sciendo Proceedings on Privacy Enhancing Technologies ; 2019 (4):272–291

Gerry Wan*, Aaron Johnson, Ryan Wails, Sameer Wagh, and Prateek Mittal

Guard Placement Attacks on Path Selection Algorithms for Tor

Abstract: The popularity of Tor has made it an attractive target for a variety of deanonymization and fingerprinting attacks. Location-based path selection algorithms have been proposed as a countermeasure to defend against such attacks. However, adversaries can exploit the location-awareness of these algorithms by strategically placing relays in locations that increase their chances of being selected as a client's guard. Being chosen as a guard facilitates website fingerprinting and traffic correlation attacks over extended time periods. In this work, we rigorously define and analyze the *guard placement attack*. We present novel guard placement attacks and show that three state-of-the-art path selection algorithms—Counter-RAPTOR, DeNASA, and LASTor—are vulnerable to these attacks, overcoming defenses considered by all three systems. For instance, in one attack, we show that an adversary contributing only 0.216% of Tor's total bandwidth can attain an average selection probability of 18.22%, 84× higher than what it would be under Tor currently. Our findings indicate that *existing location-based path selection algorithms allow guards to achieve disproportionately high selection probabilities relative to the cost required to run the guard.* Finally, we propose and evaluate a generic defense mechanism that provably defends any path selection algorithm against guard placement attacks. We run our defense mechanism on each of the three path selection algorithms, and find that our mechanism significantly enhances the security of these algorithms against guard placement attacks with only minimal impact to the goals or performance of the original algorithms.

DOI 10.2478/popets-2019-0069
Received 2019-02-28; revised 2019-06-15; accepted 2019-06-16.

***Corresponding Author: Gerry Wan:** Princeton University, E-mail: gwan@princeton.edu
Aaron Johnson: U.S. Naval Research Laboratory, E-mail: aaron.m.johnson@nrl.navy.mil
Ryan Wails: U.S. Naval Research Laboratory, E-mail: ryan.wails@nrl.navy.mil
Sameer Wagh: Princeton University, E-mail: swagh@princeton.edu
Prateek Mittal: Princeton University, E-mail: pmittal@princeton.edu

1 Introduction

Anonymous communication systems aim to protect the privacy of Internet users from untrusted entities. These systems hide the identities of users and prevent third parties from linking communication partners on the Internet. Today, Tor [10] is the most widely used anonymous communication system, serving millions of businesses, law-enforcement agencies, journalists, whistleblowers, and ordinary citizens from around the world. As of August 2018, the Tor network is comprised of over 6,000 volunteer-run relays and carries terabytes of traffic every day [47]. Each Tor client uses a public *consensus* to choose which relays to send its traffic through. The client uses onion routing [17] to send traffic, in which a sequence of relays is selected, a circuit is built through that sequence, and then encrypted data is forwarded along the circuit. This process protects user anonymity by preventing clients from being linked to their destinations on the Internet.

As a popular anonymity system, Tor is an attractive target for adversaries wishing to deanonymize users. Researchers have found that Tor is vulnerable to adversaries with visibility into Internet traffic [15, 21, 26, 27]. Passive attackers can use packet sizes and packet timings to correlate traffic on different segments of the Tor circuit. This ability can be used to associate traffic originating from the client with traffic flowing to the destination and thereby deanonymize the user [10, 13, 28, 29]. Website fingerprinting attacks can allow adversaries to recognize the encrypted traffic patterns of a client as those of specific websites [12, 33, 35, 38, 53]. Active attackers can manipulate the underlying Internet topology to place themselves along the path of Tor traffic [41]. To mitigate these threats, a number of systems have been developed that modify Tor's path selection algorithm to take into account the Internet locations of the relays, such as Counter-RAPTOR [40], DeNASA [5], and Astoria [30]. Other such path selection algorithms take into account the geographic location of relays, such as LASTor [1], which is designed to improve Tor's latency.

However, the location-awareness of these algorithms presents a new attack vector, in which malicious relays can be strategically placed in locations that make them more likely to be selected, leading to easier user deanonymization. To understand the threat of these *guard placement attacks*, we investigate their effectiveness on several proposed Tor path selection algorithms. Moreover, we precisely define the attack and give a generic defense algorithm that can be applied to all path selection algorithms. To the best of our knowledge, we are the first to systematically study guard placement attacks and the first to develop a framework for quantifying and mitigating the threat. Our contributions in this work are:

(A) *Theoretical formalization:* We formalize a general framework for analyzing security against guard placement attacks. This includes defining a formal threat model with an explicit adversary, giving untargeted and targeted attack versions, quantifying the adversary's success via a metric, and providing a definition that guarantees security against guard placement attacks.

(B) *Attack evaluations:* We demonstrate the threat by running our attacks on three state-of-the-art Tor path selection algorithms: Counter-RAPTOR [40], DeNASA [5], and LASTor [1]. For instance, we show that in LASTor an adversary with a bandwidth of just 0.216% of the Tor network can increase its average guard selection probability to almost 18.22%, 84× the current Tor selection probability. We remark that *we defeat the separate defenses against guard placement attacks that each individual algorithm already possesses* and showcase the importance of provably secure defenses.

(C) *Defense framework:* We propose a general technique to defend against guard placement attacks that can be applied to *any* path selection algorithm. We prove our approach secure under our definition, which bounds the advantage of an attack (Theorem 2). Finally, we apply our defense mechanism to the three algorithms we attack, and find that it is feasible to largely maintain the original goals of location-aware path selection algorithms while mitigating the threat.

Overall, our work provides a critical tool for the design and analysis of path selection algorithms for Tor.

2 Background

2.1 Tor

The Tor network is a widely deployed and popular anonymous communication system that primarily aims to prevent attackers from linking communication partners or associating online communication with a single user. Tor uses the onion routing protocol [17], in which data is transmitted over the network through a series of *relays*. A Tor client constructs a *circuit* by choosing an entry, middle, and exit relay to reach a destination on the Internet. Tor clients choose these relays from those listed in the current network *consensus*, which is updated hourly. A relay is chosen for a given position randomly with probability roughly proportional to its bandwidth weight as given in the consensus. We refer to the current algorithm for choosing relays in a circuit as *Vanilla Tor*. When creating and using the circuit, the layered encryption of onion routing ensures that each relay learns information about only the previous hop and the next hop in the circuit, and that no single relay is able to link the client to the destination [10].

To improve long-term security, Tor clients use *entry guards* (or simply *guards)* for the entry position of their circuits [31, 59]. Each client selects a small number of relays to use as guards for a long period of time. Currently, a typical Tor client selects one guard to be used for 3–4 months. For all circuits created during this time period, one of the selected guards will be used as the entry relay. Each guard is chosen by clients at random with probability proportional to bandwidth [45, 46].

To be a guard, a relay must satisfy a number of criteria that are chosen to ensure good performance and to raise the cost of obtaining the guard position. We highlight four criteria that affect this cost. First, the relay must be measured by Tor's bandwidth-measurement system, which can take up to two weeks after joining the network [48]. Second, the relay must have enough bandwidth for its consensus weight to be at least 2,000, which we estimate to require 35.5 Mbit/s (see Appendix A). Third, the relay must be online consistently enough to be considered "stable", which can be ensured by keeping the relay online at all times. Fourth, the relay must be online long enough to be considered "familiar", which takes at most eight days [45].

2.2 Tor Adversaries

Relay Adversaries. There are two general types of Tor adversaries: *relay adversaries* and *network adversaries*. Relay adversaries run Tor relays with the goal of performing traffic analysis attacks on the circuits that they are a part of. Since all Tor relays are run by volunteers with no restrictions, it is difficult to tell which ones can be trusted [11, 57, 58]. A well-known threat is when the adversary controls both the entry and the exit relay in a single circuit, allowing them to trivially deanonymize the client [10, 13, 21]. Another well-studied attack, called website fingerprinting, only requires the adversary to control the entry relay [22, 24, 32, 33, 35, 38, 53, 55].

To defend against attacks that can be performed by malicious entry relays, Tor uses the same guards in the entry position for all circuit creations during the lifetime of the guards (typically several months). This provides a long-term defense by preventing clients from quickly choosing a malicious entry relay [9]. Elahi et al. [14] observe that the use of guards prevents a relay adversary from compromising a large set of clients in a short amount of time, and Johnson et al. [21] observe that the frequency of guard selections limits the speed of compromise by a relay adversary. In this work, we focus on the threat of malicious guards.

Network Adversaries. Network adversaries do not run malicious relays. Instead, they leverage their position as a network operator to observe some portion of a client's circuit. These can include Autonomous Systems (ASes), Internet Service Providers (ISPs), and Internet Exchange Points (IXPs) [27]. A single such network entity can potentially observe both sides of a Tor circuit, performing traffic correlation attacks to link a client to its destination [15, 42]. Asymmetric traffic analysis (i.e. correlating traffic flows in different directions) and active attacks that exploit BGP dynamics can deanonymize users even more effectively [41]. Further work has also shown that network adversaries can exploit client mobility and user behavior over time to deanonymize users or leak information about their network location [51].

2.3 Location-Aware Path Selection

The current Tor network does little to defend against network adversaries [10]. A number of location-aware path selection algorithms have been proposed to defend against passive and active AS-level adversaries, as well as to enhance Tor's performance.

Passive AS-level adversary defenses. Edman and Syverson [13] propose an AS-aware path selection algorithm that uses AS topology snapshots to avoid ASes that appear both between the client and guard and between the exit and destination. Nithyanand et al. [30] propose Astoria, a similar AS-aware Tor path selection algorithm that also considers asymmetric attackers, colluding attackers, and load-balancing across the Tor network. Furthermore, it ensures that in the case where no safe paths are available, the Tor client chooses guard and exit relays in a way that will minimize the chance of a successful attack. Barton and Wright [5] propose a destination-naïve AS-aware path selection approach called DeNASA. DeNASA chooses guards by avoiding network paths that contain an empirically-determined list of "suspect" ASes.

Active AS-level adversary defenses. To improve Tor security against BGP hijack attacks [41], Sun et al. propose a new guard selection algorithm called Counter-RAPTOR [40]. The resilience of each candidate guard to BGP hijacks is determined based on the client and guard location, and this is factored into the guard's selection probability. Resilience is defined as the probability of a client source AS not being deceived by a false BGP advertisement launched by an attacker against the guard AS. Counter-RAPTOR is shown to improve the resilience experienced by Tor clients up to 36% on average and up to 166% for certain clients.

Performance improvements. LASTor [1] is a location-aware path selection algorithm designed to reduce Tor latency. LASTor incorporates some awareness of passive AS-level adversaries, but primarily favors relays that minimize the geographic distance from the client and through the circuit to the destination. LASTor is able to reduce median path latencies by 25%.

These location-aware path selection algorithms use client and guard location to choose guard relays in order to protect against AS-level adversaries and improve performance. However, in doing so, these systems make themselves vulnerable to the guard placement attack.

