A Moving Target Defense Approach to Disrupting Stealthy Botnets[*]

Sridhar Venkatesan
Center for Secure Information Systems
George Mason University
svenkate@gmu.edu

Massimiliano Albanese
Center for Secure Information Systems
George Mason University
malbanes@gmu.edu

George Cybenko
Thayer School of Engineering
Dartmouth College
george.cybenko@dartmouth.edu

Sushil Jajodia
Center for Secure Information Systems
George Mason University
jajodia@gmu.edu

ABSTRACT

Botnets are increasingly being used for exfiltrating sensitive data from mission-critical systems. Research has shown that botnets have become extremely sophisticated and can operate in stealth mode by minimizing their host and network footprint. In order to defeat exfiltration by modern botnets, we propose a moving target defense approach for dynamically deploying detectors across a network. Specifically, we propose several strategies based on centrality measures to periodically change the placement of detectors. Our objective is to increase the attacker's effort and likelihood of detection by creating uncertainty about the location of detectors and forcing botmasters to perform additional actions in an attempt to create detector-free paths through the network. We present metrics to evaluate the proposed strategies and an algorithm to compute a lower bound on the detection probability. We validate our approach through simulations, and results confirm that the proposed solution effectively reduces the likelihood of successful exfiltration campaigns.

Keywords

Botnets, moving target defense, detector placement

1. INTRODUCTION

In recent years, botnets have gained significant attention both in the industry and in the research community due to their extensive use in various kinds of criminal or otherwise unauthorized activities. Of particular interest are their use in large-scale attack campaigns – such as DDoS and spamming – and their potential role – within the context of advanced persistent threats (APTs) [10] – to exfiltrate sensitive data from mission-critical systems.

In response to the threats posed by botnets, researchers and practitioners have developed several detection mechanisms. The performance of these mechanisms is primarily dependent upon the set of features used to identify malicious traffic. Most of the existing research focuses on identifying a combination of packet-based, time-based, and behavior-based features to weed out bot traffic from the traffic mix [2]. For instance, BotHunter [8] exploits the ordered sequence of messages between a bot and a Command and Control (C&C) server – in a centralized botnet architecture – while [27] exploits a combination of packet-based features, such as average number of packets/bytes sent/received, and time-based properties, such as application uptime, to narrow down on hosts that may potentially belong to a P2P botnet.

With the ever-increasing accuracy of these feature-based detection techniques, researchers are investigating advanced botnet designs – few of which have been adopted by existing botnets – to evade or otherwise remain stealthy to such detection mechanisms. These evasion/stealth mechanisms can be broadly classified into two types: anti-signature stealth and architectural stealth [24]. Anti-signature stealth techniques change the characteristics of the traffic generated by bots (such as encryption, randomizing number of packets per flow) to alter the features used for detection. Stinson et al. [21] studied different anti-signature stealth/evasion tactics that can be employed by a botmaster to remain undetected.

On the other hand, architectural stealth techniques aim to build topology-aware botnets to reduce the exposure of malicious traffic to these detectors. They exploit the fact that existing detection mechanisms are expected to be installed in a location that can monitor all the traffic entering/exiting the network (such as network gateways for enterprise network) or assume the existence of several vantage points in the network that intercepts malicious traffic (such as routers in an ISP-network). As a result, attackers can design a stealthy botnet communication architecture which can reduce the observable bot traffic through these monitoring points. For instance, a botmaster can modify the botnet design such that only a single bot or a few

[*]This work was partially supported by the Army Research Office under grants W911NF-13-1-0421 and W911NF-13-1-0317, and by the Office of Naval Research under grant N00014-13-1-0703.

MTD'16, October 24 2016, Vienna, Austria
© 2016 ACM. ISBN 978-1-4503-4570-5/16/10…$15.00
DOI: http://dx.doi.org/10.1145/2995272.2995280

bots act as relays between bots inside a monitored domain and external peers or C&C site outside of that domain. Such modifications can be used to evade existing detection techniques, such as those described in [8] and [27]. Moreover, recent P2P-versions of the Zeus bot include a proxy layer bot that aggregates data exfiltrated from multiple bots before forwarding it to a C&C site [1].

To this end, Sweeney and Cybenko [23, 24] studied the importance of the physical location of bots (referred to as the *cyber high ground*) to perform stealthy missions such as data exfiltration. They designed a P2P botnet with the objective of exfiltrating data from mission-critical nodes in a network. For a given network topology, they design a P2P overlay communication structure that maximizes exfiltration effectiveness, while maintaining stealthiness (by avoiding detectors and reducing network footprint). In the remainder of this paper, we use the term *stealthy botnets* to refer to architectural stealthy botnets.

A straightforward solution to intercept exfiltrated data might be to mirror traffic traversing all the routers in the network. However, this will adversely impact the performance of the network. Therefore, in order to detect exfiltration attempts by such stealthy botnets, we propose a moving target defense approach for deploying detectors across the network – in a resource-constrained environment – which consists in dynamically and continuously changing the placement of detectors over time. Specifically, we propose several strategies based on centrality measures that capture important properties of the network. Our objective is to increase the attacker's effort and likelihood of detection by creating uncertainty about the location of detectors and forcing the botmaster to perform additional actions in an attempt to create detector-free paths through the network. We present two metrics to evaluate the proposed strategies – namely the *minimum detection probability* and the *attacker's uncertainty* – and an algorithm to compute the minimum detection probability, which represent a lower bound on the detection probability. We validate our approach through simulations on actual and synthetic networks, and the results confirm that the proposed solution can effectively reduce the likelihood of successful exfiltration campaigns.

The paper is organized as follows. Section 2 discusses related work. Section 3 provides some preliminary definitions, whereas Section 4 and Section 5 present our threat model and defender's model respectively. Then, Section 6 introduces the two metrics we define for assessing the proposed strategies, and Section 7 presents such strategies in detail. Finally, Section 8 presents the detailed results of our simulations, and Section 9 gives some concluding remarks.

2. RELATED WORK

The communication architecture of a botnet plays a crucial role in accomplishing its malicious objectives. Ever since the first appearance of a botnet in 2001, its communication architecture has evolved in response to the increasing effort to detect botnets based on their communication patterns [7, 8] and their communication structure [15, 16]. The choice of a communication architecture by a botmaster is largely influenced by its resilience to known defenses and the communication latency [4]. Traditionally, centralized architectures were adopted to minimize latency by establishing a direct communication channel between the Command and Control (C&C) server and the bots. As a result of several botnet takedowns [5, 22] – which leveraged the single-point of failure of a centralized architecture – botmasters have switched to more resilient communication architectures such as P2P botnets or hybrid P2P/centralized C&C structures [19].

To counteract P2P botnets, several detection models were developed. However, many existing detection techniques assume that the bots exhibit malicious activity [7, 8] and, hence, do not perform well against custom protocols that exhibit stealth by coexisting with legitimate applications. To detect such stealthy P2P botnets, Zhang et al. [27] derived a statistical fingerprint of different P2P communications to differentiate between hosts that belong to a legitimate P2P network from those that belong to a P2P botnet. Besides leveraging communication patterns to detect botnets, BotGrep [15] takes a graph-theoretic approach to detect P2P botnets in ISP networks. It leverages the highly structured communication between bots to isolate a P2P botnet's communication subgraph from the ISP communication graph. BotGrep improves its accuracy by sampling C&C traffic from several vantage points in the network, which are determined by considering graph-theoretic properties of the communication graph. Furthermore, Ha et al. [9] studied the effectiveness of exploiting the structural properties of P2P botnets for detection and concluded that static monitoring of botnets will not perform well for all bot distributions in a network. To address these limitations, the dynamic detector placement approach proposed in this paper leverages the topology of the network being protected and can improve effectiveness by dynamically choosing the vantage points for monitoring C&C traffic.

