via a Gigabit switched network. Figure 4 depicts the in-lab network testbed topology used to demonstrate our technique. Again, the client `client1` is the victim and is connected to the `server` via a *squid* proxy installed on the host `relaysdirs`.

As before, the client downloaded a large file and the server varied the TCP throughput. The probing hosts measured the available bandwidth on the routers along both branches of the network - one leading to `client1`, which downloads the file, and the other leading to `client2`, which is not involved in the transfer.

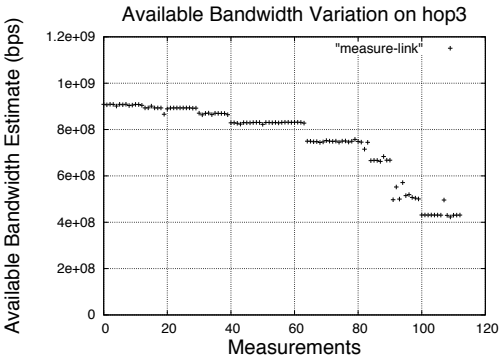


Fig. 5. Available bandwidth on `hop3` drops uniformly when we increase the traffic towards the victim.

Available bandwidth on all the routers along the path connecting `client1` to `server` fluctuated, as `server` varied the available TCP traffic to port 80, that it saw originating from the host `relaysdirs`. For brevity, we present graphically the results from probing `hop3` (Figure 5), along the path leading to the victim and `hop5` (Figure 6), leading to the idle node.

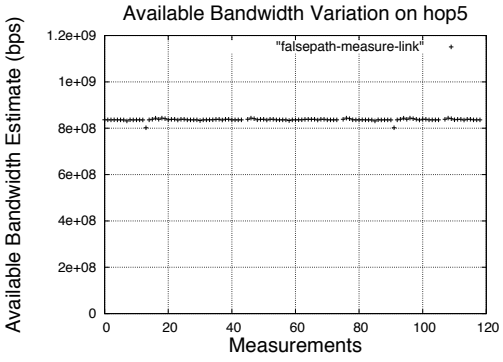


Fig. 6. There is no persistent available bandwidth fluctuation on `hop5` (unlike `hop3`, that is along path of the download traffic).

We observed reduction in available bandwidth as client-server traffic eventually occupies more bandwidth along the path via `hop3`. The available link bandwidth of `hop5` does not show any drastic change, as it was not along the actual download path. Probing router `hop3` and `client1` also showed similar bandwidth fluctuations while `hop6` and `client2` showed no definite fall in the available the link bandwidth; thus concurring with our idea of tracking link bandwidth fluctuation along the path leading up to the actual source of traffic. These results are presented in Appendix B.

4.3 Probing Tor Relays, Clients and Hidden Services

The validation of our technique in previous subsections, albeit on a controlled environment, encouraged us to believe that our technique might potentially be used to track induced available bandwidth on network routers connecting a Tor client to a Tor Entry Node. Therefore, we attempted to identify the Tor Onion Routers (ORs) participating in a real-world circuit. The server colluded with the adversary to induce fluctuations in the circuit throughput. This resulted in available bandwidth fluctuation on ORs and network routers connecting these ORs to Onion Proxies (OPs)[6]. *This experiment is elaborated in our previous paper* [13]. We concisely describe the experiment and results here. Figure 7 illustrates how the adversary probed the Tor relays involved in a circuit.

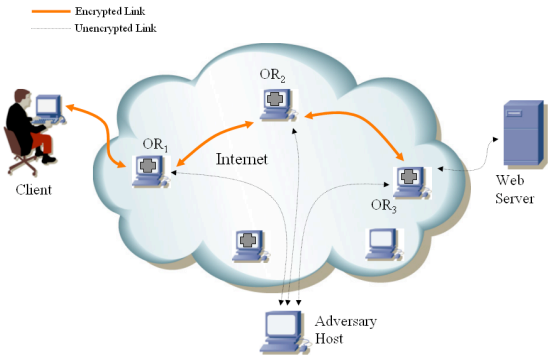


Fig. 7. Adversary probing the fluctuations in available bandwidth of ORs participating in a Tor circuit.

