2015 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks

Detection of Early-Stage Enterprise Infection by Mining Large-Scale Log Data

Alina Oprea*, Zhou Li*, Ting-Fang Yen†, Sang H. Chin‡ and Sumayah Alrwais§
*RSA Laboratories, Cambridge, MA, USA, Email: {alina.oprea; zhou.li}@rsa.com
†E8 Security, Palo Alto, CA, USA, Email: tingfang.yen@gmail.com
‡Draper Laboratory, Cambridge, MA, USA and Boston University, Boston, MA, USA, Email: schin@draper.com
§Indiana University, Bloomington, IN, USA, Email: salrwais@umail.iu.edu

Recent years have seen the rise of sophisticated attacks including advanced persistent threats (APT) which pose severe risks to organizations and governments. Additionally, new malware strains appear at a higher rate than ever before. Since many of these malware evade existing security products, traditional defenses deployed by enterprises today often fail at detecting infections at an early stage.

We address the problem of detecting early-stage APT infection by proposing a new framework based on belief propagation inspired from graph theory. We demonstrate that our techniques perform well on two large datasets. We achieve high accuracy on two months of DNS logs released by Los Alamos National Lab (LANL), which include APT infection attacks simulated by LANL domain experts. We also apply our algorithms to 38TB of web proxy logs collected at the border of a large enterprise and identify hundreds of malicious domains overlooked by state-of-the-art security products.

I. Introduction

The cybersecurity landscape is evolving constantly. More sophisticated attacks including Advanced Persistent Threats (APTs) [13], [34], [1], [2] have emerged recently targeting organizations' intellectual property, financial assets, and national security information. Well-funded attackers use advanced tools and manually orchestrate their campaigns to adapt to the victim's environment and maintain low profiles of activity. Additionally there are also more malware than ever before. A whitepaper published by Panda Labs [30] found 30 million new malware strains in circulation in 2013 alone, at an average of 82,000 malware a day. Many of these are variants of known malware designed to evade existing security products, such that existing defenses, e.g., anti-virus, firewalls, intrusion detection systems, often fail at detecting early-stage infections [26].

However, certain *infection patterns* still persist across different malware and APT families due to the typical infection vectors used by attackers. For example, during the malware *delivery stage*, victim hosts often visit several domains under the attacker's control within a short period of time as a result of redirection techniques employed by attackers to protect their malicious infrastructures [37]. After delivery, backdoors are installed on the compromised machines to allow *footholds* into the targeted organization [26], where the machines initiate outbound connections regularly to a command-and-control (C&C) server to receive instructions from the attacker. Malware communications commonly take place over HTTP/HTTPS, since web traffic is typically allowed by firewalls. More importantly, domains used in the same attack campaign are often related, sharing locality in either IP address space, time of access or set of hosts contacting them. These patterns of infections have been observed in targeted attacks (e.g., APT1 group [26], Shady RAT [20], Mirage [11]), as well as botnet infections (e.g., Zeus, Citadel [12] and ZeroAccess [23]).

In this work, we leverage these observations to detect early-stage malware and APT infections in enterprise networks, in particular suspicious communications to external destinations initiated by internal hosts. We propose a graph-theoretic framework based on belief propagation [32] to identify *small communities* of related domains that are indicative of early-stage malware infections. We first restrict our attention to traffic destined to *rare destinations*. These are "new" domains, not visited before by any host in the organization within an observation window (and thus more likely to be associated with suspicious activity), and contacted by a small number of internal hosts. In each iteration of our belief propagation algorithm, the rare domains are scored according to several features and similarity with domains detected in previous iterations. The weight of each feature used in scoring a domain is computed using linear regression during a training stage. Our techniques are unique in combining unsupervised learning techniques (belief propagation), with a supervised learning method (linear regression) for detecting new infections when limited ground truth is available.

Our algorithm can be applied either with "hints" (starting from "seeds" of known compromised hosts or domains), or without (without prior knowledge of malicious activity). In the first case, seeds can be obtained from commercial blacklists containing Indicators of Compromise (IOCs) that the enterprise security operations center (SOC) has access to. Currently, SOC security analysts manually investigate incidents starting from IOCs, and we aim here to facilitate this process. In the latter case, we first identify automated connections indicative of C&C activity using both enterprise-specific and generic features. Domains labeled as potential C&C servers are used as seeds in the same belief propagation algorithm to detect other related domains that belong to the same attack campaign.

We demonstrate the effectiveness of our techniques on two different datasets, one containing DNS records and the other web proxy logs. The first consists of two months (1.15TB) of anonymized DNS records from Los Alamos National Lab (LANL) in early 2013. This dataset also includes 20 independent APT-like infection attacks simulated by LANL domain experts and was released along with a challenge (*APT*

978-1-4799-8629-3/15 $31.00 © 2015 IEEE
DOI 10.1109/DSN.2015.14

Infection Discovery using DNS Data [14]) requesting methods to detect compromised internal hosts and external domains in the simulated attacks. The challenge included "hints" of varying details (e.g., one or multiple compromised hosts), as well as answers for validation. Our techniques proved effective at detecting the LANL simulated attacks achieving an overall 98.33% true detection rate, at the cost of low false positives.

Our second dataset contains two months of web proxy logs collected from a large enterprise in early 2014 (38.41TB of data). Through careful manual analysis in collaboration with the enterprise SOC, we confirm that a large percentage of domains identified by our algorithms (289 out of 375) are related to malicious or suspicious activities (with false positive rate on the order of $10^{-4}\%$). Interestingly, a large number of them (98) are entirely new discoveries, not yet flagged by VirusTotal even several months after we detected them. This demonstrates the ability of our techniques to detect entirely new attacks overlooked by state-of-the-art security products.

To summarize our main contributions in the paper are:

Belief propagation framework for detecting enterprise infection. We develop a graph-theoretic framework based on belief propagation for detection of early-stage enterprise infections. Given "seed" hosts or domains, we automatically infer other compromised hosts and related malicious domains likely part of the same campaign. Our approach uniquely leverages relationships among domains contacted in multiple stages of the infection process and utilizes a novel combination of unsupervised and supervised learning techniques.

Detector of command-and-control communication in enterprise. By exploiting novel enterprise-specific features and combining them with other generic features of malicious activity, we build a detector of C&C communication tailored to an enterprise setting. Domains labeled as C&C can be seeded in the belief propagation algorithm to detect other related domains.

Solve the LANL challenge. We apply the belief propagation algorithm to the LANL challenge and identify the malicious domains in the 20 simulated APT campaigns with high accuracy and low false positive and false negative rates.

Evaluate on real-world data from large enterprise. We apply our solution to a large dataset (38.41 TB) of web proxy logs collected at an enterprise's network border. We identify hundreds of domains contacted by internal enterprise hosts not detected previously by state-of-the-art security products. Through careful manual investigation, we confirm that a large percentage (77.07% out of 375 domains) are related to various malicious or suspicious activities. While 191 domains (50.93%) are also reported by VirusTotal (but unknown to the enterprise), we identify 98 (or 26.13%) new discoveries (domains not reported by VirusTotal or the enterprise).

II. Problem Statement

Our goal is to detect early-stage malware and APT infection within an organization, in particular suspicious communications to external destinations initiated by internal hosts. We describe below characteristics of common enterprise infections, why existing solutions fail against such threats and the challenges we had to overcome to detect them.

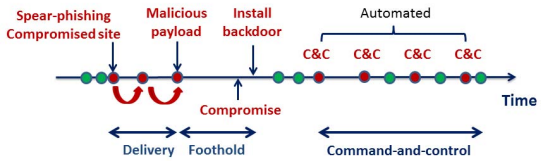

Fig. 1: Timeline of common infection patterns on a compromised host. Red (darker) circles are connections to malicious domains, and green (lighter) circles to legitimate ones.

