Ph.D. Thesis Proposal

Abusing Privacy Infrastructures: Analysis and Mitigations

Amirali Sanatinia

College of Computer and Information Science

Northeastern University

Ph.D. Committee

Guevara Noubir	Advisor, Northeastern University
Agnes Chan	Northeastern University
Alina Oprea	Northeastern University
Erik-Oliver Blass	External member, Airbus Group Innovations
Aziz Mohaisen	External member, SUNY Buffalo

July 2017

Abstract

In the last two decades, advances in privacy-enhancing technologies, including cryptographic mechanisms, standardized security protocols, and infrastructure, significantly improved the privacy of users. Tor, a byproduct of those primitives, emerged as a practical solution to protecting the privacy of citizens against censorship and tracking. At the same time, Tor's success encouraged illegal activities, including sophisticated botnets, ransomware, and a marketplace for drugs and contraband. Tor and other privacy infrastructure had a significant impact on society protecting users and are of significant importance to guarantee privacy. When such infrastructure is misused for launching malicious activities, such right becomes at risk. The goal of this thesis is to pursue the highest societal impact by providing mechanisms that will make detection of abusers of this valuable infrastructure possible.

In this thesis, we investigate the abuse of privacy infrastructures from three different perspectives. First we look at the next generation of resilient botnets that rely on Tor for their malicious activity. In the second part, we expose malicious snooping actors inside the Tor network that are integral to the functioning of the hidden services and the dark web. In the third and last part, we propose a novel privacy preserving data collection and analysis framework to study the longevity of hidden services.

Botnets rose to be a major tool for cyber-crime and their developers proved to be highly resourceful. We contend that the next waves of botnets will extensively attempt to subvert privacy infrastructures and cryptographic mechanisms. In the first part of this thesis, we will preemptively investigate the design and mitigation of such botnets, (i.e., OnionBots) that can achieve a low diameter and a low degree and be robust to partitioning under node deletions.

Tor's security relies on the fact that a substantial number of its nodes do not misbehave. In the second part of this thesis we expose a category of misbehaving Tor relays (HSDirs), that are integral to the functioning of the hidden services and the dark web. The HSDirs act as the DNS directory for the dark web. Because of their nature, detecting their malicious intent and behavior is much harder. We introduce, the concept of honey onions (honions), a framework to detect misbehaving Tor relays with HSDir capability. Furthermore, we develop an approximation algorithm to this specific problem as well as an Integer Linear Program (ILP) formulation.

Very little is known about the lifespan of hidden services. Such knowledge provides manifold benefits, such as the detection of malicious and benign domains. However, to avoid disrupting Tor and its security and privacy services, such study needs to be carried out in a privacy preserving manner. Furthermore, The distributed nature of Tor and hidden services makes such study non-trivial and introduces challenges that need to be addressed. We devise novel protocols and algorithms to draw conclusions based on the data that is distributively collected from the network, while protecting the privacy and security properties of the Tor infrastructure.

Contents

1 Introduction

Over the last decade, Tor emerged as a popular tool and infrastructure that protects users' anonymity and defends against tracking and censorship. It is used today by millions of ordinary users to protect their privacy against corporations and governmental agencies, but also by activists, journalists, businesses, law enforcements and military [1].

The success and popularity of Tor makes it a prime target for adversaries as indicated by recent revelations [4]. Despite its careful design that significantly improved users privacy against typical adversaries, Tor remains a practical system with a variety of limitations and design vulnerabilities, some of which were indeed exploited in the past [6, 12]. Due to the perceived security that Tor provides, its popularity, and potential implication on its users, it is important that the research community continues analyzing and strengthening its security.

This is specially important since users typically have a poor understanding of the privacy protection that Tor really provides as evidenced by past events. For instance, in a highly publicized case, security researchers collected thousands of sensitive e-mails and passwords from the embassies of countries including India and Russia [2]. These embassies used Tor believing it provides end-to-end encryption, sending sensitive un-encrypted data through malicious exit nodes. Other research revealed that many users run BitTorrent over Tor, which is insecure and resulted in deanonymization [22]. Finally, recent incidents revealed that the Tor network is continuously being attacked by a variety of organizations from universities to governmental agencies, with difficult to predict ramifications [6, 5]. Even more recently, the still unexplained sudden surge in the number of hidden services (`.onion`), more than tripling their number before returning to relatively smaller numbers (See Figure 1), indicates that the Tor network is not well understood, in part due to its peer-to-peer nature, the privacy services it provides that limit measurements, and the attacks that it attracts [29].

Tor and other privacy infrastructure had a significant impact on society protecting users and are of significant importance to guarantee privacy. When such infrastructure is misused for launching malicious activities, such right becomes at risk.

1.1 Focus of this Work

In this thesis, we investigate the abuse of privacy infrastructures. In the first part, we look at the next generation resilient botnets (OnionBots). We contend that the next waves of botnets will extensively attempt to subvert privacy infrastructure and cryptographic mechanisms as indicated by recent events [3, 7, 13]. We preemptively investigate the design and mitigation of such botnets, re-

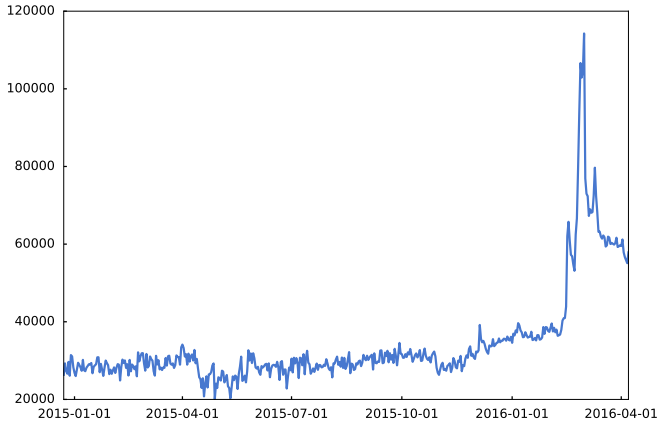


Figure 1: Recent unexplained surge in the number of Hidden Services. The number of hidden services (`.onion`) suddenly tripled, before settling at twice the number before the surge.

ferred to as `OnionBots` [29]. Chief among privacy infrastructure, Tor protects users' identity over the Internet and allows one to host Internet servers without revealing their location using Tor's Hidden Services. OnionBots can abuse these privacy features to their benefit. Evidence of these predictions can be found in the malicious use of hidden services for hosting the infamous silk road [13], the CryptoLocker ransomware's C&C server [7], and instances of the Zeus [3] botnet.

