Operational security log analytics for enterprise breach detection

Zhou Li
RSA Laboratories
Bedford, MA, USA
Email: zhou.li@rsa.com

Alina Oprea
College of Computer and Information Science
Northeastern University
Boston, MA, USA
Email: a.oprea@neu.edu

***Abstract*—Enterprises today are facing an increasing number of criminal threats ranging from financially motivated and opportunistic malware to more advanced targeted attacks organized by nation-state actors. To protect against these threats, enterprises deploy a number of perimeter defenses, including traditional controls (anti-virus software, intrusion detection systems, firewalls) and more advanced techniques (web proxies or deep packet inspection products). Organizations collect and store the log data generated by these security controls, but this data is most of the time used for forensic investigation once an attack has been discovered by an external mechanism. In this paper, we describe a security log analytics framework for proactive breach detection, which we have tested on three applications. We summarize the algorithms and detection results from our previous work ([13, 20, 21]). Compared to other research in this area, our framework analyzes multiple sources of security logs, performs large-scale analysis, and is continuously refined from feedback given by security experts. Our techniques have been successfully used in operational setting in a large organization and are currently integrated in a real-time behavior analytics product.**

I. Introduction

The attack landscape faced by organizations today is continuously evolving. A wide spectrum of threats ranging from 'crimeware" (opportunistic and financially motivated malicious activities) to sophisticated cyber-espionage and nation-sponsored campaigns (e.g., Advanced Persistent Threats or APTs) threaten most organizations. Enterprises deploy a variety of security defenses (firewalls, intrusion-detection systems, anti-virus software, etc.) to protect their perimeter against breaches. Nevertheless, well-funded attackers develop custom tools which attempt to evade the security protections in place. This results in an astonishing number of successful breaches highly publicized by media (e.g., Target [6], Sony [16], Anthem [7]) with devastating financial consequences for their victims. Verizon DBIR [18] reported that among 10,000 surveyed organizations in 2015 2,000 experienced a serious breach, and the total estimated losses exceeded 400 million dollars.

The security defenses deployed by enterprises yield large volumes of security logs that contain interesting and useful information about activities in the network. Unfortunately, most of the time these logs are consulted by security analysts for forensic investigation, once an attack has been discovered by some other mechanism. Even though attack-related activities are recorded in the logs, APT campaigns often remain undetected for long periods of time. To remediate this situation, large organizations established Security Operations Centers (SOC) or Incident Response Teams (IRT). Human analysts in these teams are tasked to examine the variety of alerts issued by security products, actively look for new attacks and mitigate the identified threats. Experienced security experts are sometimes successful in manually unveiling advanced attacks through log data analysis, but this approach is obviously not scalable to the massive scale of data collected in large organizations.

Extracting intelligence from security logs in an automated manner seems a promising direction for preventing enterprise breaches, but nevertheless it is a daunting task and introduces unforeseen challenges. In addition to the large amount of data that needs to be processed and analyzed, logs come from multiple vendors, are sometimes inconsistent in their format, and there is usually a high semantic gap between the information recorded in the logs and what is required to detect a breach. Additionally, ground truth of real intrusions is very limited in an enterprise setting due to the large number of perimeter protections employed. It has been well recognized in the community that designing machine learning techniques with high accuracy in such settings with limited ground truth is extremely challenging [15].

In this paper, we introduce an *analytics framework* that automatically analyzes log data from various sources with the goal of *proactively protecting enterprises against breaches*. We first build a *normalization layer* that pre-processes the data, creates unique identifiers and synchronizes all timestamps to a common time zone. We design multiple *profilers* that learn the legitimate user and host behavior across several dimensions. At the same time, we build a *feature extraction layer* that generates a large number of features correlated with malicious behavior, based on extensive feedback from security experts. The framework supports multiple *statistical analysis modules* including supervised, unsupervised and semi-supervised learning that address various use cases. We discuss three applications we developed related to (1) identifying outlying hosts in the global population through clustering techniques; (2) predicting host infection based on a number of indicators through a regression model; and (3) detecting initial infection in a multi-stage campaign through

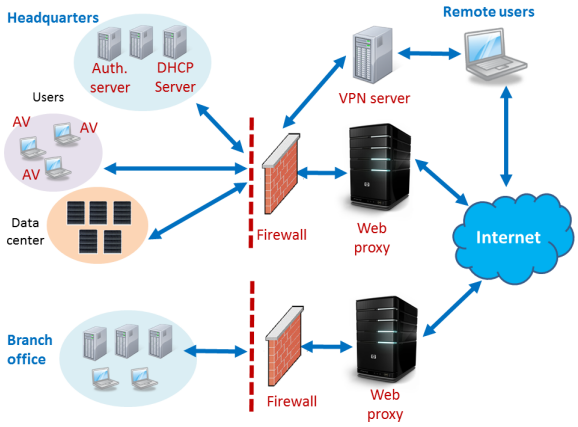

Figure 1. Typical configuration of enterprise networks.

