Counter-RAPTOR: Safeguarding Tor Against Active Routing Attacks

Yixin Sun
Princeton University
yixins@cs.princeton.edu

Anne Edmundson
Princeton University
annee@cs.princeton.edu

Mung Chiang
Princeton University
chiangm@princeton.edu

Nick Feamster
Princeton University
feamster@cs.princeton.edu

Prateek Mittal
Princeton University
pmittal@princeton.edu

ABSTRACT

Anonymity systems such as Tor are known to be vulnerable to network-level adversaries who can observe both ends of the communication to deanonymize Tor users. Recent works have shown that Tor is susceptible to the previously unknown active BGP routing attacks, which expose Tor users to more potential network-level adversaries. This paper aims at mitigating and detecting such active routing attacks against Tor. We first present a new measurement study on the resilience of the Tor network to active BGP prefix attacks. We show that Autonomous Systems that carry a large portion of Tor traffic, such as OVH, have relatively low resilience to active attacks. Next, we present a new guard relay selection algorithm that incorporates resilience of relays into considerations. We show that the algorithm successfully lowers the probability of a Tor client being affected by a hijack attack by 31%, and at the same time provides better anonymity to the Tor client. Finally, we demonstrate a live monitoring system that can detect routing anomalies on the Tor network in real time. By tuning our system for the Tor network, it can detect an attack similar to known attacks while having no false positives.

CCS Concepts

•**Security and privacy** → *Pseudonymity, anonymity and untraceability; Network security;*

Keywords

ACM proceedings; LaTeX; text tagging

1. INTRODUCTION

The Tor network [16] has been the most widely used system for anonymous communication that protects users' identities from untrusted parties who have access to user traffic. Tor serves millions of users and carries terabytes of traffic everyday with its network of over 7,000 relays [12], which makes it a popular target for adversaries who wish to break the anonymity of the users.

Tor is known to be vulnerable to traffic correlation attacks. An adversary who can observe the traffic at both ends of the communication path (i.e., between the Tor client and the entry relay, and between the exit relay and the destination server) can perform traffic analysis on packet sizes and timings to deanonymize the Tor users [32, 36]. Network-level adversaries, i.e., autonomous systems (ASes), that lie on the path between a Tor client and an entry relay, and between an exit relay and the destination server have been shown to be at such a compromising position to deanonymize Tor clients [17, 18, 22]. More recently, researchers have further exploited the dynamics of BGP routing to propose the new RAPTOR attacks [35], which exaggerate this threat by enabling more network-level adversaries to be at a compromising position, including active BGP prefix attacks which were not previously studied on Tor.

Building countermeasures to defend Tor against such malicious AS-level adversaries is a challenge facing the research community. Past works have explored AS-aware relay selection algorithms that minimize the chance of selecting Tor relays with the same AS lying on both ends of the communication paths [13, 17, 34]. However, all this work only focuses on mitigating *passive* attacks in which AS-level adversaries only passively observe traffic instead of launching any *active* attacks. Previous studies have shown that active BGP routing attacks can pose new threats to Tor users, and Tor relays have already been affected in past real-world BGP attacks [35]. These observations motivate our work on developing countermeasures against such active BGP attacks on Tor.

This paper has three contributions. First, we quantify the vulnerability of the current Tor network to active BGP prefix hijack and interception attacks. Second, we develop proactive approaches to lower the probability of being affected by such attacks, which includes a novel Tor guard relay selection algorithm. Finally, we present a live monitoring system on Tor that can detect routing anomalies in Tor relays.

Measurement on the Tor network. We measure the vulnerability of the current Tor network by calculating the resilience to BGP prefix attacks for all ASes that contain Tor relays. Based on the current Internet topology [3] and Tor consensus data [11], we first leverage an AS-resilience metric [27] to measure resilience to *all* possible hijacking scenarios. Then, we extend the metric to analyze interception attack scenarios and measure resilience to interception

WOODSTOCK '97 El Paso, Texas USA

DOI: 10.475/123_4

attacks launched by Tier 1 ASes. Our keys findings are:

- Resilience values corresponding to hijack attacks for all Tor-related ASes are similar to a normal distribution, where most ASes fall in the middle of the spectrum. However, some ASes that contain a high number of relays and/or high bandwidths also have low resilience values, i.e., AS 16276 (OVH), which contains 339 Tor relays and only has a resilience value of 0.32 on a scale of $[0, 1]$. (0 corresponds to being resilient to *no* attacking AS and 1 corresponds to being resilient to *all* attacking ASes.)
- Resilience values corresponding to interception attacks for all Tor-related ASes are skewed towards higher resilience. However, some ASes (i.e., OVH) with a lot of Tor relay bandwidth have relatively low resilience, which means that they are more susceptible to interception attacks.
- Average Tor-related AS resilience has slightly increased each year from 2008 to 2016.

Proactive approaches against active BGP attacks. First, we start a campaign by contacting Tor relay operators to move their relays into a more specific prefix range, i.e., /24. We have successfully cooperated with an anonymous relay operator to move a Tor relay into a /24 prefix from the original /16 prefix. We also show that the increase in routing table size due to announcing /24 prefixes for Tor relays is negligible. Second, we propose and implement a novel Tor guard relay selection algorithm, which considers the AS resilience metric. Our guard relay selection algorithm is the first algorithm to incorporate resilience to active BGP routing attacks on Tor [35]. The algorithm combines resilience and bandwidth into relay selection to ensure security as well as performance. Our evaluation shows that the algorithm achieves a 31% reduction in probability of being affected by a prefix hijack attack and improvement on anonymity bounds compared to the current Tor relay selection algorithm. At the same time, it does not suffer any noticeable performance loss based on a real-world evaluation of page load time from the Alexa Top 100 sites.

Reactive approaches against active BGP attacks. We build a live monitoring system that monitors routing activities on Tor relays in real time. The monitoring system consists of two parts: a BGP monitoring system on the control plane, and a `traceroute` monitoring system on the data plane. The BGP monitoring system collects live BGP updates from BGP Stream [2], as well as the latest hourly Tor consensus data, and detects any suspicious prefix announcements (affecting the Tor network) in real time. The `traceroute` monitoring system serves as a verification mechanism if the BGP monitoring system flags anomalous behavior. The system uses Planetlab nodes to send `traceroute` requests to the Tor relays flagged by BGP monitoring, and verifies the routing anomaly in the data plane. Our live monitoring system will help enhance the transparency of the Tor network with regards to active BGP attacks. Our evaluation shows that most BGP updates that involve a Tor relay are only announced by a single AS (across all updates). We implement heuristics that are well-suited for detecting attacks on the Tor network, with low false positive rates; we have tuned our heuristics to detect an injected attack while having no false positives.

The paper is organized as follows. Section 2 provides a brief overview of background and related work on Tor. Section 3 describes the metrics and methodology used to measure Tor relay resilience to active BGP prefix hijack and interception attacks. Section 4 discusses our campaign on moving Tor relays to /24 prefixes and presents our new Tor guard relay selection algorithm. Section 5 demonstrates our design for the live monitoring system and describes our deployment experience. Section 6 discusses potential obstacles and shortcomings of the current approaches and directions for future work. Finally, we conclude in Section 7.

2. BACKGROUND AND RELATED WORK

Here we discuss network level adversaries on the Tor network and past work on defending against such network level adversaries.

2.1 Network Adversaries on Tor

Feamster and Dingledine [18] first investigated AS-level adversaries in anonymity networks, and they showed that some ASes could appear on nearly 30% of entry-exit pairs. Murdoch and Zielinski [28] later demonstrated the threat posed by network-level adversaries who can deanonymize users by performing traffic analysis. Furthermore, Edman and Syverson [17] demonstrated that even given the explosive growth of Tor during the past years, still about 18% of Tor circuits result in a single AS being able to observe both ends of the communication path. In 2013, Johnson *et al.* [22] evaluated the security of Tor users over a period of time, and the results indicated that a network-level adversary with just a low-bandwidth cost could deanonymize any user within three months with over 50% probability and within six months with over 80% probability.