In the second part of this thesis, we expose malicious snooping actors inside the Tor network that are integral to the functioning of the hidden services and the dark web. Tor's security, by design, relies on the fact that a substantial number of its relays should not be malicious. It is, however, difficult to assess to what extent this condition holds true. The fact that many attacks are passive, makes it even harder to assess the significance of this threat. The HSDirs act as the DNS directory for the dark web. Because of their nature, detecting their malicious intent and behavior is much harder. We introduce, the concept of honey onions (honions) [30], a framework to detect misbehaving Tor relays with HSDir capability. By setting up and deploying a large scale honion over Tor for more than 72 days, we are able to obtain lower bounds on misbehavior among HSDirs. We propose algorithms to both estimate the number of snooping HSDirs and identify them, using optimization and feasibility techniques. Our experimental results indicate that during the period of our work at least 110 such nodes were snooping information about hidden services they host. We reveal that more than half of them were hosted on cloud infrastructure and delayed the use of the learned information to prevent easy traceback. Furthermore, we provide the most likely geolocation map of the identified snooping

Tor HSDirs.

In the third part of the thesis, we study the longevity of hidden services. This work allows researchers to gain more insight into the nature of dark web. There is no centralized authority in Tor, therefore the collection of such data is non-trivial and introduces new challenges. We will design and implement a data collection and analysis framework to study the longevity of hidden services in a privacy preserving manner. We will put safeguards in place, both in theory and practice, to ensure the security and privacy of hidden services.

1.2 Related Work

Kartaltepe et al. [20], investigated a new generation of botnets that use online social networks, such as Twitter as their C&C infrastructure. An instance of such malware, Naz, gets its commands by making GET requests to the RSS feed of its botmaster on Twitter. The tweets contain the base64 encoding of shortened URLs (e.g., bit.ly) that redirect the bot to the compressed malicious payload. Stranerger et al. [33], introduce a botnet communication protocol, called Overbot. Their design leverages Kademila peer-to-peer protocol, a distributed hash table (DHT) used by many peer-to-peer applications. They investigate the possibilities of using the existing protocol to design stealth C&C channels. The bot uses the 160-bit hash values in a search request to announce it's sequence number, which is encrypted with the public key of the botmaster. Later, this sequence number is used to send commands to the bot. Nappa et al. [24], propose a parasitic botnet protocol that exploits Skype's overlay network. Skype provides a widespread resilient network with a large install base for C&C infrastructure. The communications between the master and the bots are encrypted using adhoc schemes. The protocol broadcasts messages to all peers in the network, similar to the algorithms used in Gnutella. Once each peer receives a new message it passes it to all of its neighbors.

Previous research studied malicious traffic and misbehaving relays in the Tor network, however it was mostly limited to the traffic carrying relays and exit nodes [21, 19, 36, 10]. Winter et al. [39] expose malicious exit nodes by developing two exit relay scanners, one for credential sniffing and one for active man-in-the-middle (MITM) attacks. The authors discovered 65 malicious or mis-configured exit relays participating in different attacks. They proposed an extension to the Tor browser to thwart MITM attacks by such malicious exit nodes. In another work [38], the authors propose *sybilhunter*, a technique to detect Sybil relays based on their characteristics such as configuration, fingerprint, and uptime sequence using the consensus document. Ling et al. [23] present TorWard, a systems for the discovery and the systematic study of malicious traffic over Tor. The system allows investigations to be carried out in sensitive environment such as a university campus,

and allows to avoid legal and administrative complaints. The authors investigate the performance and effectiveness of TorWard by performing experiments and showing that approximately 10% of Tor traffic can trigger IDS alert.

Other research looked at the content and popularity of hidden services and the leakage of .onion address. Biryukovhs et al. [11] collected 39824 hidden services descriptors and scanned them for open ports. The author findings reveal that the majority of hidden services belong to botnets, followed by adult content and drug markets. Another study [37], measures the leakage of onion addresses at the root DNS servers (A and J), and provides the popularity of different hidden services categories based on the leaked requests.

Secure and private computation over data is not a new concept. Researchers have been working on this field for a long time. The holy grail is the fully homomorphic encryption [16], which allows computation over encrypted data. However, it is very slow and impractical for any real world problem. Very recently hardware innovations such as Intel SGX [8] tried to provide some solutions. However, there are other concerns and issues with such approach [27]. Previous studies have suggested schemes for privacy preserving data collection and analysis [18, 15]. We will build upon previous work to design and implement a data collection and analysis framework to study the longevity of hidden services in a privacy preserving manner.

1.3 Proposal Overview

First, in Section 2 we summarize some key mechanisms of Tor. In particular, we focus on the architecture of hidden services. In Section 3, we look at the the OnionBots, what we believe will be the next generation of resilient, stealthy botnets. Then, in Section 4, we introduce honey onions (HOnions), a framework to expose when a Tor relay with Hidden Service Directory (HSDir) capability has been modified to snoop into the hidden services that it currently hosts. Lastly, in Section 5 we look at privacy preserving longevity study of hidden services, and the proposed research plan and timeline is outlined.

2 Privacy Infrastructure: Tor

Tor [14] is an anonymity network that allows users to circumvent censorship and protect their privacy, activities and location from government agencies and corporations. Tor also provides anonymity for the service provided with hidden services, which enables them to protect their location (IP address), yet allowing users to connect to them. Hidden services have been used to protect both legitimate and legal services for privacy conscious users (e.g., Facebook), and for illicit pur-

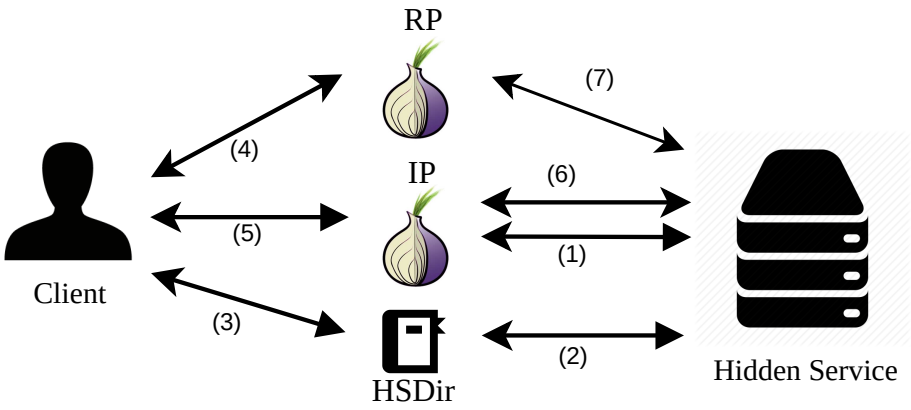


Figure 2: Tor hidden service architecture and connection setup.

poses such as drug and contraband market [13], and extortion. This attracts attacks from a variety of actors. We first summarize some key mechanisms of Tor. In particular, we focus on the architecture of hidden services, both from the client and the service provider perspective.