3 Models and Definitions

3.1 Adversary Model

To perform a guard placement attack, the adversary needs bandwidth to contribute to his malicious guards and IP addresses to host relays (Tor enforces a limit of two relays per IP). A global and competitive market exists for hosting services, and so this attack can be performed by nearly anyone with a small amount of money and an Internet connection. In particular, privileged points of network observation are not necessary. This is a weak adversary that falls within the threat models considered by the systems we attack as well as by Tor itself.

Adversary Resource Endowment. We formalize the adversary's resources using the following parameters:

1. *Total bandwidth* (B): The total amount of bandwidth the adversary can support (across all relays).
2. *Number of guards* (K): The total number of guard relays the adversary can deploy.
3. *Set of candidate guard locations* (L): The set of locations where it is feasible for the adversary to deploy relays. Possible examples of L include all ASes on the Internet, all ASes hosting at least one Tor relay, or all geographic coordinates within certain regions.

Attack Parameters. The attack itself is a function of the following parameters:

1. *Client locations* (C): The set of possible locations of clients that the adversary would like to attack.
2. *Guard selection algorithm* (A): The guard selection component of the path selection algorithm used by Tor clients. We will consider the following algorithms: (1) Vanilla Tor ($A = \mathsf{VT}$), (2) Counter-RAPTOR ($A = \mathsf{CR}$), (3) DeNASA ($A = \mathsf{DN}$), and (4) LASTor ($A = \mathsf{LT}$).

3.2 Definitions

Attack Taxonomy. A guard placement attack is a type of relay-level attack. The adversary places malicious guards in network locations with the goal of maximizing the probability that at least one of the guards is selected by a client under attack. Once a malicious guard is selected, the adversary can then mount website fingerprinting or traffic correlation attacks [6, 10, 12, 21, 33, 35, 55]. In this work, we consider two types of guard placement attacks: *untargeted* and *targeted*. In the untargeted attack, the adversary attacks all likely Tor client locations. This case can apply when an adversary wishes to attack every Tor client or when he has no information about the locations of the clients of interest. In the targeted attack, the adversary attacks clients in a single location, i.e. $|C| = 1$. Of course, an adversary could target a number of client locations in between these extremes, but we find it useful to investigate them as they estimate the best and worst cases for a client.

Attack Success Metrics. We measure the success of a guard placement attack as the probability that an attacked client selects a guard of the adversary. Given an adversary with total bandwidth B, total guards K, and candidate guard locations L, an attack strategy s is a tuple of location-bandwidth pairs representing guard placements: $s = ((\ell_1, b_1), \ldots, (\ell_K, b_K))$, where $\forall_i \ell_i \in L$, $\forall_i b_i \geq 0$, and $\sum_i b_i \leq B$. Given guard selection algorithm A and a client in location $c \in C$, let $p_A(c, s)$ be the probability that the next guard selected by the client is malicious. Then our main metric for the success of attack strategy s is

$$\sigma(A, C, s) = \frac{1}{|C|} \sum_{c \in C} p_A(c, s). \qquad (1)$$

This metric quantifies the *average* success over client locations C. This can be viewed as reflecting the average risk to clients in C or the adversary's expected chance of success against a specific client knowing only that the client location is in C. We also analyze the maximum success over client locations: $\max_{c \in C} p_A(c, s)$. Note that these metrics are general for both untargeted and targeted attacks, the latter being the case where $|C| = 1$.

Attack Goal. The adversary's goal is to find a strategy s that maximizes $\sigma(A, C, s)$ given the client locations C and the clients' guard selection algorithm A. Let $S(B, K, L)$ be the set of all attack strategies given bandwidth B, number of guards K, and candidate guard locations L: $S(B, K, L) = \{((\ell_1, b_1), \ldots, (\ell_K, b_K)) : \forall_i \ell_i \in L, \forall_i b_i \geq 0, \text{and} \sum_i b_i \leq B\}$. Then we can express the goal of the attacker as selecting some strategy $s^* \in \arg\max_{s \in S(B,K,L)} \sigma(A, C, s)$. This goal applies to both the untargeted and targeted attacks, as it is parameterized by the set of client locations C.

Security Definition. We define security against the guard placement attack in a more conservative way than we measure attack success. While attack success is measured given an adversary strategy, a meaningful security definition must take into account all possible adversary strategies. However, the resources needed by these strategies must also be incorporated into the definition. The reason is that Tor (and onion routing in general) makes no trust assumptions on the relays and prioritizes performance, and so an adversary that contributes the majority of the guard resources should have his guards selected by the majority of clients. In order to quantify the adversary's maximum possible success relative to his contribution, we unify the resources contributed by relays under a single *cost* parameter. This cost includes the time required to run the relays and obtain the necessary `GUARD` flag. We give a specific cost model in Section 3.3, but our framework is generic and applies to any cost model.

We define the cost of running a guard placement attack as the cost of running the adversary's guards relative to the total cost of running all guards in the

Tor network. Let $\mathsf{relCost}(g)$ be the cost of running guard g divided by the cost of running all Tor guards. Costs are additive in our model, and so the relative cost of running a set of guards G is $\sum_{g\in G}\mathsf{relCost}(g)$. Note that $\mathsf{relCost}(g) \in [0,1]$, and $\sum_{g\in\mathcal{G}}\mathsf{relCost}(g) = 1$, where $\mathcal{G}$ is the set of all guards in the network.

Using this cost model, we define security as the maximum success of the attacker relative to his cost. We will define security over all possible client locations $\mathcal{C}$ (i.e. not just those in C that the adversary intentionally attacks) to guarantee security to all clients. Let S be all possible attack strategies given all *possible* guard locations $\mathcal{L}$: $S = \cup_{B\geq 0, K\geq 0, L\subseteq\mathcal{L}} S(B,K,L)$. The maximum success relative to cost is

$$\overline{\sigma}(A) = \max_{s\in S}\max_{c\in\mathcal{C}} \frac{p_A(c,s)}{\sum_{g\in s}\mathsf{relCost}(g)}. \tag{2}$$

Observe that, because of the maximization over all strategies, $\overline{\sigma}(A)$ bounds the absolute success of any specific strategy s after adjusting for its cost: $\overline{\sigma}(A) \geq \sigma(A,C,s)/\sum_{g\in s}\mathsf{relCost}(g)$. Thus our security notion, given in Definition 1, simply bounds $\overline{\sigma}(A)$.

Definition 1. *Path selection algorithm A is secure against guard placement attacks with parameter θ, i.e. is* θ-GP-secure*, if $\overline{\sigma}(A) \leq \theta$.*

Definition 1 provides a strong notion of security. Because it bounds the maximum success over all strategies, it applies to all adversaries and attacks. Moreover, because it considers the maximum over all possible client locations, it provides the same security guarantee to all clients. This use of a worst case metric is in contrast to our attack metric σ, which considers success averaged over a set of client locations. While the weaker average metric is useful to understand the threat of specific attacks, it is less appropriate as a security definition as a bounded average could still leave certain client locations highly vulnerable to attack and may even allow every client location to be vulnerable when individually targeted. Note that Definition 1 applies to all strategy components—not just the malicious guard locations. In particular, the definition covers strategies that simply vary the number and bandwidths of the malicious guards.

Proving that an algorithm satisfies Definition 1 is not straightforward due to the maximization over all strategies. However, we can show that it is equivalent to a simpler condition on the path selection algorithm. Let $f_A(c,g)$ be the probability that a client using path selection algorithm A in location c chooses g as its next guard. Let $\mathcal{G}$ be the set of all guards. The maximum probability-cost ratio for path selection algorithm A is

$$\rho(A) = \max_{c\in\mathcal{C}}\max_{g\in\mathcal{G}} \frac{f_A(c,g)}{\mathsf{relCost}(g)}. \tag{3}$$

Theorem 1 shows that we can just bound $\rho(A)$ to prove that A satisfies Definition 1. Its proof is in Appendix C.

Theorem 1. *Path selection algorithm A is θ-GP-secure if and only if $\rho(A) \leq \theta$.*

3.3 Empirical Cost Model

We propose a cost model derived from empirical analysis of the prices of commercial hosting providers. Using data from Tor Metrics [47], we identify the top 10 ASes in the Tor network by total relay consensus weight (see Appendix D). Of these 10, we find 7 that provide commercial hosting and list prices online. For each of these 7 providers, we identify the cheapest server price for each bandwidth offered. Moreover, we include the possibility of running two relays on the same IP address (as limited by Tor) and splitting the bandwidth and cost between them. If the host allows purchasing extra IP addresses, we consider additional splitting of the bandwidth, up to 32 possible IP addresses, which would constitute an entire /24. Then, to determine the cheapest price for a given bandwidth, we consider the cheapest possible option among all providers with at least that bandwidth.

In every case, the cheapest provider was Online SAS. Each bandwidth could be obtained at the cheapest price from one of three of its products: a dedicated server at 1,000 Mbps for \$11.40/month, a cloud server at 200 Mbps for \$4.55/month, and a cloud server at 100 Mbps for \$2.28/month. We emphasize that these costs are for running the relays for one month, which takes into consideration the time it takes to obtain the `GUARD` flag. The exact cost model is given in Appendix D.

To obtain a guard's relative cost (i.e. $\mathsf{relCost}$), the guard's bandwidth is input to the model to determine the absolute cost, and that value is divided by the sum of the absolute guard costs. We use a linear regression on the past consensus weights and self-advertised bandwidths of Tor's guards to convert consensus weights to the bandwidths used in the cost model. The regression is necessary because the consensus weights are the result of a load-balancing algorithm [34] that causes them to differ substantially from the true bandwidths. This regression has a coefficient of determination of $r^2 = 0.86$. See Appendix A for more details.

While our cost model is based on limited data, Definition 1 is somewhat robust to any inaccuracies. If the model estimates the relative cost c of any set of guards to be $(1+\varepsilon)c$, then an algorithm with θ-GP-security under the model will instead actually have $(1+\varepsilon)\theta$-GP security. Moreover, while any cost model will not reflect the costs to all adversaries at all times, any guard selection algorithm will give an adversary some success relative to his cost, and we argue that explicitly modeling the cost increases the chance of successfully limiting his relative success.

4 Case Study I: Counter-RAPTOR

Counter-RAPTOR [40] modifies the way guards are chosen in Tor to improve security against BGP hijack attacks. In Counter-RAPTOR, the client computes, for each guard g, the guard's resilience $r(g)$, which is the probability that the client's AS is not deceived by an equally-specific prefix attack on the guard's AS. The resiliences are used as a component of the weights used by the client to select guards. To prevent a client from being too biased towards any one guard, Tille's algorithm [49] is applied to the resiliences to produce more uniform values $r'(g)$ (for details, see Appendix B). Guard g, with normalized bandwidth $b(g)$, is then given weight

$$w(g) = \alpha \cdot r'(g) + (1-\alpha) \cdot b(g), \tag{4}$$

where α is a parameter that trades off attack resilience and performance (the recommended value is $\alpha = 0.5$). We show that, despite the use of Tille's algorithm, Counter-RAPTOR is still vulnerable to guard placement attacks.

