An Unsupervised Multi-Detector Approach for Identifying Malicious Lateral Movement

Atul Bohara*‡, Mohammad A. Noureddine*‡, Ahmed Fawaz†‡, William H. Sanders†‡
*Department of Computer Science, †Department of Electrical and Computer Engineering, and ‡Information Trust Institute
University of Illinois at Urbana-Champaign
E-mail: {abohara2, nouredd2, afawaz2, whs}@illinois.edu

***Abstract*—Lateral movement-based attacks are increasingly leading to compromises in large private and government networks, often resulting in information exfiltration or service disruption. Such attacks are often slow and stealthy and usually evade existing security products. To enable effective detection of such attacks, we present a new approach based on graph-based modeling of the security state of the target system and correlation of diverse indicators of anomalous host behavior. We believe that irrespective of the specific attack vectors used, attackers typically establish a command and control channel to operate, and move in the target system to escalate their privileges and reach sensitive areas. Accordingly, we identify important features of command and control and lateral movement activities and extract them from internal and external communication traffic. Driven by the analysis of the features, we propose the use of multiple anomaly detection techniques to identify compromised hosts. These methods include Principal Component Analysis, k-means clustering, and Median Absolute Deviation-based outlier detection. We evaluate the accuracy of identifying compromised hosts by using injected attack traffic in a real enterprise network dataset, for various attack communication models. Our results show that the proposed approach can detect infected hosts with high accuracy and a low false positive rate.**

***Index Terms*—advanced persistent threat; lateral movement; command and control; anomaly detection;**

I. Introduction

There has been a surge in targeted cyber attacks that use persistent and stealthy actions to compromise victim organizations, usually high-profile companies or government agencies. Such attacks are popularly known as *advanced persistent threats (APTs)*. Typically, the goal of APT actors is to steal proprietary intellectual property or disturb mission-critical services. Although such attacks infect fewer entities than mass malware (e.g., botnets, worms) does, the estimated cost of data loss due to APT incidents is significantly higher. The expected losses incurred in the 2013 Target Corporation data breach and the 2011 RSA data breach were $248 million and $66 million, respectively [1], [2].

APT actors enter target systems by evading existing signature- or anomaly-based detection and prevention systems. Afterward, these actors establish a presence and persistence in the network by compromising multiple accounts and hosts. This process of expansion and persistence of the APT is commonly known as *lateral movement (LM)*. Lateral movement usually happens in conjunction with *command and control* (C&C) communications to gather internal system structure information, guide the expansion process, and ultimately cause data exfiltration [3], [4].

The challenges involved in detecting APTs are multifold. First, attack tools are constantly evolving and adapting to changes in cyber defense technologies [5]. Attackers can stay undetected over prolonged time periods by using traditional operating system and networking services as their attack vectors [3], [6], [7]. Second, the attackers are not bound to follow any particular defined progression in compromising a target system [7], [8]. Finally, there is a lack of data sources about the actual behavior of attackers during the lateral movement stage, which limits the research community's ability to effectively build and evaluate models of lateral movement-based attacks and their defense mechanisms. Numerous successful incidents show that the current security solutions fall short in addressing these challenges [1], [6], [7].

We observe that regardless of the attack vectors that the attackers choose to adopt, compromised machines necessarily have to communicate with external C&C servers. The malware also needs to spread laterally in the network to reach the inner segments that are not exposed to the outside Internet. In fact, C&C and lateral movement are common, and necessary, patterns of attack behaviors; they have persisted in studied APTs over the past seven years [9].

In this paper, we present an attack-vector-independent approach to detect lateral movement-based attacks (e.g., APTs) in an enterprise network system. We concentrate on the broader patterns of interactions that an attacker needs to have with the system after gaining initial entry. To analyze diverse indicators of C&C and LM compromises in a collective way, we propose a graph-based model of the target system, which we call a *host communication graph (HC graph)*. The HC graph represents the internal communication between target system hosts, along with the data collected from C&C and LM monitors.

First, we extract features from the HC graph to evaluate both the C&C and lateral movement-specific disruptions. Next, we use the features to develop two anomaly detection methods. In the first method, we identify C&C infected hosts and use principal component analysis (PCA) and k-means on the lateral movement-related features to find hosts that behave much like the infected hosts. In the second method, we perform extreme value analysis on the data transformed by PCA to identify hosts infected by malicious lateral movement. Since the two anomaly detectors use diverse indicators to determine infected hosts, we combine their outputs to generate the final set of compromised hosts. We use a parameter-based averaging ensemble method. We mix different proportions of the flagged hosts from the two detectors, and, for each host, compute the average label value. Our approach is unsupervised, since we do not use the true labels to train the ensemble approach. Therefore, we measure the accuracy of detection by changing the threshold of the average label value above which we consider a host to be compromised.

For our experiments, we use a dataset of normal network operations from the Los Alamos National Lab's enterprise network [10]. To inject attack traces in the dataset, we develop a model for C&C and LM activities using the susceptible-infected-susceptible (SIS) virus spread model [11]. The model implements a small set of tunable parameters that allow us to simulate different types of LM activities, including both benign administrative tasks and malicious APT traffic. We use these parameters to evaluate the proposed method's sensitivity to changes in the attacker's behavior. In particular, we vary the number of hosts that are part of the malicious LM and the number of hosts that are infected by C&C malware and then compute the true and false positive rates (TPR, FPR) of detection in each setting.

On average (computed over all experiment runs), we achieve a TPR of 88.7% (confidence interval 1.9%) and an FPR of 14.1% (confidence interval 3.6%). Even in the atypical case when the attack has compromised almost all the hosts in the network, the TPR realized is 82.6%. Analysis of the ensemble detector's accuracy done by varying the classification threshold of the average label value provides insights that can be used by security analysts to tune the detectors.

More specifically, we make the following contributions:

1) Feature identification and extraction: We identify characteristic patterns of malicious lateral movement and C&C, and represent them as statistical features that we extract from diverse security monitors, mainly NetFlow, LM, and C&C monitors (Sections IV-A, IV-B).

2) Anomaly detection: We use the proposed features with an ensemble of unsupervised anomaly detection techniques to build an automated method to identify compromised hosts in the target network (Section IV-C).

3) Attack trace generation: We provide a method for generating malicious as well as benign lateral movement traces, based on the SIS virus spread model. We injected the generated traces in an enterprise system dataset consisting of internal network traffic (Section V).

4) Trace-based evaluation: We perform trace-based simulations to evaluate the proposed system's accuracy in detecting compromised hosts for two attack communication models. (Sections V, VI).

II. Threat Model

In this work, we consider network-based APT attacks on enterprise systems from external actors, mainly involving C&C and lateral movement. We refer these attacks as lateral movement-based attacks. We address the problem of detecting such attacks with high accuracy and a low false alarm rate.