While all prior research, to our knowledge, focus on passive adversaries, more recently, Sun *et al.* [35] proposed a new suite of attacks, called RAPTOR attacks, that discovered the threat posed by active AS-level adversaries who can perform active BGP routing attacks to put themselves onto the path between client-entry and/or exit-destination. The paper also showed from past BGP data that during past known real-world BGP prefix attacks, Tor relays were affected as well - i.e., in the Indosat hijack in 2014, among the victim prefixes there were 44 Tor relays, and 33 of them were guard relays which had direct connections with Tor clients.

2.2 Tor Path Selection against Network Adversaries

The existence of network-level adversaries motivates the research on AS-awareness in path selection in Tor. In 2012, Akhoondi *et al.* [13] proposed LASTor, a Tor client which takes into account AS-level path and relay locations in selecting a path; although our work differs by considering relays' resilience to active attacks and relays' capacity. Recently, Nithyanand *et al.* [34] constructed a new Tor client, Astoria, which adopted a new path selection algorithm that considered more aspects - relay capacity, asymmetric routing, and colluding ASes. However, Astoria only considers a passive AS-level attacker and does not consider the case of active routing attacks. This motivates our work on defending Tor against active AS-level adversaries who can launch active routing attacks such as BGP prefix hijacks and interceptions. Towards this goal, it is important to understand AS-level Internet topology and network path predictions.

Lad *et al.* [27] investigated the relationship between Internet topology and prefix hijacking, and provided a metric for evaluating AS resilience to active prefix hijack attacks. Although the study was conducted in 2007 when there were far less ASes than now, and the study only simulated a partial attack scenario of 1000 randomly selected ASes as attackers, it provides a foundational starting point for our work. Thus, in section 3, we will start with measuring the AS resilience of the Tor network to active prefix hijack attacks using the metric, but considering *all* attack scenarios; then, we devise a novel metric to evaluate AS resilience to active prefix *interception* attacks. In Section 4.2, we incorporate the AS resilience metric into the Tor guard relay selection algorithm.

3. MEASURING TOR'S CURRENT STATE OF RESILIENCE TO BGP ATTACKS

In order to defend Tor against active BGP attacks launched by network-level adversaries, we start by investigating the vulnerability of the Tor network to BGP prefix attacks and quantifying how much of the Tor network would be affected. First, we look at how to evaluate the Tor network in terms of susceptibility to hijack attacks. Second, we extend the metric to quantify how resilient the Tor network is to prefix interception attacks by Tier 1 ASes. These steps help quantify how vulnerable the Tor network is to network-level adversaries in a novel way. Specifically, we measure:

- How resilient ASes that contain Tor relays are to prefix hijack attacks
- How resilient ASes that contain Tor relays are to prefix interception attacks

3.1 Resilience to prefix hijack attacks

Network-level adversaries can launch a BGP prefix hijack by announcing a prefix that it does not own. Consequently, some ASes would be deceived by the false announcement and thus send traffic to the false origin AS instead of the true origin AS. Previous work has tackled questions of AS resilience to prefix hijack attacks using simulations of the entire Internet [27]. We build off this work by applying these metrics to the Tor network. [27] explains the probability of a node v believing the true origin t given a false origin a announcing a route that belongs to true origin t:

$$\bar{\beta}(t, a, v) = \frac{p(v, t)}{p(v, t) + p(v, a)} \tag{1}$$

where $p(v, a)$ is the number of equally preferred paths from node v to false origin a and $p(v, t)$ is the number of equally preferred paths from node v to true origin t. Using this probability, the resilience metric is introduced – the resilience of a node t is the fraction of nodes that believe the true origin t given an arbitrary hijack against t:

$$R(t) = \sum_{a \in N} \sum_{v \in N} \frac{\bar{\beta}(t, a, v)}{(N-1)(N-2)} \tag{2}$$

where N is the total number of ASes.

To measure the resilience of Tor-related ASes, we first use an Internet topology [5] to get all of the AS relationships, and construct an AS-level graph. We identify the Tor-related ASes and simulate prefix hijacks on the graph. Algorithm 1 shows the steps we take on the AS graph to calculate the resilience of Tor-related ASes from a source AS t. Then, by summing the resilience values from *all* source ASes and dividing by $(N-1)$, we get the total resilience of each Tor-related AS.

Algorithm 1 Algorithm to calculate prefix hijack resilience for Tor-related ASes.

```
function CalcResilience(graph G, node t)
    CalcPathsFromNode(G, t)
    zeros(R)
    for each reachable node v from node t do
        if node v contains Tor guard/exit relays then
            n ← num. of less preferred nodes than node v
            R[v] ← n + β̄(v, a, t) ∀ equally preferred node a
        end if
    end for
    N ← num. of nodes in G
    return [R[i]/(N − 2) for each node i in R]
end function
```

Note that the CalcPathsFromNode(G, t) step computes paths from source AS t to all other ASes, which requires AS-level path predictions. Previous works have shown that AS level paths are determined mainly based on two preferences [20]:

- Local Preference: customer route is preferred over peer route, which is preferred over provider route.
- Shortest Path: With the highest local preference, paths with the shortest hops will be preferred.

Furthermore, the AS paths should also have the *valley free* property [19]. Thus, we use breadth first search to traverse the graph from a given source node based on this property and the preferences. We first explore provider-customer paths, which are the most preferred; next, we explore one peer-to-peer path followed by a sequence of provider-customer paths, which are the next preferred; finally, we explore customer-provider paths followed by an optional peer-to-peer path and then followed by a sequence of provider-customer paths. Note that, nodes are explored in the most preferred to least preferred order, and those which are explored in the same step are equally preferred. This ordering will help accelerate the resilience calculation.

3.2 Resilience to prefix interception attacks

Next, we measure the resilience of Tor-related ASes to prefix interception attacks. Launching a prefix interception attack requires one further step than prefix hijack attacks - the false origin AS needs to forward the hijacked traffic back to the true origin AS. Prior work [15, 30] has pointed out that to be able to do this, the false origin AS needs to satisfy a *safety condition*: none of the ASes along the existing route from false origin AS to true origin AS should choose the invalid route advertised by the false origin AS, and thus the false origin AS can still use its existing route to forward the hijacked traffic back to the true origin. Thus, when making the invalid route announcement, there are two cases to consider: (1) if the false origin AS's existing route to the true origin AS is through a peer or customer route, then

it's safe to make the false announcement to all its neighbors without affecting its existing route to the true origin; (2) if the false origin AS's existing route to the true origin AS is through a provider route, then it can only make the false announcement to its peers and customers, but not providers.

Based on the above property, we modify the Algorithm 1 to the following Algorithm 2 to evaluate resilience to interception attacks.

Algorithm 2 Algorithm to calculate prefix interception resilience for Tor-related ASes.

function CALCRESILIENCE(graph G, node t)
  CALCPATHSFROMNODE(G, t)
  $zeros(R)$
  for each reachable node v from node t **do**
    if node v contains Tor guard/exit relays **then**
      $n \leftarrow$ less preferred nodes than node v
      if existing route t to v is provider route **then**
        $n \leftarrow (n - m)$ for all nodes m for which t to m is provider route
      end if
      $a \leftarrow$ equally preferred nodes as node v
      if existing route t to v is provider route **then**
        $a \leftarrow (a - m)$ for all nodes m for which t to m is provider route
      end if
      $R[v] \leftarrow n + \bar{\beta}(v, a, t)$ $\forall$ equally preferred node a
    end if
  end for
  $N \leftarrow$ num. of nodes in G
  return $[R[i]/(N-2)$ for each node i in $R]$
end function

3.3 Hijack Resilience Results

Resilience of the current Tor network. We obtained the list of Tor relays from the Tor consensus data in January 2016 and retrieved their belonging ASes. Then, we downloaded the AS topology published by CAIDA in January 2016. The AS topology contains 52,838 ASes, in which 1,202 ASes contain a total of 6,942 Tor relays. We simulated *all* possible hijacking scenarios against each of the 1,202 Tor-related ASes, totaling $52,837 \times 1,202 = 63,510,074$ prefix hijacks. We used the methods described in Section 3.1 to evaluate the resilience of each Tor-related AS.