3. PRELIMINARIES

Let $G = (V, E)$ be a graph representing the physical topology of the network, where V is a set of network elements – such as routers and end hosts – and E captures the connectivity between them. Let $N = \{h_1, h_2, ..., h_n\}$ be a set of mission-critical hosts.

Typically, traffic between any two nodes is routed using a routing algorithm/policy. Let Π_G denote the set of all simple paths $\pi(v_i, v_j)$ between any pair of nodes $(v_i, v_j) \in V \times V$. Then, a routing algorithm can be formally defined as a mapping R_A from the set $V \times V$ of all pairs of nodes in the connectivity graph to Π_G, such that

$$R_A(u, v) = \langle u, z_1, z_2, \ldots, z_r, v\rangle, \forall (u, v) \in V \times V$$

where $\langle u, z_1, z_2, \ldots, z_r, v\rangle \in \Pi_G$ is the path followed by traffic from u to v, as determined by the routing protocol/policy. Note that we slightly abuse notation and, for the sake of presentation, we may treat a path $\pi \in \Pi_G$ as a set of nodes. Although most routing algorithms attempt to route traffic along the shortest path from source to destination, our approach does not rely on the assumption that traffic is routed along the shortest path, but rather on the more general assumption that we can predict what paths the algorithm/policy will select for routing traffic. However, for the sake of presentation, and without limiting the generality of our approach, we do assume that the networks being studied implement a shortest path routing algorithm.

In order to exfiltrate data from the set N of mission-critical nodes, the attacker compromises a set $B \subseteq V$ of network nodes – referred to as *bots* – and creates an overlay network to forward captured data to a remote C&C server. It

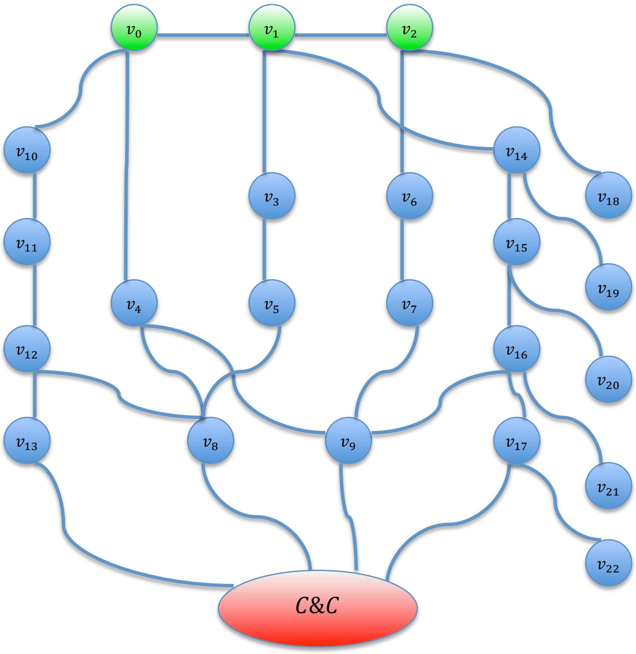


Figure 1: Example of network graph

should be noted that the traffic between any two bots in the overlay network traverses different routers in the physical network $G(V, E)$, which, as mentioned above, is determined by the routing algorithm/policy.

Definition 1 (Exfiltration Path). *Given the set $N \subseteq V$ of mission-critical nodes for a network $G = (V, E)$ and a set $B \subseteq V$ of nodes controlled by the attacker, an* overlay path *is a sequence $\pi_o(b_0, \mathsf{C\&C}) = \langle b_0, b_1, b_2, \ldots, b_r, \mathsf{C\&C}\rangle$ of bots – with $b_0 \in N$ and $b_i \in B$ for each $i \in [1, r]$ – chosen by the attacker to forward traffic from mission-critical node b_0 to a remote* C&C *site. The* exfiltration path *corresponding to an overlay path $\pi_o(b_0, \mathsf{C\&C})$ is the sequence of nodes in V traversed by traffic exfiltrated through $\pi_o(b_0, \mathsf{C\&C})$, and it is defined as:*

$$\begin{aligned}\pi_e(b_0, \mathsf{C\&C}) \;=\; &\langle b_0, v_1^0, v_2^0, \ldots, v_{l_0}^0, b_1, v_1^1, v_2^1, \ldots, v_{l_1}^1,\\ &b_2, \ldots, b_r, v_1^r, v_2^r, \ldots, v_{l_r}^r, \mathsf{C\&C}\rangle\end{aligned}$$

where $R_A(b_i, b_{i+1}) = \langle b_i, v_1^i, v_2^i, \ldots, v_{l_i}^i, b_{i+1}\rangle, \forall i \in [0, r-1]$ is the routing path from b_i to b_{i+1} and $R_A(b_r, \mathsf{C\&C}) = \langle b_r, v_1^r, v_2^r, \ldots, v_{l_r}^r, \mathsf{C\&C}\rangle$ is the routing path from b_r to C&C*.*

Example 1. *In the example of Fig. 1, if $v_5 \in B$ is a bot and the attacker chooses to exfiltrate traffic through the overlay path $\pi_o(v_1, \mathsf{C\&C}) = \langle v_1, v_5, \mathsf{C\&C}\rangle$, then the corresponding exfiltration path is $\pi_e(v_2, \mathsf{C\&C}) = \langle v_1, v_3, v_5, v_8, \mathsf{C\&C}\rangle$.*

For a given set of mission-critical nodes N, the defender's objective is to intercept and detect exfiltration traffic. In order to monitor the network for botnet activity, the defender can deploy detectors on a subset of nodes $D \subset V$ and inspect traffic flowing through them. One of several botnet detection mechanisms can be used to detect botnet activity [7, 8, 26, 27]. These detection mechanisms leverage the fact that the bots need to communicate with their peers or the C&C server to relay captured data.

Definition 2 (Detection). *Given the set $N \subseteq V$ of mission-critical nodes for a network $G = (V, E)$ and the set $B \subseteq V$ of bots controlled by the attacker, an exfiltration attempt over an exfiltration path $\pi_e = \langle v_0, v_1, v_2, \ldots, v_r, \mathsf{C\&C}\rangle$, with $v_0 \in N \cap B$, is said to be* detected *iff the exfiltrated traffic traverses a detector node, that is, $\exists d \in D$ s.t. $d \in \pi_e$. A botnet is said to be* stealthy *with respect to N, iff no exfiltration path between nodes in $N \cap B$ and a* C&C *site can be detected.*

Unfortunately, existing detection mechanisms suffer from false positives and false negatives, therefore exfiltration attempts may go undetected even when a detector is placed on a node along the exfiltration path. However, for the purpose of our analysis, the accuracy of detectors is not as critical as their placement. As mentioned earlier, a prudent attacker – who does not want any portion of the exfiltration traffic to be detected and eventually dropped before reaching a C&C site – will opt for creating more bots in order to establish a detector-free path, rather than having the traffic routed through detectors, irrespective of their false positive and false negative rate. Based on these considerations, and in order to simplify the presentation of our approach, we ignore the accuracy of detectors and assume that any exfiltration attempt going through a detector node is detected. Factoring the probability of missed detections in our model is relatively straightforward, and we plan to do so as part of our future work.

