Measuring and mitigating AS-level adversaries against Tor

Rishab Nithyanand
Stony Brook University

Oleksii Starov
Stony Brook University

Adva Zair
Hebrew University of Jerusalem

Phillipa Gill
Stony Brook University

Michael Schapira
Hebrew University of Jerusalem

Abstract

The popularity of Tor as an anonymity system has made it a popular target for a variety of attacks including blocking, denial of service, and timing attacks. In this paper, we focus on timing attacks which are no longer solely in the realm of academic research with recent revelations about the NSA and GCHQ actively working to implement them in practice. We specifically focus on recently exposed timing attacks that leverage asymmetric routing and information gained on reverse network paths (*e.g.,* via TCP ACK numbers) to deanonymize Tor users.

First, we present an empirical study which leverages scalable algorithmic simulations of routing policies on an up-to-date map of the Internet's topology, including complex AS relationships and sibling ASes. Our approach allows us to gain a high fidelity snapshot of the threat of timing correlation attacks in the wild. In our experiments we find that 58% of all circuits created by Tor are vulnerable to attacks by timing correlation and colluding sibling ASes. In addition, we find that in some regions (notably, China) there exist a number of cases where it is not possible for Tor to construct a circuit that is safe from these correlation attacks.

To mitigate the threat of such attacks, we build Astoria– an AS-aware Tor client. Astoria uses leverages recent developments in network measurement to perform path-prediction and intelligent relay selection. Astoria not only reduces the number of vulnerable circuits to 5.1%, but also considers how circuits should be created when there are no safe possibilities. Astoria also performs load balancing across the Tor network, so as to not overload low capacity relays. In addition, Astoria provides reasonable performance even in its most secure configuration.

1. INTRODUCTION

Tor is a popular anonymity system for users who wish to access the Internet anonymously or circumvent censorship [1]. The increasing popularity of Tor has recently made it a large target for blocking and denial of service [2–4] and timing attacks to deanonymize users [5–9]. Timing attacks, which correlate traffic entering the Tor network with traffic exiting it, are no longer solely in the realm of academic research with recent revelations about the NSA and GCHQ actively working to implement them in practice, in collusion with Internet Service Providers [10–12].

Timing attacks have been shown to be feasible and practical for network-level attackers. Specifically, a timing attack may be implemented by any autonomous system (AS) that lies on both, the path from the Tor client to the entry relay and the path from the exit relay to the destination. Previous studies have demonstrated the potential for this type of attack [9,13,14] and have proposed relay selection strategies to avoid common ASes (potential attackers) that may perform them [15]. However, recent work [16] has shown that these strategies perform poorly in practice.

The threat of network-level adversaries has been exacerbated by a recent study which highlights that the set of potential ASes that may perform timing analysis is potentially much larger due to asymmetric routing, routing instabilities, and intentional manipulations of the Internet's routing system [17,18]. These attacks significantly raise the bar for relay-selection systems. Specifically, they require the relay-selection system be able to accurately measure or predict network paths in both the forward and reverse direction. Measuring the reverse path between two Internet hosts is non-trivial, especially when the client does not have control over the destination, as is commonly the case for popular Web services. While solutions for measuring reverse paths have been proposed [19], they are still not widely deployed or available.

In this paper, we quantify the threat posed by these new attacks, and develop a relay selection method to minimize their impact. We leverage up-to-date maps of the Internet's topology [20] combined with algorithmic simulations [21] to predict which ASes are in a position to perform timing analysis on forward or reverse paths. We then augment our analysis with techniques to identify ASes owned by a single organization (sibling ASes) in order to gain a clearer picture of which ASes are likely to collude with each other. This provides a more accurate impression of network-level threats than previous work. Through these techniques, we make the following key observations:

- 58% of circuits constructed by Tor are vulnerable to network-level attackers.
- 43% of all sites in the local Alexa Top 500 of Brazil, China, Germany, Spain, France, England, Iran, Italy, Russia, and the United States had main content that was not reached via a *safe* path – i.e., a path that was free from network-level attackers.
- Connections from China were found to be most vulnerable to network-level attackers with 85.7% of all Tor circuits and 78% of all main content requests to sites in the local Alexa Top 500 being vulnerable to colluding network-level attackers.
- Reducing the number of entry guards results in an increase in vulnerability of Tor circuits in several countries. The most drastic loss of security was seen in Spain. In

particular, Tor with 3 guards (default) had 34.8% vulnerable circuits, Tor with 2 guards had 59.8% vulnerable circuits, and Tor with a single guard had 75.7% vulnerable circuits.

We propose, construct, and evaluate Astoria–an AS-aware Tor client that includes both, security and relay bandwidth considerations when creating Tor circuits. Astoria is the first AS-aware Tor client to consider the recently proposed asymmetric correlation attacks [17,18]. When there are safe alternatives, Astoria actively avoids using circuits on which asymmetric correlation attacks might be launched. It also leverages, methods to identify sibling ASes [22] when determining whether or not a given circuit is safe. In the absence of a safe path, Astoria uses a linear program to minimize the amount of threat posed by any adversary. Finally, Astoria considers the bandwidth capabilities of relays while making AS-aware relay selection decisions. Astoria aims to be a good network citizen and allows users to adjust how their circuits distribute load across Tor relays. Even in it's most secure configuration, Astoria will not overload slow relays.

Paper outline. In Section 2 we briefly overview how the current Tor client performs relay selection and circuit construction, describe the current state of research in relay selection for Tor, and introduce our adversary model. In Section 3 we describe the components of our measurement toolkit used for detecting network-level attackers on Tor circuits. We then present some interesting results regarding the prevalence of attackers on Tor constructed circuits and the general potential for attack by adversaries described in our model. In Section 4, we present the details of our AS-aware client – Astoria. A performance and security evaluation of Astoria is performed in Section 5. In Section 6 we discuss methods for improving Astoria performance and discuss its usability. We make our conclusions in Section 7.

2. BACKGROUND AND MOTIVATION

We now provide background on Tor relay selection, related work in this area, and introduce our adversary model.

2.1 Tor relay selection

The Tor anonymity network consists of approximately 6,000 relays (Tor routers). Most requests made through a Tor client are sent to their destination via a three-hop path known as a circuit. Each circuit consists of an entry, middle, and exit relay. The entry-relay communicates directly with the client making the request, and the exit-relay communicates with the destination of the request. The fundamental idea is that no single relay in the circuit learns the source and the destination of the request.

In its early days, Tor selected relays for each section of a circuit uniformly at random from the set of available circuits. This was changed in order to improve performance (by preferring to route through higher bandwidth relays [23]) and security [24]. In today's Tor network, based on certain performance characteristics such as reliability, bandwidth served, and up-time, relays may earn certain flags that make them a preferential choice for various roles during circuit construction.

One such flag is the guard flag. New relays joining the Tor network are monitored for stability and performance via remote measurements for a period of up to eight days [25]. At this point, relays that have shown to be stable and reliable are assigned a guard flag. Relays with a guard flag earn the ability to serve as the entry-relay to the Tor network. By default the Tor client selects three guards to be used as entry-relays for all circuits for a prolonged period of time. The main ideas behind the selection of a fixed set of entry-relays are (1) to reduce the possibility that a client will select an entry- and exit-relay operated by the same entity after prolonged use, (2) prevent attacker owned entry-relays from denying service to clients that are not also using an exit-relay owned by the attacker, and (3) increase the cost to an attacker that wishes to be chosen as an entry-relay, by requiring them to earn the guard flag [25].

In addition to picking routers that are more stable and reliable, for other locations on a circuit, the Tor client also requires that (1) no two routers on a circuit share the same /16 subnet and (2) no routers in the same family (as advertised by the router) may be chosen on the same circuit. [23].