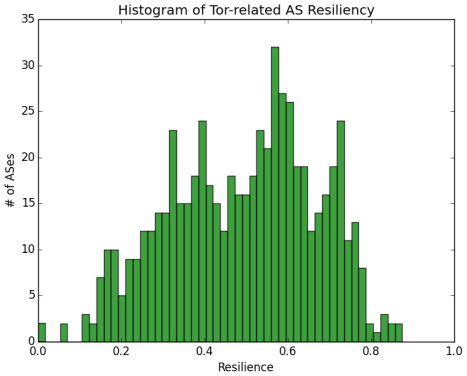
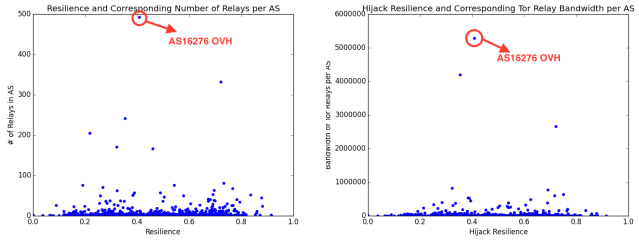


Figure 1: Histogram of hijack resilience values for all Tor-related ASes.

Figure 1 shows the prefix hijack resilience results. We can see that most Tor-related ASes lie in the middle of the spectrum for resilience. We then look at AS resilience corresponding to its involvement in Tor in terms of number of Tor relays and aggregated Tor bandwidth that it contains. Figure 2a shows the resilience value for an AS corresponding to the number of Tor relays it contains. Even though resilience values are even across ASes, we find that the most common ASes for hosting Tor guards are extremely susceptible to attack. Figure 2b shows the resilience value for an AS corresponding to its total Tor bandwidth.



(a) Hijack Resilience vs. Number of Tor Relays per AS

(b) Hijack Resilience vs. Tor Bandwidth per AS

Figure 2: Hijack resilience vs. relays/bandwidths for all Tor-related ASes.

We can see that in both figures 2a and 2b, there is one significant outlier, AS16276 (OVH), which contains 339 relays and carries a total of 5,282,512 KB/s Tor bandwidths. It has a relatively low hijack resilience value of 0.408, indicating that in a hijack event, the probability of a Tor client (who uses relays in the AS) being deceived is close to 60%. Another AS with high bandwidth yet low resilience is AS12876 (ONLINE S.A.S.), which contains 242 relays. It has an even lower resilience value of 0.35, and therefore, and even greater probability of being deceived by a hijack event.

Resilience over the years. We also conducted our measurements on past data to analyze the hijack resilience trend on the Tor network. Figure 3 shows the average AS resilience of Tor-related ASes to hijack attacks with standard deviations above and below using January data from each year. We can see that the average resilience has only slightly increased since 2008. There was a slight decrease in average resilience between 01/2013 and 01/2015, and then increased back again in 01/2016. After further investigation, we found that there was a steep increase in the number of Tor-related ASes from 2013 to 2015 - about 200 new ASes hosting Tor relays each year (18.8% to 21.6% increase), while the total number of ASes on the internet only increased marginally (0.26% to 6.4% increase). Thus, the corresponding decrease in average resilience might be due to many more stub ASes (potentially with low resilience values) starting to run Tor relays in those two years, which brought down the overall average. On the contrary, from 01/2015 to 01/2016, we saw an increase of $> 6,600$ ASes in the internet (14.4% increase), while the number of Tor-related ASes remained almost the same. A large portion of the new emerging ASes might be stub ASes with low hijacking influences, potentially resulting in the increase of total resilience values of the existing Tor-related ASes.

3.4 Interception Resilience Results

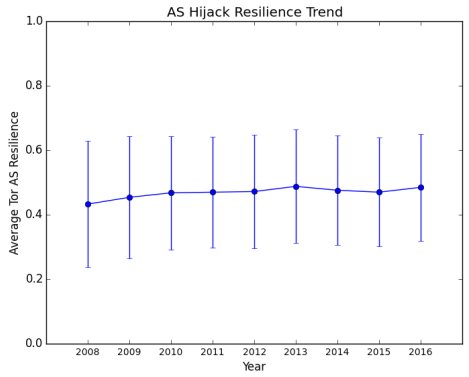


Figure 3: Average hijack resilience of Tor-related ASes each year from 2008 to 2016.

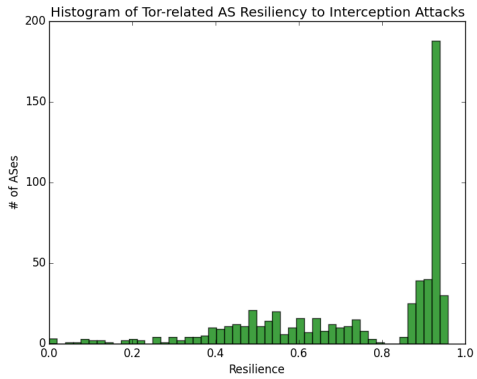


Figure 4: Histogram of interception resilience values to Tier-1 ASes for all Tor-related ASes.

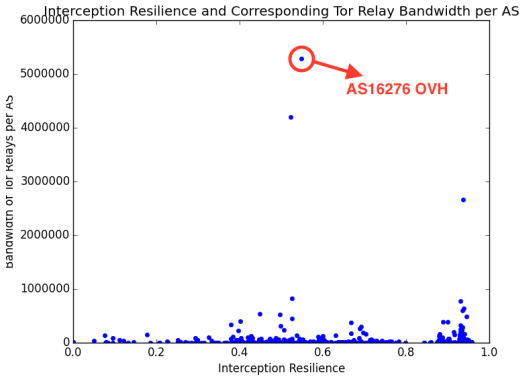


Figure 5: Interception Resilience to Tier-1 ASes vs. Tor Bandwidth per AS

Tier-1 ASes play an important role in Internet routing. They sit at the top level of the internet hierarchy and carry a large amount of network traffic. Recall from the discussion in Section 3.2 that in order to successfully intercept traffic of a true origin AS, the false origin AS needs to satisfy a *safe condition* - it cannot announce the invalid route to its providers when its existing route to true origin AS is through a provider route. This condition puts Tier-1 ASes at a powerful position - Tier-1 ASes do not have any providers and thus can always announce the invalid route to all its neighbors (peers/customers), which will be further propagated down to other ASes in the Internet hierarchy. On the contrary, ASes that are towards the bottom of the Internet hierarchy would not have much interception power. They have limited number of peers/customers, and most of their outgoing routes are through providers. Therefore, due to the difference in interception power, we only focus on measuring interception resilience to Tier-1 ASes as the attacking AS here instead of *all* ASes as the attacking AS.

We picked 17 ASes as Tier-1 ASes in our evaluation [1]. We used the Tor consensus data and CAIDA Internet topology data, both from January 2016. Figure 4 shows the distribution of interception resilience values of all Tor-related ASes. We can see that many Tor-related ASes have high resilience to interception attacks that could be carried out by Tier-1 ASes. The intuition behind this is that, even though Tier-1 ASes are at a position to intercept traffic, they also have longer paths from ASes that are close to the bottom of the hierarchy, which may prefer closer ASes with shorter paths instead of taking the longer paths to reach the Tier-1 ASes.