4.1 Untargeted Attack – Counter-RAPTOR

We first analyze the success of an untargeted attack in which the attacker seeks a high average guard-selection probability over many client locations. When placing a guard, the attacker benefits from choosing a location that has high resilience with respect to many client locations. We consider the success of the attack under different bandwidths and numbers of guards.

Experimental Setup. We attack the locations (C) in the list of 368 top Tor client ASes measured by Juen [23]. We let the set of candidate ASes for running the malicious guard (L) include all ASes that already contain at least one Tor relay. We obtain Tor network data from CollecTor [44] and retrieve relevant relay information from an August 1, 2018 consensus. All IP to AS mappings are done using Team Cymru data [43], and we use Internet topology data from CAIDA [8]. AS path prediction [16] is used to compute resiliences. This data is used to model Tor and Internet routing throughout the paper unless otherwise noted.

Varying the bandwidth. We use one malicious guard ($K = 1$) and consider seven consensus weight values B: 2,000; 3,000; 7,500; 10,000; 30,000; 75,000; and 150,000. These bandwidths are the Tor consensus weights across the range of existing guard bandwidths. Our linear regression on consensus weights and advertised bandwidths (Appendix A) shows that a guard with a weight of 2,000, which is 0.011% of the total guard weight, likely has an actual bandwidth of 35.5 Mbit/s, and a guard with a weight of 150,000, which is 0.81% of the total guard weight, has a predicted actual bandwidth of 939.8 Mbit/s. To optimize the attack success σ (Equation 1) against all client locations (C), the adversary computes the success probability for placing a malicious guard in each of the candidate ASes (L), and then chooses the location with the largest attack success. This is the same AS regardless of bandwidth, and we find that it is AS199524 (G-Core Labs).

We consider the attack success probabilities under Counter-RAPTOR from two perspectives. First, to show the increase in guard probability compared to today's Tor network, we include the success probability under Vanilla Tor. Second, we show the relative cost of the attack (Section 3.3) to present what would be the "ideal" success of the attacker under our cost model.

Figure 1 shows the untargeted attack success for a varying attacker bandwidth. The shaded areas represent the range of guard-selection probabilities over the client locations even though no specific one is being targeted. For the smallest bandwidth guard shown (2,000), we can see that it achieves a selection probability of 0.046%, which is 4.3× greater than the Vanilla Tor success probability and 1.37× greater than the relative cost of the guard. This means that it would be feasible for small adversaries (such as a single person) to have higher success rates on average than they could on today's Tor network. For all bandwidths, the highest success probability is obtained against AS28885. The adversary has the highest relative advantage against that client location for a bandwidth of 2,000, giving them a 12.7× increase in success rate compared to Vanilla Tor and a 4.1× increase over the ideal cost-based probability, even *without targeting those client locations specifically.* We

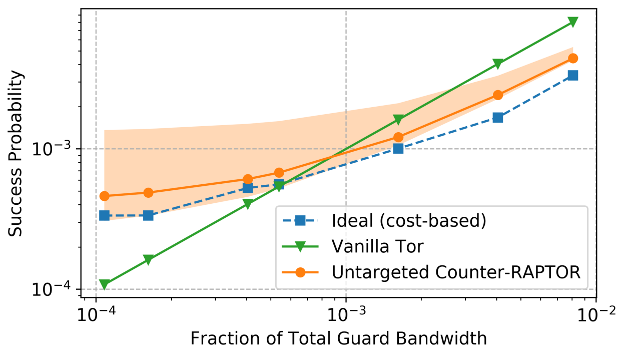

Fig. 1. Success probability of an untargeted attack on Counter-RAPTOR clients with 1 malicious guard and varying bandwidths. Shaded areas show range of success over client locations.

# Guards (K)	relCost (%)	Avg Success (%)	Max Success (%)
1	0.134	0.148	0.238
2	0.153	0.189	0.368
3	0.181	0.230	0.498
5	0.263	0.311	0.754
10	0.334	0.512	1.384
20	0.665	0.909	2.597
Vanilla	0.134	0.216	0.216

Table 1. Success probability of an untargeted attack on Counter-RAPTOR clients with a fixed bandwidth of 40,000 (0.216% of total guard bandwidth) and varying number of malicious guards.

can also see that a single-guard untargeted attack's relative success compared to the ideal cost-based model stays about the same as bandwidth increases, but loses effectiveness when compared to Vanilla Tor for bandwidths greater than 0.1% of the total. This is because Counter-RAPTOR inherently weights bandwidth less than Vanilla Tor (exactly 1/2, due to α). However, this does not exonerate Counter-RAPTOR; a larger adversary can instead run many small guards to obtain high absolute success.

Varying the number of relays. Because Tor does not place restrictions on how many relays one can run, a strategic adversary can deploy a large number of guards in the Tor network to better optimize his attack success probability. For a *fixed* bandwidth budget, the adversary can split the bandwidth among multiple guards and place them each strategically to increase the likelihood that one of them is selected. The adversary does need to take care not to reduce the individual bandwidth of each relay such that it falls below the minimum threshold (2,000) to be considered a guard. We observe that Counter-RAPTOR is particularly vulnerable to splitting bandwidth among multiple guards due its additive formula for guard weight (Equation 4). Dividing B bandwidth among K guards instead of one guard reduces the bandwidth term by a factor K but does not reduce the resilience term.

We use a fixed bandwidth weight of 40,000, which represents just 0.216% of the total guard bandwidth. This ensures that the adversary can divide the bandwidth evenly among up to 20 guards without falling below the minimum guard threshold (see Section 2.1). We analyze the optimal strategy, which places all malicious guards in the same AS location (AS199524).

Table 1 shows the effects of splitting a fixed bandwidth budget among an increasing number of malicious guards in an untargeted attack. Dividing the bandwidth evenly among 20 guards increases the average attack success to 0.909%, which is 4.2× higher than the Vanilla Tor success and 1.4× higher than the relative cost. Deploying multiple guards pushes the success probability for this moderately sized adversary above Vanilla Tor, and it also increases the advantage relative to the cost of running the attack. We can also see this trend in the maximum success over all clients, where the absolute success probability reaches nearly 2.6% for 20 guards, an increase of 12× over Vanilla Tor and 3.9× over the relative cost.

4.2 Targeted Attack – Counter-RAPTOR

We next analyze targeted guard placement attacks on Counter-RAPTOR. We use a single malicious guard with a bandwidth of 2,000 and for each of the 368 top Tor client ASes compare the targeted and untargeted success rates. For each of these locations, the adversary performs an exhaustive search over all candidate ASes (L) to find the one that has the maximum success probability for the client location.

Figure 2 shows the success rates of targeted and untargeted attacks over the client locations with just one guard. For visual clarity, we rank each client AS in increasing order first by untargeted success and then by targeted success. The largest increase in targeted success probability over all client ASes is 47%. However, for the 13 client ASes where the optimal untargeted placement happens to also be the optimal targeting location, targeting them specifically is no better than attacking all clients at once. For the client AS with the maximum adversary success (index 368), we can see that a

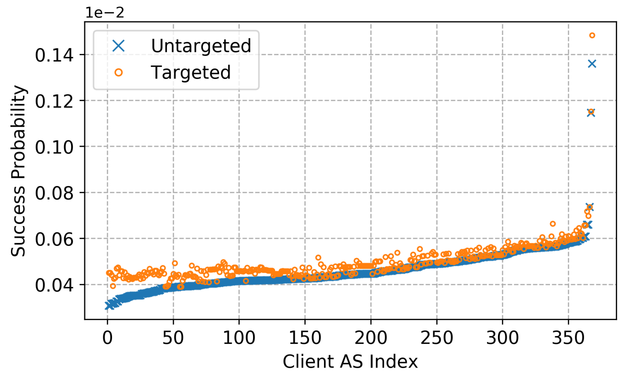

Fig. 2. Success rates of targeted attacks against Counter-RAPTOR clients using 1 malicious guard with a bandwidth of 2,000 (0.011% of total guard bandwidth).

single-guard targeted attack with a bandwidth of 2,000 can achieve a success probability of 0.15%. This is over 13.6× higher than what an adversary with the same resources could achieve against Vanilla Tor and over 4.4× higher than the relative cost.

4.3 Summary

We show that strategic guard placement attacks expose new vulnerabilities in Counter-RAPTOR. Similar to Vanilla Tor, adversaries with more bandwidth resources are able to compromise clients with higher probability. However, small adversaries with relatively little bandwidth can attack Counter-RAPTOR more successfully than they can attack today's Vanilla Tor network. We also show that targeted guard placement attacks can boost the attack's likelihood of success, and splitting the same bandwidth resource among multiple malicious guards can increase the probability that one of them is chosen.

5 Case Study II: DeNASA

DeNASA [5] is a proposal for improving security against passive AS attackers that may snoop on Tor circuits. This algorithm chooses guards weighted by bandwidth, except only among those that do not have a *Suspect AS* on the path to or from the guard. Suspect ASes are those that are frequently in a position to perform traffic correlation attacks on Tor circuits. According to the authors, the top two Suspect ASes that clients should avoid are AS3356 (Level 3) and AS1299 (Telia Company) [5]. DeNASA uses AS path inference [16] to predict whether or not a Suspect AS is on-path between client-guard pairs. Guards that do not have a Suspect AS on-path are called *suspect-free*. If there are no suspect-free guards, DeNASA resorts back to the Vanilla Tor guard selection algorithm. The number of Suspect ASes is limited to only two as a defense against guard placement attacks, but we show that this is still ineffective.

5.1 Untargeted Attack – DeNASA

We consider an untargeted attack on DeNASA in which the adversary attempts to maximize average guard-selection probability over common Tor client locations. In this attack, the adversary seeks to place a malicious guard in a location that is suspect-free from many client locations. The adversary further benefits from choosing a guard location that is suspect-free from client locations that have few other suspect-free guards and thus are more likely to choose the malicious guard.

Experimental Setup. We again analyze an untargeted attack with the 368 top Tor client ASes [23] as the client locations (C) and the ASes with at least one Tor relay as the candidate ASes (L). We use the same Internet topology and Tor data as in Section 4.

Varying the bandwidth. We again evaluate the attack success for one malicious guard ($K = 1$) and seven different guard bandwidths (B) ranging from 2,000 to 150,000 and compare it to Vanilla Tor and the ideal cost-based success. To place the guard, the adversary does an exhaustive search over all candidate ASes (L) and chooses the AS that receives the highest average selection probability over all client locations. We discover that the optimal location to place the malicious guard is in AS1659 (Taiwan Academic Network) for smaller adversaries (bandwidth weight less than 30,000), but is AS12637 (Seeweb) for larger adversaries. This difference is because once an adversary has enough bandwidth, it is better to give up on attacking client locations that are extremely vulnerable in favor of attacking many more clients that are only somewhat vulnerable.