The Tor hidden services architecture is composed of the following components:

- *Server*, that runs a service (e.g., a web server).
- *Client*, that wishes to access the server.
- *Introduction Points (IP)*, a set of Tor relays, chosen by the hidden service, that forward the initial messages between the server and the client's Rendezvous Point.
- *Rendezvous Point (RP)*, a Tor relay randomly chosen by the client that forwards the data between the client and the hidden service.
- *Hidden Service Directories (HSDir)*, a set of Tor relays chosen by the server to store its descriptors.

Server. To enable access to a server, the service provider, generates an RSA key pair. Then he calculates the SHA-1 digest of the generated public key, known as the `Identifier` of the hidden service. The `.onion` hostname is the base-32 encoding of the identifier. To connect to a hidden service, the aforementioned identifier needs to be communicated to the clients through an external out-of-band channel. As depicted in Figure 2, the hidden service, chooses a set of relays, called Introduction Points (IP), and establishes Tor circuits with them (step 1). After setting up the circuits, the hidden service calculates two service descriptors to determine which relays are the responsible HSDirs, using the below formula and uploads the descriptors to them (step 2).

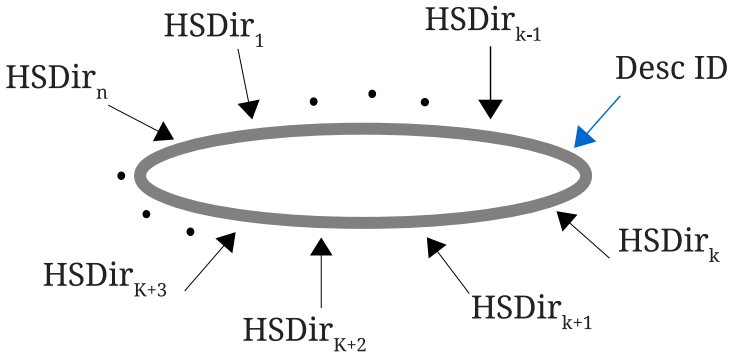


Figure 3: Fingerprints circle or Hidden Service Directories (HSDir) and placement of a hidden service descriptor.

$$
\begin{aligned}
\texttt{descriptor-id} &= \texttt{H(Identifier||secret-id-part)} \\
\texttt{secret-id-part} &= \texttt{H(time-period||descriptor-cookie||replica)} \\
\texttt{time-period} &= (\texttt{current-time} + \texttt{permanent-id-byte} * 86400/256)/86400
\end{aligned}
$$

In the above equations, `H` is the SHA-1 hash digest. `Identifier` is the 80 bit truncated SHA-1 digest of the public key of the hidden service. `Descriptor-cookie` is an optional 128 bit field which could be used for authorization. The hidden services periodically change their HSDir. The `time-period` determines when each descriptor expires and the hidden services need to calculate the new descriptors and upload them to the new corresponding HSDirs. To prevent the descriptors from changing all at the same time, the `permanent-id-byte` is also included in the calculations. The `Replica` index, takes values of 0 or 1, and results in two descriptors. Each descriptor is uploaded to 3 consecutive HSDirs, a total of 6. Consider that the circle of HSDirs is sorted based on their fingerprint (SHA-1 hash of their public key) as shown in Figure 3. If the descriptor of a hidden service falls between the fingerprint of HSDir_{k-1} and HSDir_k, then it will be stored on HSDir_k, HSDir_{k+1} and HSDir_{k+2}.

Client. When a client wishes to contact a hidden service, he first needs to compute the `descriptor-id` using the above formula, and contact the corresponding HSDirs (step 3). To communicate with a connection with the hidden services, the client first needs to choose a set of random relays as his

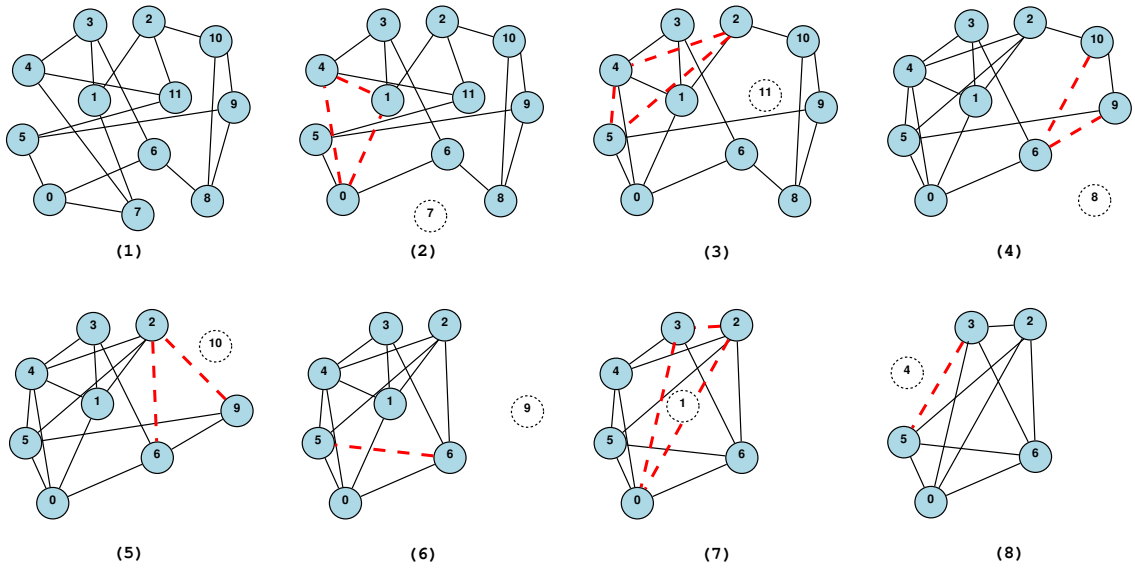
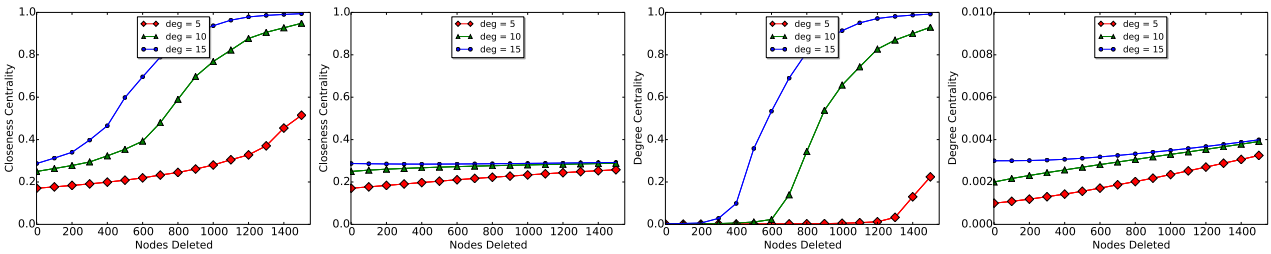