In order to exfiltrate data from a mission-critical node $h \in N$ to a C&C site in a stealthy manner, the attacker must identify a detector-free path $\pi_e^*(h, \mathsf{C\&C}) \in \Pi_G$, and forward data through it. The set of all detector-free paths represents the *exfiltration surface* of the network, which can be formally defined as follows.

Definition 3 (Exfiltration Surface). *Given the set $N \subseteq V$ of mission-critical nodes for a network $G = (V, E)$, let $D \subset V$ be a set of detector nodes. The* exfiltration surface *of G with respect to D is the set of detector-free paths $\psi_D = \{\pi_e(h, \mathsf{C\&C}) \mid h \in N \wedge \pi_e(h, \mathsf{C\&C}) \in \Pi_G \wedge \pi_e(h, \mathsf{C\&C}) \cap D = \emptyset\}$. We use Ψ to denote the set of all possible exfiltration surfaces from mission-critical nodes N to* C&C *sites.*

In [25], we proposed an approach to deploy detectors on selected network nodes, so as to reduce the exfiltration surface by either completely disrupting communication between bots and C&C nodes, or at least forcing the attacker to create more bots, thereby increasing the footprint of the botnet and the likelihood of detection. As the detector placement problem is intractable, we proposed heuristics based on several centrality measures. Specifically, it was shown that the *iterative mission-betweenness centrality strategy* yields the best results. In this strategy, after a node has been selected as a detector, the mission-betweenness centrality of all non-detector nodes that belong to the exfiltration surface is recomputed, and the node with the highest centrality is chosen for placing an additional detector. In practice, this approach prevents two or more detectors from being placed on the same high-centrality path.

Although the above strategy significantly increases an attacker's effort, the resulting exfiltration surface is static. Therefore, a persistent attacker can gather enough information to precompute the exfiltration surface of the target system and identify a detector-free path to exfiltrate data.

In this paper, we overcome this limitation by designing detector placement strategies that *dynamically* change the exfiltration surface by continually altering the placement of detectors.

4. THREAT MODEL

In our threat model, the attacker's ultimate goal is to exfiltrate data from mission-critical nodes, while remaining stealthy and persisting in the system for an extend period of time. To this end, we make the following assumptions.

- The attacker can discover the topology of the network, and is aware of what nodes are mission-critical. Reports by Kaspersky labs [11] and Mandiant [12] show that threat actors can infiltrate an organization's network and persist in the system for several years, mapping out the organization and exfiltrating sensitive and valuable intellectual property.
- Exfiltrating large volumes of data generates abnormally large network flows which in turn may trigger alerts. To avoid detection, the attacker partitions the data to be exfiltrated into m segments $d_1, d_2, \ldots, d_m$, and transmits these segments over a temporal span $\mathcal{T} = \langle t_1, t_2, \ldots, t_m \rangle \subseteq \mathbb{N}^m$, i.e., at each time point t_i, the attacker transmits data segment d_i to a C&C site. The attacker is said to have *successfully exfiltrated from a mission-critical node* if and only if all the m data segments are exfiltrated by time t_m.
- The attacker is aware of the detector placement strategy employed by the defender.

5. DEFENDER'S MODEL

To detect exfiltration over a network, the defender deploys detectors on selected network nodes. Placing detectors on different vantage points in the network reduces single-point detection failures of network perimeter devices and increases the opportunities to detect botnet activity. In our defender's model, we consider a resource-constrained setting where the defender can only deploy k detectors. In practice, an upper bound on the number of detectors can be determined by considering the number of systems in the network that can perform detection tasks without impacting the performance of other applications/services running on them. A dual problem that can be defined is that of minimizing the number of detectors needed to satisfy predefined security requirements, but we do not discuss this problem for reasons of space. In the following, we provide a formal definition of the notion of *detector placement.*

We assume that the defender is aware of the location of potential C&C sites. For an enterprise network, C&C locations could include any destination on the Internet, i.e., outside the network perimeter. Similarly, for an ISP network, potential C&C sites could be located outside the administered domain. Furthermore, research by Moreira Moura [14] and Collins et al. [3] shows that certain IP address ranges are known to participate in malicious campaigns. This information can be leveraged to identify potential C&C locations. Due to the conservative estimate on the location of potential C&C sites, blacklisting traffic destined to these locations will adversely impact the utility of the network.

Definition 4 (k-placement). *Given a network $G = (V, E)$, a k-placement over G is a mapping $pl : V \to \{0, 1\}$ such that $\sum_{v \in V} pl(v) = k$. Vertices v such that $pl(v) = 1$ are called* detector nodes. *We will use $\mathcal{P}_k$ to denote the set of all possible k-placements.*

As mentioned earlier, static placement of detectors suffers from the drawback that a persistent attacker – who is aware of the network topology – can discover the location of detectors and identify detector-free paths from mission-critical nodes to C&C sites. The resulting exfiltration surface remains static during exfiltration and, therefore, cannot detect data exfiltration. Hence, to increase the probability of detection, we propose to continually shift the exfiltration surface by dynamically changing the location of detectors. In our model, we discretize time as a finite sequence of integers $\mathcal{T} = \langle t_1, t_2, \ldots, t_m \rangle \subseteq \mathbb{N}^m$, for some $m \in \mathbb{N}$, such that for all $1 \le i < m$, $t_i < t_{i+1}$. In the following, we slightly abuse notation and, for the sake of presentation, we treat the sequence $\mathcal{T}$ as a set when appropriate. We can now define how placements can evolve over time.

Definition 5 (Temporal k-placement). *A* temporal k-placement *is a function $tp : \mathcal{T} \to \mathcal{P}_k$. We will use $\mathcal{TP}_k$ to denote the set of all possible temporal k-placements.*

Intuitively, for each time point in $\mathcal{T}$, a temporal k-placement deploys detectors on k network nodes. In order to create uncertainty for the attacker with respect to the location of detectors, we choose k-placement functions by using a probability distribution over temporal placements.

Definition 6 (Temporal probabilistic k-placement). *A* temporal probabilistic k-placement *(tp-k-placement) is a function $\tau : \mathcal{TP}_k \to [0, 1]$ such that $\sum\limits_{tp \in \mathcal{TP}_k} \tau(tp) = 1$.*

Example 2. *Figure 2 shows an example of temporal probabilistic k-placement τ for the graph of Fig. 1 and for $k = 2$. Each table in the figure represents a different temporal k-placement tp (for the sake of presentation, we assume those shown are the only possible temporal k-placements in $\mathcal{TP}_k$). Note that $\sum_{tp \in \mathcal{TP}_k} \tau(tp) = 1$. For any given temporal k-placement tp, the i-th column – with $i \in \{1, 2, \ldots, m\}$ – in the corresponding table represents the k-placement pl that tp associates with time point t_i. Note that, for each k-placement pl, $\sum_{v \in V} pl(v) = k$. This example assumes that only certain nodes, namely v_3, v_4, v_5, and v_6, can host detectors.*

Let the indicator random variable $I^v_{t_i}$ be associated with the event that node v is chosen as a detector at time t_i. Given a temporal probabilistic k-placement τ, the probability with which a node $v \in V$ will be chosen as a detector at time t_i can be derived as

$$\begin{aligned} pr^v_{t_i} &= Pr(I^v_{t_i} = 1 \mid \tau) \\ &= \textstyle\sum_{tp \in \mathcal{TP}_k \ s.t.\ (\exists pl \in \mathcal{P}_k)(tp(t_i) = pl \wedge pl(v) = 1)} \tau(tp) \end{aligned} \quad (1)$$

Therefore, at time t_i, a defender chooses k nodes for detector placement by sampling from the distribution given in Eq. 1. We denote such a strategy by $\mathcal{D}_{t_i} \sim \{pr^v_{t_i}\}_{v \in V}$.