The colluding web server transmitted a file which the client downloaded the file through a Tor circuit. During the download, the adversary used our traffic analysis technique to identify the victim relays participating in the circuit. While the server shaped the circuit's throughput to various values, the adversary measured the available bandwidth using LinkWidth. This process was repeated several times. In every iteration, the adversary changed the client's bandwidth share, increasing it each time by approximately 100 Kbps. The adversary detected decrease in measured available bandwidth that was reflected through increase in congestion and packet losses.

In our experiments, we successfully identified **all** the relay nodes in 11 out of the 50 circuits that we probed. For 14 circuits, we were able to identify only two of the three participating Tor relays. There were 12 circuits in which only one of the relays was detected. There were also 13 circuits that we are unable to detect any of the participating ORs. Finally, among all the 150 ORs probed there were 22 which filtered all probe traffic. A similar experiment was performed for obtaining

[6] Onion Proxy is the term used for Tor clients [15]

an estimate of the false positives. Initial observations yielded approximately 10% false positives. *However, on repeated iterations of the same experiment, we detected no false positives.* These very likely result from noises and errors in our measurement techniques resulting from lack of adequate network vantage hosts, background network cross-traffic and asymmetric network links.

Probe Set-up and Technique for Identifying Tor Clients: To determine the network identities of Tor clients involved in Tor circuits, we used the same setup as in Figure 7. However, in this experiment, the adversary probed routers on the network path connecting the Tor Entry Nodes to the Tor Clients. The client fetched a relatively large file from a colluding server, which shaped the bandwidth of the connection.

Circuit Number	Hops from Client –Entry Node Node	Correctly Detected Client–Entry Node Hops	Unresponsive Routers	Routers Not Reporting Enough Fluctuations	Success Rate
1	10	6	4	0	60.00%
2	15	4	0	0	26.67%
3	18	4	7	7	22.23%
4	18	5	8	5	27.78%
5	14	6	2	6	42.86%
6	14	9	1	4	64.30%
7	15	7	2	6	46.67%
8	14	7	2	5	50.00%
9	14	4	2	8	28.57%
10	15	6	4	5	40.00%

Table 1. Available-Bandwidth Fluctuation Detection in Links Connecting a Tor Client and its Entry Node.

Lacking a "network map" that has link-level connectivity information of the Internet, we relied on `traceroute` information to discover the hops between the client and its Entry Node. However, in practice, an adversary equipped with AS-to-AS and link-level network connectivity maps, needs to probe only a relatively small set of links to determine which of them might be carrying the victim's traffic. Entry and exit, to and from an AS, is through a small number of *inter-domain* routers. The adversary can look up the AS location of the Tor relays from the inter-domain routing information. This will enable him to track the induced available bandwidth variation **only** on inter-domains routers and search for the AS that hosts the victim anonymous client. Nodes in all the probable ASes, with link speeds comparable to that of the inter-domain routers', could be used for this task. To obtain higher fine-grained network position, the attacker might have to track down to the end hosts. This step will require ISP router maps through services such as as *Rocketfuel* [5]. *Though seemingly an exponential search problem, this would be reasonably tractable provided the fluctuations could be detected with high confidence.* Moreover, optimizing the adversary's search of links to probe is a different problem and we do not consider that aspect of the attack in this paper.

The results from probing the available link bandwidth variations on network routers connecting the clients to their respective Entry Nodes, is presented in Table 1. The accurate detection of bandwidth fluctuation was performed through the detection of packet loss changes. This loss is indication of decrease in available bandwidth, whenever the routers and Tor relays were probed using LinkWidth. Our technique used an un-cooperative method of available bandwidth or throughput estimation. Sometimes the probe traffic, being aggressive, prevented a Tor client, using TCP (which is "elastic" and non-aggressive), to utilize all of the allowed bandwidth. This lead to an "on-off" pattern in the the client's download. This is particularly true when the probes and the probed traffic traverse the same victim router.