Figure 4: Node removal and the self repairing process in a 3-regular graph with 12 nodes. The dashed red lines, indicate the newly established links between the nodes.

Rendezvous Point (RP), and establish a circuit with them (step 4). Then he contacts the hidden service's IPs to indicate his desire to contact the hidden service, and announcing his RPs (step 5). In the next stage, the IP will forward this information to the hidden services (step 6). At last, the hidden service establishes a circuit to the RPs, and the two can start communicating.

3 OnionBots: Next Generation of Botnets

Another potential source of abuse are Botnets. They have evaded mitigation and takeovers by adopting an increasing sophisticated strategies. Privacy infrastructures, such as Tor has opened new possibilities of abuse by malicious users. Recent statistics about hidden services clearly indicate changes in their popularity and use. We envision a next generation of cryptographic, resilient and stealthy botnets (i.e., OnionBots) that abuse privacy infrastructures for cyber attacks, by completely decoupling their operation from the infected host IP address. Furthermore, they rely on distributed self-healing network formation that is simple to implement, yet achieves a low diameter and a low degree, and is robust to partitioning attacks. As a result, the current detection and mitigation strategies would be inadequate against them. We devise a mitigation mechanism that uses OnionBots' very own capabilities to neutralize them. In light of the potential of such botnets, we believe that the research community should proactively develop detection and mitigation methods to thwart OnionBots, potentially making adjustments to privacy infrastructure.



(a) without pruning (b) with pruning (c) without pruning (d) with pruning

Figure 5: The average closeness centrality, and degree centrality of nodes in a *k-regular* graph, ($k = 5, 10, 15$) with 5000 nodes after 30% node deletions, with and without pruning.

3.1 Current Botnets Topolgy

Currently, bots are monitored and controlled by a botmaster, who issues commands. The transmission of theses commands, which are known as C&C messages, can be centralized, peer-to-peer or hybrid. In the centralized architecture the bots contact the C&C servers to receive instructions from the botmaster. In this construction the message propagation speed and convergence is faster, compared to the other architectures. It is easy to implement, maintain and monitor. However, it is limited by a single point of failure. To evade detection and mitigation, attackers developed more sophisticated techniques to dynamically change the C&C servers, such as: Domain Generation Algorithm (DGA) and fast-fluxing (single flux, double flux). The next step in the arms race between attackers and defenders was moving from a centralized scheme to a peer-to-peer C&C. Storm [17], Nugache [35], Walowdac [34] and Gameover Zeus [9] are examples of such botnets. Some of these botnets use an already existing peer-to-peer protocol, while others use customized protocols. Very recently the use of Tor received more attention from malware and botnet authors. For example, the new 64-bit Zeus employs Tor anonymity network in its botnet infrastructure [3]. It creates a Tor hidden service on the infected host and the C&C can reach these infected hosts using their unique `.onion` address through Tor.

3.2 OnionBot Communication Graph

OnionBots form a peer-to-peer, self-healing network that maintains a low degree and a low diameter with other bots to relay messages. The already existing peer-to-peer networks are generic in terms of their operations. Therefore, their design and resiliency is based on different assumptions and requirements. We propose a Dynamic Distributed Self Repairing (DDSR) graph construction that is simple, stealthy and resilient.

The DDSR construct is inspired by the knowledge of Neighbors-of-Neighbor. Where each node