A. Enterprise Infections

Common infection vectors used in targeted attacks are social engineering [26] and compromise of legitimate sites [40]. In the case of social engineering, attackers craft spear-phishing email addressed to several employees within the targeted organization including a malicious attachment or a hyperlink to a malicious file. Attack vectors employed by mainstream malware include spam emails, USB drives, and a variety of web-based attacks (e.g., drive by download, clickjacking, malvertising, etc.). Many of these attacks (both targeted and mainstream) include similar stages in the infection process [26], [20], [11], [12] depicted in Figure 1:

Delivery stage: During delivery, the victim machine gets the malicious payload, for example by an email attachment, or drive-by-download attack, etc. Typically the victim visits several malicious domains within a short time interval as a result of redirection employed by attackers [37]. Many times, the initial malware is generic (e.g., performs system reconnaissance) and downloads additional *second-stage* malware specifically crafted for the victim environment [39].

Establishing foothold: After delivery a backdoor is usually installed on the victim's machine and the attacker establishes a foothold within the organization [26]. In almost all cases, backdoors initiate outbound connections to evade firewalls that block connections from outside the network. Most communications go through HTTP or HTTPs since these ports are allowed by most enterprise firewalls [35], [28].

Command-and-control (C&C): Typically, backdoors connect regularly to the command-and-control center operated by attackers to receive further instructions and allow attackers backdoor access into the victim environment [26], [12].

Based on a thorough analysis of many published reports, and discussion with a large enterprise SOC, we extract several common characteristics of enterprise infections:

Uncommon domains: Attackers tend to use domains under their control for different stages of the campaign (e.g., delivery, C&C) [26]. These domains are uncommon destinations, with low volume of traffic directed to them globally. Additionally, [26] points out that attackers use more frequently domain names rather than direct IP connections for their C&C communication so that they can dynamically flux the domains.

Communities of domains: A compromised host usually contacts several malicious domains within a relatively short time interval. For instance, a user clicking on an embedded link in an email might visit the front-end attacker site, get redirected to a site hosting malicious payload and shortly after the backdoor is established will initiate the first connection to the C&C

server. These domains form *small communities* exhibiting similarity in connection timing, set of hosts contacting them (if multiple hosts are infected in the same campaign) and sometimes proximity in IP address space [19], [26].

Automated C&C communication: Backdoors typically communicate with C&C servers on a regular basis to allow attackers access into the victim environment. In many publicized APT campaigns (e.g., NightDragon [10], Mirage [11], Shady RAT [20]) as well as botnet infections (e.g., Zeus, Citadel [12], ZeroAccess [23]), C&C communication occurs at fairly regular time intervals (minutes or hours) with small amount of randomization. We also examined malware samples provided by Mandiant on the APT1 group. Among 43 backdoor samples, the vast majority (39) exhibit fairly regular timing patterns.

HTTP/HTTPs Communication. The communications to C&C servers is typically done through HTTP or HTTPs since other ports are blocked by enterprise firewalls [35], [28].

B. Current Defenses

Enterprises deploy different security products (anti-virus, intrusion-detection, etc.), most of which perform signature-based detection: they extract signatures from malware samples (e.g., MD5 of binary file) and match them against new activity. Additionally, the enterprise SOC relies heavily on commercial blacklists to block destinations with known malicious activities. Both signature-based detection and blacklisting can be easily evaded by attackers, e.g., by obfuscating malicious binaries or registering new domains. However, attackers tend to reuse code and successful infiltration techniques across different campaigns [15]. It is this observation that we leverage to propose new behavior-based detection methods that capture most common infection patterns reported in the literature.

C. Challenges

There were several challenges we had to overcome in the process of developing our detection methodology. First, security products deployed within the enterprise perimeter record large volumes of data daily. For example, the two datasets we used to evaluate our system are 1.15 TB and 38.14 TB, respectively. To perform efficient analysis, we describe in Section IV-A a suite of techniques that reduce the data volume by an order of magnitude while retaining the communication structure between internal hosts and external domains.

Second, sophisticated attacks tend to stay "under the radar" and easily blend in with millions of legitimate requests. There is an inherent tension between detecting stealthy attacks and reducing false positives. We carefully selected parameters of our various algorithms to achieve extremely low false positives (on the order of $10^{-4}\%$). This results in a manageable number of incidents referred to the SOC for further analysis.

Finally, limited ground truth is available for enterprise infections since the only way they are identified is when they are detected and blocked (by anti-virus, intrusion detection tools, or blacklists). To overcome this problem, we propose a novel combination of unsupervised and supervised learning techniques described in Section III-A. The evaluation shows that our approach is successful at identifying new, unknown infections not detected by state-of-the-art security products.

III. Methodology

In this section, we provide an overview of our approach to detecting early-stage enterprise infection. Our system analyzes log data collected at the enterprise border on a daily basis, maintains profiles of normal activity within the enterprise, and detects malware infections by exploiting the relationship between suspicious external destinations used in different infection stages. We introduce our main framework based on belief propagation here and present the details of our techniques in Section IV.

A. Belief propagation framework

We model the communication between internal hosts in the enterprise and external domains with a bipartite graph having two types of vertices, hosts and domains. An edge is created between a host and a domain if the host contacts the domain at least once during the observation window (e.g., one day). The communication graph is created from either the DNS or web proxy logs captured at the border of the enterprise.

To keep the size of the communication graph manageable we apply a number of data reduction techniques, mainly restricting to *rare domains* and hosts contacting them. Rare domains are those contacted by a small number of hosts and are newly observed in that enterprise's traffic (making them more prone to suspicious activities than legitimate, popular destinations). They are determined after profiling the enterprise traffic for a given period (e.g., a month) to construct a history of destinations contacted by internal hosts.

To detect the infection patterns depicted in Figure 1, our main insight is to apply a graph theoretic technique called *belief propagation* [32]. Belief propagation is a graph inference method commonly used to determine the label of a graph node given prior knowledge about the node itself and information about its graph neighbors. The algorithm is based on iterative message-passing between a node and its neighbors until convergence or when a specified stopping condition is met.

As described in Section II-A, our main goal is to detect *communities of malicious domains* with similar features that are likely part of the same campaign. We adapt the general belief propagation framework to this task, by starting from a seed of known malicious domains or hosts, and iteratively computing scores for other rare domains contacted by known compromised hosts. The score for a domain is computed based on 1) the degree to which the domain exhibits C&C-like behavior (described in Section IV-C), *and* 2) its similarity to labeled suspicious domains from previous iterations of the algorithm. The final domain score is computed as a weighted sum of features, where the weights are determined through a supervised approach (based on linear regression). More details about domain similarity scoring are provided in Section IV-D.

The algorithm proceeds iteratively and builds the communication graph *incrementally* (for efficiency reasons). In each iteration, the algorithm computes scores for those rare domains contacted by compromised hosts, and labels the domains with the highest scores as suspicious. These domains are added to the graph together with the set of hosts contacting them. The algorithm terminates when the score of the top-ranking domain is below a threshold, or when the maximum number

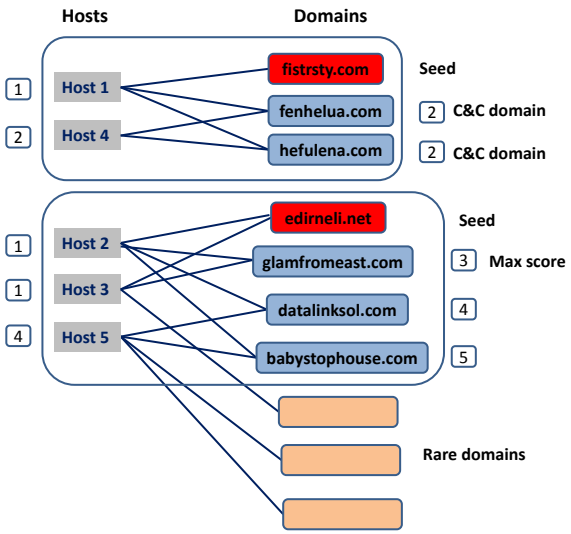

Fig. 2: Application of belief propagation.

of iterations is reached, and returns a list of labeled malicious domains ordered by suspiciousness level.

The belief propagation algorithm depends on a number of parameters (e.g., threshold for C&C communication, threshold for domain similarity, etc.). These parameters are tailored to a particular enterprise after training for a month to determine the contribution of each relevant feature in computing the specific values for that enterprise.