Figure 3 shows the attack success. For the smallest bandwidth shown (2,000), the adversary achieves on average 0.043% success. This is a relative advantage of 3.9 over Vanilla Tor and 1.29 over the relative cost. The shaded area shows the range over all client locations of the probability that the malicious guard is selected provided that the adversary places the guard in the location that maximizes average success. There is a wide range of attack success across client locations for any given bandwidth, even though this attack is un-

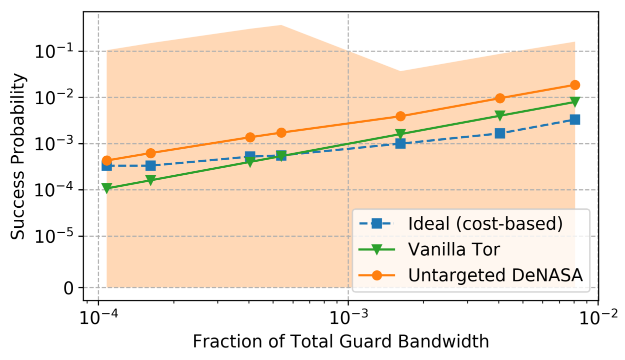

Fig. 3. Success probability of an untargeted attack on DeNASA clients with 1 malicious guard and varying bandwidths. Shaded areas show range of success over client locations.

# Guards (K)	relCost (%)	Avg Success (%)	Max Success (%)
1	0.134	0.522	4.896
2	0.153	0.555	54.16
3	0.181	0.535	61.17
5	0.263	0.544	58.64
10	0.334	0.531	62.32
20	0.665	0.531	62.32
Vanilla	0.134	0.216	0.216

Table 2. Success probability of an untargeted attack on DeNASA clients with a fixed bandwidth of 40,000 (0.216% of total guard bandwidth) and varying number of malicious guards.

targeted. Clients in the worst-case location (AS30083) have an extremely high probability of selecting the malicious guard. In particular, for a bandwidth of 2,000, a single malicious guard placed in AS1659 achieves a selection probability of 10.6% (964× that of Vanilla Tor, 316× the relative cost) for clients in AS30083. The reason for such large success probabilities is that AS30083 can only reach one non-malicious suspect-free guard. For large bandwidths, the worst-case client location becomes AS36992. A guard with a bandwidth of 150,000, which represents just 0.81% of total guard bandwidth, achieves success rates of 16.2%, even though AS36992 is not specifically targeted.

Varying the number of relays. With DeNASA, an adversary must place guards in *separate* ASes to gain an advantage in running multiple small guards with a fixed total bandwidth. Deploying multiple guards within the same AS will have no effect on average success probability because the entire AS is either suspect-free or not. By deploying a fraction of the guard bandwidth in a separate AS, the adversary may be able to capture some clients that were not able to reach the first AS due to an on-path Suspect AS. In our attack analysis, we implement a greedy algorithm that places each additional guard in the candidate AS that maximally increases the success probability. Note that this is just a heuristic and may not find the optimal strategy.

Table 2 shows the effect of splitting a bandwidth weight of 40,000 among up to 20 malicious guards. Note that the success probabilities do not strictly increase because we use a heuristic. We can see that running two guards instead of just one vastly increases the maximum success probability to 54.2%. This is 251× the Vanilla Tor success rate and 354× the relative cost. The reason for this huge improvement is that the second guard can be deployed in an AS that is suspect-free from client AS30083, while the optimal location for deploying just one guard of size 40,000 cannot reach this particularly vulnerable client AS. We further see that adding even more guards does not increase the average success or the maximum success, but does increase the relative cost. This is because while adding more guards does allow the adversary to attack more client locations, it also removes bandwidth from the most effective attack vantage points. Since there is some cost to deploying more guards, an adversary attacking DeNASA should use no more than two or three guards placed in separate ASes.

5.2 Targeted Attack – DeNASA

In a targeted guard placement attack, the adversary maximizes his success by placing his guard in an AS that is suspect-free from the target client. This is possible as long as the client is not in one of the Suspect ASes and the set of candidate guard locations L is large enough. For example, if the client's AS is in the set of candidate guard locations L, the adversary can simply place the malicious guard in that AS. If there exists no suspect-free AS for a targeted client location, then DeNASA chooses guards based on bandwidth, and so the adversary can choose any location for his guard.

We show in Figure 4 the success rate of targeted attacks using one malicious relay with a bandwidth of 2,000 against each client AS. Again, we rank each AS in increasing order first by untargeted success and then by targeted success. For the 50 of 368 client ASes (13.6%) that had zero probability of compromise in the untargeted attack, we find that all are able to reach at least one suspect-free AS, and so targeting them specifically gives a non-zero success rate. For the other 318 client ASes, targeting them does no better than generally at-

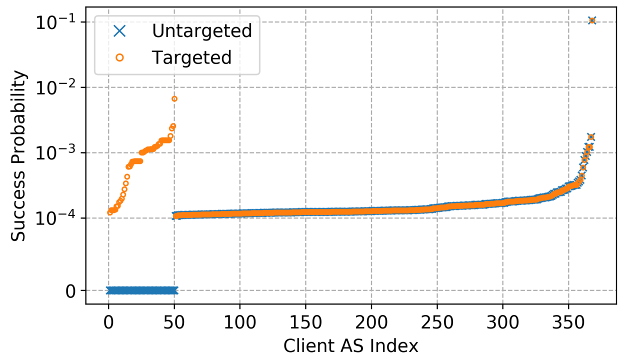

Fig. 4. Success rates of targeted attacks against DeNASA clients using 1 malicious guard relay with a bandwidth of 2,000 (0.011% of total guard bandwidth). For 86.4% client ASes, the targeted and untargeted success rates are equivalent (the overlapping points).

tacking all clients. Figure 4 also shows that the success of a targeted attack in DeNASA is heavily dependent on the target client AS. For about five percent of clients, the probability of successful attacks targeting these ASes are bigger than an order of magnitude higher than what the same adversary would be capable of in the Vanilla Tor network. While a majority of clients can reach most non-adversarial guards without traversing a Suspect AS, there are a few ASes that are especially vulnerable to adversaries that leverage DeNASA's location-awareness.

5.3 Summary

We demonstrate vulnerability in DeNASA to the guard placement attack in that an adversary can strategically place guards in suspect-free locations. We show that for an untargeted attack with a single guard of bandwidth weight 2,000, a DeNASA client chooses the malicious guard with average probability 3.9 times greater than a Vanilla Tor client does, and in the worst case the selection probability can be more than 964 times greater. This shows that the guard placement success rates of an adversary varies significantly across client locations. For large enough guards, the maximum absolute success probability is also large. We also show that splitting the bandwidth among a few guards in separate ASes can greatly increase the untargeted success rate by reaching more clients. However, it is also important to note that such deployment more comes at a higher cost to the adversary. Finally, we show that targeted attacks are successful against clients that were not vulnerable to the untargeted attack.

6 Case Study III: LASTor

LASTor [1] is a location-aware path selection algorithm that primarily aims to reduce latency of communication on Tor. While the proposal offers some additional AS-awareness to defend against passive AS adversaries, it is only incorporated in the path selection algorithm after guards are chosen. LASTor uses a Weighted Shortest Path algorithm that selects a given path with probability inversely proportional to the expected latency between the client and destination. Network latency is approximated by the end-to-end geographical distance. Thus, the algorithm attempts to select a path close to the direct line between the client and the destination.

LASTor includes relay clustering as a defense against guard placement attacks. This technique limits the success of an adversary strategy that places all of the malicious relays in the same location. However, we demonstrate that it is ineffective in general against guard placement. The clustering algorithm divides the globe into a grid of cells and includes a cluster for each cell containing all the relays within that cell. The recommended edge lengths of each cell are 2 degrees of latitude and longitude. To select guards, all guards are first clustered, and then distance from each cluster is computed as the great-circle distance [50] from the center of the cluster to the client. The client selects a "feasible" cluster uniformly at random from the closest 20% of clusters and then picks one guard uniformly at random among the guards in that cluster. Note that LASTor does not consider relay bandwidth.

6.1 Untargeted Attack – LASTor

We consider an untargeted attack on LASTor in which the adversary seeks high average selection probability over many likely physical locations for Tor clients. The adversary benefits by placing his guards close to many of the client locations. Because LASTor does not take bandwidth into account, the adversary further benefits from having as many guards as possible.

Experimental Setup. We choose the client locations (C) from the ten countries with the most directly-connecting Tor users. We use 200 cities located across these countries as client locations. The locations in the i^{th} country are the geographic coordinates of the $200 f_i$ most-populous cities in that country, where f_i is that country's fraction of Tor users among the top-ten countries. We obtain the top ten Tor countries and their f_i values using data from Tor Metrics [47]. We let the can-

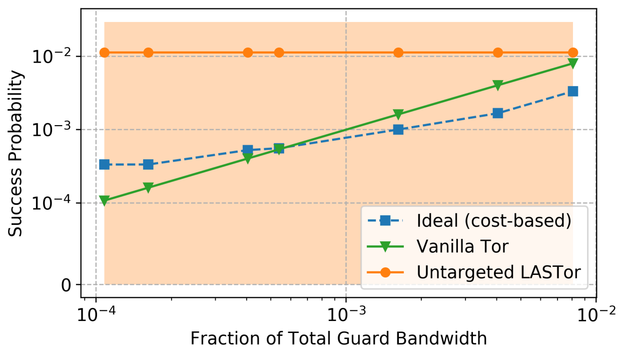

Fig. 5. Success probability of an untargeted attack on LASTor clients with 1 malicious guard and varying bandwidths. The shaded area shows ranges of success over client locations.

# Guards (K)	relCost (%)	Avg Success (%)	Max Success (%)
1	0.134	1.132	2.941
2	0.153	2.250	5.882
3	0.181	3.353	8.824
5	0.263	4.414	14.29
10	0.334	10.22	27.78
20	0.665	18.22	34.21
Vanilla	0.134	0.216	0.216

Table 3. Success probability of an untargeted attack on LASTor clients with a fixed bandwidth of 40,000 (0.216% of total guard bandwidth) and a varying number of malicious guards.

didate locations for running the guard (L) be the set of all relay clusters that already contain at least one Tor relay. We use the Maxmind GeoIP database [25] for IP to geo-location mapping.

Varying the bandwidth. We again consider a single malicious guard ($K = 1$) with seven bandwidth weights (B) from 2,000 to 150,000. To choose the guard's location, the adversary finds the optimal strategy by computing for each candidate location its potential guard selection probability from all client locations and choosing the one that maximizes the average probability (i.e. σ, see Equation 1). We find that this optimal location is just north of Moscow (57.8794, 34.9925).

Figure 5 shows the attack's success on the set of client locations. It is clear that LASTor selection probabilities have no dependency on bandwidth, giving an adversary running a small guard a significant advantage compared to both Vanilla Tor and the relative cost. A malicious guard with consensus weight 2,000, which is the minimum weight that we examine, obtains a 1.13% average success probability over all clients, 103× greater than what the same adversary would obtain under Vanilla Tor, and 34× greater than the relative cost. The shading shows that the success rates range widely from 0% to 2.94% for all bandwidths. The constant maximum success probability over all clients indicates that there is always at least one client location that has the malicious guard's cluster within its closest 20%. The constant minimum success probability, which occurs for 61% of client locations, is because it is not in the closest 20% of clusters for those locations.