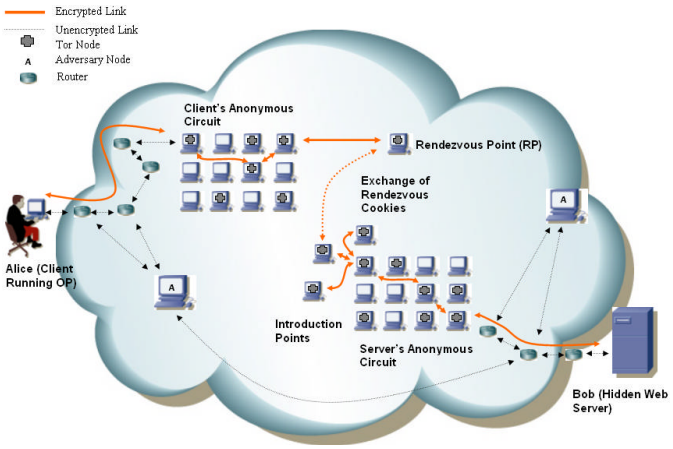


Fig. 8. The adversary measures the available bandwidth and thus detects any fluctuations in each link of the path leading to the victim.

As a caveat, modifying the available throughput of the TCP connection through a certain repeating pattern (*e.g.,* 50 Kbps, 100 Kbps, 500 Kbps, 50 Kbps, 100 Kbps, 500 Kbps), would be akin to inducing a distinct "signal" or "watermark" in the client-server traffic. If very high true positives are assured, then this "signal" could be detected always and only on relays and routers through which the victim's traffic travels[7]. This can optimize the search for links to probe while determining the source of the anonymous traffic[8].

Probe Set-up and Technique for Identifying Tor Hidden Servers: To identify a Hidden Service, we used the setup depicted in Figure 8. Here the adversary probed the routers connecting Hidden Service to its Entry Node. Con-

[7] Thereby also eliminating false positives and false negatives.

[8] Without such an optimization, the adversary might end up performing a depth-first search of large segments of the Internet rooted at the Tor entry node

trary to the previous cases, the available bandwidth fluctuation was induced by the client which is assumed to collude with the adversary. We relied solely on `traceroute` for determining which routers connect a Hidden Server to its corresponding Entry Node. Table 2 summarizes the results of this experiment:

Circuit Number	Hops from Hidden Server–Entry Node	Correctly Detected Hidden Server–Entry Node Hops	Unresponsive Routers	Routers Not Reporting Enough Fluctuations	Success Rate
1	13	4	2	7	30.70%
2	12	9	0	3	75.00%
3	11	7	1	3	63.64%
4	14	5	4	5	35.71%
5	12	9	0	3	75.00%
6	13	3	3	7	23.08%
7	16	5	4	7	31.25%
8	13	3	2	8	23.08%
9	17	4	1	12	23.53%
10	13	5	1	7	38.46%

Table 2. Available-Bandwidth Fluctuation Detection in Links Connecting a Hidden Server to Its Entry Nodes

In almost every circuit, there were some routers which filtered our probe packets. The rest of the routers were either detected correctly or not detected at all (*i.e.*, no false positives). This can be attributed to the lack of sufficient vantage points or to insufficient throughput achieved by the client in some cases (approximately 5–10 KBytes/sec). Despite of these practical problems, we were still able to trace the bandwidth fluctuations along the path (and hence the identity) of the Tor client and Hidden Server with high accuracy; over 60% and 75% in some of the circuits. The observed degradation in the client's performance whenever the adversary probed the candidate routers, are accepted as "available bandwidth fluctuations". Placing a large number of probing hosts at network vantage points would provide the adversary with better detection resolution and accuracy.

5 Issues, Discussion and Possible Counter-Measures

We initially tested our trace-back technique under various controlled environments. Our results indicate high true positives and almost zero false negatives. Small bandwidth variations, due to the introduction of a 50–100 Mbps TCP stream, were clearly discernible on a 1 Gbps link. This led us to believe that small bandwidth fluctuations on *high-capacity links* can be detected provided there is low background cross traffic that may introduce false positives or false negatives in the measurements.

LinkWidth provides very accurate available link bandwidth estimation. As presented in our technical report [14], LinkWidth can accurately detect 1 Kbps

fluctuation in available link bandwidth. Of course, this accuracy decreases when the variations decrease as a percentage of the overall link capacity. Small distortions, for instance 50 Kbps, on a 500 Kbps are easier to detect, than when they are on a 1 Gbps link.