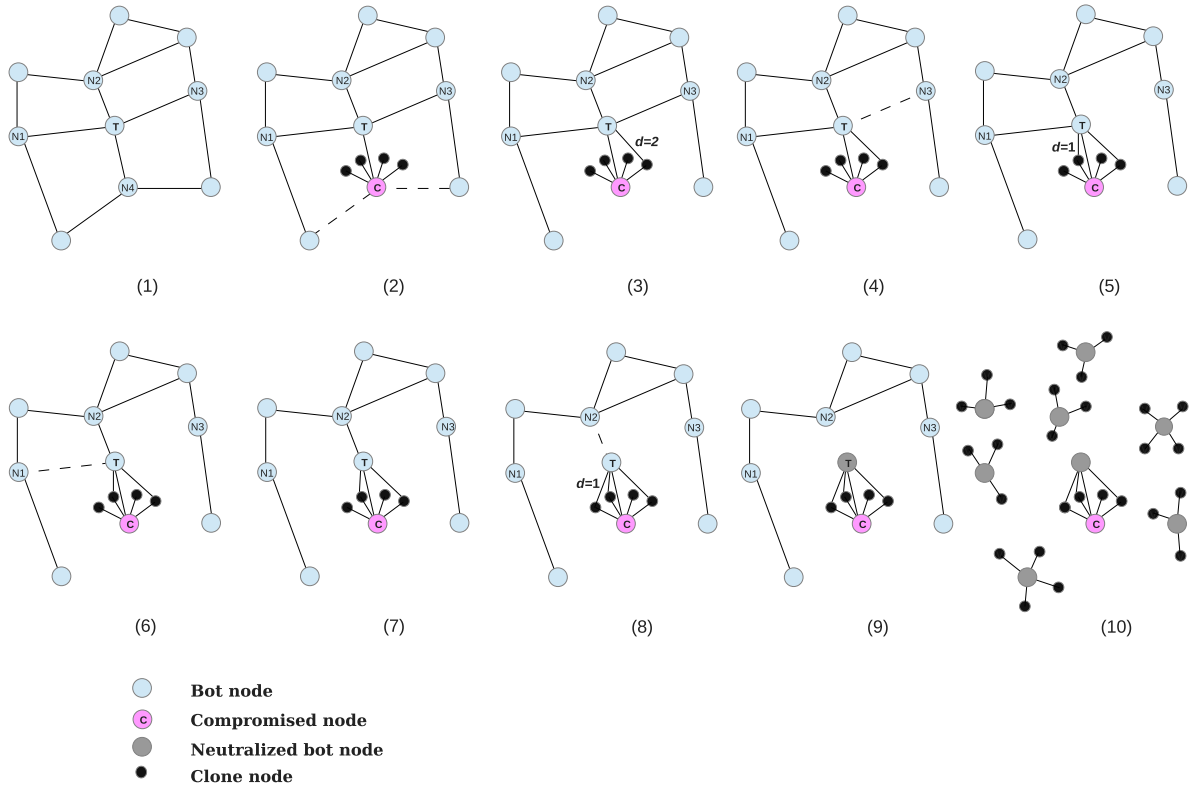


Figure 6: SOAP: node T is under attack by the compromised node C and its clones. In each step one of the clones initiates the peering process with the T, until it is contained. After several iterations, the network is partitioned, and the botnet is neutralized.

has the knowledge about its immediate neighbors. Consider graph G with n nodes (V), where each node $u_i \in V$, $0 \leq i < n$, is connected to a set of nodes. The neighbors of u_i, are denoted as $N(u_i)$. Furthermore, u_i has the knowledge of nodes that are connected to $N(u_i)$. Meaning that each node also knows the identity of its neighbor's neighbors. In the context of our work the identity is the `.onion` address. Having this information enables the botnet to repair its graph formation and maintain its connectivity in a distributed setting. When a node u_i is deleted, each pair of its neighbors u_j, u_k will form an edge (u_j, u_k) if $(u_j, u_k) \notin E$, where E is the set of existing edges. Figure 4 depicts the node deletion and graph healing process. The aforementioned basic DDSR graph does not deal with the growth in the connectivity degree of each node, denoted by $d(u)$; after multiple deletions the degree of some nodes can increase significantly. Such increase is not desirable for the resiliency and the stealthy operation of the botnet. Therefore we introduce the concept of pruning in node deletion to address this challenge. When a node is removed, each neighboring node of the deleted node (u_i), deletes the highest degree node from its peer list. If there is more than one such candidate, it randomly selects one among those for deletion, until its degree is in the desired range. Figure 5 depicts the impact of pruning on the degree and closeness centrality of the botnet's connectivity network.

In the proposed OnionBot, nodes forget the `.onion` address of the pruned nodes. Additionally, to avoid discovery, mapping and further blocking, each bot can periodically change his `.onion` address and announce the new address to his current peer list. The new `.onion` address is generated based on a secret key and time. This periodic change is possible because of the decoupling between IP address and the bots, which is provided by Tor.

3.3 Mitigation of OnionBots

Many of the current detection and mitigation mechanisms are IP-based, and rely on the network traffic patterns or DNS queries to distinguish legitimate traffic from malicious traffic. However, current solutions do not work with OnionBots, since the Tor traffic is encrypted, non IP-based, and there are no conventional DNS queries. Malicious traffic detection mechanisms in Tor [23] are the first step in the mitigation. However, we need to adapt our detection and mitigation methods to address the evasion mechanism of OnionBots. We devised a mitigation mechanism that uses OnionBots' very own capabilities (e.g., the decoupling of IP address and the host) against them. Figure 6 depicts the soaping attack in different steps. Node T is the target of the soaping attack, nodes N_i, are its neighboring bot nodes, and nodes C are the adversary, and his clones, which are represented with small black circles. In step 1, the botnet is operating normally, and none of T's neighbors are compromised. In step 2, one of its peers, N_4, is compromised. Then, N_4 (now depicted as C), makes a set of clones (the small black circles). In step 3, a subset of C's clones, start the peering process with T, and declare their degree to be a small random number, which

changes to avoid detection (e.g., d=2). Doing so increases the chances of being accepted as a new peer, and replacing an existing peer of T. In step 4, T forgets about one of its neighbors with the highest degree, N_3, and peers with one the clones. The clones repeat this process until T has no more benign neighbors (steps 5-8). As a result, T is surrounded by clones and is contained (step 9).

4 HOnions: Exposing Snooping Tor HSDir Relays

Tor's security and anonymity is based on the assumption that the large majority of the its relays are honest and do not misbehave. Particularly the privacy of the hidden services is dependent on the honest operation of Hidden Services Directories (HSDirs). In this work we introduce, the concept of honey onions (*HOnions*), a framework to detect and identify misbehaving and snooping HSDirs. After the deployment of our system and based on our experimental results during the period of 72 days, we detect and identify at least 110 such snooping relays. Furthermore, we reveal that more than half of them were hosted on cloud infrastructure and delayed the use of the learned information to prevent easy traceback.

4.1 Approach and Deployment of Honions

We introduce the concept of *honey onions* (`honions`), to expose when a Tor relay with HSDir capability has been modified to snoop into the hidden services that it currently hosts. In order to automate the process of generating and deploying honions in a way that they cover a significant fraction of HSDirs, we developed several tools. A key constraint in this process was to minimize the number of deployed honions. This derives primarily from our desire to not impact the Tor statistics about hidden services (specially given the recent surge anomaly). By considering the number of HSDirs (approximately 3000), we could infer that we need to generate around 1500 honions to cover all HSDirs with 0.95 probability. We used 1500 honions per batch (daily, weekly, or monthly) and could verify that 95% of the HSDirs were systematically covered.

HOnion back end servers: Each honion corresponds to a process that is running locally. The server behind hidden services, should not be running on a public IP address, to avoid de-anonymization [37]. We also log all the requests that are made to the server programs and the time of each visit. Recording the content of the requests allows us to investigate the snoopers' behavior and intent.