Figure 5 shows the plot of interception resilience values to Tier-1 ASes versus bandwidth of a Tor-related AS. Interestingly, the two ASes with highest Tor bandwidth - AS16276 (OVH) and AS12876 (ONLINE S.A.S.) - do not have high interception resilience. Their resilience values fall between 0.45 and 0.5, indicating that in an interception attack event by a Tier-1 AS, the probability of a Tor client (who uses relays in either of these two ASes) being deceived is $> 50\%$. Note that this result is consistent with the hijack resilience result shown in Figure 2b, with these same two ASes of high bandwidth yet low resilience values.

Bandwidth is not enough. We have shown in this section that Tor-related ASes with high bandwidth can have low resilience values to both hijack and interception attacks, and thus are more likely to be susceptible to active BGP attacks compared to high resilience ASes. What's even worse is that high bandwidth relays are more preferred by Tor users during relay selection, resulting in exposing many Tor users to the high risk of being compromised by active BGP attacks. Therefore, this vulnerability motivates our work on incorporating AS resilience into relay selection, which we will delve into details in Section 4.2.

[1] AS174, AS209, AS286, AS701, AS1239, AS1299, AS2828, AS2914, AS3257, AS3320, AS3356, AS5511, AS6453, AS6461, AS6762, AS7018, AS12956

4. PROACTIVE APPROACHES

Sun *et al.* [35] recently demonstrated a successful real-world BGP routing attack on a Tor guard relay by announcing a more-specific prefix (i.e., /24 prefix against /23 prefix), which covers the guard relay. We have shown in Section 3 that network-level adversaries can hijack a portion of the traffic by announcing an equally-specific prefix (i.e., same prefix length). To counter these two routing attacks, we propose two methods, respectively: 1) convincing relay operators to move the relay into a /24 prefix to defend against a more-specific prefix attack, and 2) introducing a new Tor guard relay selection algorithm that minimizes the likelihood that a Tor client sees a hijacked route in the case of an equally-specific prefix attack.

4.1 Using /24 Prefixes

Network-level adversaries can hijack internet traffic by announcing a more-specific prefix that belongs to the true origin AS. Since BGP routing prefers longer prefixes over shorter ones, the traffic will go to the false origin with the more-specific prefix instead. The longest prefix length that BGP usually accepts is /24, and thus any prefix that's shorter than /24 is vulnerable to more-specific prefix attacks. Sun *et al.* [35] recently found that more than 90% of BGP prefixes hosting relays are shorter than /24, making them vulnerable to a more-specific BGP prefix attack.

Therefore, one way to defend against more-specific prefix hijack attacks is to move Tor guard relays into /24 prefixes. One concern that may arise is the extra burden which can be created on the routing table if announcing /24 prefixes for all Tor guard relays. We obtained a snapshot of the routing table from RouteViews [9], and the Tor consensus data from January 2016. We found that there are 607,335 unique prefixes in the routing table, whereas there are only 1484 unique /24 prefixes covering all Tor guard relays (roughly 0.24% of the routing table prefixes). Thus, the impact on the routing table size would be negligible.

To make real world impact of this approach, we start a campaign in progress by contacting Tor relay operators, and suggesting that they move their Tor relays into /24 prefixes. We start with cooperating with an anonymous Tor relay operator, which contains a Tor relay under a /16 prefix. The approach they took was to announce an additional /24 prefix that covers the Tor relay, while still maintaining the original /16 announcement. This minimized the amount of work needed to move Tor relays into /24 prefixes.

4.2 Guard Relay Selection

Guard relays are at an important position in the Tor circuit, since they have direct connections with Tor clients. Section 4.1 discusses how to mitigate more-specific prefix hijack attacks; however, even if the guard relay belongs to a /24 prefix, it is still subject to an equally-specific prefix hijack attack. We have shown in Section 3 that high-bandwidth Tor relays could belong to ASes with low resilience to prefix hijack attacks, which put many Tor users at risk. Therefore, we propose a new guard relay selection algorithm that incorporates AS resilience in order to mitigate such active prefix hijacks on Tor.

4.2.1 Design Goals

1. *Mitigate prefix hijacks on Tor.* This is the main goal of the new guard relay selection algorithm. The algorithm computes the AS resilience against prefix hijacks of all Tor guard relays from the client source AS, and prefers the ones that have higher resilience, minimizing the likelihood that the client would be affected by a prefix hijack on its guard relay.

2. *Protect the anonymity of Tor clients.* In addition to lowering possibilities of being hijacked, the algorithm should also protect the anonymity of Tor users by balancing preferences among relays and providing rigorously assessed anonymity bounds.

3. *Performance and Load balancing.* The algorithm should incorporate relay bandwidth into the selection decision and avoid causing excessive traffic congestion on low bandwidth relays.

4.2.2 Mitigate Prefix Hijacks on Tor

In Section 3.1, we measured AS resilience of each Tor-related AS based on the metrics proposed by [27]. The algorithm computes *total* resilience of an AS by summing individual resiliencies from each source AS. *However, the Tor client would only care about the individual resilience to a Tor-related AS from the source AS where the client is located.* Thus, instead of computing the total resilience, we use Algorithm 1 to calculate resilience $R(i)$ of each Tor-related AS i, from the client AS t.

Tor relay selection is bandwidth-aware and prefers high bandwidth relays. The probability of each relay i being chosen is based on its bandwidth $B(i)$. Thus, in order to still provide clients with the load balancing option of Tor bandwidths, we offer a tunable parameter α in the relay selection algorithm combining AS resiliency $R(i)$ and bandwidth $B(i)$. Each relay i will be assigned a weight as following:

$$W(i) = \alpha \times R(i) + (1 - \alpha) \times B(i)$$

Note that, when α is set to 0, the relay selection becomes the same as bandwidth-only selection; while when α is set to 1, the selection becomes resiliency-only selection.

4.2.3 Randomization is needed

If we simply select the set of guard relays based on the probability of $W(i)/\sum W(i)$, an adversary can potentially run a relay that has an AS-level path with high local preferences and/or short path length to the Tor client, such that it has high resilience from the client AS as the source. And it therefore obtains a high probability of being chosen. Furthermore, the Tor client might also be susceptible to fingerprinting attacks due to the differences in relay selection probabilities based on the AS-location of the client. An adversary that can observe the client for a long enough time may be able to infer the AS-location of the client based on its observed relay selection choices. Thus, we need to take into account these potential vulnerabilities and protect the anonymity of clients. Note that the weight of a Tor relay depends on two components: (1) the resilience of the AS in which the relay is located, and (2) the relay's bandwidth. The relay's bandwidth is not specific to client locations, and thus would not reveal any client identities; in addition, due to resource constraints, it is not trivial to run a relay with significantly higher bandwidth than all other relays to obtain high probability of being chosen. While on the contrary, AS resilience of relays *is* client-specific, and requires much fewer resources to run a malicious relay with high AS resilience.

Instead of using resilience $R(i)$ for relay i directly in the weight calculation, we first adjust it to $R(i)\prime$ by calculating the estimated inclusion probability of the relay in a random sampling of size $(g \cdot N)$ using the algorithm proposed by Tille [37]. Note that N corresponds to the total number of ASes containing Tor guard relays, and g is a configurable parameter indicating the percentage of random sampling we want to perform. The steps are as following:

1. For each relay i, $R(i)\prime = \frac{k \cdot R(i)}{\sum R(i)}$ in which k is initially equal to the sample size $(g \cdot N)$.
2. For each relay i, if $R(i)\prime > 1$, $R(i)\prime = 1$ and $k = k - 1$.
3. Repeat the above process until each $R(i)\prime$ is in $[0, 1]$.
4. For each relay i, $R(i)\prime = \frac{R(i)\prime}{g \cdot N}$

Note that when g is set to 0, then no random sampling will be performed, while if g is set to 1, then all relays will have the same $R(i)\prime$ in their weights.