2.2 Related work

The threat of correlation attacks by AS-level adversaries on the Tor network was first identified and empirically evaluated by Feamster and Dingledine [14] in 2004, when the Tor network had only 33 relays and significantly different relay selection algorithms. The study revealed that 10-30% of all circuits constructed by Tor had a common AS that could observe both ends of the circuit. Shortly after, by constructing efficient traffic correlation attacks while considering network-level adversaries, Murdoch and Danezis [5] and Murdoch and Zieliński [7] demonstrated that the threat from AS-level attackers was one of practical concern. In 2009, Edman and Syverson [13] found that the threat of AS-level adversaries had not reduced since [14], in spite of revised relay selection strategies and substantially larger number of relays in the network.

Johnson *et al.* [9] performed an empirical evaluation of the effect of adversary bandwidth investment strategies, Tor client location, and Tor client use (*e.g.,* for IRC, browsing, BitTorrent, *etc.*). They found that a network adversary could effectively de-anonymize most Tor users within six months with very low bandwidth costs.

Most recently, Vanbever *et al.* [17] and Sun *et al.* [18], presented RAPTOR, an AS-level attack integrating BGP interception with a simple correlation attack that takes advantage of the asymmetric nature of Internet routing, to exactly de-anonymize Tor users with up to 90% accuracy in just 300 seconds. RAPTOR emphasizes the need for Tor relay selection strategies to consider ASes that lie both, on the forward and reverse paths between the (client, entry) and (exit, destination).

Perhaps most closely related to our work, Akhoondi *et al.* [15], constructed LASTor, a Tor client which explicitly considered AS-level attackers and relay locations while constructing Tor circuits. While LASTor appeared to successfully reduce path latencies and the probability of common ASes at either end of the Tor circuits, it neglected the capacity of relays selected by the system. Relay capacity is an important variable to consider to ensure that custom relay selection schemes do not overload a small set of relays – therefore reducing the performance of the entire network. Their evaluation, based on only HTTP HEAD requests (as opposed to webpage loads), did not stress the system sufficiently to reveal the issues associated with capacity-agnostic relay selection. Further, LASTor does not consider an adversary that may (1) collude with other ASes, and/or (2)

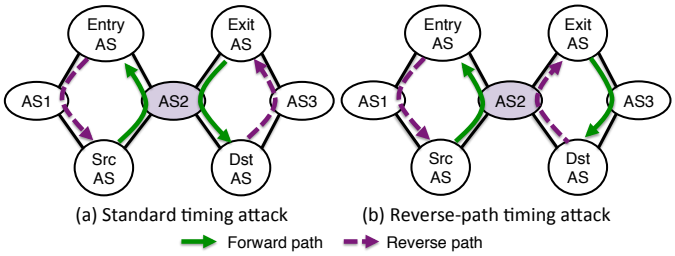


Figure 1: Standard and reverse-path timing attacks. In the standard timing attack, AS2 must observe the direction of the connection that data is flowing on (forward path). In the reverse-path timing attack AS2 can infer the data flow using ACK numbers on the reverse path.

only need to be on one of the asymmetric path segments between source and entry-relay; and exit-relay and destination (*e.g.,* RAPTOR).

2.3 Adversary model

In the standard view of timing attacks, an AS needs to lie on the forward path[1] between the source and destination (i.e., on the solid green colored path segments in Figure 1 (a)). With this view point the adversary (AS 2) can view the packet sizes as transmitted from the source to destination, going-into and coming-out-of the Tor network and directly perform a timing attack.

However, recent work by Vanbever *et al.* [17] and Sun *et al.* [18] highlights the fact that an adversary on the reverse path may also learn packet size and timing information via the TCP Acknowledgement (ACK) field. Figure 1(b) illustrates this case. AS 2 can directly observe packet timings between the source and entry-relay AS (Entry AS), but can only observe ACKs from the destination back to the exit-relay AS (Exit AS).

In this view, an adversary has the potential to launch a timing attack on a Tor circuit as long as the following criteria are satisfied:

Let $p_{src \to entry} = \{AS_1, AS_2, \ldots, AS_n\}$ be the set of ASes on the path from the source (Tor client) to the selected entry-relay (this set includes the entry-relay AS), $p_{entry \to src} = \{AS'_1, AS'_2, \ldots, AS'_m\}$ be the set of ASes on the path from the entry-relay back to the source, and $p_{entry \leftrightarrow src} = p_{entry \to src} \cup p_{src \to entry}$.

We similarly define paths to and from the exit-relay and destination (*e.g.,* a popular content provider, or other Web service) as $p_{exit \to dst}$, $p_{dst \to exit}$, and $p_{exit \leftrightarrow dst}$.

We say that a Tor circuit is subject to attack if there exists an AS A_i such that:

$$A_i \in \{p_{src \leftrightarrow entry} \cap p_{exit \leftrightarrow dst}\}$$

Similar to prior work on relay selection, we assume that our adversary is an autonomous system (AS), or an entity working with the cooperation of ASes (*e.g.,* governments). However, while all previous work only considers the standard view of network attacks, we also consider attackers that may lie on the reverse-path, as described above. In addition, we also include the possibility that some sets of ASes may collude with each other to de-anonymize Tor users. Specifically, we consider that an AS may collude with sibling ASes [22] – i.e., other ASes owned by the same organization. Finally, we consider that our adversary may run surveillance activities over a long period of time. As part of our relay selection algorithms (Section 4), we consider a probabilistic relay selection strategy that minimizes the amount of traffic that is observable by any single attacker.

[1]Here we use 'forward path' to refer to the direction of data flow in the TCP connection

3. MEASURING ADVERSARY PRESENCE

In this section, we investigate the prevalence of the adversary we describe above. We detail how prediction of AS paths between a source and a destination is performed and how sets of potential attacking ASes are generated. Then, we provide a description of our measurement setup and our findings.

3.1 Predicting attacking ASes

Adversaries that can exploit asymmetric routing present a challenge to measuring their prevalence. While forward paths are simple to measure (via forward traceroutes), the addition of potential attackers on the reverse-path between a source and destination implies the need for identifying potential attackers on the reverse-paths between the client and entry-relay (and the exit-relay and destination) path. This poses a serious problem, since obtaining information about reverse-paths is far less straightforward. While Reverse Traceroute [19] would be a useful tool for these measurements, it is not widely deployed.

Additionally, since our measurement toolkit was assembled with the goal of integration with our Tor client – Astoria (Section 4), using external measurement and control-plane mapping tools (e.g., iPlane [26]) was not an option. This is because such tools require knowledge of the clients intended destination – an undesirable option for an anonymity tool such as Tor. Thus, any measurement or path prediction needs to be performed on the Tor client without leaking any information to attackers or third party tools and service providers.

To address the aforementioned challenges we employ an efficient path prediction approach which leverages up-to-date maps of the AS-level Internet topology [20], and algorithmic simulations that take into account a common model of routing policies [21].

AS-level topology. We perform path prediction using an empirically-derived AS-level Internet topology. In this abstraction, the Internet is represented as a graph with ASes as nodes and connections between them represented as edges. Connections between ASes are negotiated as business arrangements and are often modeled as two main types of relationship: *customer-provider* where the customer pays the provider for data sent and received; and *settlement-free peering* or *peer-peer* where two ASes agree to transit traffic at no cost [27].

However, in practice AS relationships may violate this simple taxonomy *e.g.,* ASes that agree to provide transit for a subset of prefixes (*partial transit*) or ASes that have different economic arrangements in different geographic regions (*hybrid relationships*) [20]. It can also be the case that two ASes are controlled by the same organization *e.g.,* because of corporate mergers such as Level 3 (AS3356) and Global Crossing (AS3549) or organizations that leverage different AS numbers in different regions such as Verizon (AS701, 702, 703). The AS-level topology we leverage takes partial

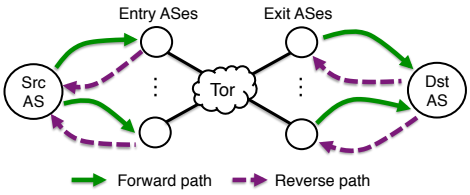


Figure 2: Illustration of the AS paths that the client needs to predict, note that these paths must be predicted for each potential entry and exit relay in both the forward and reverse direction.

transit and hybrid relationships into account, and we use techniques discussed by Anwar *et al.* [22] for detecting sibling ASes. This is done to identify ASes that are likely to collude with each other.