B. Example

Figure 2 shows an example of applying the belief propagation algorithm. Starting from two seed domains (marked in red), three hosts (Hosts 1-3) are added to the graph in the first iteration. In second iteration, two C&C domains contacted by Host 1 are detected and added to the graph, as well as another host contacting these domains. In third iteration, in absence of additional C&C activity, the domain of maximum score among all rare domains contacted by Hosts 1-4 is added to the graph. The algorithm continues to score rare domains visited by hosts in the graph and incrementally build the bipartite graph until the stopping condition is met. In this example, two distinct communities of malicious domains belonging to two attack campaigns are highlighted.

C. Modes of operation

Our detection method operates in two modes. In the first, called *SOC hints*, we use the incidents that the enterprise SOC investigated as starting points (or seeds) in the belief propagation algorithm. Given either hosts or domains confirmed malicious, the algorithm identifies other related malicious domains (likely part of the same campaign) and internal compromised hosts that were unknown previously. This mode automates the manual investigation process that the SOC team performs and captures relationships between domains used by attackers in different stages of a campaign.

In the *no-hint* mode, we don't leverage existing seeds of known malicious activity. Our insight here is that C&C communications are automated, high-frequency activities distinctive from human-generated behavior (e.g., user visiting a site or clicking a link in an email). We develop a new C&C communication detector (whose details are given in Section IV-C) that utilizes a combination of enterprise-specific and generic features. Interestingly, the detected C&C domains and the hosts contacting them can be used to seed the same algorithm and identify related suspicious domains and compromised hosts.

The output in both modes of operation is a list of suspicious domains in decreasing order of their scores and the list of hosts contacting them. These are presented to the enterprise SOC for further investigation after additional context information (e.g., domain registration) is added to help the analyst during investigation.

IV. System Details

After providing an overview of our system, we give here more technical details of our methods.

A. Datasets, normalization and reduction

LANL dataset. The first dataset we used consists of anonymized DNS logs collected from the LANL internal network over 2 months (February and March 2013). It includes DNS queries initiated by internal hosts, responses from the LANL DNS servers, event timestamps, and IP addresses of the sources and destinations. All of the IP addresses and domain names are anonymized consistently. The dataset also includes 20 simulated attack campaigns representative of the initial stages of APT infection.

The LANL dataset consists of 3.81 billion DNS queries and 3.89 billion DNS responses, amounting to 1.15 TB. To allow efficient analysis, we employ a number of data reduction techniques. We first restrict our analysis only to A records, as they record the queries to domain names and their responses (IP addresses) and information in other records (e.g., TXT) is redacted and thus not useful. This step prunes 30.4% of DNS records on average per day. We also filter out queries for internal LANL resources (as our focus is on detecting suspicious external communications), and queries initiated by mail servers (since we aim at detecting compromised hosts).

***AC* dataset.** The second dataset *AC* consists of two months (January and February 2014) of logs collected by web proxies that intercept HTTP/HTTPs communications at the border of a large enterprise network with over 100,000 hosts. The logs include the connection timestamp, IP addresses of the source and destination, full URL visited, and additional fields specific to HTTP communications (HTTP method, status code, user-agent string, web referer, etc.). We also obtained a list of domain IOCs used by the enterprise SOC.

Analyzing the *AC* web proxy dataset proved difficult due to its large scale and various inconsistencies. On average 662GB of data is generated daily, resulting in a total of 38.14TB of data over two months. This dataset is 33 times larger than the LANL dataset, and much richer in information. However, the *AC* dataset has some inconsistencies due to multiple time zones of collection devices and dynamic assignment of IP addresses. We omit here a description of our normalization procedure, but we converted all timestamps into UTC and IP addresses to hostnames (by parsing the DHCP and VPN logs collected by

the organization). We then extract the *timestamp*, *hostname*, *destination domain*, *destination IP*, *user-agent string*, *web referer* and *HTTP status code* fields for our analysis. We do not consider destinations that are IP addresses.

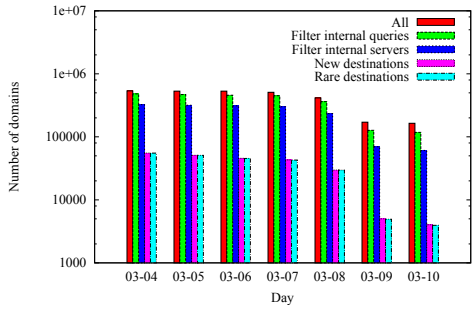

Fig. 3: The number of domains encountered daily in LANL after data reduction for the first week of March.

Rare destinations. In the analysis and results presented in the following sections, we focus on "rare" destinations in our datasets. Our insight is that popular legitimate websites (visited by a large user population) are better administered and less likely to be compromised, but connections to uncommon destinations may be indicative of suspicious behavior. More specifically, we define rare destinations as: *new domains* (not visited before by any internal hosts) that are also *unpopular* (visited by a small number of internal hosts). We set the threshold at 10 hosts based on discussion with the SOC.

To determine the rare destinations, we use the first month of data for profiling and build a history of external destinations visited by internal hosts. We "fold" the domain names to second-level (e.g., `news.nbc.com` is folded to `nbc.com`), assuming that this captures the entity responsible for the domain. We maintain a history of (folded) destinations queried by internal hosts, updated at the end of each day to include all new domains from that day. A domain is considered new on a particular day if it is not in the history.

Following the steps detailed above, we greatly reduce the volume of data as shown in Figure 3. On average, starting from 80K hosts and 400K domains in the LANL dataset, we retain only 3.3K hosts and 31.5K domains after reduction. In the *AC* dataset, we reduce from 120K hosts and 600K domains to an average of 20K hosts and 59K rare domains daily.

B. Belief Propagation Algorithm

The goal of the belief propagation (BP) algorithm, as explained in Section III-A, is to detect communities of malicious domains that belong to the same attack campaign. The BP algorithm can be applied in two modes: with hints of compromised hosts provided by SOC, or without hints. In the latter case the C&C communication detector is run first to identify a set of potential C&C domains and hosts contacting them. These are given as seeds to the same BP algorithm. Algorithm 1 gives pseudocode for BP starting from a set of compromised hosts $\mathcal{H}$, and set of malicious domains $\mathcal{M}$.

The algorithm maintains several variables: $\mathcal{R}$ the set of rare domains contacted by hosts in $\mathcal{H}$ and $\mathcal{N}$ the set of newly labeled malicious domains (in a particular iteration). In each iteration, the algorithm first detects suspicious C&C-like domains among set $\mathcal{R}$ using function Detect_C&C whose exact implementation will be provided next section. If no suspicious C&C domains are found, the algorithm computes a similarity score for all rare domains in $\mathcal{R}$ with function Compute_SimScore. The domain of maximum score (if above a certain threshold T_s) is included in set $\mathcal{M}$. Finally the set of compromised hosts is expanded to include other hosts contacting the newly labeled malicious domain(s). The algorithm iterates until the stopping condition is met: either no new domains are labeled as malicious (due to their scores being below the threshold) or the maximum number of iterations has been reached. The output is an expanded lists of compromised hosts $\mathcal{H}$ and malicious domains $\mathcal{M}$.

It's important to note that domain scores are computed as weighted sums of features, where the weights are determined through supervised learning (using linear regression). Thus, the algorithm is a novel combination of belief propagation, an unsupervised graph inference algorithm, with a supervised learning method.