Varying the number of relays. Because the LASTor guard selection algorithm does not depend on bandwidth, it is particularly susceptible to an adversary that has limited bandwidth but can run multiple relays. Moreover, an adversary with multiple relays can strategically place them in separate clusters to obtain positive success probability from more client locations. We consider an adversary implementing a greedy algorithm to insert malicious guards into the Tor network, where each added guard is placed to maximize the increase in average success probability. Note that this is a heuristic may not find the optimal strategy.

In Table 3 we show the success probabilities of untargeted attacks on the 200 client locations with a bandwidth budget of 40,000 and up to 20 relays ($K = 20$). As we can see, the success probability increases nearly linearly with the number of relays even while keeping the total bandwidth constant. Thus, the attacker obtains the highest advantage by running 20 relays, in which case he increases the average success probability 84× over Vanilla Tor and 27× over the relative cost. The maximum success probability is 158× higher than Vanilla Tor's and 51× higher than the relative cost.

6.2 Targeted Attack – LASTor

In the targeted attack, the adversary focuses on a single client location and attempts to maximize his success probability with respect to this target. LASTor first chooses a feasible cluster, then chooses a guard within that cluster. Therefore, a strategic adversary would want to place guards in feasible clusters that are geographically close to the client but that contain as few guards as possible. We find that an adversary specifically targeting any one of the 200 client locations is always able to find a feasible cluster that is within the closest 20% and that does not already contain a guard. This gives a malicious guard targeting any specific client a 2.94% chance of being selected from among the 1,935 other guards in the Tor network, *regardless* *of its bandwidth.* By comparison, the highest-bandwidth non-malicious guard has a 0.74% chance of selection under Vanilla Tor. We emphasize that this targeted attack success applies to all 200 client locations, even those that have zero chance of selecting a malicious guard in the untargeted attack.

6.3 Summary

We demonstrate serious vulnerabilities in LASTor to the guard placement attack, despite the relay clustering that LASTor uses as a defense to that very attack. With just a single malicious guard, a low-bandwidth adversary can exploit the location-awareness of LASTor to increase its guard selection probability by more than two orders of magnitude over Vanilla Tor. We also show that splitting the adversary's bandwidth among multiple guards and placing them in separate clusters drastically increases the attack success probability.

7 Countermeasures

In this section, we present a meta-algorithm that modifies the guard selection component of any path selection algorithm to provably defend against guard placement attacks. We apply this algorithm to Counter-RAPTOR, DeNASA, and LASTor and evaluate its effect on their security and performance.

7.1 Defense Mechanism

In a successful guard placement attack, the adversary obtains a success probability that is disproportionate to the fraction of Tor's guard resources that he contributes. We therefore propose a defense mechanism that bounds guard selection probabilities relative to their costs. Our defense mechanism can be applied to any guard selection algorithm, as it operates by interacting with the algorithm to produce a modified guard selection distribution. The mechanism takes as input a desired bound on the guards' probability-cost ratios, and it produces a distribution satisfying this bound while preserving as much as possible the security benefits of the original guard distribution.

Before describing the mechanism, we introduce some notation. Let A be the client's algorithm for selecting its next guard. Let $\theta \geq 1$ be a security parameter indicating the desired bound on the guards' probability-cost ratios. Intuitively, θ represents the maximum relative advantage the network is willing to give to an adversary running a guard placement attack. Let $\mathcal{G}$ be the set of guards in the Tor network. For $g \in \mathcal{G}$, recall (Section 3.2) that $\mathsf{relCost}(g)$ denotes its cost fraction. Also recall that $f_A(g)$ denotes the selection probability of g for a client using A (we drop the client parameter c for simplicity). We assume that A produces this distribution given the set of guards (i.e. $f_A = A(\mathcal{G})$).

The defense mechanism D is given in Algorithm 1. It takes in A and θ and produces a guard selection distribution f'_A that bounds the probability-cost ratio for all guards. D operates by asking A for its desired guard selection distribution, enforces the θ bound on that distribution by potentially reducing guard probabilities, removes any guards thus limited, and repeats to assign to the remaining guards the probability in excess of the bound from the removed guards. Thus D repeatedly uses A as the guide for how the current unassigned probability should be allocated among guards that haven't yet met the θ bound. D uses each distribution that A produces as much as possible, only reducing some desired probabilities if they would cause the guard to exceed the θ bound. We prove in Appendix C that this algorithm terminates (Theorem 2), as it is not obvious from the description that it does so.

7.2 Defense Framework Evaluation

We evaluate the security and performance of our defense framework from three perspectives: (1) how well it reduces vulnerability to a guard placement attack, (2) how well the modified guard selection algorithm maintains its original goal (e.g. increasing hijack resilience), and (3) how it affects the load balancing of clients over guards. We apply Algorithm 1 to Counter-RAPTOR, DeNASA, and LASTor with varying threshold values.

7.2.1 Security from Guard Placement Attack

The defense algorithm bounds the extent to which adversaries can exploit the location-awareness of the guard selection algorithm to achieve high selection probabilities. Tor clients can apply this defense to their location-aware path selection algorithm to mitigate guard placement attacks while maintaining AS-awareness or latency benefits. Theorem 2 shows that modifying a guard selection algorithm using the defense makes it robust to guard placement attacks in general, regardless of attacker strategy.

Algorithm 1: Defense algorithm D

```
   Input: Guard selection algorithm $A$, security
          parameter $\theta$
   Output: Guard selection distribution $f'_A$
1  forall $g \in \mathcal{G}$ do
2      $f'_A(g) \leftarrow 0$
3  end
4  $B \leftarrow \mathcal{G}$                  // Guards below threshold
5  $p \leftarrow 1$                  // Probability to allocate
6  repeat
7      $x \leftarrow 0$                  // Excess probability
8      $f_A \leftarrow A(B)$
9      forall $g \in B$ do
10         if $f'_A(g) + p \cdot f_A(g) \geq \theta \cdot \mathsf{relCost}(g)$ then
11             $x \leftarrow x + f'_A(g) + p \cdot f_A(g) - \theta \cdot \mathsf{relCost}(g)$
12             $f'_A(g) \leftarrow \theta \cdot \mathsf{relCost}(g)$
13             $B \leftarrow B - g$
14         else
15             $f'_A(g) \leftarrow f'_A(g) + p \cdot f_A(g)$
16         end
17     end
18     $p \leftarrow x$
19 until $x = 0$
20 return $f'_A$
```

Theorem 2. *Let A be any guard selection algorithm and $\theta \geq 1$ be the security parameter. Then using the guard selection distribution $f'_A = D(A, \theta)$ is θ-GP-secure.*

In other words, applying Algorithm 1 and using the resulting guard selection probability distribution is secure against guard placement even though the original guard selection algorithm may not be. An additional consequence of Theorem 2 is that applying Algorithm 1 limits any advantage the adversary might obtain from splitting his bandwidth among multiple relays, as Definition 1 allows the adversary to vary the number and bandwidths of guards in addition to their locations. We defer the proof of Theorem 2 to Appendix C.

We evaluate how much the defense would improve the security of using the location-aware algorithms on the existing Tor network by computing the probability-cost ratios for each algorithm. The probability-cost ratio of guard g under algorithm A is $\rho(g) = f_A(g)/\mathsf{relCost}(g)$. We determine the values of $\rho(g)$ under our cost model for each existing guard in the August 1, 2018 consensus. We do not insert any malicious guards.

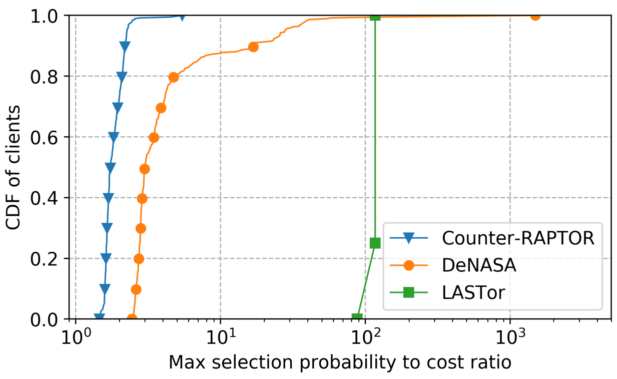

Fig. 6. Maximum selection probability to cost ratio for clients choosing guards in the August 1, 2018 Tor consensus.

Figure 6 shows the maximum probability-cost ratio for each client location under Counter-RAPTOR, DeNASA, and LASTor. These values represent the ability of existing guards to attract clients relative to their costs (recall Equation 3). For Counter-RAPTOR, the worst-case client location has at least one guard with a ratio of 5.4. Some clients under DeNASA could select a guard with a probability 1,490× the cost it took to deploy the guard, but this ratio varies greatly across clients. LASTor always gives some existing guard a large advantage, and 75% of client locations have a maximum probability-cost ratio of greater than 100.

These results show that even on the current network, applying proposed location-aware algorithms would give some guards a chance to observe clients that is disproportionate to their cost. We emphasize that these results only include relays that currently exist, and a strategic attack on a specific location-aware algorithm would yield even higher success, as shown in Sections 4–6. Furthermore, Theorem 2 shows that applying Algorithm 1 to any of these algorithms would mitigate the threat by guaranteeing that the maximum probability-cost ratio *for all client locations* would be no higher than the desired limit θ.

7.2.2 Algorithms' Original Goals

The defense meta-algorithm provably limits the advantage of a guard placement attack to a factor of θ, but it does impact the original goals of the location-aware path selection algorithm. However, we show that for reasonable values of θ that impact is generally small. In the following sections, again let $f_A(g)$ denote the probabil-

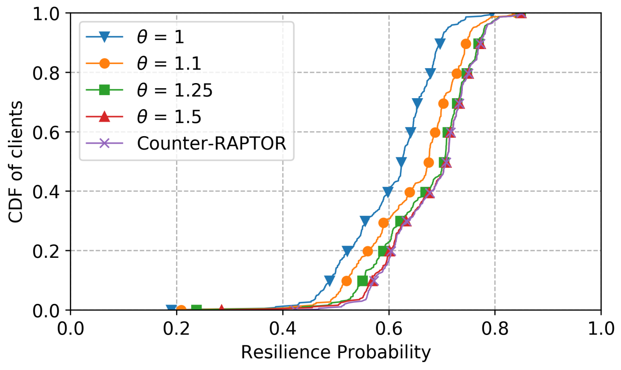

Fig. 7. Probability of being resilient to attacks with different threshold (θ) values.

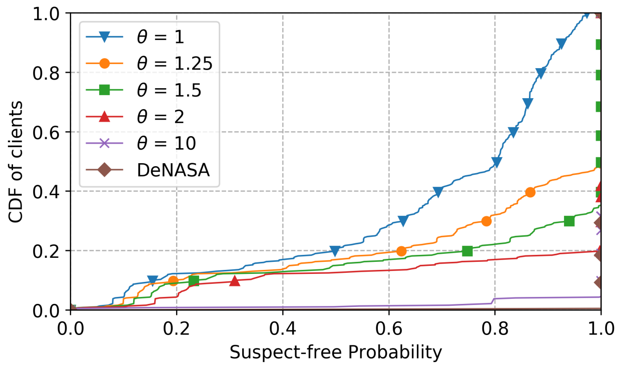

Fig. 8. Probability of choosing suspect-free guards with different threshold (θ) values.

ity that a client selects guard g under algorithm A, and let $\mathcal{G}$ denote the set of all guards in the network.