Routing policies. Routing on the AS-graph deviates from simple shortest path routing because ASes route their traffic based economic considerations. We use a standard model of routing policies proposed by Gao and Rexford [27]. The path selection process can be broken down into the following ordered steps:

- *Local Preference (LP).* Paths are ranked based on their next hop: customer is chosen over peer which is chosen over provider.
- *Shortest Paths (SP).* Among the paths with the highest local preference, prefer the shortest ones.
- *Tie Break (TB).* If there are multiple such paths, node a breaks ties: if b is the next hop on the path, choose the path where hash, $H(a, b)$ is the lowest.[2]

This standard model of local preference [27] captures the idea that an AS has incentives to prefer routing through a customer (that pays it) over a peer (no money is exchanged) over a provider (that it must pay).

In addition to selecting paths, ASes must determine which paths they will announce to other ASes based on export policies. The standard model of export policies captures the idea that an AS will only load its network with transit traffic if its customer pays it to do so [27]:

- *Export Policy (EP).* AS b announces a path via AS c to AS a iff at least one of a and c are its customers.

Computing paths following these policies using simulation platforms (*e.g.,* CBGP [28]) can be computationally expensive which limits the scale of analysis. Thus, we employ an algorithmic approach [21] that allows us to compute all paths to a given destination in $\mathcal{O}(|V| + |E|)$ where $|V|$ is the number of ASes and $|E|$ is the number of edges. Following [21], we call this computation of all paths to a given destination "computing the routing tree" for the destination.

Predicting paths. We use the routing policies and algorithmic simulations [21] as described above to compute routes between pairs of ASes using the AS-level topology published by CAIDA [20]. While it is difficult to predict the specific AS-level path between a source and destination AS, recent work shows that 65-85% of measured paths are in the set of paths which satisfy *LP* and *SP* above [22]. Thus, we modify the algorithmic simulator to return all paths satisfying *LP* and *SP* simultaneously, instead of using *TB* to produce a unique path. We consider the set of ASes in the set of paths satisfying *LP* and *SP* between a and b to be the set $p_{a \to b}$.

For standard measurement purposes, the toolkit simply takes a source and destination address and returns the set of ASes on the forward and reverse-path between the two.

However, in the context of integration with our Tor client, it must predict paths to and from each of the entry-relays for the client's AS, and paths from all exit-relays toward the destination AS (Figure 2). This results in $|En| + |Ex| + 2$ routing-tree computations where $|En|$ and $|Ex|$ are the number of entry and exit relays, respectively. In order to mitigate the risk of correlation attacks, by default, Tor restricts the number of entry-relays available to each Tor client to three (called guards [29]), and there are typically of the order of 1,000 exit-relays available to a client during circuit construction. The effect of reducing the number of entry-relays available to each client in the context of our adversary model is discussed in Section 3.2.

Fortunately, since the source AS and entry-relay ASes are relatively stable, these paths can be precomputed for later use by the client. (We observe the benefit of this in Section 5.) However, performing relay selection on a per-destination basis means that pre-building circuits, as is done by the current implementation of Tor, is no longer feasible. In Section 4, we show how integrating our path measurement toolkit impacts the performance of our Tor client.

Identifying attacked circuits. Once our path prediction toolkit returns the set of ASes that occupy each (forward and reverse-) path from the Tor client to a given entry-relay and from an available exit-relay to the destination, potential circuits are labeled as under attack *iff* there are common or sibling ASes on the (client, entry-relay) and (exit-relay, destination) path. This is in line with our adversary model described in Section 2.

3.2 Results

To understand the threat posed by the adversary described in Section 2, we performed several experiments. In particular, our goal was to understand the threat faced by the Tor client under various configurations, and in different locations.

Experimental setup. In our experiments, we consider the fact that Tor users in different countries face different levels of threats from local ASes. To this end, we performed page loads using several configurations of the Tor client from 10 different countries: Brazil (BR), China (CN), Germany (DE), Spain (ES), France (FR), England (GB), Iran (IR), Italy (IT), Russia (RU), and the United States (US). This list was obtained by considering various intersections of the number of Tor users in each country [30] and the Freedom House rankings for Internet freedom [31].

A VPN service was used to perform page loads of the Alexa Top 500 local sites [32] from within each of the aforementioned countries. Page loads were performed using the Tor client in four configurations: vanilla (default: three entry guard restricted Tor), one entry guard restricted Tor, two entry guard restricted Tor, and a modified Tor client performing entry- and exit-relay selection uniformly at random.

For each configuration, logs were maintained to track: (1) the list of available entry- and exit-relays during circuit construction, (2) the actual chosen entry and exit-relay for each

[2]In practice, this is done using the distance between routers and router IDs. Since we do not incorporate this information in our model we use a randomized tie break which prevents certain ASes from "always winning".

	Siblings	Vanilla Tor	Tor (2Guards)	Tor (1Guard)	Uniform Tor
Main	Yes	43.1%	35.9%	37.0%	36.8%
Main	No	42.4%	34.8%	36.2%	34.2%
All	Yes	59.0%	52.4%	54.4%	65.0%
All	No	58.6%	51.7%	53.7%	62.9%

Table 1: Percentage of circuits carrying main requests for the Alexa Top 500 websites (of 10 countries) that are under threat and percentage of all circuits under threat for various relay selection strategies.

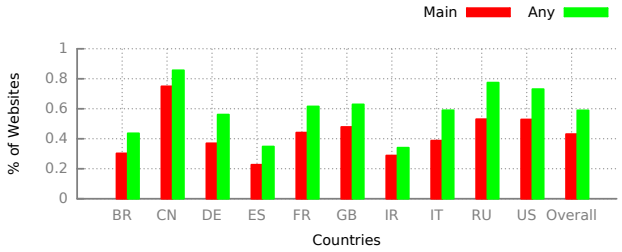


Figure 3: The fraction of circuits used by Vanilla Tor in each country (for main and any requests) that are vulnerable to asymmetric colluding attackers.

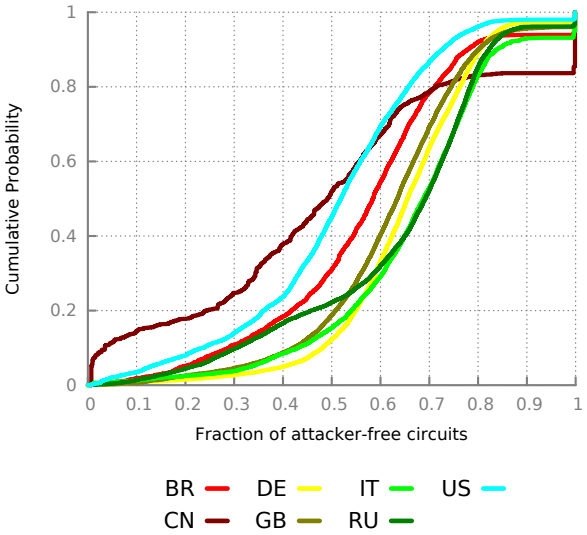


Figure 4: Distribution of of the fraction of colluding and asymmetric correlation attacker-free circuits for 100 source ASes connecting to local Alexa Top 100 sites in 7 countries of interest.

circuit constructed by the client, and (3) the list of requests made for each site and the circuit used by the Tor client to serve the request.

In addition to the source AS (AS of the VPN) and the destination AS (AS of the web content), ASes of the entry- and exit-relays (available and used) were extracted from our logs and input to our measurement toolkit to identify the set of attackers actually present on the constructed Tor circuit ("actual attackers") and attackers that could have been present had a particular entry- and exit-relay combination been selected ("potential attackers").