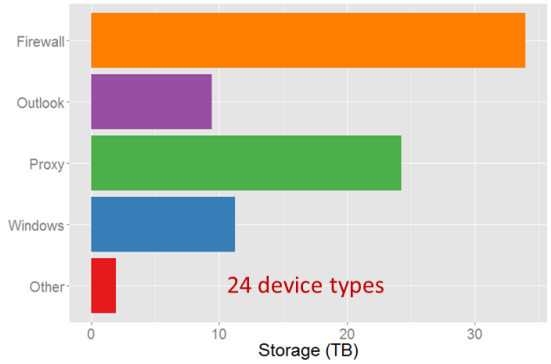

Figure 2. Volume of log data in a month.

applications of graph mining techniques.

Our techniques have been successfully deployed in production at EMC and identified hundreds of new attacks, overlooked by existing defenses. Based on feedback from security experts from EMC's Critical Incident Response Center (CIRC) related to our system design and investigation of produced incidents, our framework has been continuously refined to account for new attack patterns and new sources of data. We describe here the design of the analytics framework, and summarize the algorithms and results from our previous work [13, 20, 21].

II. Background and Challenges

Enterprise networks. Figure 1 shows a common architecture of enterprise networks deploying network-level defenses (firewalls, web proxies, VPNs, etc.), as well as host-based technologies (anti-virus software). Enterprises collect a wide range of logs generated by these various devices, including *web proxies* that log every outbound HTTP and HTTPs connection, *DHCP servers* that log dynamic IP address assignments, *VPN servers* that log remote connections to the enterprise network, *Windows domain controllers* that log authentication attempts within the corporate domain, and *antivirus software* that logs the results of malware scans on end hosts. These logs are typically stored in a commercial security information and event management (SIEM) system.

Challenges. Building security analytics algorithms for proactive breach detection using enterprise security logs is challenging. Most previous works that have successfully applied machine learning in security settings performed experiments in controlled environments, in which finer-grained data collection (system calls or process execution) is enabled and detailed ground truth information on malicious files or infected machines is available. In contrast, security products in enterprises record events with less detail and various formats, are configured by multiple vendors and are sometimes inconsistent in the information they record. We elaborate on the specific challenges encountered in our environment below:

Massive data. The security logs generated in large organizations are massive in scale. As an example, the enterprise of our study has 120K machines widely distributed geographically and generates on average 2TB of security logs daily from 24 different types of security devices. The web proxy logs include on average 600K distinct external destinations (domain names) and 10 million distinct URLs daily, reaching 24TB in a month. Figure 2 shows the volume of log data for the most prominent device types. Timely detection of critical threats in our framework requires efficient data-reduction algorithms and strategies to focus on security-relevant information in the logs.

Inconsistencies and semantic gap. The logs yielded by different security products are usually recorded with vendor-specific formats, are sometimes inconsistent and experience high semantic gaps between the information recorded in the logs and what is needed for attack detection. As an example, most web proxies record *source IP addresses* of endpoint machines, but most organizations assign the IP address space dynamically through DHCP. In these cases, DHCP logs could be used to obtain the *host name* in a proxy log, but correlating different log types in a distributed organization is not immediate as collecting devices use different configurations and time zones.

Limited ground-truth information. In contrast to experimental networks in which malware samples can be run for data collection, in enterprise settings there is limited information on malicious activities available for analysis. Perimeter-deployed defenses block a variety of known threats (such as variants of well-known crimeware attacks). As a consequence, we need to work mostly with unlabeled data, which is the most challenging scenario for designing highly accurate machine learning algorithms. Moreover, manual investigation needs to be performed for validating new

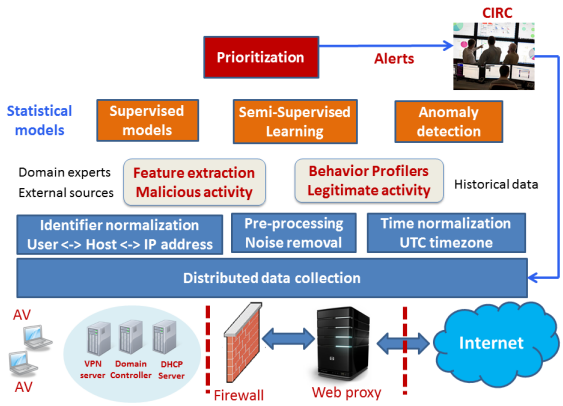

Figure 3. Analytics framework.

findings, and this imposes strict requirements on reducing false positive rates.