We abstract the lifecycle of lateral movement-based attacks into the following stages: (i) initial compromise, (ii) command and control, (iii) lateral movement, and (iv) data exfiltration or service disruption. These stages are essential parts of several models of a lifecycle of an APT [3], [4], [12] and previously studied incidents [1], [6], [7].

Initial compromise consists of reconnaissance, delivery of malware, and establishing of backdoor communication. Subsequently, the attacker establishes C&C channels between the victim machines and a set of external servers, with the goal of maintaining active control over the compromised hosts. In conjunction with C&C, the attacker attempts to move laterally to widen the compromised region. The last step includes the actual execution of the malicious actions, whether they are data exfiltration, service disruption, or physical damage.

In this work, we particularly focus on C&C and lateral movement. During C&C, a set of compromised machines send automated periodic beacons to attacker-controlled servers, awaiting future instructions. These commands intend to compromise other hosts in the network or to gather intelligence and sensitive data. The Internet Relay Channel has traditionally been the most preferred communication tool for C&C, but attackers have started opting for more commonly used protocols, such as HTTP(S), FTP, and SSH, to evade easy detection. For more robust communication infrastructure, attackers may use dynamic domains and P2P server architectures.

Attacker lateral movement consists of several tactics including internal reconnaissance, credential stealing, vulnerability exploitation, and privilege escalation. In fact, the exploit techniques that are used to achieve such goals are ever-changing and evolving. Examples of such techniques include use of application deployment software to deploy malware, exploitation of vulnerability to escalate privileges, and use of pass-the-hash to bypass standard authentication steps [5].

We believe that since the C&C and lateral movement are closely correlated during the attacks, we can better defend systems by analyzing these two stages in a holistic way. Although significant research has been done separately on the detection of initial compromise, C&C, and data exfiltration, unfortunately, there has been relatively little research effort on the detection of attacker lateral movement. Detecting attacks in C&C and lateral movement stages can thwart an APT campaign and prevent actual damage.

III. Approach Overview

A. Overview

To detect the coordinated malicious activity of C&C and lateral movement, we need to construct an estimate of system-wide state such that the different types of indicators emerging from several hosts can be correlated with one another. Additionally, to act upon the system-wide state over long time windows, we need to keep the volume of information restricted, because such attacks can persist in the victim systems from over a few weeks to several months [13].

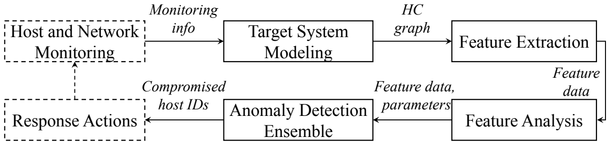


Fig. 1: High-level architecture of the proposed approach. We mainly focus on the modules shown in boxes with solid lines.

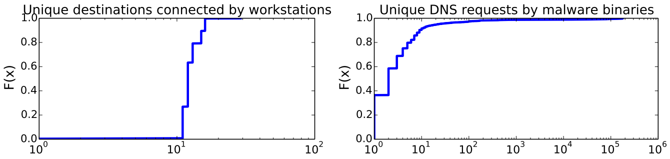
Figure 1 shows different components of the proposed system. We assume that the network under consideration is equipped with monitoring tools to record host- and network-level information. In this paper, we do not build these monitors, but use the information recorded by the monitors to model the current state of the target system. We refer to this model as an *HC graph*. We use the HC graph to extract features related to C&C and LM behavior. We then standardize these features and analyze their distributions to perform anomaly detection. In the anomaly detection step, we use a combination of different types of statistical anomaly detection techniques to detect the set of compromised hosts. The ensemble of multiple anomaly detectors allows us to achieve high accuracy while keeping the false positive rate low. Finally, we can use the results of attack detection to implement different types of attack response actions.

In what follows, we explain the three types of monitors that we use in the proposed system. The remaining modules of the proposed system are described in Sections IV, V, and VI.

B. Monitors

1) Network Flow Monitor: A network flow monitor processes internal network traffic and generates flow records. For our purposes, we assume that such a monitor produces a time series consisting of source and destination machines, as well as ports information. Examples of such monitors are an internal firewall, a network IDS, and a NetFlow-capable internal router.

2) C&C Monitor: A C&C monitor produces evidence that a host is involved in external network communication with a potential C&C server. We believe that such external traffic can be distinguished from normal traffic in a typical enterprise network. To test and validate this hypothesis, ideally, we would use a dataset consisting of both typical enterprise communications and possible C&C connections with a perfect labeling of each network flow to identify it to one of the two classes. However, such dataset is difficult to obtain in practice. Therefore, we studied the external communication behavior of hosts in two datasets, the VAST Challenge 2013 MC3 (VAST) [14] and the GT malware passive DNS data daily feeds (GT) [15]. Figure 2a shows the cumulative distribution of the number of unique destinations contacted by hosts over a three-day period in the VAST dataset, which has no APT-like attacks in it. We found that the workstations most often connect to a small number of external destinations. We observed that the majority of probability mass (between the first and third quartiles) lies between 8 and 17 unique destinations. Figure 2b shows the distribution of the number of unique DNS requests made by malware-infected hosts over the three-day period (Jan. 17–19, 2017) in the GT dataset. The malware binaries try to resolve thousands of domain names, which indicates that the infected hosts are likely to make a large number of unique external connections. We conclude that the outward communication behavior of infected hosts can be distinguished from the behavior of typical enterprise hosts.



(a) (b)

Fig. 2: Empirical CDF of (a) number of distinct destinations contacted by workstations in VAST 2013 network and (b) number of distinct DNS queries made by malware binaries in GT-MPDNS dataset.

We thus assume the presence of C&C monitors that analyze network communication of the hosts and assign a suspiciousness score to each host; we call it the `c_score`. As an example, we can compute the `c_score` simply as the number of connections that the host makes to unique destinations over the observed period. Discussion of design and analysis of the C&C monitors is beyond the scope of this paper, which can be found in [16], [17].

3) LM Monitor: We can characterize lateral movement by a set of related chains of communications between different hosts in the network. An LM monitor creates traces of connections that are chained together based on their dependence on each other. Two types of monitors are prominently used to construct these chains: (i) network-level monitors and (ii) host-level monitors. In [18], the authors propose to correlate anomalous hosts with network events and establish attack causality in the context of epidemic-spreading attacks such as worms and botnets. Fawaz et al. [19] provide a low-overhead way to obtain a more refined view of host behavior. In this approach, a host-level monitor is responsible for analyzing kernel-level activity and generating causally related connection pairs. These connection pairs are then aggregated hierarchically to produce network-wide chains. In our work, we assume the presence of LM monitors and use the system-wide communication graph constructed by them. This graph includes both benign and malicious LM chains if present. We refer the reader to [18], [19] for more details on the working of these monitors.

The aforementioned techniques can establish dependence relations between different network connections with varying levels of accuracy. They do, however, fall short of distinguishing between benign and malicious hosts. The size and complexity of modern systems and the lack of knowledge about attacker activities make the detection of malicious chains challenging. Our approach addresses these challenges by correlating lateral movement and C&C indicators to identify malicious events.