6. METRICS

To evaluate the performance of a defender strategy, we present two metrics: the *minimum detection probability* and

$$\tau\left(\begin{array}{c|ccccc} & t_1 & t_2 & t_3 & \dots & t_m \\ \hline v_3 & 1 & 1 & 0 & \dots & 1 \\ v_4 & 1 & 0 & 1 & \dots & 1 \\ v_5 & 0 & 0 & 1 & \dots & 0 \\ v_6 & 0 & 1 & 0 & \dots & 0 \end{array}\right) = 0.3$$

$$\tau\left(\begin{array}{c|ccccc} & t_1 & t_2 & t_3 & \dots & t_m \\ \hline v_3 & 0 & 1 & 1 & \dots & 1 \\ v_4 & 1 & 0 & 0 & \dots & 0 \\ v_5 & 0 & 1 & 0 & \dots & 0 \\ v_6 & 1 & 0 & 1 & \dots & 1 \end{array}\right) = 0.2$$

$$\tau\left(\begin{array}{c|ccccc} & t_1 & t_2 & t_3 & \dots & t_m \\ \hline v_3 & 0 & 0 & 0 & \dots & 1 \\ v_4 & 0 & 1 & 1 & \dots & 1 \\ v_5 & 1 & 1 & 0 & \dots & 0 \\ v_6 & 1 & 0 & 1 & \dots & 0 \end{array}\right) = 0.5$$

Figure 2: Example of temporal probabilistic k-placement

the *attacker's uncertainty*. The minimum detection probability provides a theoretical lower bound on the probability that an exfiltration activity is detected due to the detector placement strategy. On the other hand, the attacker's uncertainty is measured as the entropy in the location of the detectors from the attacker's point of view: the higher the entropy, the higher the attacker's effort required to discover the location of detectors.

6.1 Minimum Detection Probability

As mentioned earlier, the attacker's objective is to exfiltrate the data segments $d_1, d_2, \ldots, d_m$ over a temporal span $\mathcal{T} = \langle t_1, t_2, \ldots, t_m \rangle \subseteq \mathbb{N}^m$, while remaining undetected. At each time point t_i, the defender chooses a strategy, $\mathcal{D}_{t_i} \sim \{pr^v_{t_i}\}_{v \in V}$ and samples k nodes without replacement. Let D_{t_i} denote the set of detectors at time t_i. Following defender's placement of detectors, the attacker begins exfiltrating data segment d_i. For a chosen overlay path $\pi_o(h, \mathsf{C\&C})$ to transmit d_i, the traffic will traverse the corresponding exfiltration path $\pi_e(h, \mathsf{C\&C}) = \langle h, v_{i_1}, v_{i_2}, \ldots, v_{i_l}, \mathsf{C\&C}\rangle$, with $h \in N$. Therefore, the probability that the attacker's exfiltration of data segment d_i is detected is given by:

$$detectPr(\mathcal{D}_{t_i}, d_i, \pi_e(h, \mathsf{C\&C})) = 1 - \prod_{v \in \pi_e(h,\mathsf{C\&C}) \setminus \{h, \mathsf{C\&C}\}} \left(1 - pr^v_{t_i}\right) \quad (2)$$

A rational attacker – who is aware of the defender's strategy – will choose a path which minimizes the probability of detection. Therefore, the path chosen by the attacker to exfiltrate d_i is:

$$\pi^{i^*}_e(h, \mathsf{C\&C}) = \underset{\pi_e(h,\mathsf{C\&C})}{\mathrm{argmin}} \left(detectPr\left(\mathcal{D}_{t_i}, d_i, \pi_e\left(h, \mathsf{C\&C}\right)\right)\right) \quad (3)$$

In other words, Eq. 3 can be used to compute the minimum detection probability that a defender strategy $\mathcal{D}_{t_i}$ can guarantee at time t_i. Finally, an exfiltration activity is said to be *detected* when *any* of the m data flows is detected. Therefore, the minimum probability with which a strategy $\mathcal{D}_{t_i}$ detects an exfiltration activity is given by

$$eDetectPr\left(\{\mathcal{D}_{t_i}\}_{i \in [1,m]}\right) = 1 - \prod_{d_i} \left(1 - \min_{\pi_e} \left(detectPr\left(\mathcal{D}_{t_i}, d_i, \pi_e\right)\right)\right)$$

Given a graph $G(V, E)$, the minimum detection probability of a strategy $\mathcal{D}_{t_i}$ at time t_i – i.e., $\min_{\pi_e} \left(detectPr\left(\mathcal{D}_{t_i}, d_i, \pi_e\right)\right)$ – can be computed using Algorithm 1. At a high-level, the algorithm transforms the graph $G(V, E)$ into a weighted dual graph $H(V', E')$ in which the edge weights are a function of the probability that the corresponding vertex in $G(V, E)$ does not host a detector. Specifically, at time t_i, the path detection probability over any path $\pi_e(u, v)$ (given by Eq. 2) can be re-written as $1 - b^S$, where $S = \left(\sum_{x \in \pi_e(u,v)} \log_b\left(1 - pr^x_{t_i}\right)\right)$ and b is an arbitrarily small value. Here, b^S is the upper bound on the probability that the path $\pi_e(u, v)$ will be free of detectors. Therefore, each edge in E' corresponding to a node $v \in V$ is assigned a weight $\log_b(1 - pr^v_{t_i})$. Following this assignment, the algorithm determines the shortest path length between the vertices in V' that correspond to the edges incident on the mission-critical and $\mathsf{C\&C}$ vertices in V. The shortest path length represents the maximum probability that the data exfiltration is not detected and the vertices in V corresponding to edges on this shortest path form the path $\pi^{i^*}_e(h, \mathsf{C\&C})$.

In particular, after generating the dual graph $H(V', E')$ on Line 1, Algorithm 1 assigns weights to all the edges $(e, f) \in E'$ based on the probability $pr^v_{t_i}$ that the corresponding vertex $v \in V$ is chosen for detector placement (Lines 3 – 10). If a detector is placed on vertex $v \in V$ with probability 1, then any exfiltration over a path that contains v will be detected. A rational attacker will avoid such paths and hence the algorithm sets the weight of the corresponding edges in E' as ∞ (Line 8). On the other hand, if the probability is less than 1, then the corresponding edge is assigned a weight $\log_b(1 - pr^v_{t_i})$ (Line 6). Next, Line 15 computes the length of the shortest path between vertices e and c in V', which correspond to the edges in E that are incident on mission-critical nodes and $\mathsf{C\&C}$, respectively. Finally, Line 16 computes the minimum detection probability over all the paths from a mission-critical node $h \in N$ to $\mathsf{C\&C}$ that traverse edges e and c. Line 19 computes the minimum detection probability for each mission-critical node h by considering all the paths to $\mathsf{C\&C}$. Finally, the minimum detection probability with respect to all mission-critical nodes in N for graph $G(V, E)$ is computed on Line 21.