4.2.4 Implementation on Tor

Mapping the IP addresses of the Tor client and Tor relays to their respective AS is necessary before we can compute AS resilience. In order to preserve the anonymity of the Tor client and not reveal its location to outside servers or anyone who can observe it's communications, the client will perform the IP to ASN mapping locally by utilizing the Maxmind ASN database [8], which can be included in the Tor download package. Note that the vanilla Tor client uses the Maxmind GeoIP database for IP to Country mapping, which is already included in the Tor package as well. In addition, the client will use the AS topology database from CAIDA [5] for AS-level path inference in the resilience calculation.

If the Tor client specifies in the torrc configuration file that the AS resilience-aware guard relay selection is preferred, and supplies an α value as the tunable parameter, the following procedure will be invoked:

1. If the Maxmind ASN file and AS topology file have not been downloaded, the Tor client will download the two files from Maxmind and CAIDA, respectively, and save them in the local data directory.
2. The Tor client will perform IP to ASN mapping, and compute the AS resilience $R(i)$ of all candidate relays from the client AS as the source AS.
3. The Tor client will perform random sampling on all ASes containing candidate relays and adjust the resilience value to $R(i)\prime$.
4. The Tor client will compute a weight for each candidate relay using formula $W(i) = \alpha \times R(i)\prime + (1-\alpha) \times B(i)$.
5. The Tor client will proceed with the path selection. The remaining part of the circuit construction process stays the same as it is in Tor.

4.3 Security and Anonymity Evaluation

We evaluate the security and anonymity of the Tor guard relay selection algorithm from three perspectives: (1) reduction in probability of a Tor client being affected by a hijack attack on the Tor guard relay, (2) balancing preferences in relay selection to avoid favoring certain relays significantly more than the others, (3) vulnerability to client fingerprinting attacks, and (4) rigorously assessed anonymity bound for a given Tor client.

4.3.1 Probability of a Tor client being affected by a hijack attack on Tor guard relay

This is the main goal of the new relay selection algorithm. Let $P_{pick}(i)$ denotes the probability that a Tor client will choose relay i using our algorithm, and $P_{deceived}(i)$ denotes the probability that a Tor client will be deceived if relay i is being hijacked. $P_{deceived}(i)$ can be calculated using similar approach in Section 3. The overall probability of a Tor client being affected by a hijack attack on guard relay can then be expressed as:

$$\sum_{i \in \{allguardrelays\}} P_{pick}(i) * P_{deceived}(i)$$

We first evaluate the probability without random sampling for five values of $\alpha = \{0, 0.25, 0.5, 0.75, 1\}$ with 1000 randomly selected ASes as the source AS and Tor consensus data from January 2016. Figure 6 shows the result. Table 1 shows the average percentage of reduction in the probability of being affected by a hijack attack compared to $\alpha = 0$. We can see that $\alpha = 1$ has the lowest probability of being affected by an attack, with an average of 31% reduction.

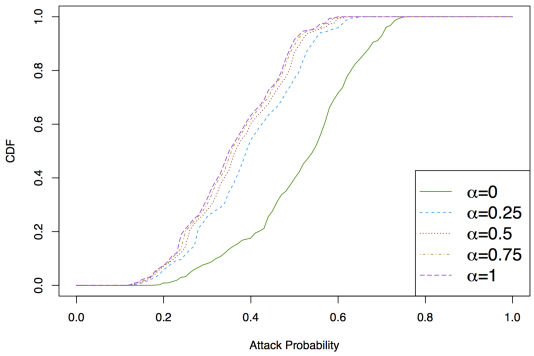


Figure 6: Attack Probability with different α values

We then evaluate it with random sampling of $g = 10\%$. Figure 7 shows the comparison of $\alpha = \{0.25, 0.5\}$ with and without random sampling. Table 1 shows the average percentage of reduction in attack probability with $g = 10\%$ random sampling. We can see that there is only a very slight decrease in the reduction percentage with the random sampling.

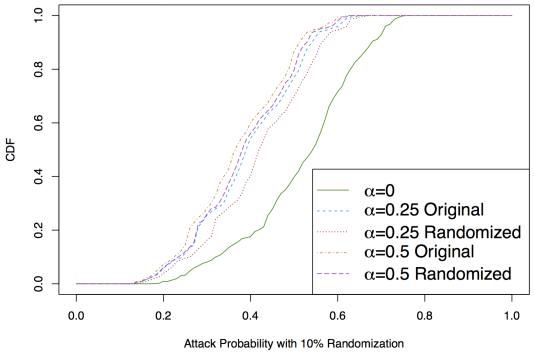


Figure 7: Attack Probability with $g = 10\%$ sampling

α	Without Random Sampling	With $g = 10\%$ sampling
0.25	24%	19%
0.5	28%	25%
0.75	30%	29%
1	31%	31%

Table 1: Percentage of reduction in attack probability compared to $\alpha = 0$

4.3.2 Balancing Relay Selection Preferences

We want to main good balance among the weights/preferences for each candidate relay to avoid strongly favoring certain relays over the others. For instance, for a given Tor

client, if relay i has 90% probability of being chosen while all other relays totals up to 10% probability, then this can create potential security vulnerabilities, let alone performance issues. An adversary may easily target relay i to launch an attack, or even manipulate the selection algorithm to strongly favor a relay run by the adversary. Thus, we want to evaluate the variance in probabilities of a given Tor client picking any particular relay and show that it's well-balanced.

We used Gini coefficient as the variance evaluation metric, as it has been used in previous work [13]. We first evaluate without random sampling, for five values of α : $\{0, 0.25, 0.5, 0.75, 1\}$ with 1000 randomly selected ASes as source AS and Tor consensus data from January 2016. Figure 8 shows the result. The green line to the right is when $\alpha = 0$, which is solely based on bandwidth, resulting in a Gini coefficient of 0.51 for all source ASes. The Gini coefficients for the other four α values which involve resilience-based selection are much lower than bandwidth-only selection, meaning that there is lower skew in probability of selecting any particular relay and thus is more well-balanced.

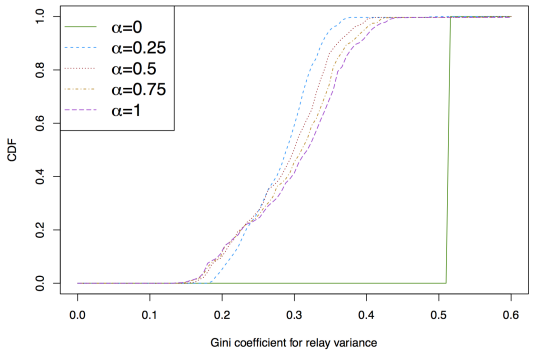


Figure 8: Gini coefficients for variance in relay selection preferences given a Tor client. Lower Gini coefficient value indicates higher variance and less skew.

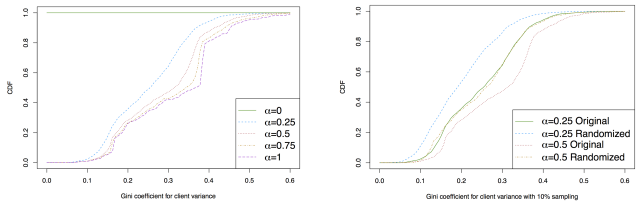
4.3.3 *Vulnerability to client fingerprinting attacks*

This is a potential security tradeoff of the relay selection algorithm. We briefly discussed in Section 4.2.3 about fingerprinting a client location based on its preferences of relays in the long term. For instance, for a given relay, if client a has 70% probability of choosing the relay while client b only has 30% probability, then an adversary can observe the client's choice of relays overtime to infer client information. The resilience component of our relay selection algorithm may be subjective to such fingerprinting attacks, which we will evaluate here.