Algorithm 1 [Belief Propagation]

```
/* H ← set of seed hosts */
/* M ← set of seed domains */
/* dom_host is a mapping from a domain to set of hosts contacting it */
/* host_rdom is a mapping from a host to set of rare domains visited */
function BELIEF_PROPAGATION(H,M):
    R ← ∪_{h∈H} host_rdom[h]
    while (not stop_condition) do
        N ← ∅      /* set of newly labeled malicious domains */
        for dom in R \ M do
            if Detect_C&C(dom) then
                N ← N∪ {dom}
                R ← R\ {dom}
        if N = ∅ then
            for dom in R \ M do
                score[dom] ← Compute_SimScore(dom)
            max_score ← max(score[dom])
            D ← domains of maximum score
            if max_score ≥ T_s then
                N ← N ∪ D
        if N ≠ ∅ then
            M ← M ∪ N
            H ← H ∪ (∪_{d∈N} dom_host[d])
            R ← R ∪ (∪_{h∈H} host_rdom[h])
```

C. Detection of C&C communication

Dynamic histograms. As discussed in Section II-A backdoors initiate automated communication with C&C domains to allow attackers access into the victim environment. We aim at detecting automated connections with fairly regular timing patterns, but be resilient to outliers (e.g., large gaps in communication) and randomization between connections. For every rare domain contacted by a host with a certain minimum frequency (set at 4) during the daily observation window we generate the histogram of inter-connection intervals and compare it to that of a periodic distribution.

To be resilient to bin alignment we propose a *dynamic histogram* method. We set up a maximum bin width W and cluster the inter-connection intervals of successive connections from a host to a domain (using a Greedy approach). We then define the bins dynamically from the generated clusters. We compare the resulting histogram with that of a periodic distribution with period equal to the highest-frequency interval. For comparing the two histograms we choose the Jeffrey

divergence metric motivated by the fact that it is "numerically stable and robust to noise and size of histogram bins" [36]. Finally we label the communications between a host and a domain automated if the statistical distance between the two histograms is at most J_T. The bin width W and threshold J_T control the resiliency of the method to outliers and randomization between connections. We discuss their selection according to the LANL dataset in Section V-B.

Additional features. For each rare automated domain we extract six additional features for the C&C detector:

Domain connectivity features: We consider the number of hosts contacting the domain (NoHosts) called *domain connectivity* and the number of hosts with automated connections to the domain (AutoHosts). The intuition here is that most rare legitimate domains are contacted by only one host, but the probability of multiple hosts contacting a rare domain increases when the hosts are under the control of the same attacker.

Web connection features: Based on discussions with SOC, web connections with no referer may indicate automated connections (not initiated by a user). To capture this, we include a feature NoRef denoting the fraction of hosts (among all hosts contacting that domain) that use no web referer.

Software configurations in an enterprise are more homogenous than in other networks (e.g., university campus), and as such we expect that most user-agent strings are employed by a large population of users. With this intuition, the *rare user-agent strings*, those used by a small number of hosts, might indicate unpopular software installed on the user machine which can potentially be associated with suspicious activities. We consider a feature RareUA denoting the fraction of hosts that use no UA or a rare UA when contacting the domain.

To determine the popularity of UA strings, we maintain a history of UAs encountered across time and the hosts using those UAs. The UA history is built during the training phase for a period of one month and then updated daily based on new ingested data. A UA is considered rare (after the training period of one month) if it is used by less than a threshold of hosts (set at 10 based on SOC recommendation).

Registration data features: Attacker-controlled sites tend to use more recently registered domains than legitimate ones [25]. In addition, attackers register their domains for shorter periods of time to minimize their costs in case the campaign is detected and taken down. We query WHOIS information and extract two features: DomAge (number of days since registration), and DomValidity (number of days until the registration expires).

Scoring automated domains. We employ a supervised learning model for computing domain scores. We found 841 automated rare domains in the AC dataset in February. We split this data into two sets, the first two weeks used for training and the last two weeks for testing. We also extract the six features described above and query VirusTotal to get an indication of the domain's status. Domains with VirusTotal score greater than 1 are labeled as "reported" and other domains as "legitimate".

Using the set of domains in the training set, we train a linear regression model to predict the label of a domain (reported or legitimate). The regression model outputs a weight

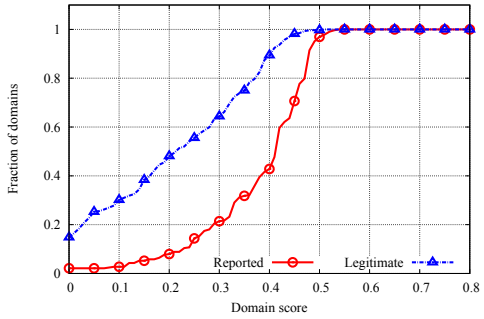

Fig. 4: CDFs of reported and legitimate domain scores.

for each feature, as well as the significance of that feature. The final score for each automated domain is a linear combination of feature values weighted by regression coefficients. The higher the score, the more suspicious the domain. Among all six features, the only one with low significance was AutoHosts, which is highly correlated with NoHosts and we decide to omit it. The most relevant features found by the model are DomAge and RareUA. DomAge is the only one negatively correlated with reported domains (as they are in general more recently registered than legitimate ones), but all other features are positively correlated.

Based on the trained model, we select a threshold for domain scores above which a domain is considered potential command-and-control. The graph in Figure 4 shows the difference between the scores of automated domains reported by VirusTotal and legitimate ones on the training set. For instance, selecting a threshold of 0.4 for labeling an automated domain suspicious results in 57.18% of reported domains being correctly predicted on the training set (at the cost of 10% false positive rate among legitimate domains). Similar results are obtained on the testing set. Our final goal is not identifying **all** automated domains reported by VirusTotal, but rather most suspicious ones to bootstrap the BP algorithm.

Thus we implement function Detect_C&C from Algorithm 1 as returning 1 if the domain score is above the threshold selected during training and 0 otherwise. We emphasize that the selection of feature weights and threshold on domain score is customized to each enterprise during the training stage.

D. Domain similarity

With the goal of capturing infection patterns from Figure 1, we consider a number of features when computing similarity of a domain D with a set of domains S labeled malicious in previous iterations of BP.

Domain connectivity. We use the domain connectivity as defined above.

Timing correlations. Second, we consider features related to the time when the domain D was visited by internal hosts. During initial infection stage of a campaign, we suspect that a host visits several domains under the attacker's control within a relatively short time period (as explained in Section II-A). We thus consider the minimum timing difference between a host visit to domain D and other malicious domains in set S. The shorter this interval, the more suspicious D is.

IP space proximity. Third, we consider proximity in IP space between D and domains in set S. Proximity in the IP /24 and /16 subnets is denoted by IP24 and IP16 respectively. The intuition here is that attackers host a large number of malicious domains under a small number of IP subnets [19], [26].

We provide measurement of the timing and IP proximity features on the LANL dataset in Section V-B.

Finally, the domain similarity score is tailored to the particular enterprise during the training stage. To obtain a list of (non-automated) rare domains and their features, we start from a set of compromised hosts (contacting C&C domains confirmed by VirusTotal). We include each rare domain contacted by at least one host in this set, extract its features, query VirusTotal to get an indication of its status (reported or legitimate), and divide the data into training and testing set, covering the first and last two weeks of February, respectively.

We apply again linear regression on the training set to determine feature weights and significance. Among the eight features described above, the only one with low significance was IP16, as it's highly correlated with IP24. The most relevant features identified by the model are RareUA, DomInterval, IP24 and DomAge. The threshold for domain score similarity is selected based on the balance between true positive and false positives (we omit the score PDF due to space limitations).

We implement function Compute_SimScore from Algorithm 1 as returning 1 if the domain similarity score is above the chosen threshold and 0 otherwise.

E. Putting it all together

Our system for detecting early-stage enterprise infection consists of two main phases: training (during a one-month bootstrapping period) and operation (daily after the training period). An overview diagram is presented in Figure 5.

Training. During the training period a benchmark of normal activity for a particular enterprise is created. It consists of several steps.

(1) *Data normalization and reduction:* The first stage processes the raw log data (either HTTP or DNS logs) used for training and applies normalization and reduction techniques.

(2) *Profiling:* Starting from normalized data, the system profiles the activity of internal hosts. It builds histories of external destinations visited by internal hosts as well as user-agent (UA) strings used in HTTP requests (when available). These histories are maintained and incrementally updated during the operation stage when new data is available.

(3) *Customizing the C&C detector:* The detector of C&C communication is customized to the particular enterprise.

(4) *Customizing the domain similarity score:* The domain similarity score used during belief propagation is also customized to the enterprise during the training phase.