HOnions generation and deployment schedule: To keep the total number of honions small, we decided on three schedules for their generation and placement, *daily*, *weekly*, and *monthly*. The three schedules allow us to detect the malicious HSDirs who visit the honions shortly (less than 24 hours) after hosting them. Since the HSDirs for hidden services change periodically, more sophisticated

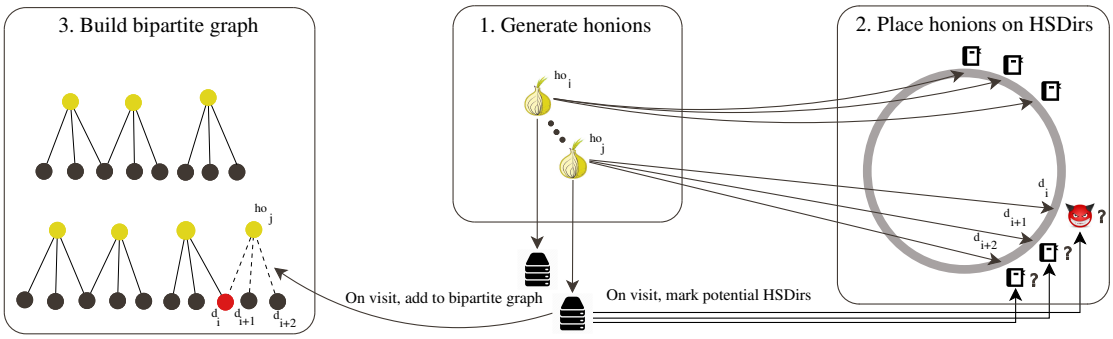


Figure 7: Flow diagram of the honion system.

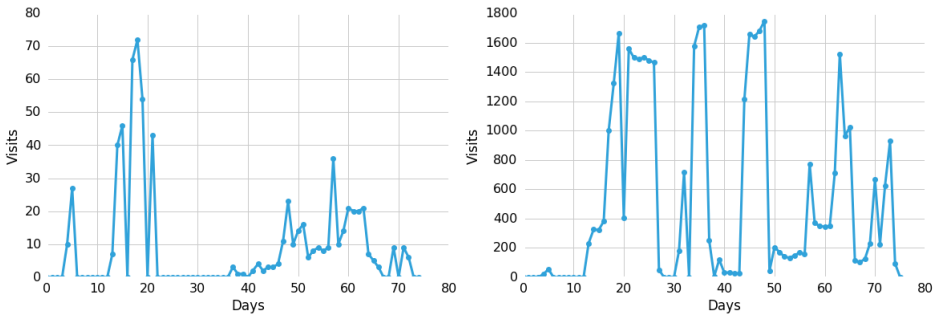
snoopers may wait for a longer duration of time, so they can evade detection and frame other HSDirs.

4.2 Estimation & Identification of Snooping HSDirs

Identifying snooping HSDirs: Based on the visited hidden service, the time of the visit, and the HSDir that have been hosting the specific onion address prior to the visit, we can mark the potential malicious and misbehaving HSDirs. Then, we add the candidates to a bipartite graph, which consists of edges between HSDirs and the visited honions. The analysis of this graph allows us to infer a lower bound on the number of malicious HSDirs as well as specific snoopers. Figure 7 depicts the architecture of the system.

HOnion Visit Graph Formation: In the following we first introduce a formal model and notation for the Honey Onions system. First, HO denotes the set of honey onions generated by the system that were visited, and HSD the set of Tor relays with the `HSDir` flag (so far referred to as HSDir relays). The visits of honions allow us to build a graph $G = (V, E)$ whose vertices are the union of HO and HSD and edges connect a honion ho_j and HSDir d_i iff ho_j was placed on d_i and subsequently experienced a visit. G is by construction a bipartite graph. We also note that each honion periodically changes descriptors and therefore HSDirs (approximately once a day). However, a HSDir currently a honion ho cannot explain visits during past days. Therefore, each time a honion changes HSDirs we clone its vertex ho to ho' and only add edges between ho' and the HSDirs who know about its existence when the visit happened.

Estimation & Set Cover: Since each honion is simultaneously placed on multiple HSDirs, the problem of identifying which ones are malicious is not trivial. We first formulate the problem of deriving



(a) Daily Visits (b) All Visits

Figure 8: Plot of the visits to the honions.

a lower-bound on their number by finding the smallest subset S of HSD that can explain all the visits. The size s of the minimal set tells us that there cannot be less than s malicious HSDirs who would explain the visits.

$$\underset{S \subseteq HSD}{\operatorname{argmin}} |S : \forall (ho_j, d_i) \in E \exists d_i' \in S \wedge (ho_j, d_i') \in E| \tag{1}$$

Finding the smallest set S as defined by Equation 1, is not trivial as one can easily see that it is equivalent to the hitting set problem, which itself is equivalent to the set cover problem, which is well known to be NP-Complete. However, it can also be formulated as an Integer Linear Program. Let $x_{1 \leq j \leq |HSD|}$ be binary variables taking values 0 or 1. Solving Equation 1, consists of finding integer assignments to the x_j such that:

$$\begin{array}{ll} \min_{(x_1,\ldots,x_{HSD})} & \sum_{j=1}^{|HSD|} x_j \\ \text{subject to } \forall ho_i \in HO & \sum_{\forall j:(ho_i,d_j)\in E} x_j \geq 1 \end{array}$$

4.3 Analysis and Findings

We started the daily honions on Feb 12, 2016; the weekly and monthly experiments on February 21, 2016, which lasted until April 24, 2016. During this period there were three spikes in the number of hidden services, with one spike more than tripling the average number of hidden services. There are some theories suggesting that this was due to botnets, ransomware, or the success of the anonymous chat service, called Ricochet. However, none of these explanations can definitely justify the current number of hidden services. Our daily honions spotted snooping behavior before the spike in the hidden services, this gives us a level of confidence that the snoopings are not only a result of the anomaly (Figure 8). Rather, there are entities that actively investigate hidden services.

Figure 9: The global map of detected misbehaving HSDirs and their geographic origin.