Since VPN vantage points only represent a single AS in a given country, we augment our results with simulations of network paths from a random sample of 100 ASes in a selected subset of the countries. Using data regarding the available entry- and exit-relays for a client, we computed the number of potential attackers in each AS.

The results of our VPN- and simulation-based experiments are presented below.

Live Tor network results. A summary of the results of our experiments on the live Tor network is illustrated in Table 1. As can be seen, the threat from de-anonymization is alarmingly high. When using the default configuration of Tor, 43% of all circuits carrying the main request (GET) for each site and 59% of all circuits are under threat from colluding network-level attackers.

Figure 3 breaks down these numbers by country, showing the fraction of circuits built for the main page and any request for the top sites that are vulnerable to the attacker. We find that the threat of asymmetric colluding attackers is not uniformly spread. Clients using Tor from three countries: China (CN), Russia (RU), and the United States (US) are found to be most vulnerable, while rather surprisingly, clients connecting from Brazil (BR), Spain (ES), and Iran (IR) are found to be the least vulnerable to our attacker.

One must remember, however, that the techniques for finding sibling (colluding) ASes does not capture the notion of governments and dictatorial regimes who may enforce traffic monitoring across networks managed by distinct organizations.

Simulation results. Since the results of our experiments on the live Tor network were highly dependent on the location of the VPN, simulations were required to understand the distribution of threat in other locations within each country. To this end, seven countries were selected for further investigation: Brazil (BR), China (CN), Germany (DE), England (GB), Italy (IT), Russia (RU), and the United States (US). In each country, 100 ASes were randomly selected as client locations and the Alexa local Top 100 sites were used as destinations. The simulation toolkit generated a list of entry- and exit-relays available to each client for performing the page load (using Tor client generated data of currently available entry- and exit-relays on the network).

Each generated entry- and exit-relay combination was then analyzed for the presence of attackers to understand how many "safe" or "attacker-free" paths were available. We see in Figure 4 the cumulative distribution function of the fraction of attacker free circuits. China (CN) stands out as the most interesting case. First, we see that there are $\approx$ 10% of all requests generated have no attacker-free circuits! Next, we also notice that there are no known attackers present on $\approx$ 20% of all requests. This appears to indicate that the threat of de-anonymization is non-uniform even within a country, with certain client locations being much safer than others.

The effect of guards. Table 2 shows the effect of reducing the number of guards available to the Tor client during circuit construction. As can be seen, the benefit of reducing the number of guards for preventing correlation attacks is completely location dependent. In countries such as England (GB) and Iran (IR) the benefit is very pronounced with fewer guards drastically decreasing circuits that are vulnerable to our attacker. However, in other countries such

as China (CN), Spain (ES), and Italy (IT) the situation is either marginally or much worse with the number of vulnerable circuits actually increasing as the number of guards decreases. This is counter to the intuition that a smaller guard set will provide better security [29]. Our result is likely impacted by the concentration of guards in specific ASes, a problem identified in previous work [18]. It is clear, however, that there cannot be a universal rule regarding the optimal number of guards that a client must use, without knowing the location of the client.

Country	Siblings	Three Guards	Two Guards	One Guard
BR	Yes	43.6%	29.2%	37.5%
	No	43.6%	28.2%	32.3%
CN	Yes	85.7%	86.7%	86.0%
	No	85.4%	86.5%	85.9%
DE	Yes	56.1%	27.0%	42.7%
	No	54.7%	27.0%	40.7%
ES	Yes	34.8%	59.8%	75.7%
	No	34.8%	58.9%	75.5%
FR	Yes	61.6%	59.8%	58.6%
	No	61.6%	59.8%	58.5%
GB	Yes	62.9%	72.0%	21.5%
	No	61.8%	67.9%	21.5%
IR	Yes	34.1%	33.1%	16.6%
	No	34.0%	33.1%	16.5%
IT	Yes	59.0%	41.2%	81.4%
	No	59.0%	40.3%	79.1%
RU	Yes	77.4%	47.0%	66.6%
	No	76.6%	47.0%	66.5%
US	Yes	73.1%	67.8%	59.5%
	No	73.0%	67.8%	59.4%

Table 2: Percentage of total circuits under threat when lowering the number of guards.

The effect of sibling ASes. From the results seen thus far, it appears that the increase in threat levels when considering sibling ASes is marginal in most cases. Table 3 shows the increase in the number of attacked requests at the corresponding fraction of attacker-free circuits, for each of the seven countries analyzed in our simulations, when siblings were considered as colluders. As expected, the impact is completely location dependent. We see that clients in China (CN) and the United States (US) face the most threat from colluding ASes, with over an additional 1% of web requests facing the scenario of having less than 10% of all available circuits be safe from the adversary, making it much more likely for these requests to be served via a vulnerable circuit, if using the vanilla Tor client.

Country	10%	25%	50%	75%
BR	0.7%	3.2%	9.2%	5.9%
CN	1.4%	1.8%	3.1%	0.7%
DE	0.1%	0.3%	4.0%	6.1%
GB	0.2%	1.3%	6.6%	6.2%
IT	0.0%	2.8%	5.5%	5.5%
RU	0.0%	0.8%	4.4%	4.4%
US	1.0%	2.5%	7.9%	3.0%

Table 3: Percentage increase in the number of requests with fewer than 10, 25, 50, and 75% attacker-free circuits, when considering sibling ASes as colluders .

4. Astoria: AN AS- AND CAPACITY-AWARE TOR CLIENT

Motivated by the observation that vanilla Tor very often selects paths that may be subject to an adversary that exploits asymmetric network paths, we seek to design a relay selection algorithm to mitigate the opportunities for such attackers. We design our relay selection system, Astoria, based on the idea of stochastic relay selection. This works by having the Tor client generate a probability distribution that minimizes the chance of attack over all possible relay selection choices, and selecting an entry- and exit-relay based on this distribution. The advantage of such a stochastic selection is that if the client has no safe options, choosing randomly can be engineered to minimize the amount of information gained by any adversary (as we show below). Further, it allows clients to skew their relay selection towards relays with higher capacity.

4.1 Astoria Goals

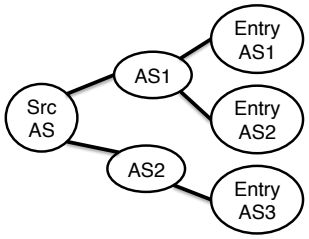
Astoria is constructed with several goals in mind:

- *Deal with asymmetric attackers.* Astoria avoids constructing circuits involving common ASes on the forward- or reverse-paths between the client to the entry-relay and the exit-relay and the destination. This mitigates the threat from RAPTOR style [18] asymmetric correlation attacks.
- *Deal with the possibility of colluding attackers.* Astoria considers the evermore real threat of ASes that may collude to de-anonymize users of anonymity tools. Astoria can be configured to build circuits that do not contain known to be colluding ASes on the forward- or reverse-path between the client and entry-relay and exit-relay and destination.
- *Consider the worst case possibility.* Astoria uses a probabilistic relay selection algorithm that ensures, even in the worst-case (where there are no safe paths to and from the entry- and exit-relay), that no single AS (or, family of ASes) is able to de-anonymize a large number of the client generated circuits. This is done by minimizing the number of circuits that route through each attacker AS.
- *Minimize performance impact.* It is clear that any client which aims to be AS-aware and considers the above threats, will lose the ability to perform many optimizations such as pre-constructing circuits. Our goal is to minimize the effect of the above considerations on the performance of the Tor client.

4.2 Minimizing information gained by any adversary

While there often are cases when there is a relay selection that will completely eliminate the risk of our adversary, we develop our relay selection to be robust, even if this is not the case. Further, with attacks implemented using BGP hijacking and interception the number of unsafe paths may be higher than what we observe in or analysis (we discuss this more in Section 6).