C. Target System Model

We use a graph-based model of the target system to combine the information collected by the three monitors: internal network communication, C&C, and lateral movement activity. We represent the model as a directed graph $G(V, E)$, called the *host communication* (HC) graph. The set of vertices, $V = \{1, 2, \ldots, n\}$, represents the hosts in the system. The set of edges, $E \subseteq V \times V$, represents the network communication between the hosts. E consist of all types of communications happening in the network, including normal system operation and attacker actions. In practice, an HC graph can be generated, for example, by analyzing NetFlow records or firewall logs of the internal network traffic.

We define a vertex labeling function as $f_{\mathrm{c}} : V \to [0, 1]$, which labels every vertex of the graph G with its c_score between 0 and 1. The c_score quantifies the extent to which a particular node is under C&C attack. This approach allows us to model different C&C attack patterns by generating different c_score distributions over the graph G.

We next define a lateral movement graph as $l(V_{\mathrm{l}}, E_{\mathrm{l}}) \subseteq G$ such that $\exists u, v, w \in V_{\mathrm{l}}$ if the connections (u, v) and (v, w) are chained together by the LM monitors. Graph l represents a separate LM behavior. $\mathcal{L}$, which is the set of all these graphs, represents all types of lateral movements in the system. It is important to note here that we do not distinguish between benign and malicious lateral movement at this point.

The HC graph thus allows us to combine diverse indicators and construct a system-wide state.

IV. Detecting Lateral Movement-Based Attacks

The proposed lateral movement detection approach is as follows. First, we identify the features that are most useful for detecting the attacks. We then extract the features using data collected via the monitors and the HC graph. We standardize the feature values and analyze their distributions and finally use an ensemble of two anomaly detectors to identify compromised hosts.

A. Feature Extraction

We extract the following features to represent the current state of every host:

C&C Score: The first feature extracted ($\mathbf{f}_{cs}$) is the c_score of a host that is generated by the C&C monitor. Using $\mathbf{f}_{cs}$, we aim to identify suspicious external communication of hosts.

Lateral movement-related features: The next set of features is based on the lateral movement graphs that are generated by the LM monitor. Each lateral movement graph is a subgraph of the overall host communication graph G.

The second feature is the number of lateral movement graphs ($\mathbf{f}_{num}$) passing through a host. We formulate the extraction of $\mathbf{f}_{num}$ as a set of matrix operations. Consider a lateral movement graph $l = (V_l, E_l) \in \mathcal{L}$ represented by an $n \times n$ adjacency matrix M, where n is the number of nodes in the graph G. We compute the feature vector $\mathbf{f}_{num}$ as follows:

$$\mathbf{f}_{num}(i) = \sum_{l \in \mathcal{L}} (\bigvee_{j} \mathbf{M}_{ij} \vee \bigvee_{j} \mathbf{M}_{ji}),\ i, j \in \{0, 1, \ldots, n-1\} \tag{1}$$

Equation 1 first determines whether a host i is part of a chain l. To do that, it performs a Boolean sum of all the elements in i^{th} row and i^{th} column of the matrix $\mathbf{M}$ to generate a binary output. A result 1 then indicates the node is part of chain l. Finally, to obtain the number of chains that pass through the node i, the equation computes an arithmetic sum of the binary outputs for all $l \in \mathcal{L}$. By following the same process for each host, we obtain the $(n \times 1)$ vector $\mathbf{f}_{num}$.

Next, we extract four features, (i) $\mathbf{f}_{mean}$, (ii) $\mathbf{f}_{range}$, (iii) $\mathbf{f}_{iqr}$, and (iv) $\mathbf{f}_{mad}$, corresponding to the mean, range, interquartile range (IQR), and median absolute deviation (MAD) of the lengths of lateral movement graphs, respectively. The motivation for including these features is that in addition to the disparity induced in the number of graphs passing through different hosts, attacker lateral movement is also likely to cause variations in the sizes of these graphs. We believe that these four features will enable us to adequately summarize the properties of lateral movement graphs to separate benign and malicious lateral movements.

We start by computing mean and range to account for the central tendency and dispersion of the distribution of lengths of lateral movement graphs, respectively. We define the length of a lateral movement graph by the number of vertices ($|V_l|$). To compute $\mathbf{f}_{mean}$, we sum together the lengths of the graphs passing through each host and then use the $\mathbf{f}_{num}$ feature vector to compute the average. Similarly, for each host h, $\mathbf{f}_{range}$ is simply the difference between the max and the min values of the lengths of the chains passing through h.

$\mathbf{f}_{mean}$ and $\mathbf{f}_{range}$ help in comparing characteristics of different hosts, but they are overly sensitive to outliers. Also, since the distribution of lengths of lateral movement graphs passing through a given host may not be a Gaussian distribution, standard deviation (or variance) will not be a suitable statistic to compute. To obtain the insights missed by $\mathbf{f}_{mean}$ and $\mathbf{f}_{range}$, we also compute IQR and MAD.

The IQR of a sample, also known as the 25% trimmed range, is defined as the difference between the 75th and 25th percentiles of the sample. Compared to the range, which is based solely on the two most extreme values, IQR measures the dispersion of the central 50% of samples, and provides insights that cannot be obtained from the range alone.

The MAD of a dataset is defined as the median of the absolute values of the differences between the individual samples and the overall median. For a host h, let $\mathbf{len}_h$ be the vector of the length of the LM chains passing through h. We compute the value of feature $\mathbf{f}_{mad}$ as follows:

$$\mathbf{f}_{mad}(h) = med\left(\{|len - med(\mathbf{len}_h)|, \text{for } len \in \mathbf{len}_h\}\right) \tag{2}$$

where $med()$ is the median of a sample. In summary, we use the information in the host communication graph to extract six $n \times 1$ feature vectors $\mathbf{f}_{cs}$, $\mathbf{f}_{num}$, $\mathbf{f}_{mean}$, $\mathbf{f}_{mad}$, $\mathbf{f}_{iqr}$, and $\mathbf{f}_{range}$.

B. Feature Analysis

After the extraction, we scale each feature vector using its maximum absolute value. This method does not shift the centroid of the data, and thus does not eliminate the sparsity that is present in the feature vectors.