Insights. Given all the above challenges and the difficulty of making machine learning successful in operation settings [15], how can we overcome these obstacles and proactively detect enterprise breaches with high accuracy and limited false positives? Here are our main insights:

- To address the massive data challenges, we *profile different aspects of user behavior over time* and obtain a baseline of their legitimate activity. For example, we profile the legitimate network traffic to determine common, popular destinations visited by users. Then we focus on *unpopular destinations* (visited by a small number of users) and *new destinations* (not visited before by any user), since they are more suspicious.

- To solve the semantic gap issue, we normalize all timestamps to a common time zone and create unique identifiers for hosts and users. This normalization enables the *accurate correlation of multiple sources of data*.

- To overcome the machine-learning challenges, we use the *domain knowledge of security experts* in the incident response team at EMC (referred to as CIRC) for extracting features highly correlated with malicious activities. We use a number of generic features indicative of malicious activities, but we are in the unique position to derive features *specific to an enterprise*, where most users have common software installations. For instance, most user-agent (UA) strings in an enterprise have a large user base, and therefore unpopular UAs are suspicious in this setting.

- To compensate for limited access to labeled data, we use anomaly-detection techniques such as clustering and outlier detection, as well as semi-supervised learning methods such as graph inference and community detection. We present applications of these techniques in the next section.

III. System Design

We developed a multi-layer analytics framework that processes multiple sources of data and supports a variety of machine-learning algorithms for proactive breach detection. An overview of our architecture is given in Figure 3. We would like to highlight that our system was deployed in production at EMC, a largely distributed organization with 60,000 employees and identified many threats overlooked by existing defenses. For a period of more than a year, we generated prioritized alerts for the EMC CIRC and got feedback from security analysts that helped us refine our system. Additionally, some components of this framework are currently being integrated in behavioral analytics modules of RSA's products. We give first an overview of our framework and then elaborate different components below.

Overview. As illustrated in Figure 3, our framework analyzes the log data produced from different products (e.g., web proxies, DHCP, VPN, domain controllers) stored in a centralized SIEM repository managed by enterprise IT. Our first task is to perform data pre-processing and normalization, so that all hosts have unique identifiers and all timestamps are under the same time zone. We then build a suite of behavior profiling modules for obtaining a baseline of legitimate behaviors across multiple dimensions. At the same time, we build a feature extraction layer in which we use internal data sources as well as external ones (e.g., domain WHOIS) to extract features highly correlated with malicious activity. Feedback from security experts in CIRC has been extensively used for the feature extraction layer. We implement different statistical analysis modules, based on supervised, semi-supervised and unsupervised learning to support different applications. Each module generates a list of prioritized alerts, enriched with contextual information, that are investigated by human analysts.

Data pre-processing. In a globally distributed enterprise, log-collection devices record timestamps differently according to their configuration. They could use local time zones, UTC, or some other representations. As a result, the associated logs could bear inconsistent timestamps and have to be normalized before being used in analytics applications. We found most collection devices not reporting their timezones, therefore we normalize all timestamps into UTC through the following approach. We inspect the timestamps of the logs tagged by the local device and the timestamps of same logs tagged by the central SIEM (which is configured in UTC timezone). Then, we compute the difference between these two set of timestamps and use the value representing the majority of logs as the timezone difference to UTC. This process is repeated regularly to account for daylight savings and other geographical factors. Finally, all logs are corrected using the inferred timezone difference. More details of this approach can be found in [21].

As described in Section II, source IP addresses commonly

logged by network appliances have to be converted to host names. We tackle this problem by leveraging complimentary data sources. Particularly, we first analyze the logs from DHCP and VPN servers to create IP-to-host bindings for hosts with dynamic IP addresses. Next, we create bindings for hosts using static IP addresses. We retrieve all source IPs seen in enterprise (denoted by A) and the dynamic IPs recorded in DHCP and VPN logs (denoted by D). The static IP addresses can be derived by computing the set difference $S = A - D$ and we perform reverse DNS lookup using S to create the bindings for static hosts. The host name associated with each log entry can be identified by leveraging these bindings (details also elaborated in [21]).

Legitimate activity profiling. We implement a number of behavioral profiles that capture different aspects of legitimate activities on the network. The profiles are built from historical data (the organization keeps three months of historical data at any time), and then continuously updated to account for concept drift. Some examples of profiles we implemented are: the external destinations (i.e., domain names) visited by users, the user-agent strings employed by various enterprise hosts, applications installed on end hosts, users' typical working hours, users' login patterns to internal machines and servers and users' VPN activities. We design compact data structures for these profiles (e.g., aggregating domains and timestamps per host) to allow loading them into memory for fast processing.