We used Gini coefficient again as the variance evaluation metric, but evaluating *the variance in the probability of each client choosing a given relay* instead. Figure 9a shows the result for 1000 randomly selected ASes as client AS.

We can see that the bandwidth-only selection in Vanilla Tor ($\alpha = 0$) has perfect Gini coefficient value of 0 for all relays, since given a relay, the probability of it being chosen is the same across all clients. With resilience-based selection, the skews in probabilities are higher. When $\alpha = 1$ (resilience-only), 80% of the relays have coefficients > 0.4. This skew in probability might be exploited by adversaries who can observe the client over a period of time to infer client locations given the differences in relay selection.

We then evaluate it again with $g = 10\%$ random sampling. Note that since $\alpha = \{0.75, 1\}$ does not have a significant advantage in attack resilience over $\alpha = \{0.25, 0.5\}$ in Section 4.3.1, while resulting in higher skew in probabilities, so we focus on evaluating the later here. Figure 9b shows that by performing 10% random sampling, when $\alpha = 0.25$, 80% of the relays have coefficients reduced to < 0.28. Although this may still pose some vulnerability to client fingerprinting, we argue that since Tor clients only select guard relays at bootstrapping time, and would then use the same guard relays over several months or until the relays become unavailable, so fingerprinting the client could not be done in a reasonably short time, especially given the low variance of probabilities (although not perfect) as shown.



(a) Without random sampling (b) With $g = 10\%$ sampling

Figure 9: Gini coefficients for variance in client's probability of choosing a given relay.

4.3.4 *Formal anonymity assessment*

Finally, we evaluate the anonymity of a given Tor client using MATor, a framework for assessing the degree of anonymity in Tor with rigorously proved anonymity bounds [14]. We implemented our new relay selection algorithm into MATor source code, and evaluated it in comparison with vanilla Tor. Note that we picked the top Tor client location AS6128 [23] to evaluate here. We used default configuration of multiplicative factor $\epsilon = 1.3$, ports setting of HTTPS+IRC vs. HTTPS, and 0.5% of total nodes as compromised nodes. We evaluated using Tor consensus files from 2/1/2016 - 2/9/2016 and server descriptor from February 2016. Figure 10 shows the result.

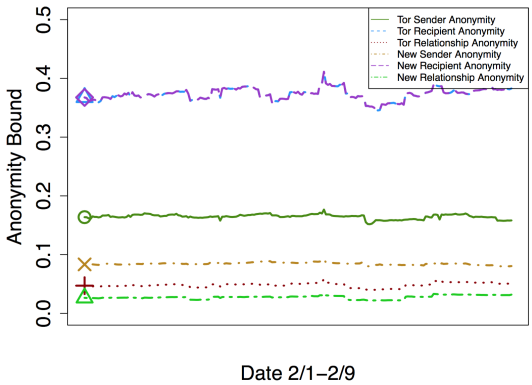


Figure 10: MATor Anonymity Bound 2/1/2016 - 2/9/2016

MATor evaluates three anonymity notions (sender, recipient, and relationship anonymity). The full details of the anonymity definitions are described in [14]. The result shows that our new guard relay selection has tighter anonymity bounds on sender and relationship anonymities compared to current Tor path selection, indicating better

anonymity guarantees. The recipient anonymity remains the same as vanilla Tor, which is expected since we do not alter selection algorithm for exit relays.

4.4 Performance Evaluation

We first evaluate the runtime of the AS resilience calculation given a source AS. We pick 1000 ASes randomly as the source AS, and record how much time it takes each of them to complete the calculation. Most of the source ASes finish within 0.6 second.

Next, we installed our new Tor client on 26 Planetlab nodes located in different ASes, and performed page loads of Alexa top 100 websites. We compared the performance of $\alpha = \{0.25, 0.5\}$ with $g = 10\%$ random sampling against vanilla Tor. Figure 11 shows the page load time. We can see that all three have very similar performance. This can be explained in the following ways. First, we do not restrict the relay selection to a smaller set of relays, but rather *preferring* certain relays over the others; second, the α value was tuned to be $\{0.25, 0.5\}$, which gives sufficient weight to the bandwidth in relay selection; thirdly, the relay selected has a relatively short AS path (and is likely geographically closer);finally, the algorithm is only for guard relay selection, whereas exit relay selection remains the same as purely bandwidth-weighted selection. These are some reasons why the new guard relay selection algorithm does not suffer any performance loss in page load time.

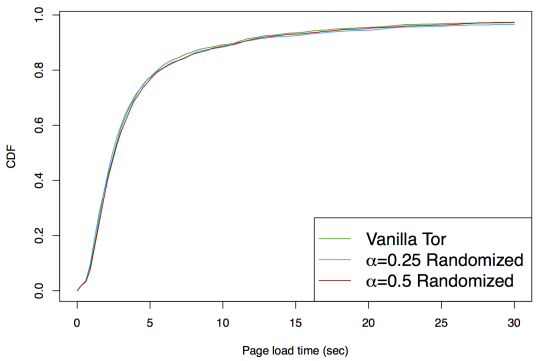


Figure 11: Page load time of Alexa top 100 sites

5. REACTIVE APPROACHES

The new Tor guard relay selection algorithm in Section 4.2 *proactively* mitigates the affect of active BGP hijacks on the Tor client. In this section, we focus on a live monitoring system that *reactively* provides information of an attack on Tor while it's happening, whether or not there are Tor clients being affected.

5.1 A Live Monitoring System for Tor

A possible reactive countermeasure against routing attacks on the Tor network is attack detection. The live monitoring system aims to detect suspicious routing attacks that affect Tor relays, and consists of two parts: BGP monitoring and `traceroute` monitoring.

5.1.1 BGP Monitoring

Our system requires information about current Tor relays. The Tor Project releases up-to-date information about current running relays every hour. Our system automatically fetches this consensus data. We focus on guard relays and exit relays, which reside at the two ends of the communication path and can easily be the target of an adversary. Furthermore, since we focus on AS-level adversaries, it is unnecessary to monitor each individual relay by its IP address. Instead, we monitor the /24 prefixes which contain Tor guard and exit relays. There is no need to monitor a more specific prefix than /24, since generally /24 is the longest prefix accepted in BGP announcement.

The system also uses IP to ASN mappings from Team Cymru [10] to find the ASes that contain Tor relays, as well as map traceroutes (IP-level paths) to AS-level paths. We also use Team Cymru [10] to obtain AS ownership of prefixes. Some prefixes are owned by an organization with multiple AS numbers, so we take this into consideration and store all AS origins of these prefixes. One caveat of using Team Cymru is the potential inaccuracy and incompleteness of the data; the system could easily be augmented to check multiple registries and compare results.

The BGP Monitoring system collects live BGP updates in combination with the latest Tor relay data. We monitor all the /24 prefixes we obtain. We check if any activity related with the prefixes exhibit any anomaly. These anomalies can be detected by applying heuristics, such as the amount of time that a BGP path is used or the frequency that a path is announced; if certain anomalies fall under a threshold for a given heuristic, they should be flagged as potential attacks. This analysis will require saving inactive relay BGP info for some period of time. This system helps:

- Identify suspicious prefix announcements.
- Differentiate between potential attacks and "normal" behavior, such as multiple origin AS conflicts, backup paths, etc [40].

The implementation of the BGP Monitoring system is based on BGP Stream [2]. We analyze the live stream of BGP announcements and withdrawals, focusing just on the prefixes that contain a Tor guard or exit. We monitor the prefixes that are reported through Team Cymru, as well as the /24 that contains each relay; we do this in order to detect sub-prefix hijack attacks – we must monitor longer prefixes in addition to the reported prefix.