To limit the risk of timing attacks, we define a linear program which generates a probability for each relay selection with the objective to minimize the maximum probability of a circuit encountering any attacker. Recall in our adversary model that we consider a long-lived adversary and that minimizing the probability of an attacker may also be seen



	Uniform	Optimal
Entry AS1	1/3	1/4
Entry AS2	1/3	1/4
Entry AS3	1/3	1/2

Figure 5: Example of optimizing relay selection. Simplified to unidirectional paths and only entry-relay selection.

as minimizing the number of circuits the adversary is able to observe over a long period of time and numerous circuit construction cycles.

Figure 5 shows an example of relay selection to give intuition about how the LP minimizes the risk from any attacker. In this example, we consider unidirectional paths and only entry-relay selection for clarity. In the figure, if the source were to choose uniformly at random across the three entry-relays, there is a 2/3 chance that $AS1$ will be able to observe traffic and only a 1/3 chance that $AS2$ will. In this case, the optimal selection is intuitive, that the source should choose entry-relays 1 and 2 with probability 1/4 each and entry-relay 3 with probability 1/2. This lowers the probability that $AS1$ can observe a circuit from 2/3 to 1/2. This probability of the most likely adversary is the quantity that our LP minimizes.

We use the following notation:

- Let $ADV_{i,j}$ be the set of attackers on the circuit using entry-relay i and exit-relay j to destination $dest$ – i.e., $\forall A \in ADV_{i,j} : A \in (p_{src \leftrightarrow entry_i}) \cap A \in (p_{exit_j \leftrightarrow dest})$.
- Let $X_{i,j,A}$ be an indicator random variable for attacker A on the circuit using entry-relay i and exit-relay j – i.e., $X_{i,j,A} = 1 \iff A \in ADV_{i,j}$, and 0 otherwise.
- Let $P_{i,j}$ be the probability that a circuit using entry-relay $entry_i$ and exit-relay $exit_j$ is utilized by the client.

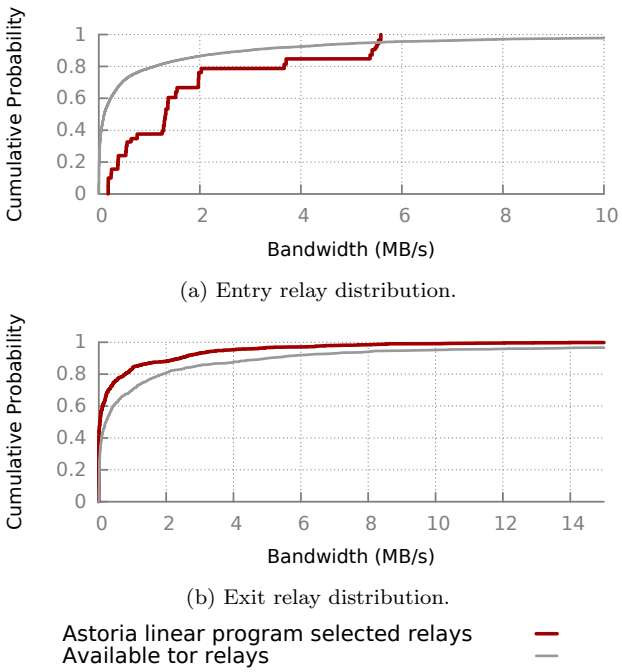
The following linear program is used to minimize the probability of the most likely attacker (i.e., the number of circuits visible to any attacker).

$$\begin{aligned} &\text{minimize } z \\ &\text{subject to } z \geq \sum_{i,\ j}(P_{i,j}\ X_{i,j,A}) \qquad \forall A \\ &P_{i,j} \in [0,\ 1] \qquad \forall i,\ \forall j \\ &\sum_{i,\ j} P_{i,j} = 1 \end{aligned}$$

Essentially, given information about the presence of attackers for each $p_{source \leftrightarrow i}$ and $p_{j \leftrightarrow dest}$ path, the linear program seeks to find the probability distribution $(P_{i,j})$ over available choices of entry- and exit-relays, for which the expected number of circuits visible to each attacker is minimized. Entry- and exit- relays are chosen according to this distribution during circuit construction.

4.3 Security is not enough

While our LP produces a relay selection distribution that minimizes the probability of each adversary, it does not take into account the resources available at the selected relays.



(a) Entry relay distribution.

(b) Exit relay distribution.

Figure 6: Distribution of Astoria linear program selected entry and exit relay bandwidths, compared to available relay bandwidths.

Given that Tor is a system run using community resources contributed by volunteers, load balancing users across these resources is important to ensure that they are used efficiently and no single relay or set of relays become overloaded. Figure 6 shows a snapshot of the distribution of relay capacities available during the period of this study, for all relays in the Tor system and for relays selected using the linear program described previously.

We observe that since our relay selection method does not take relay bandwidth distributions into account, it often selects lower capacity relays as exits, during circuit creation (Figure 8b). This skew towards lower capacity relays is rectified by augmenting the relay selection algorithm with information about relay capacities during circuit construction. This is done by introducing a user tunable parameter $\alpha \in [0,\ 1]$. This tuning parameter allows circuits to select relays with a probability equal to some combination of advertised relay bandwidth (obtained via Tor consensus data available within the client for each relay) and the linear program outputted probability. As α increases, relays are selected more in line with their bandwidth capacity distribution than the linear program generated distribution.

This is done by generating a weighted combination of the two distributions. First, the linear program generates a distribution for relay selection (D_{lp}). Next, a bandwidth based relay selection distribution (D_{bw}) is generated. For this, information from the Tor consensus[3] data is used to obtain advertised relay capacities. Then, for each possible entry- and exit-relay combination (i, j), a probability of selection is done by the normalization of $(.15 \times BW(i)) + (.85 \times BW(j))$

[3]Information regarding relay performance that is generated by Tor directory authorities and received by clients once each hour.

values. Here $BW(i)$ denotes the advertised bandwidth of the i^{th} relay. Weights of .15 and .85 were chosen in order to increase the exit-relay throughput (experiments revealed that exit-relays were often the bottleneck of the circuit – perhaps because of their relatively small number). Given these distributions, the α distribution (D_α) is computed as follows:

$$D_\alpha(i,j) = norm(\alpha \times D_{bw}(i,j) + (1-\alpha) \times D_{lp}(i,j))$$

Therefore, security conscious users may select $\alpha = 0$ to ensure that all circuits are constructed with the goal of minimizing risk from each attacker, while users willing to tradeoff some security for performance may increase α to create a proportional number of circuits using faster and potentially more dangerous relays. In Section 5, we show how changes to α affect Astoria's security and performance.

4.4 Implementing Astoria

In order to make the current Tor client AS-aware and to integrate our measurement toolkit (Section 3), the following modifications were made.

AS-aware on demand circuits. First, the Tor client was modified to perform offline IP to ASN mapping using a database downloaded from APNIC [33] for every incoming request. Note that since the entire database (9 MB) is downloaded, the client does not reveal its intended destination to any lookup services.

Next, modifications were made to the way requests were allocated to circuits. The vanilla Tor client performs preemptive circuit construction in order to serve requests as they arrive (increasing performance significantly). This is unfortunately generally infeasible for a destination- and AS-aware client. Although one may consider pre-constructing AS-aware circuits for a set of frequently served requests ASes, the performance benefit is marginal, at best. This is caused by third party requests for less popular AS destinations embedded in popular Web pages. Astoria, therefore, only performs on demand circuit construction. For each incoming request, Astoria first checks if there are existing circuits serving the same destination AS. The request is attached to the most suitable such circuit if it exists.