Analytics applications. We build multiple analytics applications for proactive breach detection. We provide more details on three of them that leverage different machine-learning algorithms. First, we designed `Beehive`, a system that detects outlying hosts with widely distinguishing behavior than the majority of host population. This application is completely unsupervised, and we use *Principal Component Analysis* (PCA) for dimensionality reduction and a clustering algorithm applied to network-traffic features, to successfully detect a number of crimeware activities and policy violations (see Section IV-A). Second, we build a predictive model that leverages a variety of features from network traffic, user demographic information and VPN activity, to predict when a host machine becomes infected by malware. This system uses labeled data from anti-virus systems deployed on enterprise machines, and is based on a supervised logistic regression model (see Section IV-B). Lastly, we address the problem of detecting communication with malicious destinations during the malware delivery or command-and-control (C&C) stages in a multi-stage attack. We use techniques from graph theory based on belief propagation, in combination with regression models, to detect small communities of infected hosts and malicious destinations. Our techniques successfully addressed the LANL *APT infection discovery using DNS data* challenge [3] and identified hundreds of unknown enterprise infections (see Section IV-C). In current work, we are developing other algorithms for detecting different stages of an attack lifecycle, e.g., C&C communication, lateral movement and data exfiltration, and plan to get a comprehensive view on an entire attack campaign.

Enrichment and reporting. In all our applications, we prioritize the detected incidents and enrich the alerts with contextual information when presenting them to CIRC analysts. Based on experts' feedback, we add information from external data sources (e.g., WHOIS, ASN), other internal data sources (e.g., organization chart or endpoint reports), and AV/Blacklists labels (e.g., trojan, adware). This information is helpful to analysts to perform quick incident triage as the number of alerts greatly exceeds the capabilities of human analysts. In addition, our system takes the results of investigation from analysts into account to tune the system parameters, include new features and experiment with new learning methods.

IV. Analytics applications

In this section, we summarize the algorithms and results from three analytics applications for identifying hosts exhibiting outlying behaviors, predicting malware encounters on hosts and detecting malicious external communications during the delivery and C&C communication phases in a multi-stage attack.

A. Outlying host detection

For generating reports of anomalous host behavior relative to the entire host population, we design a system called `Beehive` [21], based on unsupervised learning methods. `Beehive` is the first system we are aware of that extracts meaningful intelligence from different sources of security log data in an enterprise setting and identifies unknown malware infection and policy violations. Our main insight is that hosts in an enterprise network are constrained by company policies and employee job functions, and exhibit more homogeneity than those on the open Internet. `Beehive` focuses on monitoring the behavior of dedicated hosts, which are machines utilized by a single user (and does not consider machines or servers that are shared among users – since those tend to experience different behavior).

For identifying hosts with anomalous activity, we extracted 15 features from four categories from web proxy logs as follows:

1) *Destination-based features.* We are interested in identifying hosts that communicate with *new external destinations* that have not been previously contacted by other hosts in the enterprise. Such obscure destinations have higher chance of being controlled by attackers (e.g., participate in malware delivery infrastructures or act as C&C centers) than the popular, common destinations visited on a regular basis. To identify the new destinations, we build a history (or profile) of external destinations contacted by internal hosts

over time. We had to employ techniques such as custom whitelisting and domain folding to keep the history size manageable. We are also interested in unpopular external destinations that are *raw IP addresses*. These might indicate suspicious activity, as legitimate services can usually be reached by their domain names. We thus include features related to counts on number of new destinations contacted (with and without the web referer field), and number and fraction of *unpopular* raw IP addresses visited by hosts daily.

2) *Host-based features.* User-agent (UA) strings in HTTP traffic are usually customized by applications making a web request to report the application configuration. In an enterprise with most machines having common software base, we expect that most UAs are employed by a large number of machines. We maintain a profile of UAs observed in network traffic and the hosts using them over time, and are interested in *new* user-agent strings relative to the global history. We accommodate UA modifications resulting from software updates by using the edit distance to account for similar UAs. We count the number of new UAs (according to certain threshold in the edit distance) for each host.
3) *Policy-based features.* Strict network policies are usually enforced in enterprise networks by web proxies that monitor and block suspicious connections at the enterprise border. Sites with low reputation or those that are not business-related are either blocked or, in some cases, require explicit consent by employees to visit them. Hosts visiting such sites on a regular basis are suspicious, and therefore we count the number of blocked and consented domains and connections per host.
4) *Traffic-based features.* Spikes in traffic volume or number of contacted domains in a small time window could indicate automated services (streaming, chatting) or malware (e.g., bot client searching for valid C&C centers or scanning the network). To account for these, we build profiles of legitimate traffic volumes and include the number of connections (or domain) spikes and bursts (prolonged intervals of intense activities) as features in our system.