6.1.1 Analysis

In the worst case, Algorithm 1 takes $O(|E|^2)$ time to generate the edge-dual graph $H(V', E')$ as all pairs of edges in E are checked for a common vertex. As a result, in the worst case $|E'| = O(|E|^2)$. Lines 3 – 10 run in time $O(|E'|)$ and Line 11 can be computed in time $O(|V|^2)$ by traversing the adjacency matrix of G. To compute the shortest paths between vertices in H (Line 15) that correspond to mission-critical node h and $\mathsf{C\&C}$ in G, we can leverage the Fibonacci heap implementation of Dijkstra's single-source shortest path algorithm [6]. The complexity of computing the shortest path lengths for a node $h \in N$ (Lines 13 – 19) is given by $O\left(\mathcal{E}(h) \cdot (|E'| + |V'| \log |V'|)\right)$. Therefore, in the worst case, the time complexity for computing the shortest path lengths for all mission-critical nodes is $O(|E| \cdot (|E|^2 + |E| \cdot \log |E|))$.

As the time complexity of the algorithm is dominated by the shortest paths computation, the time complexity is $O(|E| \cdot (|E|^2 + |E| \cdot \log |E|))$. The worst-case time complexity for computing the minimum detection probability

Algorithm 1 *minimumDetectionProb*$(G, \mathcal{D}_{t_i}, N, \mathsf{C\&C})$

```
Input: a connectivity graph $G(V,E)$, a defender strategy, $\mathcal{D}_{t_i} \sim \{pr^v_{t_i}\}$, a set $N \subseteq V$ of mission-critical nodes, a potential C&C location
Output: the minimum detection probability of strategy $\mathcal{D}_{t_i}$ at time $t_i$ for graph $G(V,E)$ with respect to mission-critical nodes $N$ and the
    potential C&C location
$H(V', E') \leftarrow$ dual graph of $G(V,E)$, where $V' = E$ and $(e,f) \in E'$ iff $e$ and $f$ share a common vertex $v \in V$
$b \leftarrow \epsilon$        // an arbitrarily small value
for all $(e,f) \in E'$ do
    $v \leftarrow$ the common vertex of $e$ and $f$ in $V$
    if $pr^v_{t_i} < 1$ then
        $W'(e,f) \leftarrow \log_b(1 - pr^v_{t_i})$
    else
        $W'(e,f) \leftarrow \infty$
    end if
end for
// $\forall v \in V$, let $\mathcal{E}(v)$ denote the set $\{e \mid e \in E \wedge e$ is incident on $v \in V\}$
for all $h$ in $N$ do
    for all $e$ in $\mathcal{E}(h)$ do
        for all $c$ in $\mathcal{E}(\mathsf{C\&C})$ do
            $S \leftarrow$ length of the shortest path from $e$ to $c$ in $H$
            $detectPr(h,e,c) \leftarrow 1 - b^S$
        end for
    end for
    $detectPr(h) \leftarrow \min_{(e,c) \in \mathcal{E}(h) \times \mathcal{E}(\mathsf{C\&C})} (detectPr(h,e,c))$
end for
return $\min_{h \in N}(detectPr(h))$
```

is $O(|V|^6)$. However, for practical network topologies, our simulation results indicate that the running time does not exceed $O(|V|^3)$.

6.2 Attacker's Uncertainty

Probabilistic deployment of detectors introduces uncertainty for the attacker with respect to the location of the detectors. Depending upon the nature of the deployed detector (either active or passive), the attacker may progressively learn the location of these detectors through probing. For instance, if an enterprise network deploys an active IDS, a simple probing strategy could be to send malicious packets to a node suspected of hosting a detector and depending on whether the node accepts or rejects the packets, the attacker can determine the node's detection state. In an ISP network, an attacker can leverage probing strategies described by Shinoda et al. [17], and by Shmatikov and Wang [18] to identify the presence of detectors in a network. Here, the attacker probes the nodes of interest with attacks whose characteristics are recognizable from the resulting alert.

The uncertainty introduced by a dynamic placement strategy can be quantified by measuring the entropy in locating the detectors at any time t_i. Let $X^-_{t_i}$ be the random variable that maps the set V of potential locations to the corresponding probability of being chosen for detector placement. Therefore, the entropy due to a strategy $\mathcal{D}_{t_i} \sim \{pr^v_{t_i}\}_{v \in V}$ is given by:

$$H(X^-_{t_i} \mid \mathcal{D}_{t_i}) = -\sum_{x \in V} P(X^-_{t_i} = x) \log(P(X^-_{t_i} = x)) \qquad (4)$$

where $\log(P(X^-_{t_i} = x)) = 0$, when $P(X^-_{t_i} = x) = 0$. Note that, by the above definition of entropy, the higher the entropy, the higher the advantage for the defender over the attacker.

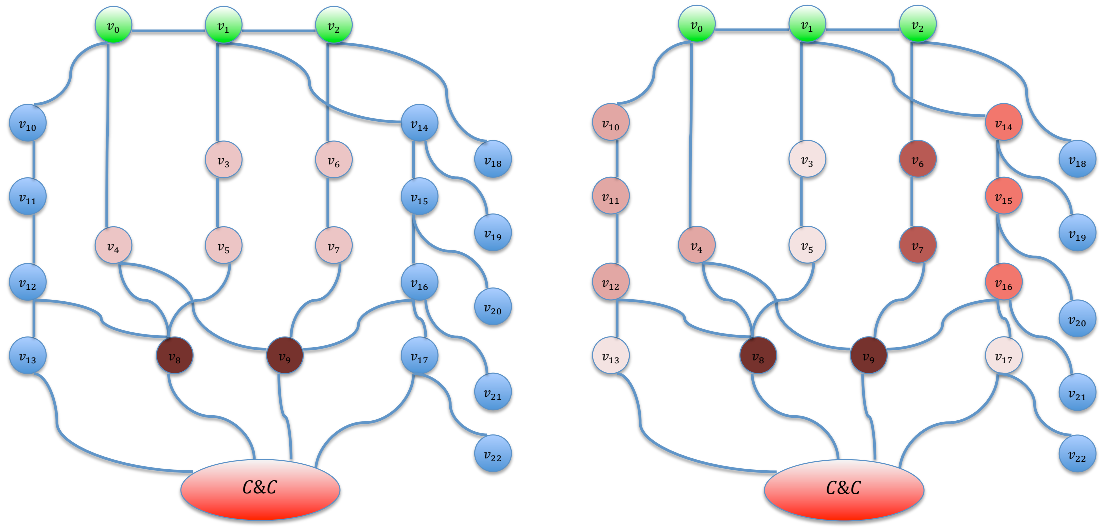
7. DEFENDER'S STRATEGIES

To illustrate the effectiveness of different defender strategies, consider the network shown in Fig. 1, which includes mission-critical nodes $N = \{v_0, v_1, v_2\}$. The attacker's objective is to relay data from any node in N to the C&C node. To protect mission-critical nodes from data exfiltration, we consider the following strategies for placing k detectors.