Circuit construction and toolkit integration. Astoria creates a new circuit if and only if a request arrives requiring a circuit to connect to an IP address whose AS has no existing or usable circuits. In such cases, the client and destination ASNs were passed to the circuit construction and relay selection algorithms. Circuit construction is performed as follows:

- First, a list of entry- and exit-relays meeting the requirements set by the request were obtained. If the Tor client is configured to utilize only guards as entry-relays, the list of guards is obtained. Next, if Astoria is configured with $\alpha > 0$, information from the most recent Tor consensus is obtained to generate the distribution D_{bw} for each entry- and exit-relay combination.
- The Astoria client performs lookups to the offline IP-ASN database to perform mapping between entry- and exit-relay IP address and AS numbers. These, along with the client and destination AS numbers are then passed to our AS-path prediction and attacker measurement toolkit (Section 3) via a socket connection.
- The toolkit returns the list of ASes on each forward- and reverse-path between the client and every potential entry-relay and the destination and every potential exit-relay. In order to improve performance, paths were cached for frequently queried destinations. Precomputation or caching of paths between the client and the high-uptime entry-relays and destinations and high-uptime exit-relays also help improve performance.
- The returned paths are checked for the presence of common ASes in the entry and exit AS path sets. If there are paths without an attacker, the linear program need not be invoked. Instead, one of the attacker free entry- and exit-relay combinations is chosen at random since this is a valid output for the linear program in such cases. Alternately, Astoria may also choose a safe entry- and exit-relay combination according to the generated D_{bw} probability distribution. We see the impact of this modification in Section 5.
- If there are no attacker-free relay combinations, the linear program is invoked in order to generate the distribution (D_{lp}) that minimizes the probability of the most likely attacker.
- D_α is generated by normalizing and weighting the D_{bw} and D_{lp} distributions. Finally, an entry- and exit-relay combination are selected in accordance to this distribution. The remainder of the circuit construction process remained unchanged from Tor.

Although the above implementation appears computationally intensive, we will show in Section 5, that the performance loss over vanilla Tor is reasonable, and the security benefits are high.

5. ASTORIA EVALUATION

We evaluate Astoria along multiple axes. First, we consider the performance of Astoria by measuring the time required to load webpages and its ability to be a good Tor citizen by selecting bandwidth-rich relays. Second, we evaluate the security enhancements provided by Astoria. We show that Astoria constructed circuits are a good defense against the adversary model described in Section 2.

5.1 Evaluation methodology

For our evaluation of Astoria, we performed page loads of the Alexa local Top 100 websites in the following 10 countries: Brazil (BR), China (CN), Germany (DE), Spain (ES), France (FR), England (GB), Iran (IR), Italy (IT), Russia (RU), and the United States (US). As before, a VPN service was used to connect to servers in each of the above countries.

The Astoria client was tested under various configurations. First, the value of α was set at 0 (using only the security-oriented LP for selection), .5 (hybrid relay selection), and 1 (relay selection based only on relay bandwidth). Second, in the case where there exists a safe relay selection option (*i.e.,* we can build a circuit with no attacker), we consider selecting a safe relay pair uniformly at random and selecting a relay pair based on the distribution of the advertised bandwidth of the routers.

For each configuration, the Astoria client maintained logs to record the entry- and exit-relays available for each circuit being constructed, the bandwidth advertised by each of these relays, the number of relay-selection combinations with and without attackers, and the actual circuit utilized to

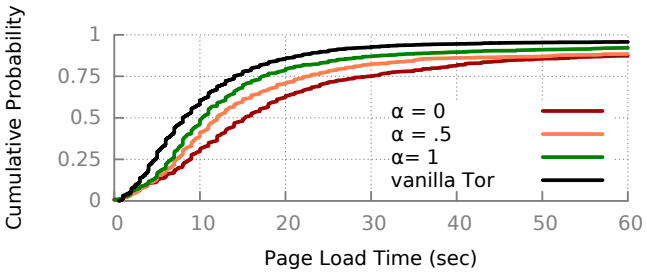


Figure 7: CDF of page load times for vanilla Tor and Astoria with $\alpha = 0$, .5, and 1 (cumulative for Alexa local Top 100 of 10 countries).

serve each incoming request. Network traces were recorded to allow analysis of page load times and round-trip times. Additionally, to understand the performance of our measurement toolkit, logs were maintained to keep track of the time spent on AS-path calculation described in Section 3.

In all configurations, the default setting of 3 entry guards was used. To compare performance with Tor, the same page loads were performed using the default Tor client.

5.2 Performance Evaluation

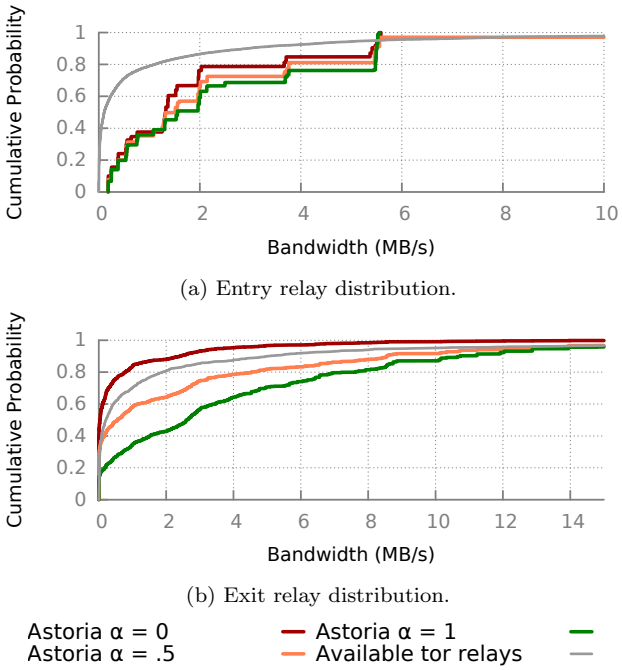
Page load times. Figure 7 shows the distribution of page load times when using vanilla Tor and Astoria under several configurations of α. Interestingly, the median performance penalty of using the most secure configuration ($\alpha = 0$) of Astoria is approximately eight seconds, while the median penalty to less secure configurations (α =.5 and 1) are two and six seconds, respectively. This property stems from the fact that Astoria cannot perform proactive circuit construction and must wait for the request to come in before creating a circuit. The majority of the performance impact of Astoria stems from this lack of circuit preconstruction and not the selection of more secure relays.

Load balancing. Figure 8 shows the bandwidth capacity distributions of the relays selected by the Astoria client for varying levels of α. When Astoria prioritizes relay bandwidth above security ($\alpha = 1$) the 75th percentile of exit-relay bandwidth is much higher than the security-oriented prioritization, with a difference of 7 MB/s.

Interestingly, one might conclude from Figures 7 and 8 that the actual performance bottleneck arises from Astoria's inability to pre-emptively construct circuits.

Overhead of path prediction. Figure 9 shows the CDF of the total amount of time spent on computing AS paths, for each site. We see that for about 50% of all sites (Top 100 sites of 10 countries), the time spent on path computation is negligible. This is due to the high frequency of repeated occurrences of destination ASes in the Top 100 sites – resulting in the AS path for each exit-relay to that destination being in the toolkit's cache. In 60% of the cases where responses were not cached (and $\approx$ 80% of the cases, overall), computing AS paths required under 3.8 seconds.

Alternate strategies for selection of safe paths. As we observed in Section 3, in a majority of cases there are several attacker free entry- and exit-relay selections that may be used for circuit construction. While Astoria, by default, selects between these safe pairs uniformly at random, in order to improve performance and load balancing efforts even further, one may select from them according to their available bandwidth capacity. Such an optimization increases throughput and reduces page load time without impacting security. The performance benefit of this optimization is illustrated in Figure 10. Specifically, this optimization reduces the median page-load time penalty by a sizable 1.6 seconds when $\alpha = 0$.



(a) Entry relay distribution.

(b) Exit relay distribution.

Figure 8: Distribution of Astoria selected entry- and exit- relay bandwidths, compared to available relay bandwidths.