Counter-RAPTOR. Let $r(g)$ indicate the hijack resilience of guard g. The aggregated probability of a client being resilient to a BGP hijack attack can be expressed as $\sum_{g\in\mathcal{G}} f_{\mathsf{CR}}(g)r(g)$. Figure 7 shows the aggregated resilience for $\theta \in \{1, 1.1, 1.25, 1.5\}$, as well as for "pure" Counter-RAPTOR without the guard placement defense. We use the 368 top client ASes [23] and guard data from the August 1, 2018 consensus. Naturally, $\theta = 1$ has the lowest probability of being resilient to an equally-specific prefix hijack attack, while pure Counter-RAPTOR is the most resilient. For $\theta = 1$, 50% of client locations have at least 0.62 resilience probability, and in pure Counter-RAPTOR, 50% of client locations have at least 0.7 resilience probability. Observe that for $\theta = 1.25$ the resilience probabilities over all client locations are nearly identical to those of clients using pure Counter-RAPTOR. This means that we can relax the θ bound to 1.25 while maintaining the hijack resilience benefits of Counter-RAPTOR. We recommend this value to mitigate guard placement attacks as it bounds the adversary's probability-cost advantage to 1.25 and does not significantly impact the original goals of Counter-RAPTOR.

DeNASA. Let $\mathbb{1}_{\text{sf}}(g)$ be the indicator function for the AS paths between the client and guard being suspect-free, i.e., $\mathbb{1}_{\text{sf}}(g) = 1$ if the paths in both directions are suspect-free and 0 otherwise. The aggregated suspect-free probability can then be expressed as $\sum_{g\in\mathcal{G}} f_{\mathsf{DN}}(g)\mathbb{1}_{\text{sf}}(g)$. Figure 8 shows the suspect-free probability for $\theta \in \{1, 1.25, 1.5, 2, 10\}$, as well as for pure DeNASA. We use the same Tor client ASes [23] and guards as in the Counter-RAPTOR evaluation. Naturally, as θ increases, the algorithm behaves more similarly to pure DeNASA, which has the highest probability for clients to choose suspect-free guards. For pure DeNASA, 366 out of 368 client locations are guaranteed to select suspect-free guards; the only clients not selecting suspect-free guards are clients located in the two Suspect ASes themselves. As we reduce θ, the modified distribution begins to give guards that are not suspect-free non-zero probability of being selected. This is the trade-off between protecting clients from the threat of Suspect ASes and from guard placement attacks. We recommend $\theta = 2$, which has 80% of client locations still only choosing suspect-free guards while ensuring that no guard is more than twice as likely to be chosen relative to the fraction of the Tor network it contributes.

LASTor. LASTor uses geographical distance as a substitute for network latency [1]. Let $d(g)$ denote the distance to the guard from a client. The expected distance from the client to a chosen guard can then be expressed as $\sum_{g\in\mathcal{G}} f_{\mathsf{LT}}(g)d(g)$. Figure 9 shows the expected distance in kilometers for $\theta \in \{1, 2, 5, 10, 20\}$, as well as for pure LASTor. We evaluate the set of 200 client locations used in Section 6.1 and use the same guards as in the previous evaluations. Naturally, as θ decreases (mitigating guard placement attacks), the expected distance of a guard selected by a client increases. The median expected distances range between 1,348 km to 4,375 km, depending on choice of θ. We recommend $\theta = 5$ for LASTor, which has 50% of client locations still choosing guards within an 3,104 km.

7.2.3 Performance Impact

We will show that our defense mechanism can be applied to Counter-RAPTOR, DeNASA, and LASTor without negatively impacting the performance of Tor's load balancing. In fact, application of the defense can improve

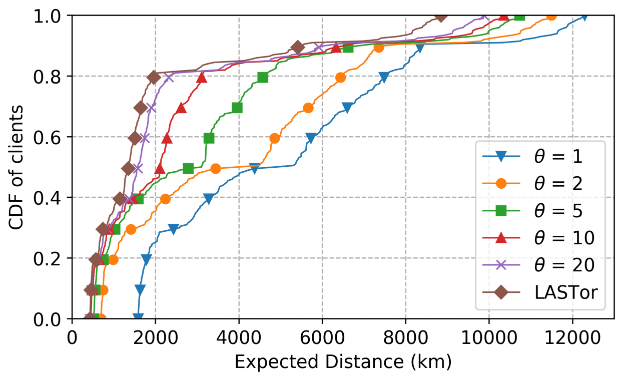

Fig. 9. Expected guard distance with different threshold (θ) values.

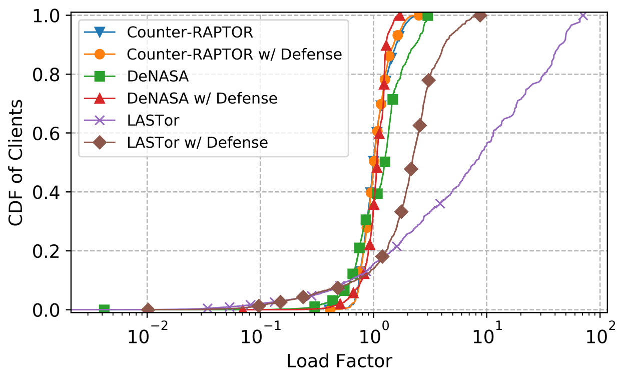

Fig. 10. Distribution of expected guard load factors for location-aware algorithms with and without guard placement defense applied.

the network load balance because it prevents attractive relays from being overloaded with client traffic.

We examine guards' expected load under these path-selection algorithms, both with and without the application of our defense mechanism. When computing expected guard loads for Counter-RAPTOR and DeNASA, we assume clients are distributed in the top 368 client ASes according to the densities measured by Juen [23]. When computing expected loads for LASTor, we assume clients are geographically distributed according to our experimental setup described in Section 6.1. Following Tor's existing load-balancing strategy, we consider ideal load balancing to be when clients are distributed proportionally to bandwidth, which is most reasonable under the assumption that clients produce similar amounts of traffic. When applying the defense, we use our recommended values of θ for each algorithm: $\theta = 1.25$ for Counter-RAPTOR, $\theta = 2$ for DeNASA, and $\theta = 5$ for LASTor. Under each algorithm, we compute each guard's expected *load factor*, which is the ratio of the guard's fraction of clients to the guard's fraction of bandwidth; for example, if a guard is used by 4% of Tor clients and contributes 2% of Tor's bandwidth, then the guard's load factor is $^{.04}/_{.02} = 2$. Under ideal load balancing, guards would have load factors close to 1.

Figure 10 shows the distribution of expected guard load under the location-aware algorithms. The CDFs are over *clients* and not client locations; e.g., the point at $(x = 10^0, y = 0.5)$ on the Counter-RAPTOR lines indicates that 50% of clients choose a guard with a load factor of at most 1 in Counter-RAPTOR.

Applying our defense to Counter-RAPTOR produces a nearly identical load distribution; therefore, 1.25-GP-security can be achieved without disturbing the load balance in Counter-RAPTOR. Applying our defense to DeNASA slightly improves network balance—the median load factor experienced by clients is reduced from 1.25 to 1.07, and the worst load factor is reduced from 3.00 to 1.70. Akhoondi et al. note that one of LASTor's deficiencies is its poor load balancing [1], which is reflected in our results. Applying our defense significantly improves the load balance of LASTor: the median load factor experienced by clients is reduced from 7.91 to 2.19, and the worst load factor is reduced from 70.1 to 8.70. Our analysis suggests that defending a path selection algorithm against the guard placement attack may result in desirable performance benefits, in addition to improving the security of the algorithm.

8 Related Work

Guard Placement Attacks. While previous work on location-aware path selection algorithms in Tor has mentioned guard placement attacks, it has not rigorously studied them. We showed that the defenses proposed for each algorithm we attacked are ineffective. Sun et al. [40] note that an adversary can run a relay that has a short AS-path length to the client to obtain a high resilience value, making it more likely for a Counter-RAPTOR client to choose the adversary's relay. To combat this, the resilience of each relay is somewhat randomized using Tille's algorithm [49]. Nithyanand et al. [30] mention that Astoria clients may be manipulated into connecting to malicious guards if there are few or no safe paths available. In these cases, the authors suggest having a minimum threshold of safe paths to choose from, but no further analysis is provided. Akhoondi et al. [1] also mention that an adversary may try to place relays close to the direct line between a LASTor client and destination, and they introduce a clustering algorithm as a defense. Perhaps most explicitly, Barton and Wright [5] coin the term "guard placement attack". To mitigate the attack in DeNASA, they limit the list of Suspect ASes to just two (AS3356 and AS1299), as doing so increases the number of guards a client can use.

In contrast to previous works, we formally define and analyze guard placement attacks using a metric that quantifies the attack success. Our work is the first to obtain quantitative estimates of the security of several location-aware Tor path selection algorithms against guard placement attacks. We also present a defense technique that modifies guard selection algorithms to make them provably secure.

Other Path Selection Algorithms. There have also been many alternate path selection algorithms proposed to improve the security and/or performance of Tor [2, 4, 36, 37, 39, 52]. Our security definitions apply to more than just location-aware algorithms in that adversaries can choose any strategy that maximizes their guard placement attack success, which includes choosing the number and bandwidths of their guards in addition to their locations. Because these other path selection algorithms do not address guard placement attacks, they may also be vulnerable and benefit from our defense.

Traffic Analysis Attacks. Onion routing systems such as Tor are vulnerable to adversaries that can observe traffic as it enters and exits the anonymity network [6, 21, 26–29, 31]. Both relay and network adversaries could monitor this traffic; a relay adversary may control both the entry and exit relay, while a network adversary might observe links on both sides of the circuit. The low-latency requirement of Tor makes correlating traffic patterns easy. This vulnerability has been known since the initial development of Tor [10], and has been demonstrated in a number of works [15, 26–28, 41]. A guard placement attack would allow an adversary to obtain the powerful guard position, making subsequent traffic correlation attacks on the client much easier.

Website fingerprinting is another deanonymization attack in which the adversary aims to associate a client's traffic patterns with those of specific websites [7, 18, 19, 22, 24, 32, 33, 35, 38, 53–56]. This is a powerful attack because the adversary only needs access to the client's traffic. Machine learning techniques can be used on features such as timing, volume, and direction to classify encrypted traffic as representative of a certain website. Like traffic correlation attacks, both relay and network adversaries could perform website fingerprinting attacks; a relay adversary would need to control the guard relay, and a network adversary would need to observe the link between the client and its guard. Guard placement attacks make website fingerprinting easier by making it easier for an adversary to induce a client to choose his guard.