- *Static Iterative Centrality Placement*: In this strategy, the defender chooses nodes based on the iterative mission-betweenness centrality algorithm proposed in [25]. The defender first computes the mission-betweenness centrality of a node v as $C_M(v) = \sum_{(s,t) \in N \times \mathsf{C\&C}\ s.t.\ v \neq t} \frac{\sigma_{st}(v)}{\sigma_{st}}$, where σ_{st} is the number of shortest paths between s and t and $\sigma_{st}(v)$ is the number of those paths that go through v. Upon computing the mission-betweenness centrality for all the nodes, the defender chooses the node with the highest centrality for detector placement. For each subsequent detector placement, the centrality $C_M(v)$ of all non-detector nodes is re-computed and the node with the highest centrality among the non-detector nodes is picked for placing the next detector. In the example of Fig. 1, assume that the defender can place $k = 2$ detectors. Then, node v_9 (or v_8) will be chosen to the place the first detector followed by v_8 (or v_9) to place the second detector.

- *Uniform Random Placement*: The static nature of the above strategy enables an attacker to pre-compute the location of detectors and compromise nodes along a detector-free path to exfiltrate data. Therefore, in order to create uncertainty about the exact location of detectors, in this strategy, the defender chooses k nodes to place detectors uniformly at random.

- *Centrality-Weighted Placement*: Although the uniform random strategy introduces uncertainty for an attacker, it may consider nodes that do not lie on any simple path between mission-critical nodes and C&C. As a result, the uniform strategy may provide a low minimum detection probability. In this strategy, to



(a) (b)

Figure 3: Candidate detector locations for the network of Fig. 1, based on the (a) centrality-weighted strategy, and (b) expanded centrality-weighted strategy

improve detection guarantees, the defender places k-detectors by randomly choosing nodes according to a probability distribution that weights nodes based on their mission-betweenness centrality, i.e., nodes with higher values of $C_M(v)$ have more chances of being chosen for detector placement. For the example shown in Fig. 1, the nodes shaded in brown in Fig. 3(a) are the only nodes considered for detector placement by this strategy (the color intensity is proportional to the relative weight of the corresponding node).

- *Expanded Centrality-Weighted Placement*: One of the major drawbacks of the centrality-weighted strategy is that coverage of the exfiltration surface is limited. In fact, the strategy considers only the nodes on the set of all shortest paths between the mission-critical nodes and C&C. Therefore, in this strategy, the coverage of the exfiltration surface is expanded by considering all the nodes in the paths that are δ-times longer than the shortest paths. Let Π_e be the set of all such paths. The revised centrality of a node v is computed as $C_E(v) = \sum_{(s,t)\in N\times \mathsf{C\&C}\; s.t.\; v\neq t} \frac{\sigma'_{st}(v)}{\sigma'_{st}}$, where σ'_{st} is the number of simple paths in Π_e between s and t and $\sigma'_{st}(v)$ is the number of those paths that go through v. In the example of Fig. 1, when $\delta = 0.25$, the nodes shaded with different intensities of brown in Fig. 3(b) will be considered for randomizing the placement.

8. SIMULATION RESULTS

We evaluated the proposed strategies using real ISP network topologies obtained from the Rocketfuel dataset [20] and synthetic topologies generated using graph-theoretic properties of typical ISP networks.

The Rockfuel dataset provides router-level topologies for 10 ISP networks. For each network, Table 1 provides a summary of the total number of routers within the network and the number of external routers (located outside the ISP) to which the ISP routers are connected. As connections to external routers are outside the monitored domain, in our simulations, we considered a worst-case scenario and assumed that all the external routers are potentially routing traffic to C&C servers.

ASN	Name	No. of routers	No. of ext. routers
1221	Telstra	2998	329
1239	Sprintlink	8341	1004
1755	Ebone	605	310
2914	Verio	7102	2432
3257	Tiscali	855	444
3356	Level3	3447	1827
3967	Exodus	895	520
4755	VSNL	121	80
6461	Abovenet	2720	2066
7018	AT&T	10152	722

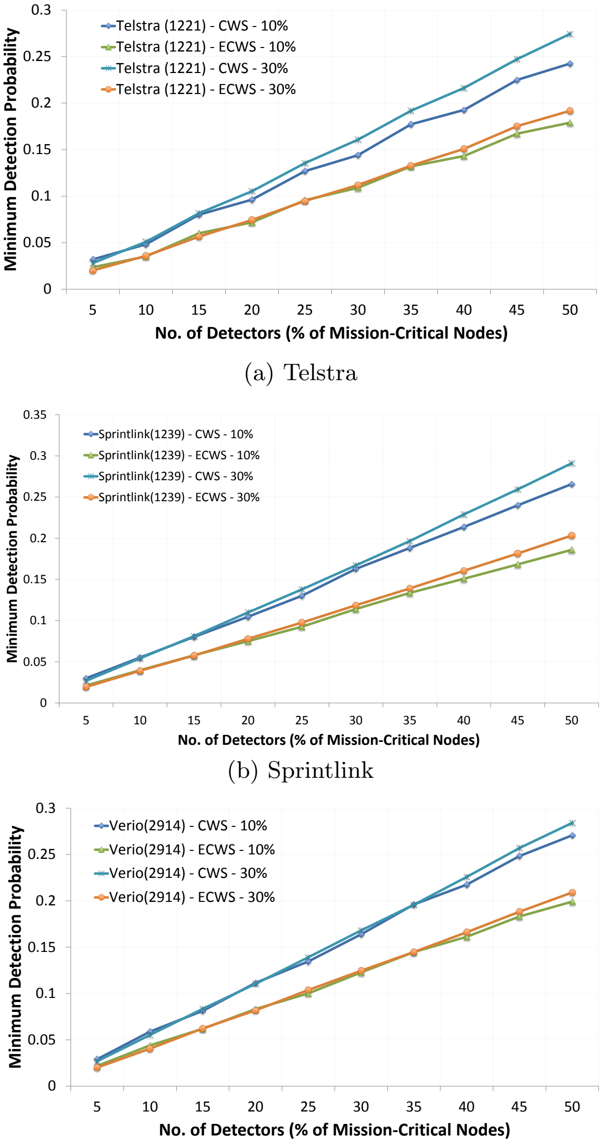
Table 1: Summary of ISP networks from [20]

To study the influence of network topology on the performance of a strategy, we evaluated these strategies using simulated medium-scaled ISP networks comprising 3,000 nodes. At the router-level, such networks are known to exhibit scale-free network properties wherein the degree distribution follows a power-law distribution [20]. In order to accurately capture the connectivity of an ISP network at the router level, the BRITE network topology generator [13] was used to generate these networks. Ten such networks were considered, with mission-critical nodes varying between 10%-30% of the network size and 500 C&C locations chosen at random for each network.

For the ISP networks from the Rocketfuel dataset, we varied the size of the detector set as a fraction of the number of mission-critical nodes while, for synthetic topologies, we varied the size of the detector set as a fraction of the network size. These simulations were intended to study the impact on the amount of resources that a network administrator might be willing to invest (either proportional to no. of nodes that need to be protected or size of the network) to detect exfiltration. In all our simulations, we set $\delta = 0.5$ for the expanded centrality-weighted strategy and tested the statistical significance of the presented results using paired t-test at 95% confidence interval. Finally, due to space constraints, we show the results for a subset of the topologies from the Rocketfuel dataset.

8.1 Minimum Detection Probability

As illustrated in Fig. 4 and Fig. 5, the probability of detecting exfiltration attempts increases linearly with the number of detectors. We observed that variations in the detection probability for different synthetic networks were less than 1% and hence, for sake of presentation, we only show mean values.