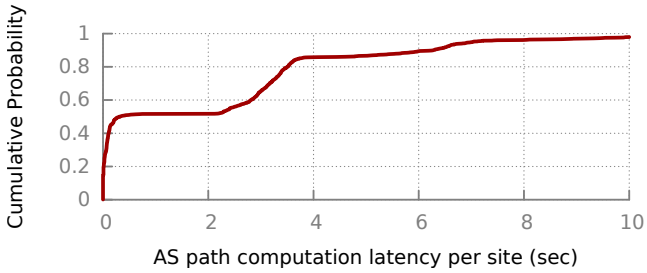
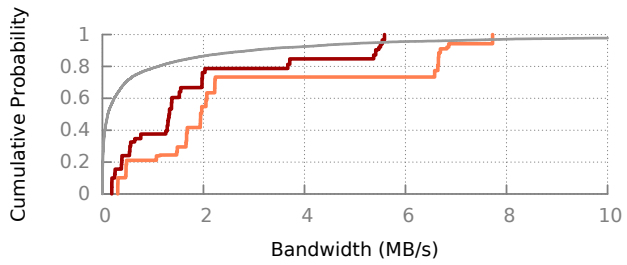


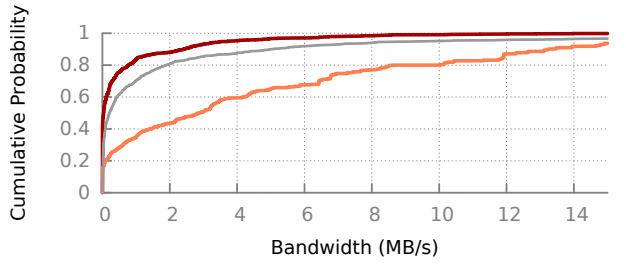
Figure 9: CDF of time spent on AS path computation per site.

5.3 Security Evaluation

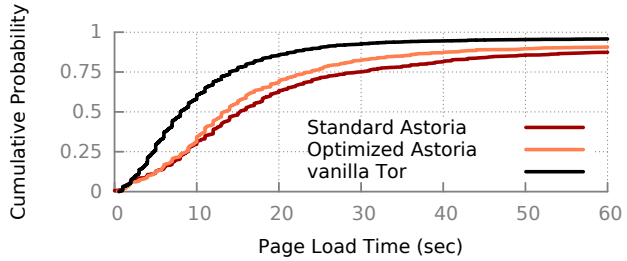
Table 4 shows the percentage of circuits under the threat of attack for various configurations of Astoria and compares them with the Tor client. We see that in it's most secure configuration, Astoria achieves its goal of completely avoiding selection of attacker occupied paths, when there are alternatives. For 5.1% of all source and destination pairs that were evaluated, Astoria was unable to find a safe entry- and exit-relay pair and resorted to relay selection based on the linear program described in Section 4. Therefore, even in these cases, there were no attackers that were able to per-



(a) Entry relay distribution.



(b) Exit relay distribution.



(c) Distribution of page load times (sec).

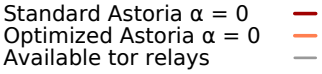


Figure 10: Performance benefits of load balancing across safe circuits.

form de-anonymization on a large number of client generated circuits.

Effect of the tuning parameter (α) on security. We find that the paths to and from high capacity relays are often occupied by common (or, sibling) ASes. Therefore, as the Astoria client attempts to skew towards higher capacity relays, the fraction of attacked circuits increases quite rapidly. At $\alpha = .5$, the security provided by Astoria is comparable with that provided by the Tor client.

6. DISCUSSION

We now discuss how Astoria could be augmented and improved with more recent and ongoing developments from the network measurement community. We also discuss potential performance enhancements for Astoria.

Protecting against other classes of attack. We note that Astoria focuses on adversaries who may lie on asymmetric network paths between the client and entry; and exit and destination, respectively. However, Sun *et al.* [18] highlight attacks based, not only on static path properties, but also dynamics of BGP (*e.g.,* hijacks, routing instability). Taking this sort of attack into account is challenging as it requires realtime access to interdomain routing data and intelligent analysis to identify incidents that may impact the safety of the client's path. In the future, we may integrate subscriptions to BGP hijack data sources (*e.g.,* Argus [34], or more recent efforts at building a real-time interception detector [35]) into Astoria to allow it to operate on dynamic BGP paths.

	Vanilla Tor	Astoria ($\alpha = 0$)	Astoria ($\alpha = .5$)	Astoria ($\alpha = 1$)
Main	42.4%	1.5%	27.2%	43.8%
All	58.6%	5.1%	54.8%	71.2%

Table 4: Percentage of circuits carrying main requests for the Alexa Top 100 websites (of 10 countries) that are under threat and percentage of all circuits under threat for various configurations of Astoria and Vanilla Tor.

Improving path prediction. Astoria relies on paths predicted via simulations of the BGP decision process on empirically derived maps of the AS-level topology. However, while these paths will capture the actual measured path 65-85% of the time [22], they do not precisely indicate the path that will be taken. We note that novel path measurement tools are on the horizon (*e.g.,* Sibyl [36]) that take into account richer vantage point sets than prior work (*e.g.,* PlanetLab used by iPlane [26] vs. RIPE Atlas [37] used by Sibyl). An interesting future direction is determining how such measurement planes can be integrated into a Tor client (*e.g.,* to operate in an offline mode or via a secured querying interface).

Performance enhancements. While the performance of Astoria in its most secure configuration is reasonable for the benefits provided, and in line with expectations of Tor users, we note that there are two primary avenues for improvement – in addition to the alternate relay selection approach discussed and evaluated in Section 5.

- Currently, our assembled measurement toolkit is not completely integrated and built with Astoria, and communication is performed using socket reads and writes. We expect sizable performance benefits from a complete native integration of the toolkit.
- From our performance evaluation it is clear that even inspite of selecting faster relays, Astoria is unable to match the performance of Tor. This difference in performance is attributed almost completely to our inability to preemptively construct circuits. One direction for future research is smart caching and pre-constructing of circuits for Astoria.

Usability of Astoria. From our evaluation of Astoria, it is clear that the performance-security trade-off is favorable only in its higher security configurations. At high security configurations, Astoria is able to perform good load balancing, achieve reasonable throughput, avoid asymmetric colluding attackers whenever possible, and even handle situations where safe circuits are not possible.

However, at lower security configurations, the performance offered by Tor is clearly better, and its security, only slightly worse. Therefore, Astoria is a usable substitute for the vanilla Tor client only in scenarios where security is a high priority.

7. CONCLUSIONS

In this paper, we have leveraged current AS-level topologies and models of interdomain routing to quantify the potential for timing attacks where an adversary can leverage asymmetric Internet routing and collude with others within the same organization. Specifically, we have shown that 58% of the time, while loading pages of common websites, Tor constructs circuits that are vulnerable to such attackers.

To mitigate the threat of asymmetric correlation attacks by colluding attackers, we also developed Astoria– an AS-aware Tor client. In addition to providing high-levels of security against such attacks, Astoria also has performance that is within a reasonable distance from the current Tor client. Unlike other AS-aware Tor clients, Astoria also considers how circuits should be built in the worst case – i.e., when there are no safe relays that are available. Further, Astoria is a good network citizen and works to ensure that the all circuits created by it are load-balanced across the volunteer driven Tor network.

Our work highlights the importance of applying current models and data from network measurement to inform relay-selection and help avoid timing attacks. Astoria also opens multiple avenues for future work such as integrating real-time hijack and interception detection systems (to fully counter RAPTOR [18] attacks) and understanding how new measurement services can be leveraged by a Tor client without defeating anonymity.

Source code: The source code of the Astoria client will be made available under the CRAPL[4] license soon.

Acknowledgments

We would like to thank Ruwaifa Anwar, Haseeb Niaz, and Abbas Razaghpanah for their help with integrating sibling detection algorithms into our measurement toolkit.

This material is based upon work supported by the National Science Foundation under Grant No. CNS-1350720 and a Google Faculty Research Award. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation or Google.