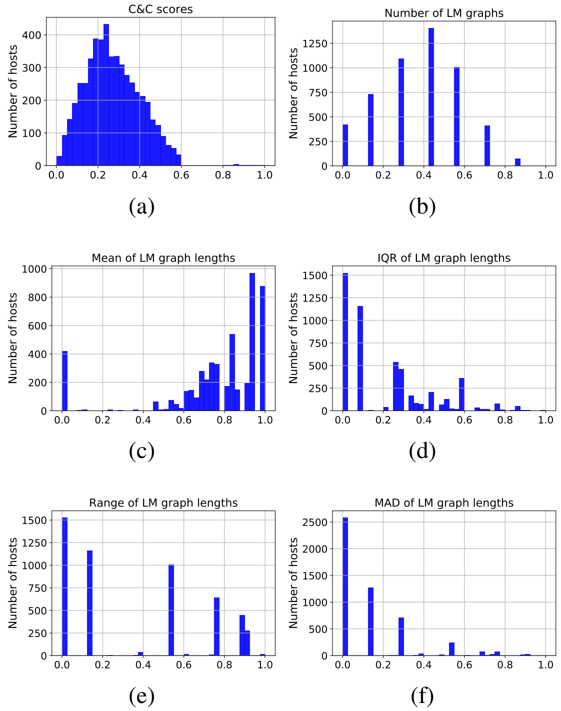
Figure 3 shows the distributions of the scaled feature vectors for experiment configuration E1.3 (ref. Table II). The distributions of the features $\mathbf{f}_{cs}$, $\mathbf{f}_{iqr}$, $\mathbf{f}_{range}$, and $\mathbf{f}_{mad}$ show a positive skew as the mass of distribution is concentrated on the left of the plots. The distribution of $\mathbf{f}_{mean}$, on the other hand, has a negative skew. A possible reason for the negative skewness of $\mathbf{f}_{mean}$ may be the fact that a larger variability in the lengths of chains affects the mean more than it affects the IQR and MAD. A few very long chains (probably benign) can significantly skew the distribution of the mean. For different types of attack, the compromised hosts can be in different regions. For example, if an attacker compromises a small number of hosts with a C&C malware, these hosts are then likely to have larger values of $\mathbf{f}_{cs}$ (e.g., greater than 0.5 in Figure 3a). As another example, if the attacker generates several LM chains of different lengths, then the hosts that have higher values of $\mathbf{f}_{num}$, $\mathbf{f}_{iqr}$, and $\mathbf{f}_{range}$ are more likely to be compromised, since the normal nodes should have fewer chains, most of which will be of similar lengths.

Next, we perform PCA to analyze the correlation among the features and reduce the dimensionality of the dataset. During all the experiments we performed, we could project the data to a two-dimensional space (using the first two PCs) while retaining more than 95% of the original variance in the data. In addition to improving the performance of analyses, the lower-dimensional data also helps in visualizing the distributions. We use these two properties to build an anomaly detection method and select values for different thresholds required during the process. We provide more details on the specific experimental configurations in Section V.

C. Anomaly Detection Methodology

We now present an approach to classify the network hosts as normal or compromised. Doing so involves two important considerations. First, we aim to build unsupervised techniques because of the unavailability of labeled datasets of network traffic. Second, we seek a classification method that performs accurately even when attacks deviate from the assumed pattern. Although we use an assumed threat model as a guiding template to build the defense mechanism, we do not make strong assumptions related to the size of attack or the sequence of the attacker's actions.

The traditional anomaly detection methods that assume that the size of the anomaly (e.g., the number of compromised hosts) is significantly smaller than the size of normal activity do not fit directly in our case [20]–[22]. We thus propose to use a combination of multiple anomaly detection algorithms to achieve high accuracy and robustness while facing variations in attacker behavior. Intuitively, the approach is first to find the hosts that exhibit at least one of the two types of suspicious indicators—high `c_score` or high disparity in lateral movement chains—and then use the relative similarity of hosts to identify compromised hosts that do not directly show any of the suspicious indicators. Our hypothesis is that the features will help us perform both tasks.



(a) (b)

(c) (d)

(e) (f)

Fig. 3: Distributions of scaled feature values computed using the host communication graph from a network of $5,151$ computers that we used in the experiments.

1) Anomaly detection using PCA and k-means: First, we perform PCA using the five lateral movement-based features. In the evaluation dataset that we use, the first two principal components (PCs) capture more than 95% of the original variance in the data. Therefore, we project the original data to a two-dimensional space. We then use the k-means algorithm to form clusters of similar data points. To select the optimal value of k, which is the number of assumed clusters in the dataset, we use silhouette analysis [23]. The silhouette value of a data point is a measure of how well it lies within its cluster as compared to the other clusters. We start by choosing a range of possible values of k. For each value of k, we perform the clustering and compute the average silhouette score of each cluster, which can range from -1 to 1; values closer to 1 indicate well-separated and cohesive clusters. We thus choose the value of k that results in the maximum average silhouette score for all points.

The next task is to identify clusters that contain compromised hosts. Since we perform clustering using lateral movement-related features, we use suspicious C&C behavior for the identification. We use MAD-based anomaly detection on $\mathbf{f}_{cs}$. MAD has a 50% breakdown point; the estimate remains bounded when fewer than 50% of the data points are replaced by arbitrary numbers. Because of this high stability in the presence of outliers, MAD can be used as a tool for anomaly detection [24]. The MAD-based outlier score is computed as follows:

$$score(i) = \frac{|\mathbf{f}_{cs}(i) - median(\mathbf{f}_{cs})|}{MAD(\mathbf{f}_{cs})} \tag{3}$$

Equation 3 generates a numeric score for each point in the vector $\mathbf{f}_{cs}$. We then flag all the points that have values greater than a threshold (θ_{mad}) to represent anomalies. Finally, we identify the clusters that cover at least 90% of the anomalous points and output all the hosts that are part of these clusters.

2) Anomaly detection using PCA and extreme value analysis: In the second method, we identify anomalous hosts using indicators of suspicious lateral movement. Our first insight is that the compromised hosts will have a large number of chains of different lengths passing through them. Similar observations have been presented in [18], [25], [26]. For example, Sekar et al. [18] found that the communication graph generated by normal hosts is sparse and significantly follows a client-server model. Therefore, a presence of diverse graph-structures in the host communication pattern is suspicious. Such behavior will result in large values for features $\mathbf{f}_{num}$, $\mathbf{f}_{iqr}$, and $\mathbf{f}_{range}$. However, it is relatively difficult to reason about the values in three-dimensional space, and since these three features are highly correlated, we perform PCA. We find that the first PC alone captures more than 90% variance. We project the data on the first PC and analyze the distribution. Doing this allows us to easily filter out the points that have large values on the first PC. These points correspond to the anomalous hosts. Specifically, we consider the points that have values greater than a threshold ($\theta_{pc.low}$) to be anomalous.

Our second insight is that the benign hosts should be part of either no chains or benign chains that originate from an administrative activity. From the viewpoint of $\mathbf{f}_{mean}$ and $\mathbf{f}_{mad}$, these two cases should result in either very small or very large values for these features. For example, admin workflows such as automated scripts in Windows Management Instrumentation and patch management via Windows Update Server would result in large graphs originating from the admin server. Conversely, the suspicious hosts should be found in the middle portion of the distributions. Following the same approach as in the previous case, we project data on the first PC after doing PCA on $\mathbf{f}_{mean}$ and $\mathbf{f}_{mad}$. This provides us with the ability to easily identify extreme values. Specifically, we compute the mid-range of the data projected on the first PC and flag as anomalous all the hosts that lie within a threshold ($\theta_{pc.mid}$) distance from the mid-range.