(a) Telstra

(b) Sprintlink

(c) Verio

Figure 4: Minimum detection probability for different networks using centrality-weighted (CWS) and expanded centrality-weighted (ECWS) strategies

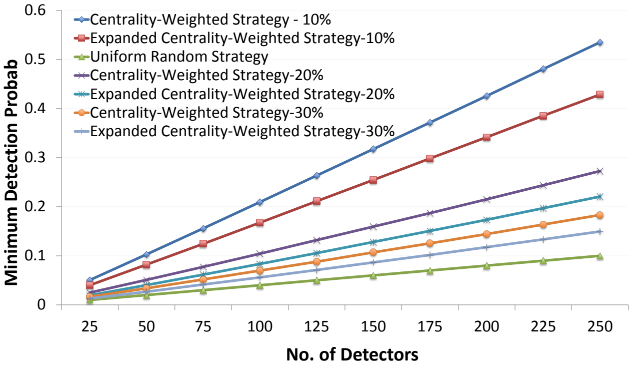


Figure 5: Minimum detection probability for different strategies using synthetic topologies

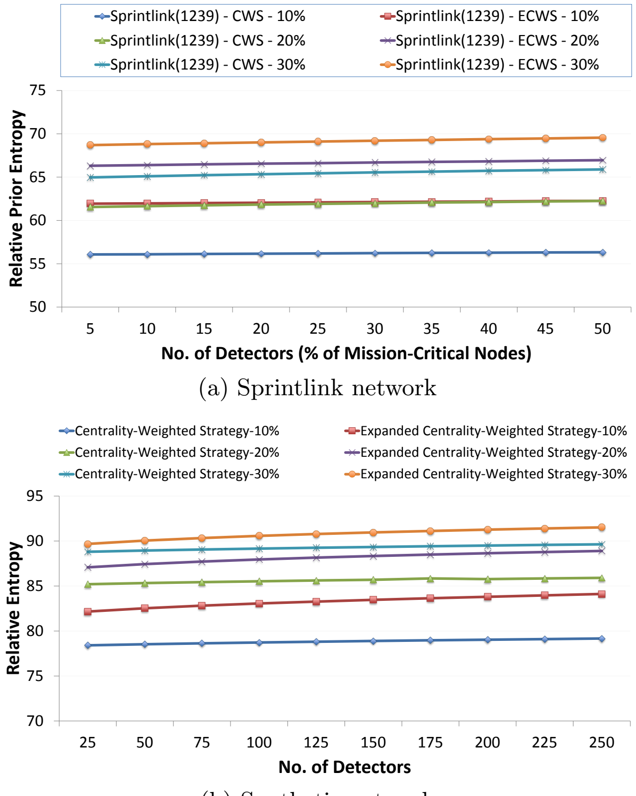
It can be observed that, independently of the number of mission-critical nodes and detectors, the centrality-weighted strategy outperforms the expanded centrality-weighted strategy. This trend can be attributed to the scale-free nature of the topology in which most of the paths traverse a small portion of the nodes. As the expanded strategy considers a larger sample space of paths and distributes the placement probability across the nodes on these paths, the nodes with high centrality will be chosen with a lower probability than in the case of the centrality-weighted strategy. In these simulations, we observed that the static iterative centrality strategy could not detect the exfiltration of the data segments in any of the networks as there was at least one detector-free path between one of the mission-critical nodes and a C&C location.

8.2 Attacker's Uncertainty

As mentioned earlier, among the detector placement strategies, the static iterative centrality strategy does not introduce any uncertainty whilst the uniform random strategy introduces the highest uncertainty in the location of detectors to an attacker. To study the attacker's uncertainty in the location of detectors due to the proposed strategies, we computed the relative entropy introduced by the centrality-weighted and the expanded centrality-weighted strategy w.r.t. the uniform random strategy. As shown in Fig. 6, the centrality-weighted strategy and its expanded version create an uncertainty that lies in-between the two ends of the entropy spectrum. In particular, as expanded strategy potentially considers more nodes, the number of combinations of detector locations, and hence the uncertainty introduced by it is higher than the centrality-weighted strategy. Therefore, for ISP networks, the choice of different centrality-weighted strategies potentially trades-off entropy for detection probability guarantees.

8.3 Running Time

In this section, we evaluate the performance of Algorithm 1 in computing the minimum detection probability and the performance of various detector placement strategies as a function of the network size. For each network size, we generated 10 ISP-type topologies, with 10% of network size as mission-critical nodes and 500 C&C locations chosen at random. We varied the number of detectors from 3% to 7% of network size and observed similar trend in the running time. For the sake of presentation, we present



(a) Sprintlink network

(b) Synthetic network

Figure 6: Relative increase in entropy for the attacker introduced by different strategies

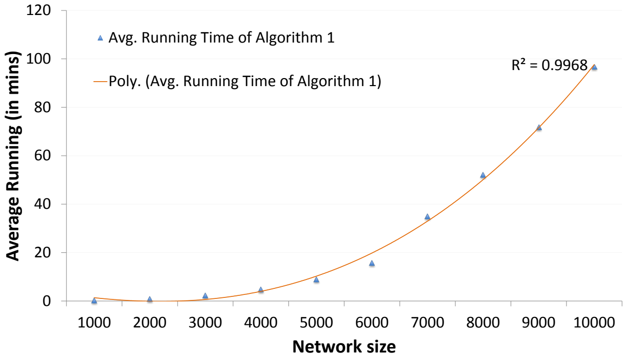


Figure 7: Running time for computing the minimum detection probability using Algorithm 1.

the results with 3% of network size for the total no. of available detectors. The runtime was averaged over the 10 graph settings for each network size. All experiments were conducted on an AMD Opteron processor with 4GB memory running Ubuntu 12.04.

Although, in theory, the worst-case running time of Algorithm 1 is $O(|V|^6)$, for practical network settings, it can be seen (line marked in orange in Fig. 7) that the execution time grows as $O(|V|^3)$ with an R^2 value of 0.99. Finally, as shown in Fig. 8, the dynamic strategy computes the detector locations faster than its static alternative. This is because, the static iterative centrality strategy has to re-compute the shortest paths multiple times to determine the location of the detectors.

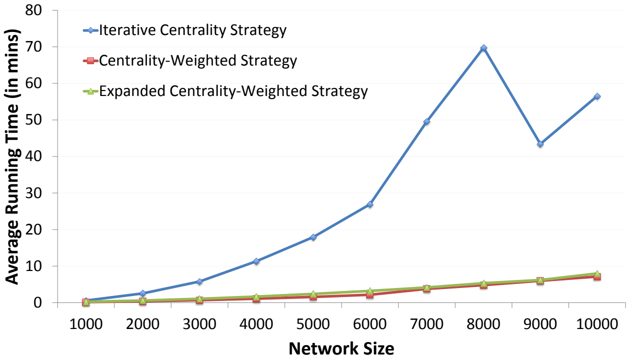


Figure 8: Running time for different detector placement strategies

9. CONCLUSIONS AND FUTURE WORK

In this paper, we have presented the key findings of our work on disrupting stealthy botnets through the use of a novel moving target defense approach. Specifically, we have targeted botnets that are being used for exfiltrating sensitive data from mission-critical systems. Defending against such botnets is challenging, as research has shown how they have become increasingly sophisticated and have the capability of operating in stealth mode by minimizing their footprint.