Simple fluctuations on network links of the in-lab testbed could be detected within 15-20 seconds. The probing speeds were adjustable run-time parameters. Faster probing caused greater contention to the client-server traffic, thereby slightly decreasing the detection accuracy and granularity.

Having obtained high true positives under controlled environments, it seemed intuitive that an adversary could potentially detect available bandwidth fluctuation on an anonymizing proxy and its propagation to corresponding clients or servers via network routers. It is important that adversarial nodes are located at network vantage points where they can filter out traffic that causes unwanted distortion to the probes. It is also essential that a Tor client achieves high end-to-end throughput through Tor relays which is comparable to the installed link capacity of the network routers.

When applied to detect available link bandwidth variations on real Tor ORs, we were able to detect with some success, fluctuations on network routers connecting the client to its respective ORs. However, we restricted our selection of Tor relays within the US, to position our US-based probing host at a better network vantage point, when probing Tor relays. Probing nodes residing across trans-oceanic links seems infeasible and provided erratic results. Consequently, we were limited by the number of Exit Nodes within the US. Out of the approximately 150 exit relays at the time of our experiments less than 100 allow clients to setup their own circuits. Moreover, less than a fifth of these allow Hidden Servers to communicate with anonymous clients. This is mostly due to intermittent quality of service, node availability, and exit policies that restricted connectivity to a small set of permitted users. Probing Tor relays and network routers required considerably more measurements than the in-lab measurements (approximately 2-5 minutes per relay or router). High Internet cross-traffic and low Tor traffic necessitates longer probing and more measurements to avoid false positives and false negatives as much as possible.

In real world scenarios, there maybe various ways to entice a Tor client to connect to such malicious servers. Tempting commercials on a website, luring a Tor client to click on, could be one such tactic. This could download applications, like multiple Adobe Flash animations, on the client's host, resulting in a sudden change in his/her available network link bandwidth. An adversary running multiple co-ordinated probing hosts, probing suspected links, could detect such a sudden sharp change in available link bandwidths on all links connecting the anonymous party to its anonymizing service; thereby revealing its identity. The adversary would require to own only a frame of a popular website, say a blog or an online forum, visited frequently by users who wish to stay anonymous.

Apart from the lack of accuracy in detecting small variations of available link bandwidth, Reardon and Goldberg have described why current Tor circuits offer low end-to-end throughput [28]. This is primarily because of multiplexing many

TCP streams through a single Tor circuit connecting a Tor client to a relay or between the relays themselves, if such circuits already exist. Therefore, TCP congestion control and flow control mechanics affect the performance of all Tor circuits that are multiplexed over a single TCP connection. Packet losses and reordering affecting the *cells* of one Tor circuit reduced the overall performance of the TCP connection. Such losses cause the *cells* from unrelated Tor circuits to be delayed as well.

These inherent measurement limitations can be potentially leveraged to create countermeasures or even narrow the applicability of our attack. For instance, an anonymous client can utilize parallel connections in a round-robin fashion to access the same server. This would diffuse the ability of the server to generate detectable traffic variations: traffic spikes would be distributed across all the parallel connections. Likewise, traffic smoothing by anonymizing proxies is another potential countermeasure. Tor allows relay operators to use such techniques. Another option is to use shorter circuit lifetime. This would impose some time limitations on the duration of the communication path, making it harder for an adversary to completely trace the target through the anonymizing network. Anonymous connections using longer paths by employing more relays do not appear to make the attack appreciably more difficult. However, as discussed in [12], it can significantly affect the client's perception of the connection throughput.

6 Conclusions

We proposed a new traffic analysis technique that can expose the network identity of end-points (users) of low-latency network anonymity systems. Our technique involves an adversary who can probe links from many network vantage points using single-end controlled bandwidth estimation tools. In addition, the adversary employs a colluding server or is able to otherwise induce fluctuations in the victim's TCP connection. It is not essential to own such colluding servers: using carefully crafted online ads and pop-ups can be used judiciously to trick users to click on specific links and result in traffic spikes. Using the vantage points, the adversary measures the effects of this fluctuation as it "trickles" through the anonymizing relays and intermediate routers to the victim.