Operation. After the initial training period, the system enters into daily operation mode. Several stages are performed daily:

(1) *Data normalization and reduction:* The system performs normalization and reduction for new log data.

Case	Description	Dates	Hint Hosts
1	From one hint host detect the contacted malicious domains.	3/2, 3/3, 3/4, 3/9, 3/10	One per day
2	From a set of hint hosts detect the contacted malicious domains.	3/5, 3/6, 3/7, 3/8, 3/11, 3/12, 3/13	Three or four per day
3	From one hint host detect the contacted malicious domains and other compromised hosts.	3/14, 3/15, 3/17, 3/18, 3/19, 3/20, 3/21	One per day
4	Detect malicious domains and compromised hosts without hint.	3/22	No hints

TABLE I: The four cases in LANL challenge problem.

(2) *Profile comparison and update:* New data is compared with historical profiles, and rare destinations, as well as rare UAs (used by a small number of hosts) are identified. Histories are updated with new data, to capture drift in normal behavior.

(3) *C&C detector:* The C&C detector is run daily, and scores of automated domains are computed with weights determined during training. Automated domains with scores above a threshold are labeled as potential C&C domains.

(4) *Belief propagation:* The belief propagation algorithm is run in either of two modes. The output is an ordered list of suspicious domains presented to SOC for further investigation.

V. Evaluation on the LANL Dataset

We start by describing the LANL challenge and then we show how we adapted our techniques to the anonymized LANL dataset. Still, using fewer features we demonstrate that our belief propagation framework achieves excellent results on the LANL challenge.

A. The LANL Challenge Problem

The LANL dataset includes attack traces from 20 independent infection campaigns simulated by LANL domain experts. Each simulation is an instance of the initial first-day infection stage of an independent campaign. LANL issued the *APT Infection Discovery Challenge* to the community requesting novel methods for the detection of malicious domains and compromised hosts involved in these attacks [14]. Each of the simulated attacks belongs to one of four cases in increasing order of difficulty, described in Table I. Cases 1-3 include "hints" about the identity of one or multiple compromised hosts, while no hint is given in case 4. Answers (i.e., the malicious domains) in each attack are provided for validation.

B. Parameter selection

When selecting various parameters for our algorithms, we separate the 20 simulated attacks into two equal-size sets, and use one for training, and the other for testing. We try to include attacks from each case in both training and testing sets, with the only exception of case 4, simulated only on one day. We deliberately add this most challenging attack (in which no hint is provided) to the testing set. We use the training set for selecting parameters needed for different components of the algorithm. We show that parameters chosen according to the training set perform well on the testing set.

Thresholds for dynamic histograms. As described in Section IV-C the dynamic histogram method can be configured with two parameters: bin width (W), and the threshold (J_T)

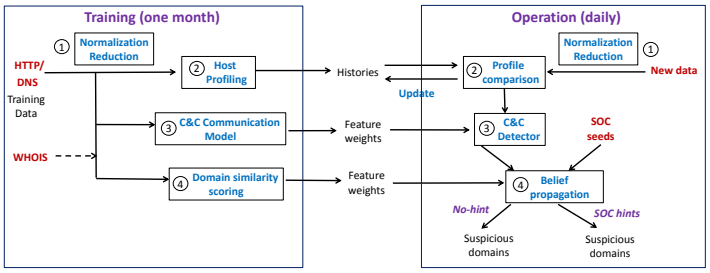

Fig. 5: Overview of training and operation stages in our system for detecting enterprise infection. Training stage is on the left and operation on the right. Input data is shown in red, processing steps in blue and various outputs in black.

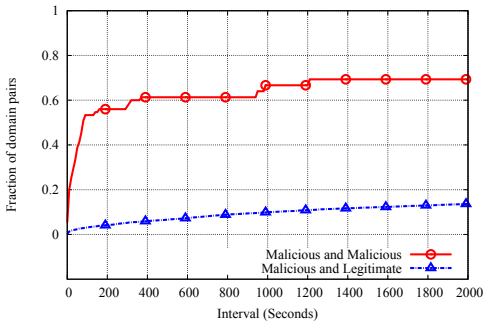

Fig. 6: CDFs of time interval between connection to two malicious domains and a malicious and legitimate domain by same host.

denoting the maximum Jeffrey distance between the two histograms. A connection with histogram at distance less than J_T from the periodic histogram is considered automated. Intuitively, the larger W and J_T, the more resilience the method provides against randomization and outliers, but more legitimate connections are labeled automated. We choose W at 10 seconds and J_T at 0.06 in order to capture all malicious pairs in the training set, while labeling fewest legitimate connections automated. Detailed results are included in the full version [29].

Timing and IP Features. We measure the relevance of the timing and IP similarity features among malicious domains. For compromised hosts in the training set, we extract the timestamp of their first connection to every rare domain visited. We plot in Figure 6 CDFs of the distributions of the time difference between visits to malicious domains and a legitimate and malicious domain by the same host. The graph confirms that connection intervals between two malicious domains are much shorter than between a malicious and a legitimate domain. For example, 56% of visits to two malicious domains happen at intervals smaller than 160 seconds, while only 3.8% of malicious-to-legitimate connection intervals are below this threshold (similar results are observed on testing dataset).

Next we measure similarity in IP space for malicious and legitimate domains in the training set. We found that 7 malicious domain pairs are in the same /24 subnet, while 18 share a /16 subnet. We observed few cases of legitimate domains residing in the same subnet with malicious ones. With the exception of 3/7, when more than 2000 pairs of malicious and legitimate domains share the same /24 or /16 subnet (due to a single malicious domain belonging to a popular service), the rest of days we observe 20 pairs in the same /24 subnet and 155 pairs in the same /16 subnet.

Domain scoring. Since domain names in the LANL dataset are anonymized and the data contains only DNS requests, we have access to a smaller number of features than in the *AC* web proxy dataset. We thus apply simple heuristics for domain scoring. We label an automated domain as C&C if it is contacted by at least 2 hosts at similar time periods (within 10 seconds). For computing domain similarity, we employ a simple additive function of three features: domain connectivity, timing correlation with a known malicious domain (value 1 if the domain is contacted close in time to a malicious domain and 0 otherwise), proximity in the IP space with malicious domains (value 2 if same /24 subnet with a malicious domain, 1 if same /16 subnet with a malicious domain and 0 otherwise).

C. Results

The summary of our results on the four cases of the LANL challenge are given in Table II. We use standard metrics from machine learning literature: *precision* is the fraction of true positives among all detected domains, *false positive rate* (FPR) is the fraction of false positives among benign domains; and *false negative rate* (FNR) is the fraction of malicious domains labeled as legitimate by our detector. Overall, we achieve a precision of 98.33% (97.06% on the testing set), with an FPR of $3.72{\cdot}10^{-5}$% over all 2.7M domains ($5.76{\cdot}10^{-5}$% over 1.7M domains in the testing set) and an FNR of 6.35% (2.94% on the testing set).

Interestingly, the BP algorithm trained on case 3 delivered very good results on case 4, where we did not have an opportunity for training (case 4 was simulated only on one day). All the five domains identified by BP were confirmed malicious, and the algorithm did not have any false positives.

Case	Malicious domains	True Positives		False Positives		False Negatives	
		Train	Test	Train	Test	Train	Test
1	12	6	4	0	0	2	0
2	22	8	12	0	0	1	1
3	24	12	12	0	1	0	0
4	5	-	5	-	0	-	0
Total	63	26	33	0	1	3	1

TABLE II: Results on LANL challenge.

VI. Evaluation on Enterprise Data

We implemented a fully operational system running in production starting from January 1 2014 to process the web proxies logs from the *AC* dataset. We use the data collected in January for training various components of the system (e.g., the C&C detector, the domain scoring module, etc.) and profiling external destinations and user-agent strings used by enterprise hosts in HTTP communication. Starting from February 1 the system enters into the daily operation mode, in which it processes new web proxies logs, applies normalization and reduction techniques, compares the data with the profiles (which are also updated) and applies our detection techniques.