Our analysis involves three different detection techniques:

1. Origin AS check. We collect the origin AS in the live BGP update and compare it to the owner AS reported by Team Cymru. If these don't match up, then we flag the update (and prefix) as a suspicious update, otherwise we ignore it.
2. Frequency heuristic. Routing attacks can be characterized by an AS announcing a path once (or extremely rarely) to a prefix that it does not own. The frequency heuristic detects attacks that exhibit this behavior. It measures the frequency of each AS that originates a given prefix; if the frequency is extremely low (below some threshold), then it could be a potential hijack attack.
3. Time heuristic. Many real-world attacks last a relatively short amount of time in comparison to life span

of a prefix [6, 1, 4, 7]. The time heuristic measures the amount of time each path to a prefix is announced for; if the amount of time is extremely small (below some threshold), then there is the possibility of it being a routing attack.

A threshold value for the frequency and time heuristic can be calculated by evaluating old BGP data, and considering known attacks. The lowest threshold value that provides no false negatives should be chosen, and can then be used for future applications of the heuristics. Any false positives can be verified with the `traceroute` monitoring system.

5.1.2 `traceroute` Monitoring

The `traceroute` Monitoring system monitors the data plane of Internet routing, i.e., how packets travel through the Internet in reality. `traceroute` is used as a verification mechanism if the BGP monitoring system flags anomalous behavior. There may be many false positives in detecting hijack attacks due to the nature of BGP. With many false positives, the `traceroute` monitoring system will be used often for verification. We discuss false positives more in Section 5.2.

`traceroute` monitoring retrieves updated Tor relay data hourly from the Tor consensus. Since running large numbers of continuous traceroutes to relay IP addresses may create unnecessary extra traffic on the Tor network, we run traceroutes only when necessary. When there is suspicious activity report by the BGP monitoring system, the `traceroute` monitoring system can be "triggered" to target the prefix in the suspicious announcement to verify or disregard the anomaly. If we detect any anomaly from the BGP monitoring system, we will immediately send traceroutes to the suspicious prefixes to verify whether there is truly a path change happening on the data plane to the Tor relays.

Our system's threat model is one where the attacker is performin active BGP routing attacks on the Tor network. Prior research has discussed a different type of attacker — one that can manipulate responses to `traceroute` [29]. Our system runs traceroutes from approximately 60 different locations; a consensus can then be taken among the traceroutes. An attacker close to the destination of the `traceroute` can potentially modify all responses. Future improvements on the system can include advanced techniques to detect malicious `traceroute` responses.

5.2 Accuracy Evaluation

The BGP monitoring system has been running since Febuary 4th, 2016; for the purposes of our evaluation, we analyze data collected between February 4th, 2016 and February 16th, 2016. It has recorded 2,248 updates that include a Tor guard or exit relay, and 28 announcements that include a Tor relay and have an origin AS that disagrees with Team Cymru's data. After implementing the frequency and time heuristics described in Section 5.1.1, we applied them to the data collected between February 4th and 16th. The origin AS check occurs in real-time as part of the BGP monitoring system. For our analysis, we assume that there were no hijack attacks on the Tor network during the time we collected data. We injected an attack to verify that we do not have false negatives; we modeled the attack after a real-world hijack attack [1]. We injected 3 updates into our data to make it appear that an attack occurred for four minutes on February 9th. The hijacked prefix was 185.13.36.0/22, it was an equal length prefix hijack attack. There were two announcements for the same path, but a different origin; the path was the same length as the true announcement.

5.2.1 Origin AS Check Evaluation

The analysis done within our system (in real-time) is the origin AS check; we compare the origin AS of an announcement with the owner of the prefix (according to Team Cymru). From the data collected, there were 28 BGP updates that were flagged by this check; 4 unique prefixes were announced among these 28 updates. By manually analyzing the anomalous updates, we found that all four prefixes contained sub-prefixes announced by the owner (accourding to Team Cymru) and were in the log of non-suspicious updates. For example, `50.116.48.0/20` was flagged as suspicious because Team Cymru reports AS63949 as the owner, but the AS path in the prefix announcement was:

286 1299 8001

When comparing to the non-suspicious updates, we found prefix `50.116.49.0/24`, a sub-prefix of `50.116.48.0/20`, being announced by AS63949 with the AS path:

286 1299 8001 63949

Because the origin AS of the update in question, AS8001, is already on the path for a sub-prefix, it does not appear to be a BGP hijack attack. We found this was the case for the remaining 3 prefixes that were flagged, and therefore categorized them as non-suspicious updates. This method does not provide false negatives in our evaluation; it would flag our injected attack because the origin AS, AS29386, does not match Team Cymru's record of AS197922.

5.2.2 Frequency Heuristic Evaluation

The frequency heuristic is applied to past data, and can be run consistently and automatically throughout the day. For our evaluation, we applied this heuristic to the updates collected between February 4th and 16th, and included our injected attack. This data included 2,251 updates (3 from the attack) that involved 293 unique prefixes that contain Tor guard or exit relays. We assume that there were no routing attacks on the Tor network during this time period, therefore, our system would have a false positive rate of 0 if we detected 1 attack (the injected attack). Our system will have 0 false negatives if it detects the attack.

The number of false positives is directly related to the threshold value that is set for the heuristic; the higher the threshold value, the more false positives are reported. On the other hand, setting the threshold value too low can cause false negatives (actual attacks that are not detected). Table 2 shows the number of percentage of updates flagged as false positives for varying threshold values. We can see that a threshold value of .3 or lower will result in 0 false positives and 0 false negatives.

The results for the frequency heuristic highlight an important characteristic about the Tor network: most prefixes are announced by a single AS in all updates, causing the frequency of the (prefix, origin AS) pair to 1.0. Only two prefixes are ever announced by more than one AS in our dataset. This allows us to conclude that this heuristic is accurate and precise for use in monitoring the Tor BGP updates.

Threshold	False Positive Percentage (Frequency)	False Positive Percentage (Time)	Detects Attack (Frequency)	Detects Attack (Time)
.1	0.00%	0.00%	Yes	Yes
.2	0.00%	0.00%	Yes	Yes
.3	0.00%	0.00%	Yes	Yes
.4	0.34%	0.00%	Yes	Yes
.5	0.34%	0.34%	Yes	Yes
.6	0.34%	0.68%	Yes	Yes
.7	0.68%	0.68%	Yes	Yes
.8	0.68%	0.68%	Yes	Yes
.9	0.68%	0.68%	Yes	Yes

Table 2: The false positive rates for different thresholds used in the heuristics, as well as false negative rates.

5.2.3 *Time Heuristic*

Using the same data that the frequency heuristic was evaluated on, we found the threshold values and their corresponding false positive rates for the time heuristic (as shown in Table 2). According to these values, any threshold below .4 would provide optimal results, but it might be wiser to select a higher threshold to minimize the false negatives (this would be selecting .4 instead of .1 in this scenario).

This time heuristic results in similar percentages to the frequency heuristic, showing that it is also well-suited for monitoring the Tor network.

6. DISCUSSION

Accuracy of AS path inference. Part of our AS resilience calculation involves AS-level path inference from the network topology. Recent work has shown that path inferences using local preference and shortest path may not be accurate [24], and thus path selection algorithms such as [34] that rely on the accuracy of AS path inferences could be affected. However, we only use path inference as an indicator of network connectivity to calculate aggregated resilience instead of predicting any *precise* routes. This is also the reason why even after we "perturb" the AS resiliencies by doing unequal probability sampling, we still have relatively good results in reducing attack probability as compared to without the sampling. Thus, our resilience calculation is robust to a certain degree of AS path inference inaccuracy and/or AS path churn.

BGP Security. There is a large body of research with the goal of defending against and detecting prefix hijacks and interceptions. These include both defensive and detection tools [26, 21, 31, 39, 41, 33, 38] as well as mechanisms such as PGBGP, which allow network administrators more time to determine if an attack is happening before using new routes [25]. Our work is orthogonal, as it targets the Tor network specifically.