8. REFERENCES

[1] Roger Dingledine, Nick Mathewson, and Paul Syverson. Tor: The second-generation onion router. In *Proceedings of the 13th USENIX Security Symposium*, August 2004.

[2] Norman Danner, Sam DeFabbia-Kane, Danny Krizanc, and Marc Liberatore. Effectiveness and detection of denial of service attacks in Tor. *Transactions on Information and System Security*, 15(3):11:1–11:25, 2012.

[3] Philipp Winter and Stefan Lindskog. How the Great Firewall of China is blocking Tor. In *Proceedings of the USENIX Workshop on Free and Open Communications on the Internet (FOCI 2012)*, August 2012.

[4] Jon McLachlan and Nicholas Hopper. On the risks of serving whenever you surf: Vulnerabilities in Tor's blocking resistance design. In *Proceedings of the Workshop on Privacy in the Electronic Society (WPES 2009)*. ACM, November 2009.

[5] 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, pages 183–195, Washington, DC, USA, 2005. IEEE Computer Society.

[6] Vitaly Shmatikov and Ming-Hsiu Wang. Timing analysis in low-latency mix networks: Attacks and defenses. In *Proceedings of ESORICS 2006*, September 2006.

[7] Steven J. Murdoch and Piotr Zieliński. Sampled traffic analysis by internet-exchange-level adversaries. In *Proceedings of the 7th International Conference on Privacy Enhancing Technologies*, PET'07, pages 167–183, Berlin, Heidelberg, 2007. Springer-Verlag.

[8] Amir Houmansadr and Nikita Borisov. Swirl: A scalable watermark to detect correlated network flows. In *Proceedings of the Network and Distributed Security Symposium - NDSS'11*. Internet Society, February 2011.

[9] Aaron Johnson, Chris Wacek, Rob Jansen, Micah Sherr, and Paul Syverson. Users get routed: Traffic correlation on tor by realistic adversaries. In *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security*, CCS '13, pages 337–348, New York, NY, USA, 2013. ACM.

[10] How the nsa attacks tor/firefox users with quantum and foxacid. `https://www.schneier.com/blog/archives/2013/10/how_the_nsa_att.html`. Accessed: 2015-05.

[11] 'tor stinks' presentation âĂŞ read the full document. `http://www.theguardian.com/world/interactive/2013/oct/04/tor-stinks-nsa-presentation-document`. Accessed: 2015-05.

[12] Nsa stores metadata of millions of web users for up to a year, secret files show. `http://www.theguardian.com/world/2013/sep/30/nsa-americans-metadata-year-documents`. Accessed: 2015-05.

[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, pages 380–389, New York, NY, USA, 2009. ACM.

[14] 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, pages 66–76, New York, NY, USA, 2004. ACM.

[15] Masoud Akhoondi, Curtis Yu, and Harsha V. Madhyastha. Lastor: A low-latency as-aware tor client. In *Proceedings of the 2012 IEEE Symposium on Security and Privacy*, SP '12, pages 476–490, Washington, DC, USA, 2012. IEEE Computer Society.

[16] Chris Wacek, Henry Tan, Kevin S. Bauer, and Micah Sherr. An empirical evaluation of relay selection in tor. In *20th Annual Network and Distributed System Security Symposium, NDSS 2013, San Diego, California, USA, February 24-27, 2013*, 2013.

[17] Laurent Vanbever, Oscar Li, Jennifer Rexford, and Prateek Mittal. Anonymity on quicksand: Using bgp

[4] `http://matt.might.net/articles/crapl/`

to compromise tor. In *Proceedings of the 13th ACM Workshop on Hot Topics in Networks*, HotNets-XIII, pages 14:1–14:7, New York, NY, USA, 2014. ACM.

[18] Yixin Sun, Anne Edmundson, Laurent Vanbever, Oscar Li, Jennifer Rexford, Mung Chiang, and Prateek Mittal. RAPTOR: routing attacks on privacy in tor. *CoRR*, abs/1503.03940, 2015.

[19] Ethan Katz-Bassett, Harsha V. Madhyastha, Vijay Kumar Adhikari, Colin Scott, Justine Sherry, Peter van Wesep, Thomas E. Anderson, and Arvind Krishnamurthy. Reverse traceroute. In *Proceedings of the 7th USENIX Symposium on Networked Systems Design and Implementation, NSDI 2010, April 28-30, 2010, San Jose, CA, USA*, pages 219–234, 2010.

[20] Vasileios Giotsas, Matthew Luckie, Bradley Huffaker, and kc claffy. Inferring complex as relationships. In *Proceedings of the 2014 Conference on Internet Measurement Conference*, IMC '14, pages 23–30, New York, NY, USA, 2014. ACM.

[21] Phillipa Gill, Michael Schapira, and Sharon Goldberg. Modeling on quicksand: Dealing with the scarcity of ground truth in interdomain routing data. *SIGCOMM Comput. Commun. Rev.*, 42(1):40–46, January 2012.

[22] Ruwaifa Anwar, Haseeb Niaz, David Choffnes, Italo Cunha, Phillipa Gill, and Ethan-Katz Bassett. Investigating interdomain routing policies in the wild. *University of Southern California Technical Report #15-958*.

[23] Torspec – tor's protocol specifications. `https://gitweb.torproject.org/torspec.git/tree/path-spec.txt`. Accessed: 2015-05.

[24] Nikita Borisov, George Danezis, Prateek Mittal, and Parisa Tabriz. Denial of service or denial of security? In *Proceedings of the 14th ACM Conference on Computer and Communications Security*, CCS '07, pages 92–102, New York, NY, USA, 2007. ACM.

[25] The lifecycle of a new relay – the tor blog. `https://blog.torproject.org/blog/lifecycle-of-a-new-relay`. Accessed: 2015-05.

[26] Harsha V. Madhyastha, Tomas Isdal, Michael Piatek, Colin Dixon, Thomas Anderson, Arvind Krishnamurthy, and Arun Venkataramani. iPlane: an information plane for distributed services. In *OSDI*, 2006.

[27] Lixin Gao and Jennifer Rexford. Stable internet routing without global coordination. *IEEE/ACM Transactions on Networking (TON)*, 9(6):681–692, 2001.

[28] B. Quoitin and S. Uhlig. Modeling the routing of an autonomous system with c-bgp. *Netwrk. Mag. of Global Internetwkg.*, 19(6):12–19, November 2005.

[29] Roger Dingledine. Improving tor's anonymity by changing entry guard parameters. *The Tor Blog*.

[30] Tor Metrics Project website. Tor project: Anonimity online. Available at https://metrics.torproject.org.

[31] Freedom House. Freedom on the Net 2014. `https://freedomhouse.org/sites/default/files/FOTN_2014_Full_Report_compressedv2_0.pdf`.

[32] Alexa top sites. `http://www.alexa.com/`. Accessed: 2015-04.

[33] Philip Smith. Bgp routing table analysis. http://thyme.apnic.net/.

[34] Xingang Shi, Yang Xiang, Zhiliang Wang, Xia Yin, and Jianping Wu. Detecting prefix hijackings in the internet with argus. In *Proceedings of the 2012 ACM Conference on Internet Measurement Conference*, IMC '12, pages 15–28, New York, NY, USA, 2012. ACM.

[35] CAIDA. HIJACKS: Detecting and Characterizing Internet Traffic Interception based on BGP Hijacking . `http://www.caida.org/funding/hijacks/`.

[36] Ethan Katz-Bassett and Pietro Marchetta and Matt Calder and Yi-Ching Chiu and Italo Cunha and Harsha Madhyastha and Vasileios Giotsas. Sibyl: A Practical Internet Route Oracle. `http://www.caida.org/workshops/aims/1503/slides/aims1503_katzbassett1.pdf`.

[37] RIPE NCC. RIPE atlas. `http://atlas.ripe.net`.