The final output of the second anomaly detector is the intersection of the two sets of hosts obtained above. These hosts exhibit both indicators of a suspicious lateral movement.

3) Combining the outputs of two anomaly detectors: We presented two detectors that use different types of indicators to identify anomalous hosts. Since (i) we do not have access to the true labels to train the anomaly detection and (ii) we cannot assume the number of compromised hosts for either C&C or lateral movement attacks, we need to rely on an ensemble of the two detectors to achieve high accuracy.

Algorithm 1 shows the proposed parametric ensemble method. Our approach is inspired by the work presented in [27]. The algorithm takes as input (i) the two sets of anomalous hosts ($\mathcal{A}$ and $\mathcal{B}$); (ii) the number of hosts in the network (N); and (iii) a step size ($step$). The main parameter of the algorithm is $frac$, which denotes the fraction of points to select from $\mathcal{A}$ and, correspondingly, the $(1 - frac)$ fraction of points to select from $\mathcal{B}$.

Algorithm 1 Anomaly Detection Ensemble

```
 1: Input: $\mathcal{A} \leftarrow$ detector1 output; $\mathcal{B} \leftarrow$ detector2 output; $N$; $step$
 2: Output: $\mathcal{F}$
 3:
 4: $l_1 \leftarrow |\mathcal{A}|$
 5: $l_2 \leftarrow |\mathcal{B}|$
 6: $frac = 1.0$
 7: $\mathcal{F} = [0]_{1 \times N}$
 8: while $frac \geq 0.0$ do
 9:     $n_1 \leftarrow frac * l_1$
10:     $tmp \leftarrow$ getBest $(\mathcal{A}, n_1)$
11:     for all $i \in tmp$ do
12:         $\mathcal{F}[i] \leftarrow \mathcal{F}[i] + 1$
13:     end for
14:     $n_2 \leftarrow (1 - frac) * l_2$
15:     $tmp \leftarrow$ getRandom $(\mathcal{B}, n_2)$
16:     for all $i \in tmp$ do
17:         $\mathcal{F}[i] \leftarrow \mathcal{F}[i] + 1$
18:     end for
19:     $frac \leftarrow frac - step$
20: end while
```

The approach to select the points from $\mathcal{A}$ (in line 10) is based upon the *goodness* of each point, which is equal to the silhouette score of the point with respect to the clusters formed by the first detector. Therefore, *goodness* is a score within the range $[-1, 1]$, and a value closer to 1 is better. We then select remaining points from set $\mathcal{B}$ at random (line 15). For every point selected by this method, we increment the corresponding value in the output set $\mathcal{F}$. The variable $step$ decides the number of times the main loop executes, which is equal to $\lfloor \frac{1}{step} \rfloor$. When the algorithm ends, the values that we get in $\mathcal{F}$ indicate the number of times a point was flagged as anomalous. Finally, to classify a point as normal or compromised, we use the threshold θ_{avg}. A smaller value of θ_{avg} would classify a large number of hosts as compromised, and would also lead to a higher false positive rate. During the evaluation of the proposed approach, we used the results of the ensemble algorithm with different values of θ_{avg} to analyze the trade-off between accuracy and the false alarm rate.

4) Parameter selection for anomaly detection algorithms: In addition to deciding an optimal value of k for clustering, we need the following parameters for anomaly detection. The first is the threshold of the MAD-based outlier score (θ_{mad}) above which we consider a `c_score` anomalous. We always use a value 2 for θ_{mad}. Since MAD is a very robust measure of dispersion, a value of 1.5 or above for the outlier score (Eq. 3) works well in most applications [24]. By choosing a value of 2, we pick a relatively strict threshold and ensure that the flagged points are really outliers. Next, the second anomaly detector needs two thresholds, $\theta_{pc.low}$ and $\theta_{pc.mid}$. To decide on good values of these two thresholds, we perform PCA on the dataset for different settings and analyze the distribution of the data projected on the first PC. We find that a value of

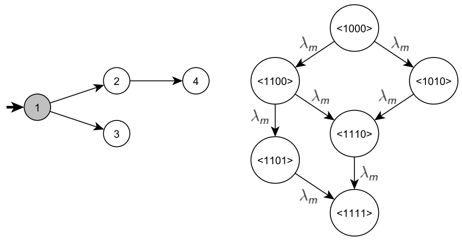


Fig. 4: Example of an LM chain and its corresponding CTMC.

0.25 for each of the two thresholds provides a good separation of different groups of points.

V. Experimental Evaluation

We performed experiments to evaluate the accuracy of the proposed anomaly detection approach in identifying compromised hosts. The experiments used internal network flow logs obtained from the Los Alamos National Lab's (LANL's) Comprehensive Muti-Source Cyber-Security Events dataset [10]. The complete dataset contains anonymized traces of 58 consecutive days from LANL's enterprise network, which is composed of 17,684 host computers. We used network flow logs only for the first day; they include a total of 8,096 active hosts and 73,150 network flows among the hosts. The dataset does not contain any traces of lateral movement-based attacks. We used the LANL NetFlow logs and injected C&C and lateral movement traces to generate the HC graph.

A. Host Communication (HC) Graph Generation

During the analysis of the NetFlow records, we found that out of a total of 8,096 hosts, only 5,151 were part of a single connected component. More than 99% of the remaining 2,945 hosts did not connect to other hosts over the one-day period. Therefore, we considered the large connected cluster of $n = 5151$ hosts and constructed a communication graph consisting only of the NetFlow information. Furthermore, since the proposed approach does not require knowledge of port numbers and flow sizes, we aggregated all the flows between a pair of hosts as a single edge in the generated NetFlow graph. Next, we augmented the NetFlow graph with simulated attack traces. As mentioned earlier, we assume the presence of benign lateral movement activity in the network, such as patching and Windows Management Instrumentation traffic. Our distinction between benign and malicious LM traces was based on the observation that the APT attackers are more targeted towards escalating their privileges and moving deeper into the network. The attackers' spread in the network is slower than that of benign administrative tasks because the attackers need to avoid detection by having their malicious traffic masquerade as benign traffic.

Now we describe the approach for generating $\mathcal{L}$ chains of communication, out of which we consider $\mathcal{M} \subseteq \mathcal{L}$ to be malicious. Each chain $l \in \mathcal{L}$ is a subgraph of the NetFlow graph generated earlier. We generate the communication chains (both benign and malicious) according to the continuous-time N-intertwined *susceptible-infected-susceptible* (SIS) model [11], which is widely adopted by researchers for modeling the spread of epidemics and computer worms and viruses [28].