Snooping HSDirs Nature and Location: In total we detected at least 110 malicious HSDir using the ILP algorithm, and about 40000 visits. More than 70% of these HSDirs are hosted on Cloud infrastructure. Around 25% are exit nodes as compared to the average, 15% of all relays in 2016, that have both the HSDir and the Exit flags. This can be interesting for further investigation, since it is known that some Exit nodes are malicious and actively interfere with users' traffic and perform active MITM attacks [39]. Furthermore, 20% of the misbehaving HSDirs are, both exit nodes and are hosted on Cloud systems, hosted in Europe and Northern America. The top 5 countries are, USA, Germany, France, UK, and Netherlands. Figure 9 depicts the spread and the geolocation of the malicious HSDirs.

HSDirs Behavior and Intensity of the Visits: Most of the visits were just querying the root path of the server and were automated. However, we identified less than 20 possible manual probing, because of a query for favicon.ico, the little icon that is shown in the browser, which the Tor browser requests. Some snoopers kept probing for more information even when we returned an empty page. For example, we had queries for `description.json`, which is a proposal to all HTTP servers inside Tor network to allow hidden services search engines such as Ahmia, to index websites. One of the snooping HSDirs (5.*.*.*:9011) was actively querying the server every 1 hour asking for a server-status page of Apache. It is part of the functionality provided by mod_status in Apache, which provides information on server activity and performance. Additionally, we detected other attack vectors, such as SQL injection, targeting `the information_schema.tables`, username enumeration in Drupal, cross-site scripting (XSS), path traversal (looking for `boot.ini` and `/etc/passwd`), targeting Ruby on Rails framework (rails/info/properties), and PHP Easter Eggs (?=PHP*-*-*-*-*).

5 Future Work and Timeline

5.1 Privacy Preserving Longevity Study of Hidden Services

Previous work studied the nature of hidden services and dark web [11]. However, very little is known about the lifespan of hidden services. Studying the lifespan of hidden services can provide manifold benefits. For example, it allows investigation of the maliciousness of domains, based on their lifespan. Short-lived hidden services are more likely not to be legitimate domains, as compared to long-lived domains. Furthermore, such knowledge provides insights into the performance and resource allocation requirements of privacy infrastructures.

The distributed nature of Tor and hidden services makes such study non-trivial and introduces challenges that need to be addressed. For example, the privacy and security services of Tor should not be undermined or compromised. Furthermore, schemes and protocols should be devised to draw conclusions based on the data that is distributively collected from the network.

As mentioned earlier, every onion address is uploaded to 6 random HSDirs, and every 24 hours the responsible HSDirs change. We use the number of times a hidden service descriptor has been uploaded to the HSDirs as an indicator of lifespan. By running enough HSDirs, we will be able to estimate the lifespan of of hidden services.

To maintain the privacy services of Tor, we only seek an aggregate PDF/CDF of the lifetime of the hidden services with the following features.

- The raw (original) onion addresses should stay hidden
- No information about a specific onion is leaked
- Even if an adversary knows the .onion, he should not be able to infer information about it
- Once the data is encrypted even the participant holding it should not be able to retrieve it without the collaboration of the other participants
- Only when all the parties are involved, we will be able to decrypt. If one party is not participating the data should be irretrievable.

From the theoretical aspect we will ensure these properties by using threshold cryptography and multi party computation (MPC). On the operational security side we deploy set of relays over cloud. The data is encrypted and only resides in the RAM. No information is written to persistent storage. This allows us to easily remove all the data, in case one of the participants is compromised. Furthermore, we run the relays controlled by three different entities over two continents. This is to insure the privacy and security of the collected data at different jurisdictions.

5.2 Timeline

The following table is the proposed timeline to complete the research:

To-do tasks	Completion Date (end of)
Deployment of HSDirs over Tor	July 2017
Design of the privacy preserving longevity study	August 2017
Implementation and Data Collection	October 2017
Analysis and Evaluation	November 2017
Dissertation Defense	December 2017

The OnionBots work is published in IEEE/IFIP International Conference on Dependable Systems and Networks (DSN 2015) [30], and the HOnions work is published in IEEE Conference on Communications and Network Security (CNS 2016) [30].

My other work that is not part of this thesis is published and presented at ACM Conference on Security and Privacy in Wireless and Mobile Networks (WiSec) [25], Hot Topics in Privacy Enhancing Technologies (HotPETs) [31], Virus Bulletin (VB) [27], DEF CON [26], IEEE Symposium on Technologies for Homeland Security (HST) [28], and New England Security Day (NESD) [32].

References

[1] Tor metrics. https://metrics.torproject.org/userstats-relay-country.html.

[2] Security researcher intercepts embassy passwords from tor. http://www.infoworld.com/article/2649832/security/security-researcher-intercepts-embassy-passwords-from-tor.html, September 2007.

[3] The inevitable move - 64-bit zeus enhanced with tor. http://securelist.com/blog/events/58184/the-inevitable-move-64-bit-zeus-enhanced-with-tor/, December 2013.

[4] Tor stinks presentation. https://edwardsnowden.com/wp-content/uploads/2013/10/tor-stinks-presentation.pdf, 2013.

[5] Deanonymizing tor hidden service traffic through hsdir is a cake walk, say researchers: Hitb presenters showcase new threats. http://www.idigitaltimes.com/deanonymizing-tor-hidden-service-traffic-through-hsdir-cake-walk-say\researchers-hitb-445328, May 2015.

[6] Confirmed: Carnegie mellon university attacked tor, was subpoenaed by feds. http://motherboard.vice.com/read/carnegie-mellon-university-attacked-tor-was-subpoenaed-by-feds, February 2016.

[7] Cryptolocker ransomware. http://www.secureworks.com/cyber-threat-intelligence/threats/cryptolocker-ransomware/, December 2013.

[8] Ittai Anati, Shay Gueron, Simon Johnson, and Vincent Scarlata. Innovative technology for cpu based attestation and sealing.

[9] D. Andriesse, C. Rossow, B. Stone-Gross, D. Plohmann, and H. Bos. Highly resilient peer-to-peer botnets are here: An analysis of Gameover Zeus. In *Malicious and Unwanted Software: "The Americas" (MALWARE), 8th International Conference on*, 2013.

[10] Kevin Bauer, Damon McCoy, Dirk Grunwald, Tadayoshi Kohno, and Douglas Sicker. Low-resource routing attacks against tor. In *Proceedings of the 2007 ACM workshop on Privacy in electronic society*, 2007.