7. CONCLUSION

In this work, we have presented countermeasures to safeguard Tor against active BGP routing attacks.

First, we evaluated the Tor network for its current state of resilience to hijack and interception attacks. We observed that some ASes with high Tor bandwidth have relatively low resilience. Next, we presented proactive countermeasures that include a new Tor guard relay selection algorithm. The algorithm successfully reduces the probability of a Tor client being affected by a prefix hijack attack, and at the same time protects the security and anonymity of the Tor client. Finally, we presented a live monitoring system that can detect routing anomalies in Tor relays in real time.

Overall, our work is the first work on mitigating active routing attacks on Tor.

8. ACKNOWLEDGMENTS

9. REFERENCES

[1] Bgp hijack incident by syrian telecommunications establishment. http://www.bgpmon.net/bgp-hijack-incident-by-syrian-telecommunications-establishment/.

[2] Bgp stream. http://bgpstream.caida.org/.

[3] Caida internet topology map. https://www.caida.org/research/topology/.

[4] Hijack event today by indosat. http://www.bgpmon.net/hijack-event-today-by-indosat/.

[5] The ipv4 routed /24 topology dataset. http://www.caida.org/data/active/ipv4_routed_24_topology_dataset.xml.

[6] Large scale bgp hijack out of india. http://www.bgpmon.net/large-scale-bgp-hijack-out-of-india/.

[7] Massive route leak causes internet slowdown. http://www.bgpmon.net/massive-route-leak-cause-internet-slowdown/.

[8] Maxmind geolite asn database. http://dev.maxmind.com/geoip/legacy/geolite/.

[9] Route views project. http://www.routeviews.org/.

[10] Team-cymru. http://www.team-cymru.org/.

[11] Tor consensus. https://collector.torproject.org/recent/relay-descriptors/consensuses/.

[12] Tor metrics. https://metrics.torproject.org/.

[13] M. Akhoondi, C. Yu, and H. V. Madhyastha. Lastor: A low-latency as-aware tor client. In *Security and Privacy (SP), 2012 IEEE Symposium on*, pages 476–490. IEEE, 2012.

[14] M. Backes, A. Kate, S. Meiser, and E. Mohammadi. (nothing else) mator (s): Monitoring the anonymity of tor's path selection. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, pages 513–524. ACM, 2014.

[15] H. Ballani, P. Francis, and X. Zhang. A study of prefix hijacking and interception in the internet. In *ACM SIGCOMM Computer Communication Review*, volume 37, pages 265–276. ACM, 2007.

[16] R. Dingledine, N. Mathewson, and P. Syverson. Tor: The second-generation onion router. Technical report, DTIC Document, 2004.

[17] M. Edman and P. Syverson. As-awareness in tor path selection. In *Proceedings of the 16th ACM conference on Computer and communications security*, pages 380–389. ACM, 2009.

[18] N. Feamster and R. Dingledine. Location diversity in anonymity networks. In *Proceedings of the 2004 ACM workshop on Privacy in the electronic society*, pages 66–76. ACM, 2004.

[19] L. Gao. On inferring autonomous system relationships in the internet. *IEEE/ACM Transactions on Networking (ToN)*, 9(6):733–745, 2001.

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

[21] X. Hu and Z. M. Mao. Accurate real-time identification of ip prefix hijacking. In *Security and Privacy, 2007. SP'07. IEEE Symposium on*, pages 3–17. IEEE, 2007.

[22] A. Johnson, C. Wacek, R. Jansen, M. Sherr, and P. Syverson. Users get routed: Traffic correlation on tor by realistic adversaries. In *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, pages 337–348. ACM, 2013.

[23] J. Juen. Protecting anonymity in the presence of autonomous system and internet exchange level adversaries. 2012.

[24] J. Juen, A. Johnson, A. Das, N. Borisov, and M. Caesar. Defending tor from network adversaries: A case study of network path prediction. *Proceedings on Privacy Enhancing Technologies*, 2015(2):171–187, 2015.

[25] J. Karlin, S. Forrest, and J. Rexford. Pretty good bgp: Improving bgp by cautiously adopting routes. In *Network Protocols, 2006. ICNP'06. Proceedings of the 2006 14th IEEE International Conference on*, pages 290–299. IEEE, 2006.

[26] M. Lad, D. Massey, D. Pei, Y. Wu, B. Zhang, and L. Zhang. Phas: A prefix hijack alert system. In *Usenix Security*, 2006.

[27] M. Lad, R. Oliveira, B. Zhang, and L. Zhang. Understanding resiliency of internet topology against prefix hijack attacks. In *Dependable Systems and Networks, 2007. DSN'07. 37th Annual IEEE/IFIP International Conference on*, pages 368–377. IEEE, 2007.

[28] S. J. Murdoch and P. Zieliński. Sampled traffic analysis by internet-exchange-level adversaries. In *Privacy Enhancing Technologies*, pages 167–183. Springer, 2007.

[29] V. N. Padmanabhan and D. R. Simon. Secure traceroute to detect faulty or malicious routing. *ACM SIGCOMM Computer Communication Review*, 33(1):77–82, 2003.

[30] A. Pilosov and T. Kapela. Stealing the internet: An internet-scale man in the middle attack, 2008.

[31] X. Shi, Y. Xiang, Z. Wang, X. Yin, and J. Wu. Detecting prefix hijackings in the internet with argus. In *Proceedings of the 2012 ACM conference on Internet measurement conference*, pages 15–28. ACM, 2012.

[32] V. Shmatikov and M.-H. Wang. Timing analysis in low-latency mix networks: Attacks and defenses. In *Computer Security–ESORICS 2006*, pages 18–33. Springer, 2006.

[33] K. Sriram, O. Borchert, O. Kim, P. Gleichmann, and D. Montgomery. A comparative analysis of bgp anomaly detection and robustness algorithms. In *Conference For Homeland Security, 2009. CATCH'09. Cybersecurity Applications & Technology*, pages 25–38. IEEE, 2009.

[34] O. Starov, R. Nithyanand, A. Zair, P. Gill, and M. Schapira. Measuring and mitigating as-level adversaries against tor. *arXiv preprint arXiv:1505.05173*, 2015.

[35] Y. Sun, A. Edmundson, L. Vanbever, O. Li, J. Rexford, M. Chiang, and P. Mittal. Raptor: routing attacks on privacy in tor. *arXiv preprint arXiv:1503.03940*, 2015.

[36] P. Syverson, G. Tsudik, M. Reed, and C. Landwehr. Towards an analysis of onion routing security. In *Designing Privacy Enhancing Technologies*, pages 96–114. Springer, 2001.

[37] Y. Tillé. An elimination procedure for unequal probability sampling without replacement. *Biometrika*, 83(1):238–241, 1996.

[38] Z. Zhang, Y. Zhang, Y. C. Hu, and Z. M. Mao. Practical defenses against bgp prefix hijacking. In *Proceedings of the 2007 ACM CoNEXT conference*, page 3. ACM, 2007.

[39] Z. Zhang, Y. Zhang, Y. C. Hu, Z. M. Mao, and R. Bush. Ispy: detecting ip prefix hijacking on my own. In *ACM SIGCOMM Computer Communication Review*, volume 38, pages 327–338. ACM, 2008.

[40] X. Zhao, D. Pei, L. Wang, D. Massey, A. Mankin, S. F. Wu, and L. Zhang. An analysis of bgp multiple origin as (moas) conflicts. In *Proceedings of the 1st ACM SIGCOMM Workshop on Internet Measurement*, pages 31–35. ACM, 2001.

[41] C. Zheng, L. Ji, D. Pei, J. Wang, and P. Francis. A light-weight distributed scheme for detecting ip prefix hijacks in real-time. In *ACM SIGCOMM Computer Communication Review*, volume 37, pages 277–288. ACM, 2007.

APPENDIX