The algorithm is first run in the *SOC hints* mode, where we use malicious domains from the IOC list provided by SOC as seeds. Second, we run our C&C detector to identify suspicious domains with automated activities. Third, these domains are given as seed to belief propagation in the *no-hint* mode. The detection results are thoroughly validated through a combination of tools and manual analysis. The system is configurable with different parameters (e.g., scoring thresholds, number of iterations in belief propagation, etc.) according to the SOC's processing capacity. We present our validation methodology and the results in different modes of operation.

A. Validation methodology

The domains output by our detector in both modes were validated as follows. We first query VirusTotal and the IOC domain list to verify their status (three months after they were detected – to allow anti-virus and blacklists to catch up). If the domain is alerted upon by at least one scanner used by VirusTotal or it's an IOC we consider it *known malicious*. For other domains, we collect additional information and hand them to a security analyst for manual investigation.

Specifically, the analyst retrieves the associated URLs from the log data and crawls them to examine the responses. The URLs are also manually submitted to McAfee SiteAdvisor. Based on the URLs, the response to our crawler and the result from SiteAdvisor, the analyst classifies the remaining domains into four categories: *new malicious* (e.g., same URL patterns as known malicious domains, returning malicious content or flagged by SiteAdvisor), *suspicious* (not resolvable when crawled, parked or having some questionable activities), *legitimate* (no suspicious behavior or code observed) and *unknown* (504 HTTP response code, a sign of server error). Since we only have a few unknowns (6 in total), we remove them from the final results. When reporting our results we use the same precision and FPR metrics from Section V-C, and *new-discovery rate* (NDR) defined as the percentage of new malicious and suspicious domains detected by our approach (and not yet identified by VirusTotal and SOC). We present graphs in Figure 7 and statistics of our findings in Table III.

B. Results for the SOC hints mode

We first present results in Figure 7(a) for the belief propagation algorithm in *SOC hints* mode seeded with 28 IOC domains. The graph shows the total number of detected domains and their categories for different domain similarity score thresholds. We do not include the seed domains in the final results. When computing domain registration features, we can not parse WHOIS information for 27% of domains. For these domains, we set default values for the DomAge and DomValidity features at average values across all other domains.

Score	Total	Malicious		Susp.	Legit.	Prec.	NDR
		known	new				
SOC hints							
0.33	137	79	15	14	29	78.8%	21.1%
0.37	114	76	15	6	17	85.1%	18.4%
0.4	91	70	3	5	13	85.7%	8.8%
0.41	86	69	2	5	10	88.4%	8.1%
0.45	73	65	1	3	4	94.5%	5.5%
C&C communication							
0.4	114	74	5	18	17	85.1%	20.1%
0.42	74	51	4	9	10	86.4%	17.5%
0.44	57	41	3	6	7	87.7%	15.8%
0.45	46	33	3	4	6	86.9%	15.2%
0.46	36	27	2	3	4	88.8%	13.9%
0.48	19	15	1	2	1	94.7%	15.8%
No hints							
0.33	265	132	27	43	63	76.2%	26.4%
0.5	169	93	17	24	34	79.3%	24.2%
0.65	152	88	16	23	25	83.5%	25.6%
0.75	135	81	13	22	19	85.9%	25.9%
0.85	114	74	5	18	17	85.1%	20.2%

TABLE III: Statistics and metrics on detected domains.

When we vary the domain similarity score threshold between 0.33 and 0.45, we detect between 137 and 73 domains, with precision ranging from 78.8% to 94.6%. Among the 137 detected domains, 108 turn out to be malicious (either known or new) and suspicious, which is about four times larger than the malicious set of domains used for seeding. The FPR is low at $3.97 \cdot 10^{-4}\%$ over 7.3M domains.

Among the 108 malicious and suspicious domains, 79 are confirmed by SOC or VirusTotal, leaving 29 domains as our new findings. We inspect the new findings and identify an interesting group of domains generated through Domain Generation Algorithm (DGA). Hosts infected with DGA malware generate a large number of domains (using a predefined algorithm) to hide the actual rendezvous points with the C&C center, which is only a handful of the contacted domains. The attacker knows the DGA algorithm used by the bots, and so only registers domains to communicate with the bots at specific times.

This group consists of 10 domains under the top-level domain (TLD) `.info` and the name for each domain has 20 characters (e.g., `f0371288e0a20a541328.info`). Surprisingly, the registration dates for most of the domains are later than the detection time. This demonstrates that our techniques have an advantage against attackers by inferring the next rendezvous point and taking preventive measures early.

C. Results for C&C detector

To evaluate the C&C detector, we compute scores for all automated domains visited daily. We vary the domain score threshold for labeling automated connections from 0.4 to 0.48

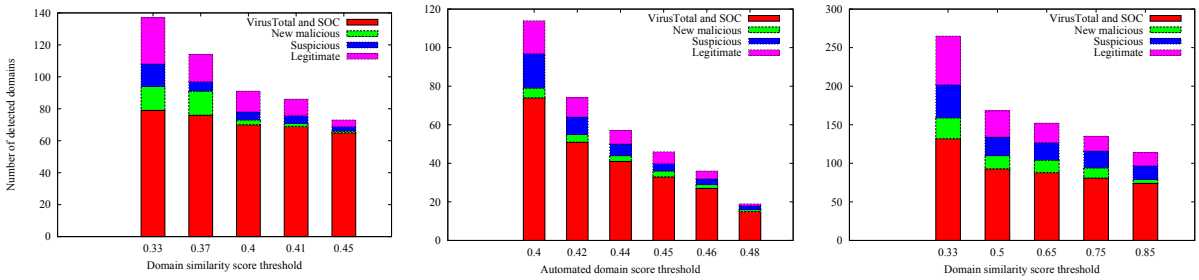

Fig. 7: Categories of detected domains. (a) SOC hints. (b) C&C communication. (c) No hints.

and present results for domains detected as C&C domains (with score above the threshold) in Figure 7(b). The results demonstrate that as we increase the threshold on automated domain scores from 0.4 to 0.48 the number of domains labeled as C&C drops from 114 to 19, while precision increases from 85.1% to 94.7%. Though FPR is higher for threshold 0.4 (at $2.33{\cdot}10^{-4}\%$), more malicious domains (including 23 new ones not known to VirusTotal or SOC) are detected.

D. Results for the no-hint case

We fix the automated domain score threshold at 0.4 to evaluate belief propagation in the *no-hint* mode. We vary the domain similarity score threshold from 0.33 to 0.85 and the result (Figure 7(c)) shows that the number of all detected domains varies from 265 to 114, with precision ranging from 76.2% to 85.1%. Altogether in the most challenging case (when no hint is available), we detect 202 malicious and suspicious domains in February, associated with 945 hosts. Though the majority of the detected domains are already alarmed by SOC and VirusTotal (132 for threshold 0.33), only 13 are reported in the IOC list and the remaining ones are unknown to the enterprise. More interestingly, we identified many new malicious and suspicious domains not known to the community (a total of 70 new domains for threshold 0.33 resulting in an NDR of 26.4%). This result suggests that our detector could complement existing security tools by discovering new suspicious activities. Its main advantage is that it has the ability to detect new campaigns without traces of known malicious behavior.

We thoroughly examined the domains labeled as new malicious and suspicious and found several prominent and interesting clusters. Among the new malicious domains, we found 5 domains hosting URLs with the same pattern `/logo.gif?` later confirmed by the SOC as related to Sality worm. We also found 15 domains with the same URL pattern reported by VirusTotal. Moreover, we identified a cluster of 10 DGA domains with none of them reported by VirusTotal and SOC, demonstrating our detector's ability in capturing new malicious campaigns. All the malicious domains are under the TLD `.info` and their names have 4 or 5 characters (e.g., `mgwg.info`). 9 out of the 10 domains hosts URLs with pattern `/tan2.html` and visiting them will be redirected to the remaining domain `1.tv990.info`.

We labeled legitimate 63 domains belonging to categories like Ad-network, Gaming, Toolbar and Torrent Tracker, resulting in a FPR of $8.63 \cdot 10^{-4}\%$ over 7.3M domains. They are captured by our detector because they exhibit suspicious features, like automated connections or are registered recently. Though they do not pose serious harm to the enterprise, some of them are policy violations (e.g., Gaming, Torrent Tracker). We did not discover any suspicious activities from examining log data, but we believe these domains still need to be vetted.