Our approach works well when the end-to-end throughput of the anonymizing network allows for bandwidth variations that can be detected by the vantage points. This motivated us to test our attack technique in real-world Tor circuits. Our experiments show that we were able to expose real-world Tor relays with a true positive rate of 49.3%. Furthermore, we can successfully traceback to the Tor clients and Hidden Servers by probing the network routers connecting them to their Entry Nodes. On average, we could correlate 49.5% of the network routers to the victim Tor traffic that they carried. Further correlations were not always feasible due to bandwidth limitations imposed by relay operators and Tor's poor performance owing to circuit scheduling and management [28]. We believe that our work exposes a real weakness in proxy-based anonymity schemes. *This threat*

will become increasingly more apparent and accurate as future networks and hosts participate in higher end-to-end throughput circuits.

References

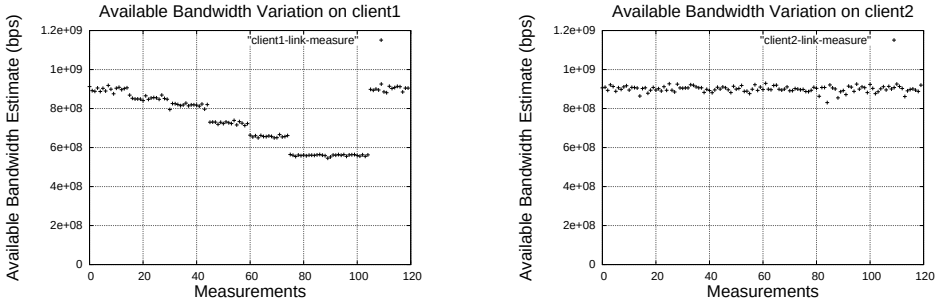
1. AS Peering Analysis. `http://www.netconfigs.com/general/anoverview.htm`.
2. CAIDA Router Measurements. `http://www.caida.org/tools/taxonomy/routing.xml`.
3. DETER Network Security Testbed. `https://www.isi.deterlab.net`.
4. I2P Anonymous Network. http://www.i2p2.de/.
5. Rocketfuel: An ISP Topology Mapping Engine. `http://www.cs.washington.edu/research/networking/rocketfuel/`.
6. The Internet Mapping Project. `http://www.cheswick.com/ches/map/`.
7. Agrawal, D., and Kesdogan, D. Measuring Anonymity: The Disclosure Attack. *IEEE Security & Privacy 1*, 6 (November/December 2003), 27–34.
8. Back, A., Möller, U., and Stiglic, A. Traffic analysis attacks and trade-offs in anonymity providing systems. In *Proceedings of the 4th International Workshop on Information Hiding(IHW)* (London, UK, 2001), Springer-Verlag, pp. 245–257.
9. Bauer, K., McCoy, D., Grunwald, D., Kohno, T., and Sicker, D. Low-resource routing attacks against tor. In *Proceedings of the 2007 ACM Workshop on Privacy in Electronic Society (WPES)* (2007), pp. 11–20.
10. Borders, K., and Prakash, A. Web Tap: Detecting Covert Web Traffic. In *Proceedings of the* 11^{th} *ACM Conference on Computer and Communications Security (CCS)* (October 2004), pp. 110–120.
11. Burch, H., and Cheswick, B. Tracing Anonymous Packets to Their Approximate Source. In *Proceedings of the* 14^{th} *USENIX Conference on System Administration (LISA)* (December 2000), pp. 319–328.
12. Chakravarty, S., Stavrou, A., and Keromytis, A. D. Approximating a Global Passive Adversary Against Tor. Computer Science Department Technical Report (CUCS Tech Report) CUCS-038-08, Columbia University, August 2008.
13. Chakravarty, S., Stavrou, A., and Keromytis, A. D. Identifying Proxy Nodes in a Tor Anonymization Circuit. In *Proceedings of the* 2^{nd} *Workshop on Security and Privacy in Telecommunications and Information Systems (SePTIS)* (December 2008), pp. 633–639.
14. Chakravarty, S., Stavrou, A., and Keromytis, A. D. LinkWidth: A Method to Measure Link Capacity and Available Bandwidth using Single-End Probes. Computer Science Department Technical Report (CUCS Tech Report) CUCS-002-08, Columbia University, January 2008.
15. Dingledine, R., Mathewson, N., and Syverson, P. Tor: The Second-Generation Onion Router. In *Proceedings of the* 13^{th} *USENIX Security Symposium (USENIX Security)* (August 2004), pp. 303–319.
16. Evans, N., Dingledine, R., and Grothoff, C. A practical congestion attack on tor using long paths. In *Proceedings of the* 18^{th} *USENIX Security Symposium (USENIX Security)* (August 2009), pp. 33–50.
17. G.Danezis, R.Dingledine, and N.Mathewson. Mixminion: A type iii anonymous remailer. `http://mixminion.net/`.
18. Gerla, M., Sanadidi, M. Y., Wang, R., and Zanella, A. TCP Westwood: Congestion Window Control Using Bandwidth Estimation. In *Proceedings of IEEE Global Communications Conference (Globecomm), Volume 3* (November 2001), pp. 1698–1702.