We represent each internal host with a 15-dimensional feature vector collected on a daily basis. Since some of the features exhibit correlations, we first apply PCA for dimensionality reduction. Subsequently, we use a variant of k-means clustering that does not require the number of clusters to be specified in advance. We ran `Beehive` daily for a two-week period in 2013 on the dedicated hosts (27-35K were active on weekdays and 9-10K on weekends). We found the vast majority of hosts fall into one large cluster and the remaining clusters usually include hosts with abnormal behaviors (i.e., outlying hosts). `Beehive` generated 784 incidents in two weeks and, in absence of ground truth, we performed manual investigation in collaboration with security experts in CIRC. Among the detected incidents, we classified 25.25% as malware-related, 39.41% as policy violations and 35.33% as related to unrecognized software or services. In addition, only 1.02% of incidents were detected by other security tools. The results suggest that our approach could complement existing defenses in enterprise settings.

B. Predicting malware encounters

While all enterprise hosts are exposed to malware threats, the risk of encountering malware are varying widely across hosts. By leveraging security logs generated by anti-virus software deployed on end hosts as ground truth, we performed an extensive study to measure the impact of different factors to the risk of host infection [20]. We considered factors from three categories (user demographics, VPN activity and web usage) and present a number of interesting findings. For instance, we discovered that malware encounters happen more frequently outside the enterprise perimeter, malware encounters are highly correlated with geographical region of the user, and the most prominent infection vector in that enterprise is the USB drive.

These findings inspired us to build a predictive model to infer the risk of an employee's machine encountering malware proactively based on these features. The results from the model provide valuable insights to CIRC analysts and can be used to prioritize investigation on machines more prone to infection and remediate malware infections in early stages. Towards this end, we extracted three categories of features from organization chart, VPN logs and web proxy logs:

1) *Demographic features.* We take into account user's gender (inferred from employee's first name), country of employee's office, level in management hierarchy, and technical level of employee's job type (e.g., engineer, architect or manager). We found that all these features have significant correlation with malware encounters, but country is the most predictive demographic feature.
2) *VPN activity features.* Since 77% of malware encounters happen outside the corporate network, where we do not have much visibility, we considered VPN usage as an approximate metric for activities off-network. In particular, we consider the number of VPN connections, total duration, sent and received bytes and number of distinct external IP addresses initiating VPN connections. All these features, except received bytes, exhibit high correlation with malware encounters.
3) *Web activity features.* Web-based malware is a prevalent threat, and therefore web activity features are of

interest in our model. Our intuition is that employees visiting high-risk sites more frequently or having higher Internet exposure are more likely to encounter malware. To capture this, we investigate features related to categories of web sites visited, aggregate volumes of web traffic, and connections to blocked or low-reputation sites. The web proxies deployed by the enterprise automatically tag visited web sites into different categories (e.g., business, entertainment, social networks, etc.). We found several of these (file transfer, social networks, chat) having high correlation with malware encounters.

As the majority of features are categorical, we employed a two-stage feature selection process. First, we use a standard binary representation of categorical features and build a logistic regression model separately for each category with the goal of finding the binary features correlated with malware encounters. Second, we combine the statistically significant features selected in the first stage to build the final model. Our final model uses a total of 20 features from all categories.

We collected four months of data in 2013 and used the first two months for training the risk model and the remaining for testing. We ranked the machines in testing set by the predicted risk score and evaluated against the ground truth. It turns out that the machines ranked highly by the predictive model have a much higher rate of encountering malware: 51% of the top 1000 machines in the testing set encountered malware, which is three times higher than the overall malware encounter rate (15.31%). This demonstrates that our model could prioritize effectively the set of hosts at high risk of infection, and can be used by the enterprise to take proactive remediation measures.

C. Detecting malicious communication

Modern advanced malware and targeted attacks span multiple stages in the attack lifecycle [9]. APT campaigns tend to advance very slowly to evade detection and are able to persist in victim environments for years without detection. During the *malware delivery* stage, a victim machine gets malicious payload through a number of infection vectors, such as social engineering, USB drives or web-based attacks. Subsequently, backdoors are typically installed on the victim machine to initiate regular communication with a command-and-control (C&C) center and allow attackers remote access to the enterprise network (as inbound connections are blocked by enterprise policies). Once attackers are inside the victim network, they start to propagate to other machines, elevate their privileges and reach the target of interest to perform data exfiltration or reach their objectives.