E. Comparison and performance

We compare the results of the two modes of operation. Only 21 domains are detected in both modes, which is a small percentage compared to 202 and 108 domains detected separately. When deployed by the enterprise, we suggest our detector configured to run in both modes, in order to have better coverage. As we have shown, starting from a seed of known malicious domains or hosts, the algorithm in *SOC hints* mode can identify suspicious domains with high accuracy. The C&C communication detector has the unique capability of identifying C&C domains used in new attack campaigns. To reduce the false positive rate in the *no-hint* mode, we recommend that the detected C&C domains are first vetted by the SOC and then belief propagation is seeded only with confirmed malicious C&C domains.

In terms of performance, our system proves scalable to the logs generated by a large enterprise (average 662GB data daily). The data is stored on a parallel Greenplum database with 90TB storage and is processed on a Cisco UCS C200 M2 server with 48GB of RAM. The normalization and profiling stages take around 2 hours every day (this includes the time to query the database, create normalized representations and write the normalized data to disk). Belief propagation is extremely fast (taking on average 5 seconds) since we build the bipartite graph incrementally and only add to the graph a small number of suspicious domains and hosts in each iteration.

Both variants of our detector include configurable options for various parameters (e.g., thresholds for domain scoring). These parameters can be chosen by the SOC according to the capacity of the team performing manual investigation, and various tradeoffs between accuracy and larger coverage as shown by our experimental evaluation.

VII. Related Work

Our work focuses on detecting early-stage infections within enterprise perimeters, including communications related to malware delivery and C&C. There has been a large body of work in this area, but to the best of our knowledge, we are the first to exploit the relationship between malicious domains associated with the same attack campaign, and to detect them

by a graph-theoretic framework based on belief propagation. We describe here related work in the literature.

Detection of C&C communication. Some of the previous work detecting C&C domains in botnets require malware samples as input to detect connections with similar patterns (e.g., BotFinder[38], Jackstraws[22]). Anomaly-based botnet detection systems (e.g., BotMiner[16], BotSniffer[18] and TAMD [42]) typically detect clusters of multiple synchronized hosts infected by the same malware. In contrast to these, our approach does not require malware samples and can detect campaigns with few hosts communicating to a C&C server.

DISCLOSURE [7] identifies C&C traffic using features extracted from NetFlow records but incorporates external intelligence sources to reduce false positives. Our C&C detector is different in that it leverages enterprise-specific features extracted from HTTP connections. From that perspective, ExecScent [28] is close to our work in detecting C&C communications in large enterprise network. However, ExecScent needs malware samples to extract templates representing malicious C&C connections. The templates are adapted to a specific enterprise considering the popularity of different features (URL patterns, user-agent strings, etc.). Our work complements ExecScent in detecting new unknown malware that can be provided as input to the template generation module.

Detection of malware delivery. Nazca [21] analyzes web requests from ISP networks to identify traffic related to malware delivery and unveils malicious distribution networks. CAMP [33] determines reputation of binary downloads in the browser and predicts malicious activities. BotHunter [17] identifies sequences of events during infection, as observed from a network perimeter.

Detection of malicious domains. Domains used in malicious activities are backed by highly resilient infrastructures to deal with takedowns or blacklisting, and hence exhibit unique characteristics distinct from benign sites. Another branch of work detects domains involved in malicious activities by patterns observed in DNS traffic (e.g., EXPOSURE [8], Notos [4], Kopis [5], and Antonakakis et al. [6]). Paxson et al. [31] detect malicious communication established through DNS tunnels. Carter et al. [9] use community detection for identifying highly localized malicious domains in the IP space.

Anomaly detection in enterprise network. Beehive [41] is an unsupervised system identifying general anomalies in an enterprise setting including policy violations and malware distribution. Our work is specifically targeting enterprise infections which pose high risk and potential financial loss.

Targeted attacks. The threats in cyberspace keep evolving and more sophisticated attacks recently emerged. Some targeted attacks (APT) are well-funded, carefully orchestrated and persist in the victim environments for years before detection.

Detecting targeted attacks in general is a very challenging task. These attacks are usually very stealthy and able to evade existing defenses [3]. However during the automated infection stage many campaigns (e.g., Shady RAT [20], Mirage [11], APT1 [26]) exhibit similar infection patterns. Recent studies have shown that even though in theory APTs could be arbitrarily sophisticated, in practice goal-oriented attackers use relatively low levels of sophistication [39], [27], [24]. We leverage some common patterns observed during the infection stage to build a detector tailored to an enterprise. Our detection result on the LANL's APT infection discovery challenge indicates that our techniques have potential in detecting infections originated from targeted attacks.

VIII. Limitations and Discussion

As reported by Mandiant, the infection patterns that we detect are quite prevalent in many APT attacks. Nevertheless, attackers could in principle use a number of techniques to evade our detectors, such as:

- Attackers may attempt to communicate through protocols other than HTTP(S) but most other communication is blocked at an enterprise border.

- Attackers could add more randomization to the timing of C&C communications evading our dynamic histogram detector. In that case, communication patterns will be less predictable to attackers orchestrating a campaign.

- Attackers could use standard user-agent strings but they need to determine a popular UA for an enterprise (since we measure UA popularity within the enterprise). Popular UAs limit functionality of malware that encodes host status, configuration and other information in the UAs.

- Attackers could register domains in advance before their use.

- Attakers could communicate directly with IP addresses at an increased risk of their infrastructure being discovered.

All these evasion methods come at an increased cost of operation for attacker, limiting malware functionality and increasing the risk of campaign discovery.

Our proposed approach is meant to complement existing tools rather than replace them. The results from §VI demonstrate that our belief propagation algorithm in both variants (*SOC hints* and *no-hint*) detects new suspicious activities overlooked by deployed defense mechanisms. These include both domains associated with existing malware campaigns (and identified by VirusTotal), but with new presence in the enterprise of our study, as well as entirely new malware campaigns (not yet detected by anti-virus technologies). Since our methods are focused on detecting the initial infection stages of a campaign it is difficult to determine how many of these suspicious activities are related to more advanced attacks, and how many are mainstream malware variants. We believe that monitoring activity to these suspicious domains over longer periods of time, as well as correlating with information from other data sources will answer this question, and we leave this as an interesting avenue for future work.

Acknowledgments

We are grateful to Kevin Bowers, Michael Fikes, Robert Griffin, Christopher Harrington, Engin Kirda, Silvio La Porta, Todd Leetham, James Lugabihl, Robin Norris, Martin Rosa, and Ronald L. Rivest for their many useful comments and suggestions on the system design and evaluation. We also thank the enterprise who permitted us access to the web proxies dataset and helped with investigation of suspicious activities. We are grateful to LANL for releasing the anonymous DNS

dataset and the C3E 2014 organizers for supporting the APT infection discovery challenge. Finally, we thank our shepherd Michel Cukier and anonymous reviewers for their feedback on drafts of this paper.

REFERENCES

[1] Hackers in China attacked The Times for last 4 months. http://www.nytimes.com/2013/01/31/technology/chinese-hackers-infiltrate-new-york-times-computers.html, 2013.

[2] Target's data breach: The commercialization of APT. http://www.securityweek.com/targets-data-breach-commercialization-apt, 2014.

[3] Verizon 2014 data breach investigations report. http://www.verizonenterprise.com/DBIR/2014/, 2014.

[4] M. Antonakakis, R. Perdisci, D. Dagon, W. Lee, and N. Feamster. Building a dynamic reputation system for DNS. In *Proc. 19th USENIX Security Symposium*, 2010.

[5] M. Antonakakis, R. Perdisci, W. Lee, N. Vasiloglou, II, and D. Dagon. Detecting malware domains at the upper DNS hierarchy. In *Proc. 20th USENIX Security Symposium*, 2011.