Algorithm 2 LM Chain Generation

```
**Input:** $G(V,E)$, $v_0$, $\lambda$, $is\_malicious$
**Output:** $C(V_c, E_c)$

$Visited(v) := \bot \quad \forall v \in V$
$V_c := \phi, \quad E_c := \phi, \quad C :=< V_c, E_c >$
$Visited(v_0) := \top$
$\mathcal{E} := \{e \in E \mid e = (v_0, w) \in E\}$
Let $len = length(\mathcal{E})$
**while** not done **do**
    Sleep for $\tau \sim Exp(len \times \lambda)$
    Choose $e = (u, w)$ target edge from $\mathcal{E}$ w.p. $1/len$
    **if** $is\_malicious$ **then**
        Compromise $w$ w.p. $p_c(w)$
        **if** not successful **then**
            **continue**
        **end if**
    **end if**
    $Visited(w) := \top$
    $V_c := V_c \cup \{w\}$
    $E_c := E_c \cup \{e\}$
    $\mathcal{E} := \mathcal{E} \setminus \{e' = (v, w) \in \mathcal{E}$ for any $v \in V_c\}$
    $\mathcal{E} := \mathcal{E} \cup \{e \in E \mid e = (w, v) \in E \wedge (\neg\, Visited(v))\}$
**end while**
```

Given a communication graph $G(V, E)$, we represent the state of each vertex $v \in V$ with a Bernoulli random variable $X_v \in \{0, 1\}$ such that for each chain $l(V_l, E_l) \in \mathcal{L}$, $X_v = 1$ iff $v \in V_l$. A node $v \in V$ is part of a chain l at time t with probability $p_v(t) = P[X_v(t) = 1]$, and is healthy with probability $1 - p_v(t)$. Once a chain l passes a node v, spread from v to a neighboring node w occurs at an exponential rate of β_m for malicious chains and β_b for benign chains. The spreading rate per edge is thus a Poisson process with rates β_m and β_b for malicious and benign chains, respectively. In fact, the N-intertwined SIS model can be represented as a continuous-time Markov chain (CTMC) with $2^{|V|}$ states. We assume that $\beta_b > \beta_m$ since attackers tend to be slower and stealthier than administrators as previously discussed.

Figure 4 shows a sample lateral movement chain, along with its equivalent CTMC. The initial state corresponds to the attacker's establishing a point of entry into the system through node 1 (or an admin's starting of a machine, in the case of a benign chain). The CTMC then captures the attacker's possible moves and corresponding rates, according to the N-intertwined SIS model. The attacker can compromise either node 2 or node 3, each with a rate of compromise λ_m. For a fully connected graph with n nodes, the size of the CTMC is 2^n, since from any node, the attacker can compromise any other node. However, in practice, network graphs are not fully connected (given segmentation into domains), and thus the state space would be significantly reduced.

To capture the correlation between C&C and lateral movement, we start the trace generation with an initial distribution of `c_scores` of hosts, such as uniform random or triangular distribution. During the lateral spread, we adopt a probabilistic update of `c_score` values. In particular, we set the `c_score` of a newly infected destination node as the maximum of its old `c_score` and the source's `c_score` with a probability

TABLE I: Summary of the simulation parameters

Parameter	Parameter
Number of nodes ($n = 5,151$)	Initial c_score distribution*
Benign rate ($\beta_b = 0.8$)	Initially compromised nodes ($k = 15$)
Malicious rate (β_m)*	Compromise success probability (p_c)*
Update score probability (p_{cs})*	Simulation time ($T = 10$)
Simulation runs ($runs = 100$)	

TABLE II: Configurations for the first experiment (E1)

Configuration	E1.1	E1.2	E1.3	E1.4	E1.5
β_b/β_m	8.0	4.0	2.0	1.3	1.0
# nodes in benign LM	4369	4418	4342	4325	4294
# nodes in malicious LM	66	326	1403	3594	4504

p_{cs}. Different values of p_{cs} allow us to simulate different types of C&C behaviors of attackers. We also make sure that each node $v \in \mathcal{V}$ that has a very high c_score (e.g., 0.8 or more) has at least one chain passing through it.

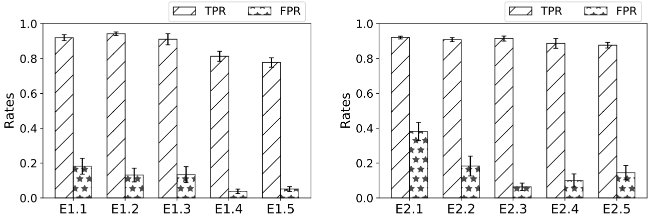
We show the attack trace generation in Algorithm 2. The simulation approach that we used was inspired by the work presented in [29]. At any time step t, let k be the number of vulnerable nodes to compromise; the simulation would then generate a random variable τ with an exponential distribution with rate $k \times \beta_m$ for a malicious chain, and $k \times \beta_b$ for benign chains. We would then advance the simulation time to $t + \tau$, sample a node w to compromise from the k susceptible nodes uniformly at random (i.e., each node would have a probability of $1/k$ of being selected), and then add w and its corresponding edge to the generated chain. All neighbors of the newly compromised node w would then be added to the set of susceptible nodes. We note that in the case of a malicious chain, the compromise of the node w could fail with probability $1 - p_c(w)$, so we restart the simulation from the same state at time $t + \tau$.

Attack trace generation thus generates an HC graph consisting of network flows, lateral movement chains (both benign and malicious), and c_scores.

B. Results

We parsed the LANL dataset and implemented in Python the code for attack trace generation, feature extraction, feature analysis, and anomaly detection. We used the implementations of PCA and k-means provided by Python's scikit-learn machine learning module [30].

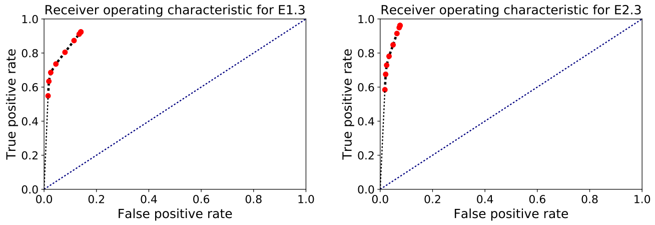
As we mentioned in Section IV-C, the traditional anomaly detection methods (e.g., PCA-based and clustering-based) do not directly fit since we cannot assume the number of compromised hosts to be very small as compared to the number of benign hosts. Moreover, to the best of our knowledge, there are no unsupervised approaches proposed previously to detect lateral movement-based attacks. Therefore, we were unable to compare our results with other methods. Instead, we focused on measuring the effect of changing attacker behavior on the accuracy of detecting compromised hosts. We evaluated the true and false positive rates (TPR and FPR) in two different experimental configurations (configs) that we created by using different values of the model parameters marked with an asterisk in Table I. To compute TPR and FPR, we used the ground truth obtained from the attack injection. We consider a host to be benign only when no malicious chain traverses it.



(a) Changing the number of hosts in malicious LM chains.

(b) Changing the number of C&C-infected hosts.

Fig. 5: TPR and FPR of detecting compromised hosts.



(a) Config E1.3.

(b) Config E2.3.