[11] A. Biryukov, I. Pustogarov, F. Thill, and R. P. Weinmann. Content and popularity analysis of tor hidden services. In *ICDCSW*, 2014.

[12] A. Biryukov, I. Pustogarov, and R. Weinmann. Trawling for tor hidden services: Detection, measurement, deanonymization. In *Security and Privacy (SP), 2013 IEEE Symposium on*, 2013.

[13] Nicolas Christin. Traveling the silk road: A measurement analysis of a large anonymous online marketplace. In *Proceedings of the 22Nd International Conference on World Wide Web*, WWW, 2013.

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

[15] Tariq Elahi, Geroge Danezis, and Ian Goldberg. Privex: Private collection of traffic statistics for anonymous communication networks. In *21st ACM Conference on Computer and Communications Security*, 2014.

[16] Craig Gentry et al. Fully homomorphic encryption using ideal lattices. 2009.

[17] Thorsten Holz, Moritz Steiner, Frederic Dahl, Ernst Biersack, and Felix Freiling. Measurements and mitigation of peer-to-peer-based botnets: A case study on storm worm. In *Proceedings of the 1st Usenix Workshop on Large-Scale Exploits and Emergent Threats*, LEET, 2008.

[18] Rob Jansen and Aaron Johnson. Safely measuring tor. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, 2016.

[19] Rob Jansen, Florian Tschorsch, Aaron Johnson, and Björn Scheuermann. The sniper attack:

Anonymously deanonymizing and disabling the tor network. Technical report, DTIC Document, 2014.

[20] Erhan J. Kartaltepe, Jose Andre Morales, Shouhuai Xu, and Ravi Sandhu. Social network-based botnet command-and-control: Emerging threats and countermeasures. In *Proceedings of the 8th International Conference on Applied Cryptography and Network Security*, ACNS, 2010.

[21] Albert Kwon, Mashael AlSabah, David Lazar, Marc Dacier, and Srinivas Devadas. Circuit fingerprinting attacks: Passive deanonymization of tor hidden services. In *24th USENIX Security Symposium (USENIX Security 15)*, 2015.

[22] Stevens Le Blond, Pere Manils, Abdelberi Chaabane, Mohamed Ali Kaafar, Claude Castelluccia, Arnaud Legout, and Walid Dabbous. One bad apple spoils the bunch: Exploiting p2p applications to trace and profile tor users. In *Proceedings of the 4th USENIX Conference on Large-scale Exploits and Emergent Threats*, LEET'11.

[23] Z. Ling, J. Luo, K. Wu, W. Yu, and X. Fu. Torward: Discovery, blocking, and traceback of malicious traffic over tor. In *IEEE Transactions on Information Forensics and Security*, 2015.

[24] Antonio Nappa, Aristide Fattori, Marco Balduzzi, Matteo Dell'Amico, and Lorenzo Cavallaro. Take a deep breath: A stealthy, resilient and cost-effective botnet using skype. In *Proceedings of the 7th International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*, DIMVA, 2010.

[25] Sashank Narain, Amirali Sanatinia, and Guevara Noubir. Single-stroke language-agnostic keylogging using stereo-microphones and domain specific machine learning. In *ACM Conference on Security and Privacy in Wireless and Mobile Networks (ACM WiSec)*, 2014.

[26] Guevara Noubir and Amirali Sanatinia. Honey onions: Exposing snooping tor hsdir relays. In *DEFCON 24*, 2016.

[27] Guevara Noubir and Amirali Sanatinia. Trusted code execution on untrusted platform using intel sgx. In *Virus Bulletin (VB)*, 2016.

[28] Amirali Sanatinia, Sanket Deshpande, Apoorv Munshi, Daniel Kohlbrenner, Michael Yessaillian, Sarada Symonds, Agnes Chan, and Guevara Noubir. Hyperdrive: a flexible cloud testbed for research and education. In *IEEE Symposium on Technologies for Homeland Security (HST)*, 2017.

[29] Amirali Sanatinia and Guevara Noubir. Onionbots: Subverting privacy infrastructure for cyber attacks. In *The Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, 2015.

[30] Amirali Sanatinia and Guevara Noubir. Honey onions: a framework for characterizing and

identifying misbehaving tor hsdirs. In *IEEE Conference on Communications and Network Security (CNS)*, 2016.

[31] Amirali Sanatinia and Guevara Noubir. Honions: Towards detection and identification of misbehaving tor hsdirs. In *Workshop on Hot Topics in Privacy Enhancing Technologies (HotPETs)*, 2016.

[32] Amirali Sanatinia and Guevara Noubir. Off-path man-in-the-middle attack on tor hidden services. In *New England Security Day*, NESD, 2017.

[33] Guenther Starnberger, Christopher Kruegel, and Engin Kirda. Overbot: A botnet protocol based on kademlia. In *Proceedings of the 4th International Conference on Security and Privacy in Communication Netowrks*, SecureComm, 2008.

[34] Ben Stock, Jan Göbel, Markus Engelberth, Felix C. Freiling, and Thorsten Holz. Walowdac - analysis of a peer-to-peer botnet. In *Proceedings of the European Conference on Computer Network Defense*, EC2ND, 2009.

[35] S. Stover, D. Dittrich, J. Hernandez, and S. Dietrich. Analysis of the storm and nugache trojans: P2P is here. In *;login*, 2007.

[36] Yixin Sun, Anne Edmundson, Laurent Vanbever, Oscar Li, Jennifer Rexford, Mung Chiang, and Prateek Mittal. Raptor: Routing attacks on privacy in tor. In *24th USENIX Security Symposium (USENIX Security 15)*, 2015.

[37] Matthew Thomas and Aziz Mohaisen. Measuring the leakage of onion at the root: A measurement of tor's. onion pseudo-tld in the global domain name system. In *Proceedings of the Workshop on Privacy in the Electronic Society*, 2014.

[38] Philipp Winter, Roya Ensafi, Karsten Loesing, and Nick Feamster. Identifying and characterizing sybils in the tor network. *arXiv preprint arXiv:1602.07787*, 2016.

[39] Philipp Winter, Richard Köwer, Martin Mulazzani, Markus Huber, Sebastian Schrittwieser, Stefan Lindskog, and Edgar Weippl. Spoiled onions: Exposing malicious tor exit relays. In *Privacy Enhancing Technologies*, 2014.