We have developed semi-supervised learning methods to detect unusual communication patterns during the automated early stages of these campaigns [13]. Our main insight is that typical infection patterns are quite similar across many attack vectors and can be distinguished from legitimate connections. For instance, attackers might use different domains or IP addresses for communicating with victim machines, but they typically exhibit similarities in the IP address space, hosts that contact them, web connections and certain timing patterns. These domains are most of the time uncommon destinations, yielding low volumes of traffic from enterprise hosts.

We aim to detect malicious external communications by applying a graph inference technique called *belief propagation*, a message-passing algorithm designed to infer the label of a node (in our case, probability of the node being malicious), based on prior knowledge of the node state and information of its neighbors. We construct a bipartite communication graph that models the relationships between internal hosts and the external destinations they visit. Starting from a set of *seed domains* (known malicious domains), we add to the graph all hosts visiting them, and then compute a risk score for each unpopular domain visited by those hosts. To keep the size of our graphs small, we only add to the graph domains with score above a certain threshold (those are at higher risk). We iterate this process until the algorithm converges or a maximum number of iterations is reached. For computing each domain's risk score, we use a regression model trained on data labeled with the VirusTotal cloud anti-virus engine. The model is built with features related to the domain's connectivity, web connection information, registration data and timing patterns.

To seed the belief propagation algorithm we use either known malicious domains or known infected hosts. We also built a detector for C&C communication in [12, 13] that is based on another regression model (trained on a month of labeled data using VirusTotal), and can be used to initialize belief propagation. Alternatively, we used known malicious domains from different commercial blacklists, or known infected hosts already investigated by the CIRC to initialize the algorithm. The features leveraged by our two regression models are:

1) *Domain connectivity.* This measures the number of hosts contacting the domain. Intuitively, unpopular domains are more likely to be involved in malicious activities.
2) *Web connection features.* Connections without web referer might denote automated communication (as expected in the C&C stage), thus we consider the fraction of hosts contacting a domain without the web referer field. We also look for hosts using unpopular user-agent (UA) strings (those observed in a small number of hosts). To determine the UA popularity, we leverage the UA profiler we built.
3) *Registration data features.* Sites under attacker's control tend to use more recently registered domains than legitimate ones, and these domains have shorter

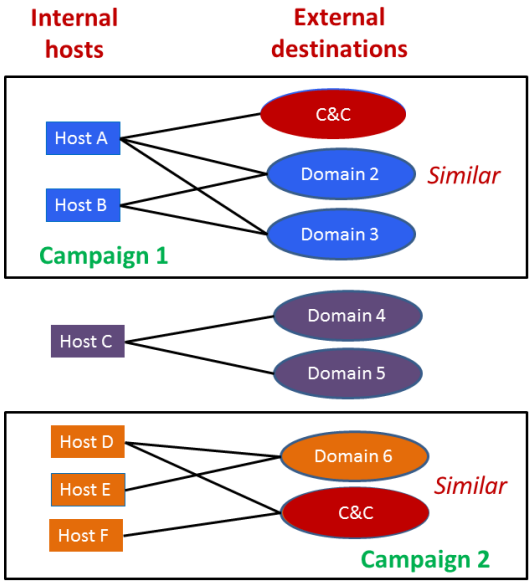

Figure 4. An example of two communities of domains related to two campaigns in the bipartite communication graph.

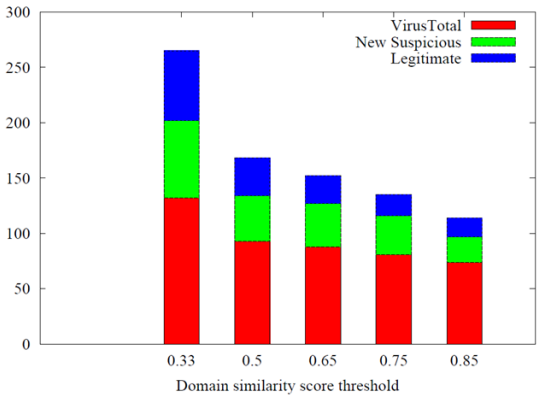

Figure 5. Detection results of belief propagation on one month of data.

validity period. We extract two features from public WHOIS data: domain age (days since registration) and domain validity (days until registration expires).