Fig. 6: ROC plots with changing average label classification threshold from 1.0 to 0.1.

1) Varying the number of nodes in malicious LM chains: In the first experiment, we varied β_m while keeping β_b fixed. We also set both p_c and p_{cs} equal to 0.2. This experiment corresponds to the different speeds with which the attacker moves laterally by compromising network hosts. We show the resulting configs in Table II.

The proposed approach was able to detect the compromised hosts with high accuracy, as shown in Figure 5a. For example, there were 66 compromised hosts in config E1.1, out of which 60 hosts were detected by the proposed approach. In config E1.3, in which a larger number of hosts were part of malicious chains, the average FPR was only 13.13%, while an average TPR of 91.08% was still achieved.

Although the average FPR is relatively high in these cases (i.e., 18.21% in E1.1 and 13.40% in E1.3), we can tune the classification threshold (θ_{avg}) of the ensemble algorithm to decrease the FPR. We implemented the ensemble approach from Algorithm 1 with $step = 0.1$ to generate ten sets of output labels. We then varied the θ_{avg} from 0.1 to 1.0 and computed TPR and FPR in each case. We show the results for config E1.3 as an ROC curve in Figure 6a. For the case in which a host was considered compromised once it had been flagged as anomalous by all ten runs of the ensemble detector ($\theta_{avg} = 1.0$), the average TPR and FPR were 54.84% and 1.52%, respectively. Both the TPR and FPR increased as we decreased the threshold by 0.1 in each experiment. Finally, with $\theta_{avg} = 0.1$, we got TPR = 92.39% and FPR = 14.06%. These results provide insights that will help us select a desirable balance for TPR and FPR.

TABLE III: Configurations for the second experiment (E2)

Configuration	E2.1	E2.2	E2.3	E2.4	E2.5
$p_c = p_{cs}$	0.2	0.4	0.6	0.8	1.0
# nodes in benign LM	4407	4360	4494	4410	4448
# nodes in malicious LM	87	342	850	2072	2422

2) Varying the number of C&C-infected nodes: Table III shows the configs generated when we increased the probabilities of successfully moving to a host (p_c) and compromising it with C&C malware (p_{cs}). We used $\beta_b/\beta_m = 4.0$. Here we model the attackers that can install C&C malware on several hosts, which would then start connecting to potential C&C servers. Therefore, the identification of compromised hosts mainly happens through use of outlier c_scores. The number of hosts that are part of malicious LM chains, however, does not increase as fast as in experiment E1.

We show the results for average TPR and average FPR in Figure 5b, and a ROC curve for varying the classification threshold θ_{avg} for config E2.3 in Figure 6b. We achieved an average TPR of 90.10% across all the configs. The average FPR is relatively high in config E2.1 when both the C&C and the lateral movement indicators are not strong. The FPR goes to an acceptable level (e.g., 6.39 in E2.3 and 10.05 in E2.4) because the C&C indicators are visible in the network.

In summary, by using an ensemble of the two anomaly detectors, we achieved an overall average TPR (computed over all experiments) of 88.7% even when attacks happened with different intensities. We achieved low FPR when at least one type of suspicious indicators was present in the network. The classification threshold is a tunable parameter, and we can adjust it to decrease the FPR at the cost of a small decrease in the TPR.

VI. Limitations and Potential Solutions

The proposed approach can identify compromised hosts by using simple statistical features related to C&C and LM indicators. However, we lack context information from a real-world enterprise system. We believe that we can further improve the approach by including information related to topology and segregation architecture, users, and different host-types. Access to such information would allow us to white-list the known normal behavior, extract features related to the underlying network, and evaluate the performance of the proposed approach under dynamic network workload.

Lateral movement occurs after a successful initial compromise of the system; a well-equipped attacker may be able to act quickly on a target, even before sufficient data have been collected for our system to detect the attack. We can easily extend the approach and push the anomaly detection to early in the progression of an APT. Specifically, we can use the features to detect targeted malicious e-mails [31] and inbound network scan traffic [32] to correlate the indicators of initial compromise with those of lateral movement.

Like any other network-based anomaly detection system, the proposed approach would fail to detect an attack that results in no behavioral changes. For instance, if an attacker can perfectly replicate typical enterprise communication behavior during its C&C and regular network services during its LM, the approach will not be able to detect it. However, we believe that it is infeasible for an attacker to hide all the malicious traces and go undetected.

Finally, we address the problem of the lack of validation data by using the proposed attack trace generation method. This approach allows us to model a number of attacker behaviors and test our approach. The proposed anomaly detection shows good performance since (i) the feature extraction is based on simple matrix operations and (ii) the sizes of the feature vector are bounded by the number of hosts in the network. However, we want to test the time it takes to detect an attack from the point when the initial compromise happened. We leave such evaluation to future work.

VII. Related Work

Researchers have primarily focused on detecting individual stages of an APT. For example, Amin et al. [31] used a random forest classifier to identify coordinated and persistent campaigns of malicious emails that are used by APT actors to facilitate computer network exploitation. Smutz et al. [33] used features related to document metadata and structure to detect malicious PDF documents. Opera et al. [17] proposed a C&C detection system that identifies compromised hosts and suspicious external domains by using a belief propagation algorithm. Hagberg et al. [25] studied authentication graphs in a large network to assess the risk with respect to credential-hopping attacks. Johnson et al. [34] presented an approach to measure the potential vulnerability of a network to LM attacks.

Several recent papers have also explored the idea of using diverse information to detect botnet and worm-based attacks. To detect bot-infected hosts, BotHunter [32] uses correlation of multiple network-level anomaly and signature-based IDSes, whereas BotMiner [35] performs clustering and correlation of indicators observed at different hosts. Sekar et al. [18] proposed to temporally correlate host-based anomaly detection results with network-related events to establish causality among the events that happen during the propagation of computer worms. Unlike the epidemic-spreading attacks, APT campaigns are slow and silent, particularly in the phases after initial compromise. APT actors maintain a low profile while moving laterally within a victim system. Therefore, we need to build defense methods specifically tailored towards APTs. Our approach correlates the features related to C&C and lateral movement and detects compromised hosts by using unsupervised anomaly detection methods.

To respond to an attack, a human security analyst or an automatic response selection module [36], [37] can use the results of the proposed system. Examples of possible responses include blocking or throttling a suspected remote address, isolating a subset of hosts that are part of malicious chains from rest of the network, and learning the target of an attack by actively disturbing the lateral movement chains and evaluating the changes in attacker's path.

Finally, the anomaly detection techniques that inspired our approach are presented in [20], [22], [24], [27].