In order to defeat exfiltration attempts by modern botnets, we have proposed a moving target defense approach for placing detectors across the network – in a resource-constrained environment – and dynamically and continuously changing the placement of detectors over time. Specifically, we have proposed several strategies based on centrality measures that capture important properties of the network. Our objective is to increase the attacker's effort and likelihood of detection by creating uncertainty about the location of detectors and forcing the botmaster to perform additional actions in an attempt to create detector-free paths through the network.

We have presented two metrics to evaluate the proposed strategies – namely the *minimum detection probability* and the *attacker's uncertainty* – and an algorithm to compute the minimum detection probability. We validated our approach through simulations, and the results confirmed that the proposed solution can effectively reduce the likelihood of successful exfiltration campaigns.

Our future plans include but are not limited to: (i) introducing a probabilistic model to account for false negatives in the deployed detectors; (ii) defining and evaluating the performance of the proposed detector placement strategies against more sophisticated attacker's strategies; and (iii) casting the model in a game-theoretic framework to study the Nash equilibria and dominant strategies.

10. REFERENCES

[1] D. Andriesse and H. Bos. An analysis of the Zeus peer-to-peer protocol, 2013.

[2] E. B. Beigi, H. H. Jazi, N. Stakhanova, and A. A. Ghorbani. Towards effective feature selection in

machine learning-based botnet detection approaches. In *Proceedings of the 2014 IEEE Conference on Communications and Network Security (CNS 2014)*, pages 247–255. IEEE, 2014.

[3] M. P. Collins, T. J. Shimeall, S. Faber, J. Janies, R. Weaver, M. De Shon, and J. Kadane. Using uncleanliness to predict future botnet addresses. In *Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement*, pages 93–104. ACM, 2007.

[4] D. Dagon, G. Gu, C. P. Lee, and W. Lee. A taxonomy of botnet structures. In *Proceedings of the 23rd Annual Computer Security Applications Conference (ACSAC 2007)*, pages 325–339, 2007.

[5] D. Dittrich. So you want to take over a botnet. In *Proceedings of the 5th USENIX Conference on Large-Scale Exploits and Emergent Threats*, pages 6–6. USENIX Association, 2012.

[6] M. L. Fredman and R. E. Tarjan. Fibonacci heaps and their uses in improved network optimization algorithms. *Journal of the ACM (JACM)*, 34(3):596–615, 1987.

[7] G. Gu, R. Perdisci, J. Zhang, W. Lee, et al. BotMiner: Clustering analysis of network traffic for protocol-and structure-independent botnet detection. In *Proceedings of the 17th USENIX Security Symposium*, volume 5, pages 139–154, 2008.

[8] G. Gu, P. A. Porras, V. Yegneswaran, M. W. Fong, and W. Lee. BotHunter: Detecting malware infection through ids-driven dialog correlation. In *Proceedings of the 16th USENIX Security Symposium*, volume 7, pages 1–16, 2007.

[9] D. T. Ha, G. Yan, S. Eidenbenz, and H. Q. Ngo. On the effectiveness of structural detection and defense against p2p-based botnets. In *Proceedings of the IEEE/IFIP International Conference on Dependable Systems and Networks (DSN 2009)*, pages 297–306, 2009.

[10] A. Juels and T.-F. Yen. Sherlock Holmes and the case of the advanced persistent threat. In *Proceedings of the 5th USENIX Workshop on Large-Scale Exploits and Emergent Threats*, 2012.

[11] Kaspersky Labs. Kaspersky lab and ITU research reveals new advanced cyber threat. http://usa.kaspersky.com/about-us/press-center/press-releases/kaspersky-lab-and-itu-research-reveals-new-advanced-cyber-threat, May 2012.

[12] D. Mcwhorter. APT1: Exposing one of china's cyber espionage units. http://intelreport.mandiant.com/, 2013.

[13] A. Medina, A. Lakhina, I. Matta, and J. Byers. BRITE: An approach to universal topology generation. In *Proceedings of the 9th International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems*, pages 346–353. IEEE, 2001.

[14] G. C. Moreira Moura. *Internet Bad Neighborhoods*. PhD thesis, University of Twente, The Netherlands, March 2013.

[15] S. Nagaraja, P. Mittal, C. Hong, M. Caesar, and N. Borisov. BotGrep: Finding P2P bots with structured graph analysis. In *Proceedings of the 19th USENIX Security Symposium*, pages 95–110, 2010.

[16] C. Rossow, D. Andriesse, T. Werner, B. Stone-Gross, D. Plohmann, C. J. Dietrich, and H. Bos. Sok: P2pwned-modeling and evaluating the resilience of peer-to-peer botnets. In *Proceedings of the IEEE Symposium on Security and Privacy (SP 2013)*, pages 97–111. IEEE, 2013.

[17] Y. Shinoda, K. Ikai, and M. Itoh. Vulnerabilities of passive internet threat monitors. In *Proceedings of the 14th USENIX Security Symposium*, pages 209–224, 2005.

[18] V. Shmatikov and M.-H. Wang. Security against probe-response attacks in collaborative intrusion detection. In *Proceedings of the 2007 Workshop on Large Scale Attack Defense*, pages 129–136. ACM, 2007.

[19] G. Sinclair, C. Nunnery, and B. B. Kang. The waledac protocol: The how and why. In *Proceedings of the 4th International Conference on Malicious and Unwanted Software (MALWARE 2009)*, pages 69–77. IEEE, 2009.

[20] N. Spring, R. Mahajan, and D. Wetherall. Measuring ISP topologies with Rocketfuel. In *ACM SIGCOMM Computer Communication Review*, volume 32, pages 133–145. ACM, 2002.

[21] E. Stinson and J. C. Mitchell. Towards systematic evaluation of the evadability of bot/botnet detection methods. In *Proceedings of the 2nd USENIX Workshop on Offensive Technologies*, page 5. USENIX Association, 2008.

[22] B. Stone-Gross, M. Cova, L. Cavallaro, B. Gilbert, M. Szydlowski, R. Kemmerer, C. Kruegel, and G. Vigna. Your botnet is my botnet: analysis of a botnet takeover. In *Proceedings of the 16th ACM Conference on Computer and Communications Security (CCS 2009)*, pages 635–647. ACM, 2009.

[23] P. Sweeney and G. Cybenko. Identifying and exploiting the cyber high ground for botnets. In *Cyber Warfare*, volume 56 of *Advances in Information Security*, pages 37–56. Springer, 2015.

[24] P. J. Sweeney. *Designing Effective And Stealthy Botnets for Cybet Espionage And Interdiction - Finding the Cyber High Ground*. PhD thesis, Thayer School of Engineering, Darthmouth College, 2014.

[25] S. Venkatesan, M. Albanese, and S. Jajodia. Disrupting stealthy botnets through strategic placement of detectors. In *Proceedings of the 3rd IEEE Conference on Communications and Network Security (CNS 2015)*, pages 95–103. IEEE, 2015.

[26] Y. Zeng, X. Hu, and K. G. Shin. Detection of botnets using combined host- and network-level information. In *Proceedings of the the IEEE/IFIP International Conference on Dependable Systems and Networks (DSN 2010)*, pages 291–300, June 2010.

[27] J. Zhang, R. Perdisci, W. Lee, X. Luo, and U. Sarfraz. Building a scalable system for stealthy P2P-botnet detection. *IEEE Transactions on Information Forensics and Security*, 9(1):27–38, 2014.