Our results on two large-scale datasets demonstrate that belief propagation is capable of revealing the full span of a malicious campaign. We participated in the LANL *APT infection discovery discovery challenge* [3] and successfully identified the simulated attacks in the 1.15TB anonymized DNS log dataset released by LANL (with accuracy of 98.33% and false positive rate of $3.72 \cdot 10^{-5}\%$). We also scaled our method to process 38TB of enterprise proxy logs and discovered in one month 265 malicious connections involving 945 hosts. For validation, we used VirusTotal scanning and manual investigation by security experts, and show our results in Figure 5. We confirmed 132 domains (49.81%) malicious with VirusTotal (denoted *known malicious*), but these were new findings for the enterprise. Interestingly, 70 domains (26.4%) were confirmed through manual investigation (denoted *new malicious*). These are entirely new malicious domains not known to the community. In a month-long timeframe, we flagged 63 domains as false positives (bringing our false positive rate to $3.41 \cdot 10^{-4}\%\%$). This results in 2 false positives on average per day, which is very manageable for the CIRC. Nevertheless, our system can be configured with different risk score thresholds to tune the number of alerts.

V. Related work

A number of research groups also looked into web proxy log analysis in enterprise networks and developed applications aiming to uncover malicious activities. Manadhata et al. [8] applied belief propagation on the host-domain bipartite graph built using enterprise log data to detect malicious domains. Similarly, Carter et al. [2] modeled the domain-to-IP relations in a bipartite graph and applied a method called Probabilistic Threat Propagation to find malicious domains that share the same IP infrastructure. We design a variant of belief propagation that builds the communication graph incrementally starting from known malicious domains or known infected hosts, and can scale to the volume of logs collected by a large enterprise.

Malware downloads can be detected as well from analyzing network traffic, as suggested by previous research. Invernizzi et al. [5] developed a system called Nazca to identify malware distribution network from HTTP logs collected by Internet Service Providers (ISPs). Vadrevu et al. [17] developed AMICO which reconstructs the relations between downloaders and files from the academic network traffic and applied a provenance classifier to detect malware downloads. Nelms et al. [10] built WebWitness, a system inspecting the download path recovered from the academic network traffic and identifies the paths leading to drive-by-download attacks. Following this work, the authors developed a detection system targeting social-engineering download attacks and measured the characteristics of such attacks [11]. Since malware authors continuously produce new malware variants, the rules learnt from training samples could be easily evaded. A recent work by Bartos et al. [1] showed that network logs could be converted into representations robust against common changes of malware behaviors.

The closest to our framework is a system developed at IBM Research processing different enterprise data sources, such as logs from web proxies, DNS servers, firewall and VPN servers. Multiple applications have been developed on top of the framework. For instance, Kaleido [19] attributes network traffic to users and MUSE [4] scores the security risk of entities in enterprise network. They also proposed an analysis engine named FCCE to correlate data with

low latency [14]. The focus of their work was mainly to reduce latency for forensic analysis, while our work aims to automatically detect unknown malicious activities with high accuracy and low false positive rates. Moreover, we worked closely with domain experts (from EMC CIRC) for refining our methods and the reports provided by our framework have been extensively used in operational settings. Some components are currently being integrated into RSA's products.

VI. Conclusions

We present a security log analytics framework to proactively detect enterprise breaches and generate prioritized alerts for investigation by human analysts. Our framework is able to process massive and heterogeneous datasets from different collection devices efficiently and pinpoint various malicious activities with high accuracy. We summarized the algorithms and results from our previous work for detecting outlying hosts, predicting malware encounters and identifying malicious external communication. Our techniques have been successfully deployed in production at EMC for more than a year and are currently integrated into RSA's products.

Yet, there are many remaining challenges in designing effective security analytics solutions. Detecting stealthy, persistent attacks is very difficult given the low volume of traffic and limited availability of ground truth. Currently, it is impossible to compare algorithms developed by different groups, and understand their tradeoffs and limitations. We believe that creating standardized datasets in the security community (similarly to the machine learning community) would be beneficial and contribute to more advances in research in this area. Finally, building machine learning models resilient against advanced attackers is a perpetual challenge to our community.

VII. Acknowledgements

We would like to thank Ting-Fang Yen and Kaan Onarlioglu for working on the analytics framework in its early days, all members of RSA Laboratories, RSA Office of the CTO and Prof. Ronald L. Rivest for providing valuable feedback and suggestions over many iterations, and the EMC CIRC (Robin Noris, Todd Leetham and Christopher Harrington) for the joint collaboration without which this work would not have been possible.

References

[1] K. Bartos, M. Sofka, and V. Franc. Optimized invariant representation of network traffic for detecting unseen malware variants. In *25th USENIX Security Symposium (USENIX Security 16)*, pages 807–822, Austin, TX, Aug. 2016. USENIX Association.