[6] M. Antonakakis, R. Perdisci, Y. Nadji, N. Vasiloglou, S. Abu-Nimeh, W. Lee, and D. Dagon. From throw-away traffic to bots: Detecting the rise of DGA-based malware. In *Proc. 21st USENIX Security Symposium*, 2012.

[7] L. Bilge, D. Balzarotti, W. Robertson, E. Kirda, and C. Kruegel. DISCLOSURE: Detecting botnet Command-and-Control servers through large-scale NetFlow analysis. In *Proc. 28th Annual Computer Security Applications Conference*, ACSAC, 2012.

[8] L. Bilge, E. Kirda, K. Christopher, and M. Balduzzi. EXPOSURE: Finding malicious domains using passive DNS analysis. In *Proc. 18th Symposium on Network and Distributed System Security*, NDSS, 2011.

[9] K. M. Carter, N. Idika, and W. W. Streilein. Probabilistic threat propagation for network security. *IEEE Transactions on Information Forensics and Security*, 9, 2014.

[10] Command Five Pty Ltd. Command and control in the fifth domain. http://www.commandfive.com/papers/C5_APT_C2InTheFifthDomain.pdf, 2012.

[11] Dell SecureWorks. The Mirage campaign. http://www.secureworks.com/cyber-threat-intelligence/threats/the-mirage-campaign/, 2012.

[12] Dell SecureWorks. Top banking botnets of 2013. http://www.secureworks.com/cyber-threat-intelligence/threats/top-banking-botnets-of-2013/, 2014.

[13] N. Falliere, L. O. Murchu, and E. Chien. W32.Stuxnet dossier. http://www.symantec.com/security_response/whitepapers.jsp, 2011.

[14] P. Ferrell. APT infection discovery using DNS data. C3E Challenge Problem. http://permalink.lanl.gov/object/tr?what=info:lanl-repo/lareport/LA-UR-13-23109, 2013.

[15] C. Grier, L. Ballard, J. Caballero, N. Chachra, C. J. Dietrich, K. Levchenko, P. Mavrommatis, D. McCoy, A. Nappa, A. Pitsillidis, N. Provos, M. Z. Rafique, M. A. Rajab, C. Rossow, K. Thomas, V. Paxson, S. Savage, and G. M. Voelke. Manufacturing compromise: The emergence of Exploit-as-a-Service. In *Proc. 19th ACM Conference on Computer and Communications Security*, CCS, 2012.

[16] G. Gu, R. Perdisci, J. Zhang, and W. Lee. BotMiner: Clustering analysis of network traffic for protocol and structure-independent botnet detection. In *Proc. 17th USENIX Security Symposium*, 2008.

[17] G. Gu, P. Porras, V. Yegneswaran, M. Fong, and W. Lee. BotHunter: Detecting malware infection through IDS-driven dialog correlation. In *Proc. 16th USENIX Security Symposium on USENIX Security Symposium*, SS'07, 2007.

[18] G. Gu, J. Zhang, and W. Lee. BotSniffer: Detecting botnet command and control channels in network traffic. In *Proc. 15th Network and Distributed System Security Symposium*, NDSS, 2008.

[19] S. Hao, N. Feamster, and R. Pandrangi. Monitoring the initial DNS behavior of malicious domains. In *Proc. ACM Internet Measuremnt Conference*, IMC '11, 2011.

[20] Hon Lau. The truth behind the Shady RAT. http://www.symantec.com/connect/blogs/truth-behind-shady-rat, 2011.

[21] L. Invernizzi, S. Miskovic, R. Torres, S. Saha, S.-J. Lee, C. Kruegel, and G. Vigna. Nazca: Detecting malware distribution in large-scale networks. In *Proc. ISOC Network and Distributed System Security Symposium (NDSS '14)*.

[22] G. Jacob, R. Hund, C. Kruegel, and T. Holz. Jackstraws: Picking command and control connections from bot traffic. In *Proc. 20th USENIX Security Symposium*, 2011.

[23] James Wyke. The ZeroAccess rootkit — Naked Security. http://nakedsecurity.sophos.com/zeroaccess/, 2012.

[24] S. Le Blond, A. Uritesc, C. Gilbert, Z. L. Chua, P. Saxena, and E. Kirda. A look at targeted attacks through the lense of an NGO. In *Proc. 23rd USENIX Security Symposium*, 2014.

[25] J. Ma, L. K. Saul, S. Savage, and G. M. Voelker. Beyond blacklists: Learning to detect malicious web sites from suspicious URLs. In *Proc. 15th ACM International Conference on Knowledge Discovery and Data Mining*, KDD, 2009.

[26] MANDIANT. APT1: Exposing one of China's cyber espionage units. Report available from www.mandiant.com, 2013.

[27] W. Marczak, J. Scott-Railton, M. Marquis-Boire, and V. Paxson. When governments hack opponents: A look at actors and technology. In *Proc. 23rd USENIX Security Symposium*, 2014.

[28] T. Nelms, R. Perdisci, and M. Ahamad. ExecScent: Mining for new C&C domains in live networks with adaptive control protocol templates. In *Proc. 22nd USENIX Security Symposium*, 2013.

[29] A. Oprea, Z. Li, T.-F. Yen, S. H. Chin, and S. Alrwais. Detection of early-stage enterprise infection by mining large-scale log data. http://arxiv.org/abs/1411.5005, 2014.

[30] Panda Security. Annual report PandaLabs - 2013 summary. http://press.pandasecurity.com/wp-content/uploads/2010/05/PandaLabs-Annual-Report_2013.pdf, 2014.

[31] V. Paxson, M. Christodorescu, M. Javed, J. Rao, R. Sailer, D. Schales, M. P. Stoecklin, K. Thomas, W. Venema, and N. Weaver. Practical comprehensive bounds on surreptitious communication over DNS. In *Proc. 22nd USENIX Security Symposium*, 2013.

[32] J. Pearl. Reverend Bayes on inference engines: A distributed hierarchical approach. In *Second National Conference on Artificial Intelligence*, 1982.

[33] M. A. Rajab, L. Ballard, N. Lutz, P. Mavrommatis, and N. Provos. CAMP: content-agnostic malware protection. In *Proc. ISOC Network and Distributed System Security Symposium (NDSS '13)*, 2013.

[34] U. Rivner. Anatomy of an attack. http://blogs.rsa.com/rivner/anatomy-of-an-attack, 2011.

[35] RSA. Stalking the kill chain. http://www.emc.com/collateral/hardware/solution-overview/h11154-stalking-the-kill-chain-so.pdf, 2012.

[36] Y. Rubner, C. Tomasi, and L. J. Guibas. The earth mover's distance as a metric for image retrieval. *International Journal of Computer Vision*, 40:99–121, 2000.

[37] G. Stringhini, C. Kruegel, and G. Vigna. Shady Paths: Leveraging surfing crowds to detect malicious web pages. In *Proc. 20th ACM Conference on Computer and Communications Security*, CCS, 2013.

[38] F. Tegeler, X. Fu, G. Vigna, and C. Kruegel. BotFinder: Finding bots in network traffic without deep packet inspection. In *Proc. 8th International Conference on Emerging Networking Experiments and Technologies*, CoNEXT '12, 2012.

[39] O. Thonnard, L. Bilge, G. OGorman, S. Kiernan, , and M. Lee. Industrial espionage and targeted attacks: Understanding the characteristics of an escalating threat. In *Proc. 15th International Symposium on Recent Advances in Intrusion Detection*, RAID, 2012.

[40] WebSense Security Lab. WebSense 2014 Threat Report. http://www.websense.com/assets/reports/report-2014-threat-report-en.pdf, 2014.

[41] T.-F. Yen, A. Oprea, K. Onarlioglu, T. Leetham, W. Robertson, A. Juels, and E. Kirda. Beehive: Large-scale log analysis for detecting suspicious activity in enterprise networks. In *Proc. 29th Annual Computer Security Applications Conference*, ACSAC '13, pages 199–208, New York, NY, USA, 2013. ACM.

[42] T.-F. Yen and M. K. Reiter. Traffic aggregation for malware detection. In *Proc. Intl. Conf. Detection of Intrusions and Malware, and Vulnerability Assessment*, DIMVA, 2008.