19. Goldschlag, D. M., Reed, M. G., and Syverson, P. F. Hiding Routing Information. In *Proceedings of* 1^{st} *International Information Hiding Workshop (IHW)* (May 1996), R. Anderson, Ed., Springer-Verlag, LNCS 1174, pp. 137–150.
20. Hopper, N., Vasserman, E. Y., and Chan-Tin, E. How Much Anonymity does Network Latency Leak? In *Proceedings of ACM Conference on Computer and Communications Security (CCS)* (October 2007), pp. 82–91.
21. Huang, D., and Agarwal, U. Countering Repacketization Watermarking Attacks on Tor Network. In *Proceedings of the* 8^{th} *International Conference on Applied Cryptography and Network Security (ACNS '10)* (Beijing, China, June 2010).
22. JAP. `http://anon.inf.tu-dresden.de/`.
23. Madhyastha, H. V., Isdal, T., Piatek, M., Dixon, C., Anderson, T. E., Krishnamurthy, A., and Venkataramani, A. iplane: An information plane for distributed services. In *Proceedings of* 7^{th} *USENIX Symposium on Operating Systems Design and Implementation (OSDI)* (November 2006), pp. 367–380.
24. Murdoch, S. J. Hot or not: Revealing hidden services by their clock skew. In *Proceedings of ACM Conference on Computer and Communications Security (CCS)* (October 2006), pp. 27–36.
25. Murdoch, S. J., and Danezis, G. Low-Cost Traffic Analysis of Tor. In *Proceedings of IEEE Symposium on Security and Privacy (IEEE S and P)* (May 2005), pp. 183–195.
26. Raymond, J.-F. Traffic Analysis: Protocols, Attacks, Design Issues and Open Problems. In *Proceedings of the International Workshop on Design Issues in Anonymity and Unobservability* (2001), pp. 10–29.
27. R.Dingeldine, M.Edman, A.Lewman, N.Mathewson, S.Murdoch, P.Palfrader, M.Perry, and P.Syverson. "tor: anonymity online". `https://www.torproject.org/`.
28. Reardon, J., and Goldberg, I. Improving tor using a tcp-over-dtls tunnel. In *Proceedings of* 18^{th} *USENIX Security Symposium 2009 (USENIX Security)* (August 2009).
29. Tirumala, A., Qin, F., Dugan, J., Feguson, J., and Gibbs, K. IPERF. `http://dast.nlanr.net/projects/Iperf/`, 1997.
30. V.Pappas, E.Athanasopoulos, S.Ioannidis, and E.P.Markatos. Compromising Anonymity Using Packet Spinning. In *Proceedings of the* 11^{th} *Information Security Conference (ISC)* (September 2008), pp. 161–174.
31. Wang, X., Chen, S., and Jajodia, S. Network flow watermarking attack on low-latency anonymous communication systems. In *Proceedings of IEEE Symposium on Security and Privacy (IEEE S and P)* (2007), pp. 116–130.
32. Wang, X., and Reeves, D. S. Robust Correlation of Encrypted Attack Traffic Through Stepping Stones by Manipulation of Interpacket Delays. In *Proceedings of the* 10^{th} *ACM Conference on Computer and Communications Security (CCS)* (October 2003), pp. 20–29.
33. Wessels, D., A.Rousskov, H.Nordstrom, Chadd, A., and A.Jeffries. Squid. `http://www.squid-cache.org/`.
34. Wright, M. K., Adler, M., Levine, B. N., and Shields, C. An analysis of the degradation of anonymous protocols. In *Proceedings of the Network and Distributed Security Symposium (NDSS)* (2002).
35. X.Fu, and Z.Ling. One cell is enough to break tor's anonymity. In *Proceedings of Black Hat Technical Security Conference* (February 2009), pp. 578–589.
36. Zander, S., and Murdoch, S. An Improved Clock-skew Measurement Technique for Revealing Hidden Services. In *Proceedings of* 17^{th} *USENIX Security Symposium (USENIX Security)* (San Jose, USA, July 2008), pp. 211–225.