Cost-based Adversary Models. Backes et al. [3] include an analysis of Tor's security against a "monetary" adversary, for which they produce a per-relay cost model based on hosting prices and bandwidth-cost statistics. In addition to considering a relay's bandwidth, their model also takes into account the specific hosting provider or country, but it does not consider the purchase of additional IP addresses. Jansen et al. [20] produce a cost model for a Tor adversary, but they do not require hosting providers to support Tor relays and do not consider additional IP addresses.

9 Conclusion

In this work, we formalize the guard placement attack, in which an adversary strategically places relays to compromise large fractions of Tor users at relatively low cost. We are the first to systematically study the guard placement attack and show that it is highly effective against location-aware path selection algorithms. We provide a definition of security against this attack and describe a general method that modifies a path-selection algorithm to satisfy this definition while minimizing the impact on the algorithm's original goals.

Our work motivates the following directions for future work: (1) *The design of θ-GP-secure path-selection algorithms without requiring the application of our defense.* If an algorithm is explicitly designed to achieve θ-GP-security, it may be able to achieve improved traffic-analysis resistance or network performance. (2) *An improved cost model.* Although we provide a concrete cost model through a study of current hosting providers, defining more sophisticated cost models can further refine our understanding of placement attacks and Tor security in general [36]. (3) *The generalization of guard placement attacks to relay placement attacks.* Many path-selection algorithms modify the way that clients choose middle and exit relays, making these positions potentially vulnerable to placement attacks. (4) *The development of techniques to quantify the security of Tor path selection.* Our work highlights the importance of designing Tor algorithms that consider multi-dimensional trade-offs among different types of adversaries. Techniques that help researchers balance the many considerations in path selection can greatly benefit the development of new Tor algorithms.

Acknowledgements

This work has been supported by the Office of Naval Research, the Army Research Office Young Investigator Prize (YIP), and National Science Foundation grants CNS-1704105, CNS-1553437, and CNS-1617286. We thank Florentin Rochet for valuable feedback.

References

[1] Masoud Akhoondi, Curtis Yu, and Harsha V. Madhyastha. LASTor: A Low-latency AS-aware Tor Client. *IEEE/ACM Transactions on Networking*, 22(6), 2014.

[2] Mashael AlSabah, Kevin Bauer, Tariq Elahi, and Ian Goldberg. The Path Less Travelled: Overcoming Tor's Bottlenecks with Traffic Splitting. In *Privacy Enhancing Technologies*, 2013.

[3] Michael Backes, Sebastian Meiser, and Marcin Slowik. Your choice mator (s). *Proceedings on Privacy Enhancing Technologies*, 2016(2), 2016.

[4] Armon Barton, Mohsen Imani, Jiang Ming, and Matthew Wright. Towards Predicting Efficient and Anonymous Tor Circuits. In *Proceedings of the 27th USENIX Conference on Security Symposium*, 2018.

[5] Armon Barton and Matthew Wright. DeNASA: Destination-Naive AS-Awareness in Anonymous Communications. In *Proceedings on Privacy Enhancing Technologies*, 2016.

[6] Kevin Bauer, Damon McCoy, Dirk Grunwald, Tadayoshi Kohno, and Douglas Sicker. Low-resource Routing Attacks Against Tor. In *Proceedings of the 2007 ACM Workshop on Privacy in Electronic Society*, WPES '07, 2007.

[7] Xiang Cai, Rishab Nithyanand, Tao Wang, Rob Johnson, and Ian Goldberg. A Systematic Approach to Developing and Evaluating Website Fingerprinting Defenses. In *ACM Conference on Computer and Communications Security (CCS)*, 2014.

[8] CAIDA Data. http://www.caida.org/data.

[9] Roger Dingledine and George Kadianakis. One fast guard for life (or 9 months). In *HotPETs*, 2014.

[10] Roger Dingledine, Nick Mathewson, and Paul Syverson. Tor: The Second-generation Onion Router. In *Proceedings of the 13th Conference on USENIX Security Symposium*, 2004.

[11] John R. Douceur. The Sybil Attack. In *Revised Papers from the First International Workshop on Peer-to-Peer Systems*, 2002.

[12] Kevin P. Dyer, Scott E. Coull, Thomas Ristenpart, and Thomas Shrimpton. Peek-a-Boo, I Still See You: Why Efficient Traffic Analysis Countermeasures Fail. In *Proceedings of the 2012 IEEE Symposium on Security and Privacy*, SP '12, 2012.

[13] Matthew Edman and Paul Syverson. AS-awareness in Tor Path Selection. In *Proceedings of the 16th ACM Conference on Computer and Communications Security*, CCS '09, 2009.

[14] Tariq Elahi, Kevin Bauer, Mashael AlSabah, Roger Dingledine, and Ian Goldberg. Changing of the Guards: A Framework for Understanding and Improving Entry Guard Selection in Tor. In *Proceedings of the 2012 ACM Workshop on Privacy in the Electronic Society*, WPES '12, 2012.

[15] Nick Feamster and Roger Dingledine. Location Diversity in Anonymity Networks. In *Proceedings of the 2004 ACM Workshop on Privacy in the Electronic Society*, WPES '04, 2004.

[16] Lixin Gao and Jennifer Rexford. Stable Internet Routing Without Global Coordination. *IEEE/AM Transactions on Networking*, 9(6), 2001.

[17] David M. Goldschlag, Michael G. Reed, and Paul F. Syverson. Hiding Routing Information. In *Proceedings of the First International Workshop on Information Hiding*, 1996.

[18] Jamie Hayes and George Danezis. k-fingerprinting: A Robust Scalable Website Fingerprinting Technique. In *25th USENIX Security Symposium (USENIX Security 16)*, 2016.

[19] Andrew Hintz. Fingerprinting Websites Using Traffic Analysis. In *Proceedings of the 2nd International Conference on Privacy Enhancing Technologies*, 2003.

[20] Rob Jansen, Tavish Vaidya, and Micah Sherr. Point Break: A Study of Bandwidth Denial-of-Service Attacks against Tor. In *28th USENIX Security Symposium*, 2019.

[21] Aaron Johnson, Chris Wacek, Rob Jansen, Micah Sherr, and Paul Syverson. Users Get Routed: Traffic Correlation on Tor by Realistic Adversaries. In *ACM Conference on Computer and Communications Security (CCS)*, CCS '13, 2013.

[22] Marc Juarez, Sadia Afroz, Gunes Acar, Claudia Diaz, and Rachel Greenstadt. A Critical Evaluation of Website Fingerprinting Attacks. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, CCS '14, 2014.

[23] Joshua Juen. Protecting anonymity in the presence of Autonomous System and Internet exchange level adversaries. Master's thesis, University of Illinois at Urbana-Champaign, 2012.

[24] Shuai Li, Huajun Guo, and Nicholas Hopper. Measuring Information Leakage in Website Fingerprinting Attacks and Defenses. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, 2018.

[25] Maxmind GeoLite2 Database. https://dev.maxmind.com/geoip/geoip2/geolite2/.

[26] Steven J. Murdoch and George Danezis. Low-Cost Traffic Analysis of Tor. In *Proceedings of the 2005 IEEE Symposium on Security and Privacy*, SP '05, 2005.

[27] Steven J. Murdoch and Piotr Zieliński. Sampled Traffic Analysis by Internet-Exchange-Level Adversaries. In *Privacy Enhancing Technologies Symposium (PETS)*, 2007.

[28] Milad Nasr, Alireza Bahramali, and Amir Houmansadr. DeepCorr: Strong Flow Correlation Attacks on Tor Using Deep Learning. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, 2018.

[29] Milad Nasr, Amir Houmansadr, and Arya Mazumdar. Compressive Traffic Analysis: A New Paradigm for Scalable Traffic Analysis. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, CCS '17, 2017.

[30] Rishab Nithyanand, Oleksii Starov, Adva Zair, Phillipa Gill, and Michael Schapira. Measuring and mitigating AS-level adversaries against Tor. In *Symposium on Network and Distributed System Security (NDSS)*, 2016.

[31] Lasse Overlier and Paul Syverson. Locating Hidden Servers. In *Proceedings of the 2006 IEEE Symposium on Security and Privacy*, 2006.
[32] Andriy Panchenko, Fabian Lanze, Jan Pennekamp, Thomas Engel, Andreas Zinnen, Martin Henze, and Klaus Wehrle. Website Fingerprinting at Internet Scale. In *Symposium on Network and Distributed System Security (NDSS)*, 2016.
[33] Andriy Panchenko, Lukas Niessen, Andreas Zinnen, and Thomas Engel. Website Fingerprinting in Onion Routing Based Anonymization Networks. In *Proceedings of the 10th Annual ACM Workshop on Privacy in the Electronic Society*, WPES '11, 2011.
[34] Mike Perry. TorFlow: Tor Network Analysis. In *HotPETs*, 2009.
[35] Vera Rimmer, Davy Preuveneers, Marc Juárez, Tom van Goethem, and Wouter Joosen. Automated Website Fingerprinting through Deep Learning. In *Symposium on Network and Distributed System Security (NDSS)*, 2018.
[36] Florentin Rochet and Olivier Pereira. Waterfilling: Balancing the Tor network with maximum diversity. *Proceedings on Privacy Enhancing Technologies*, 2017(2), 2017.
[37] Micah Sherr, Matt Blaze, and Boon Thau Loo. Scalable Link-Based Relay Selection for Anonymous Routing. In *Proceedings of the 9th International Symposium on Privacy Enhancing Technologies*, 2009.
[38] Payap Sirinam, Mohsen Imani, Marc Juarez, and Matthew Wright. Deep Fingerprinting: Undermining Website Fingerprinting Defenses with Deep Learning. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, 2018.
[39] Robin Snader and Nikita Borisov. A Tune-up for Tor: Improving Security and Performance in the Tor Network. In *Proceedings of 16th Annual Network and Distributed System Security Symposium*, 2008.
[40] Yixin Sun, Anne Edmundson, Nick Feamster, Mung Chiang, and Prateek Mittal. Counter-RAPTOR: Safeguarding Tor Against Active Routing Attacks. In *IEEE Symposium on Security and Privacy*, 2017.
[41] Yixin Sun, Anne Edmundson, Laurent Vanbever, Oscar Li, Jennifer Rexford, Mung Chiang, and Prateek Mittal. RAPTOR: Routing Attacks on Privacy in Tor. In *Proceedings of the 24th USENIX Conference on Security Symposium*, SEC'15, 2015.
[42] Paul Syverson, Gene Tsudik, Michael Reed, and Carl Landwehr. Towards an Analysis of Onion Routing Security. In *International Workshop on Designing Privacy Enhancing Technologies: Design Issues in Anonymity and Unobservability*, 2001.
[43] Team-Cymru. http://www.team-cymru.com.
[44] CollecTor - Tor Project. https://metrics.torproject.org/collector.html.
[45] Tor Directory Protocol. https://gitweb.torproject.org/torspec.git/tree/dir-spec.txt.
[46] Tor Guard Specification. https://gitweb.torproject.org/torspec.git/tree/guard-spec.txt.
[47] Tor Metrics Portal. https://metrics.torproject.org/.
[48] Torflow Protocol Specification. https://gitweb.torproject.org/torflow.git/tree/NetworkScanners/BwAuthority/README.spec.txt.
[49] Yves Tillé. An elimination procedure of unequal probability sampling without replacement. *Biometrika*, 83, 1996.
[50] Thaddeus Vincenty. Direct and Inverse Solutions of Geodesics on the Ellipsoid with Application of Nested Equations. In *Survey Review*, 1975.
[51] Ryan Wails, Yixin Sun, Aaron Johnson, Mung Chiang, and Prateek Mittal. Tempest: Temporal Dynamics in Anonymity Systems. In *Privacy Enhancing Technologies Symposium (PETS)*, 2018.
[52] Tao Wang, Kevin Bauer, Clara Forero, and Ian Goldberg. Congestion-Aware Path Selection for Tor. In *International Conference on Financial Cryptography and Data Security*, 2012.
[53] Tao Wang, Xiang Cai, Rishab Nithyanand, Rob Johnson, and Ian Goldberg. Effective Attacks and Provable Defenses for Website Fingerprinting. In *USENIX Security Symposium*, 2014.
[54] Tao Wang and Ian Goldberg. Improved Website Fingerprinting on Tor. In *Proceedings of the 12th ACM Workshop on Workshop on Privacy in the Electronic Society*, WPES '13, 2013.
[55] Tao Wang and Ian Goldberg. On realistically attacking Tor with website fingerprinting. In *Privacy Enhancing Technologies Symposium (PETS)*, 2016.
[56] Tao Wang and Ian Goldberg. Walkie-Talkie: An Efficient Defense Against Passive Website Fingerprinting Attacks. In *26th USENIX Security Symposium (USENIX Security 17)*, 2017.
[57] Philipp Winter, Roya Ensafi, Karsten Loesing, and Nick Feamster. Identifying and Characterizing Sybils in the Tor Network. In *25th USENIX Security Symposium (USENIX Security 16)*, 2016.
[58] Philipp Winter and Stefan Lindskog. Spoiled Onions: Exposing Malicious Tor Exit Relays. In *Privacy Enhancing Technologies Symposium (PETS)*, 2014.
[59] Matthew Wright, Micah Adler, Brian N. Levine, and Clay Shields. Defending Anonymous Communications Against Passive Logging Attacks. In *Proceedings of the 2003 IEEE Symposium on Security and Privacy*, SP '03, 2003.