[2] K. M. Carter, N. Idika, and W. W. Streilein. Probabilistic threat propagation for network security. *IEEE Transactions on Information Forensics and Security*, 9(9):1394–1405, Sept 2014.

[3] 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.

[4] X. Hu, T. Wang, M. Stoecklin, D. Schales, J. Jang, and R. Sailer. MUSE: asset risk scoring in enterprise network with mutually reinforced reputation propagation. *EURASIP Journal on Information Security*, 2014(1):17, 2014.

[5] L. Invernizzi, S. Miskovic, R. Torres, C. Kruegel, S. Saha, G. Vigna, S. Lee, and M. Mellia. Nazca: Detecting malware distribution in large-scale networks. In *21st Annual Network and Distributed System Security Symposium, NDSS 2014, San Diego, California, USA, February 23-26, 2014*, 2014.

[6] B. Krebs. The Target Breach, By the Numbers. http://krebsonsecurity.com/2014/05/the-target-breach-by-the-numbers, 2014.

[7] B. Krebs. Data Breach at Health Insurer Anthem Could Impact Millions. http://krebsonsecurity.com/2015/02/data-breach-at-health-insurer-anthem-could-impact-millions/, 2015.

[8] P. Manadhata, S. Yadav, P. Rao, and W. Horne. Detecting malicious domains via graph inference. In *Proceedings of the 2014 Workshop on Artificial Intelligent and Security Workshop*, AISec '14, pages 59–60, New York, NY, USA, 2014. ACM.

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

[10] T. Nelms, R. Perdisci, M. Antonakakis, and M. Ahamad. Webwitness: Investigating, categorizing, and mitigating malware download paths. In *24th USENIX Security Symposium (USENIX Security 15)*, pages 1025–1040, Washington, D.C., Aug. 2015. USENIX Association.

[11] T. Nelms, R. Perdisci, M. Antonakakis, and M. Ahamad. Towards measuring and mitigating social engineering software download attacks. In *25th USENIX Security Symposium (USENIX Security 16)*, pages 773–789, Austin, TX, Aug. 2016. USENIX Association.

[12] A. Oprea. Connecting the dots in a malicious campaign with graph analysis. Blog available at https://blogs.rsa.com/connecting-dots-malicious-campaign-graph-analysis/, 2015.

[13] A. Oprea, Z. Li, T. Yen, S. H. Chin, and S. A. Alrwais. Detection of early-stage enterprise infection by mining large-scale log data. In *45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, DSN 2015, Rio de Janeiro, Brazil, June 22-25, 2015*, pages 45–56, 2015.

[14] D. L. Schales, X. Hu, J. Jang, R. Sailer, M. P. Stoecklin, and T. Wang. FCCE: highly scalable distributed feature collection and correlation engine for low latency big data analytics. In *Proceedings of the 31st IEEE International Conference on Data Engineering*, ICDE '15, pages 1316–1327, Washington, DC, USA, 2015. IEEE Computer Society.

[15] R. Sommer and V. Paxson. Outside the closed world: On using machine learning for network intrusion detection. In *Proc. IEEE Symposium on Security and Privacy*, pages 305–316. IEEE, 2010.

[16] J. Steinberg. Massive Security Breach At Sony – Here's What You Need To Know. http://www.forbes.com/sites/josephsteinberg/2014/12/11/massive-security-breach-at-sony-heres-what-you-need-to-know, 2014.

[17] P. Vadrevu, B. Rahbarinia, R. Perdisci, K. Li, and M. Antonakakis. Measuring and detecting malware downloads in live network traffic. In *Computer Security - ESORICS 2013 - 18th European Symposium on Research in Computer Security, Egham, UK, September 9-13, 2013. Proceedings*, pages 556–573, 2013.

[18] Verizon. 2015 data breach investigations report. http://www.verizonenterprise.com/DBIR/2015/, 2015.

[19] T. Wang, F. Wang, R. Sailer, and D. Schales. Kaleido: Network Traffic Attribution using Multifaceted Footprinting. In *Proceedings of the 2014 SIAM International Conference on Data Mining*, SDM '14, 2014.

[20] T. Yen, V. Heorhiadi, A. Oprea, M. K. Reiter, and A. Juels. An epidemiological study of malware encounters in a large enterprise. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, Scottsdale, AZ, USA, November 3-7, 2014*, pages 1117–1130, 2014.

[21] T. Yen, A. Oprea, K. Onarlioglu, T. Leetham, W. K. Robertson, A. Juels, and E. Kirda. Beehive: large-scale log analysis for detecting suspicious activity in enterprise networks. In *Annual Computer Security Applications Conference, ACSAC '13, New Orleans, LA, USA, December 9-13, 2013*, pages 199–208, 2013.