APPENDIX

A Results from Probing Host/Routers on DETER Testbed

This section presents the results omitted in subsection 4.1. The graph in Figure 9(a) exhibits the results from probing the network interface of `client1`. As expected, we observe a very similar visible available bandwidth fluctuation pattern as we saw in Figure 5 for `router3`. Also evident from results in Figure 9(b), we don't observe any obvious induced available bandwidth variations when the `client2` is probed from `prober5`.



(a) Available bandwidth on `client1` varies much like `router1` and `router3`, as the download rate changes

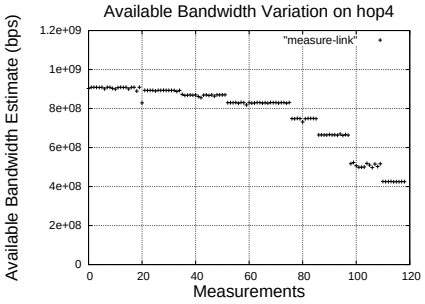
(b) Available bandwidth on `client2` doesn't vary like that on network entities along the actual download path

Fig. 9. Available Bandwidth Variation on `client1` and `client2` of DETERLAB testbed

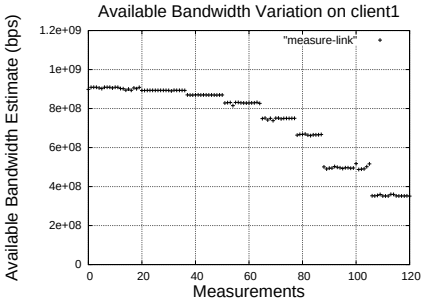
B Results from Probing Host/Routers on Lab Test-bed

All of the results obtained from probing for available bandwidth variation of the network entities were not presented in subsection 4.2. For the sake of completeness, we present the remainder of the results here in Figures 10 and 11

The results in Figures 10(a) and 10(b) are along the path which carries the download traffic. The plots in Figures 11(a) and 11(b) are for hosts that do not carry the download traffic (hence no observed variations in available link bandwidth).

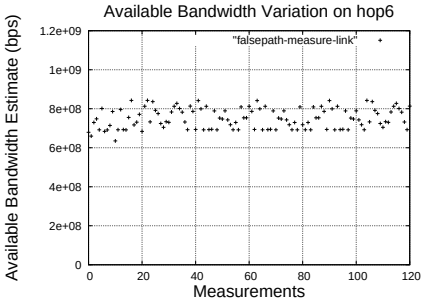


(a) Available bandwidth variation on `hop4` similar to that of `hop3`, along the actual download path

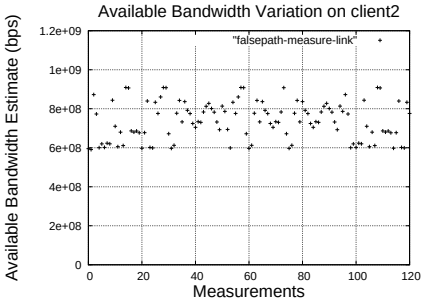


(b) Available bandwidth on `client1` varies much like `hop3` and `hop4`,as the download rate changes

Fig. 10. Available Bandwidth on Routers and End Hosts of the In-Lab Network Along the Download Path



(a) No uniform available bandwidth variation seen in `hop6`, similar to what we see in `hop3`



(b) Absense of the uniform available bandwidth variation that is observed on the actual source (`client1`)

Fig. 11. Available Bandwidth on Routers and End Hosts of the In-Lab Network Not Along the Download Path