A Interpreting Consensus Weights

Tor has a non-trivial method for computing consensus weights [34, 45]. While these values are ostensibly in units of KByte/s, they differ substantially from the actual bandwidths that relays report in their descriptors. We observe that the consensus weight is correlated to these self-advertised relay bandwidths, however. Therefore, we can use a linear regression to convert the consensus weights to relay bandwidths. The resulting linear regression ($y = 0.7638x + 2908.2712$) expresses relay bandwidth in units of KBytes/s and has a coefficient of determination of $r^2 = 0.86$. The conversion from weights to actual bandwidth can be found in Table 4.

Consensus weight	Weight fraction (%)	BW (Mbit/s)
2,000	0.0108	35.5
3,000	0.0162	41.6
7,500	0.0404	69.1
10,000	0.0539	84.4
30,000	0.162	206.6
40,000	0.216	267.7
75,000	0.404	481.5
150,000	0.809	939.8

Table 4. Conversion from consensus weights to actual bandwidth values.

B Tille's Algorithm

Counter-RAPTOR sought to provide a preliminary defense against guard placement attacks by adjusting the resilience of guards using Tille's algorithm [49]. Since high resilience guards have a higher probability of selection, Counter-RAPTOR instead simulates the process of first choosing a resilience-weighted sampling of size $g \cdot N$ and then choosing uniformly within that sample, where $g \in [0, 1]$ indicates the fraction of sampled guards and N is the total number of guards. Adding the simulated sampling step makes the selection distribution more uniform. To simulate this process, each guard's resilience is adjusted from $r(i)$ to $r'(i)$ using Tille's algorithm and then a guard is selected using Equation 4. Counter-RAPTOR uses a default value of $g = 0.1$ [40]. The steps of Tille's algorithm applied to the guards are as follows:

1. For each guard i, $r'(i) = \frac{k \cdot r(i)}{\sum_{j \in \mathcal{G}} r(j)}$, where k is initially equal to the sample size $(g \cdot N)$ and set $\mathcal{G}$ initially includes all available guards.
2. For each guard i, if $r'(i) > 1$, set $r'(i) = 1$, set $k = k - 1$, and exclude relay i from the set $\mathcal{G}$.
3. Repeat the above process until each $r'(i)$ is in $[0, 1]$.
4. For each relay i, $r'(i) = \frac{r'(i)}{g \cdot N}$

C Theorems and Proofs

Theorem 1. *Path selection algorithm A is θ-GP-secure if and only if $\rho(A) \le \theta$.*

Proof. Assume that $\rho(A) \le \theta$. Consider any adversary strategy $s \in S$ and client location $c \in \mathcal{C}$. Observe that $f_A(c, g) \le \mathsf{relCost}(g) \cdot \theta$ by the definition of ρ. Moreover, $p_A(c, s) = \sum_{g \in s} f_A(c, g)$, and so $p_A(c, s) \le \theta \cdot \sum_{g \in s} \mathsf{relCost}(g)$. Therefore, $p_A(c, s) / \sum_{g \in s} \mathsf{relCost}(g) \le \theta$. Because s and c were arbitrary, $\overline{\sigma}(A) \le \theta$.

To prove the other direction of the equivalence, assume that A is θ-GP-secure. For any guard g, let the adversary strategy s consists of just that guard. Then $f_A(c, g) = p_A(c, s)$. By the definition of θ-GP-secure, $p_A(c, s) \le \theta \cdot \mathsf{relCost}(g)$. Therefore, $f_A(c, g)/\mathsf{relCost}(g) \le \theta$. Because g and c were arbitrary, $\rho(A) \le \theta$. □

Theorem 2. *D halts.*

Proof. Every loop in Algorithm 1 has a constant number of iterations except for the loop in Lines 6–19. This loop terminates if the condition in Line 10 applies to no guard in B. B contains all guards at the beginning of the loop, and if the loop doesn't terminate at least one guard is removed. Thus, the loop iterates at most $|G|$ times. □

Theorem 3. *Let A be any guard selection algorithm and $\theta \ge 1$ be the security parameter. Then using the guard selection distribution $f'_A = D(A, \theta)$ is θ-GP-secure.*

Proof. At the beginning of Algorithm 1, each guard is in the set B. The assigned probability of g (i.e. $f'_A(g)$) only changes while g is in B. $f'_A(g)$ only increases because its assignments occur in Lines 12 and 15, where the former assignment is ensured to be an increase by the fact that $\theta \ge 0$ and the latter is ensured by the fact that always $p \ge 0$. The condition in Line 10 guarantees that if increasing $f'_A(g)$ would violate the θ bound, then instead $f'_A(g)$ is set to meet the θ bound, and g is removed from B. Therefore, if D terminates, f'_A is such that every guard satisfies the θ bound. Moreover, every iteration of the loop in Lines 6–19 must occur with a non-empty B because $\theta \ge 1$ guarantees that some guard remains strictly below the θ bound while there is unassigned probability. This fact implies that some positive amount of the unassigned probability p gets assigned during each iteration of that loop, which also ensures that Line 11 executes and thus any probability unassigned during the iteration is assigned to x, causing another iteration if necessary. By Theorem 2, D does terminate, and so the output f'_A must be a probability distribution satisfying the θ bound. Thus, by Theorem 1, it is θ-GP-secure to use $f'_A = D(A, \theta)$ as the guard selection distribution. □

Autonomous System	Consensus Weight (%)	Relays	Hosting Prices
OVH SAS (AS16276)	14.06	586	ovh.com
Hetzner Online (AS24940)	12.57	408	hetzner.com
Online SAS (AS12876)	10.84	332	online.net scaleway.com
JP McQuistan (AS200052)	3.95	48	N/A
Next Layer (AS1764)	1.94	17	nextlayer.at
netcup (AS197540)	1.86	71	netcup.eu
Quintex (AS62744)	1.78	23	N/A
iomart (AS20860)	1.66	20	N/A
myLoc (AS24961)	1.59	38	myloc.de
DigitalOcean (AS14061)	1.54	228	digitalocean.com
Total	51.78	1,771	-

Table 5. Top 10 ASes in Tor by consensus weight of hosted relays as of 2019-2-26. When hosting prices were available online, sites with pricing information are indicated.

D Cost Model Details

The top 10 ASes in the Tor network are listed in Table 5. These ASes contained relays with the largest total consensus weight. Of the 10 top ASes, 7 provide commercial hosting and make their pricing available online. For these ASes, the sites with that pricing information are indicated in the table.

The exact cost model is given in Table 6. Data was obtained from hosting provider sites 2019-2-26–27. The number of relays indicates the number of relays the product's bandwidth and cost is split among to achieve the given per-relay bandwidth and cost. Costs are given in USD. Prices given in Euros are converted to USD at a rate of 1.14 USD/Euro. The cost for a given bandwidth B is the cost listed in the table for the smallest bandwidth not smaller than B. Note that neighboring bandwidths may appear to have identical costs due to rounding.

Provider	Product	Number of relays	Bandwidth (Mbps)	Cost ($/month)
Online SAS	Dedicated	1	1,000	11.4
Online SAS	Dedicated	2	500	5.7
Online SAS	Dedicated	3	333.33	4.56
Online SAS	Dedicated	4	250	3.42
Online SAS	Dedicated	5	200	3.19
Online SAS	Dedicated	6	166.67	2.66
Online SAS	Dedicated	7	142.86	2.61
Online SAS	Dedicated	8	125	2.28
Online SAS	Dedicated	9	111.11	2.28
Online SAS	Dedicated	10	100	2.05
Online SAS	Dedicated	12	83.33	1.9
Online SAS	Dedicated	14	71.43	1.79
Online SAS	Dedicated	16	62.5	1.71
Online SAS	Dedicated	18	55.56	1.65
Online SAS	Cloud 1-XS	2	50	1.14
Online SAS	Cloud 1-S	6	33.33	1.14
Online SAS	Cloud 1-XS	4	25	0.85
Online SAS	Cloud 1-XS	6	16.67	0.76
Online SAS	Cloud 1-XS	8	12.5	0.71
Online SAS	Cloud 1-XS	10	10	0.68
Online SAS	Cloud 1-XS	12	8.33	0.66
Online SAS	Cloud 1-XS	14	7.14	0.65
Online SAS	Cloud 1-XS	16	6.25	0.64
Online SAS	Cloud 1-XS	18	5.56	0.63
Online SAS	Cloud 1-XS	20	5	0.63
Online SAS	Cloud 1-XS	22	4.55	0.62
Online SAS	Cloud 1-XS	24	4.17	0.62
Online SAS	Cloud 1-XS	26	3.85	0.61
Online SAS	Cloud 1-XS	28	3.57	0.61
Online SAS	Cloud 1-XS	30	3.33	0.61
Online SAS	Cloud 1-XS	32	3.12	0.61

Table 6. Cost model derived from hosting prices of top Tor ASes. Product in each case is from the Start line.