VIII. Conclusion

In this paper, we present an unsupervised multi-detector approach to detecting lateral movement-based attacks in enterprise systems. With it, we extract useful features using NetFlow, C&C, and specialized lateral movement monitors. We then propose a graph-based model to combine the diverse features to evaluate the current security state of the system. We show that by using the proposed features with an ensemble of PCA, k-means clustering, and extreme value analysis enables the identification of compromised hosts in an unsupervised manner. The accuracy of detection is about 88.7% and always stays above 82.6%, even in extreme deviations of the attacker from expected behavior. In each case, the number of false positives is very small. We evaluate how sensitive the approach is in detecting attacks as attacker capabilities change. The insights that we obtain can be used by researchers to test new defense mechanisms before deploying them on real systems.

Of the multiple avenues of future research opened by this work, we list the following. We plan to extend our feature extraction and attack detection by incorporating more context from the target system. We also want to make our system work in incremental batches of smaller time windows to support broader response actions.

Acknowledgments

This work is partly supported by AFOSR/AFRL FA8750-11-2-0084. We thank Zbigniew Kalbarczyk, Jenny Applequist, Arjun Athreya, and the anonymous reviewers for their feedback on the paper.

References

[1] N. E. Weiss and R. S. Miller, "The Target and other financial data breaches: Frequently asked questions," 2015. [Online]. Available: https://fas.org/sgp/crs/misc/R43496.pdf.

[2] TrendMicro, "APT myths and challenges," 2012. [Online]. Available: http://blog.trendmicro.com/trendlabs-security-intelligence/infographic-apt-myths-and-challenges/.

[3] TrendMicro, "Lateral movement: How do threat actors move deeper into your network?" TrendMicro, Tech. Rep., 2013.

[4] E. M. Hutchins, M. J. Cloppert, and R. M. Amin, "Intelligence-driven computer network defense informed by analysis of adversary campaigns and intrusion kill chains," *Leading Issues in Information Warfare and Security Research*, vol. 1, pp. 80–106, 2011.

[5] MITRE, "ATT&CK: Adversarial Tactics, Techniques & Common Knowledge," 2017. [Online]. Available: https://attack.mitre.org/wiki/Lateral_Movement.

[6] Mandiant, "APT1: Exposing one of China's cyber espionage units," 2013. [Online]. Available: https://www.fireeye.com/content/apt1-report.pdf.

[7] B. Bencsáth, G. Pék, L. Buttyán, and M. Felegyhazi, "The cousins of Stuxnet: Duqu, Flame, and Gauss," *Future Internet*, vol. 4, no. 4, pp. 971–1003, 2012.

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

[9] Verizon, "2016 data breach investigation report," 2016. [Online]. Available: http://www.verizonenterprise.com/verizon-insights-lab/dbir/2016/.

[10] A. D. Kent, "Cybersecurity data sources for dynamic network research," in *Dynamic Networks in Cybersecurity*, vol. 1. World Scientific, 2015, pp. 37–65.

[11] P. Van Mieghem, "The N-intertwined SIS epidemic network model," vol. 93, no. 2, 2011, pp. 147–69.

[12] R. Brewer, "Advanced persistent threats: Minimizing the damage," *Network Security*, vol. 2014, no. 4, pp. 5–9, 2014.

[13] Microsoft, "Microsoft advanced threat analytics," 2016. [Online]. Available: https://www.microsoft.com/en-us/cloud-platform/advanced-threat-analytics.

[14] V. A. Community, "VAST challenge, 2013," 2013. [Online]. Available: http://vacommunity.org/VAST+Challenge+2013.

[15] "Impact Cybertrust: GT Malware Passive DNS dataset," 2017. [Online]. Available: https://www.impactcybertrust.org/.

[16] D. Dash, B. Kveton, J. M. Agosta, E. Schooler, J. Chandrashekar, A. Bachrach, and A. Newman, "When gossip is good: Distributed probabilistic inference for detection of slow network intrusions," in *Proceedings of AAAI*, vol. 21, no. 2, 2006, pp. 1115–22.

[17] 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," in *Proceedings of the 2015 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*. IEEE, 2015, pp. 45–56.

[18] V. Sekar, Y. Xie, M. K. Reiter, and H. Zhang, "Is host-based anomaly detection + temporal correlation = worm causality?" CMU-CS-07-112, Carnegie Mellon University, Tech. Rep., 2007.

[19] A. Fawaz, A. Bohara, C. Cheh, and W. H. Sanders, "Lateral movement detection using distributed data fusion," in *Proceedings of the 2016 IEEE 35th Symposium on Reliable Distributed Systems*, 2016, pp. 21–30.

[20] A. Lakhina, M. Crovella, and C. Diot, "Diagnosing network-wide traffic anomalies," in *ACM SIGCOMM Computer Communication Review*, vol. 34, no. 4, 2004, pp. 219–30.

[21] S. Ramaswamy, R. Rastogi, and K. Shim, "Efficient algorithms for mining outliers from large data sets," in *ACM SIGMOD Record*, vol. 29, no. 2, 2000, pp. 427–38.

[22] V. Badrinath Krishna, G. A. Weaver, and W. H. Sanders, "PCA-based method for detecting integrity attacks on advanced metering infrastructure," in *Proceedings of the 12th International Conference on Quantitative Evaluation of Systems*, 2015, pp. 70–85.

[23] P. Rousseeuw, "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis," *Journal of Computational and Applied Mathematics*, vol. 20, no. 1, pp. 53–65, 1987.

[24] P. J. Rousseeuw and C. Croux, "Alternatives to the median absolute deviation," *Journal of the American Statistical Association*, vol. 88, no. 424, pp. 1273–83, 1993.

[25] A. Hagberg, N. Lemons, A. Kent, and J. Neil, "Connected components and credential hopping in authentication graphs," in *Proceedings of the International Conference on Signal-Image Technology and Internet-Based Systems*, 2014, pp. 416–23.

[26] J. Lambert, "Defenders think in lists. Attackers think in graphs." 2015. [Online]. Available: https://blogs.technet.microsoft.com/johnla.

[27] C. C. Aggarwal, "Outlier ensembles: Position paper," *ACM SIGKDD Explorations Newsletter*, vol. 14, no. 2, pp. 49–58, 2013.

[28] P. V. Mieghem, J. Omic, and R. Kooij, "Virus spread in networks," vol. 17, no. 1, 2009, pp. 1–14.

[29] D. T. Gillespie, "A general method for numerically simulating the stochastic time evolution of coupled chemical reactions," *Journal of Computational Physics*, vol. 22, no. 4, pp. 403–34, 1976.

[30] F. Pedregosa et al., "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–30, 2011.

[31] R. Amin, J. Ryan, and J. van Dorp, "Detecting targeted malicious email," *IEEE Security & Privacy*, vol. 10, no. 3, pp. 64–71, 2012.

[32] G. Gu, P. A. Porras, V. Yegneswaran, M. W. Fong, and W. Lee, "Bothunter: Detecting malware infection through IDS-driven dialog correlation," in *USENIX Security Symposium*, vol. 7, 2007, pp. 1–16.

[33] C. Smutz and A. Stavrou, "Malicious PDF detection using metadata and structural features," in *Proceedings of the 28th Annual Computer Security Applications Conference*. ACM, 2012, pp. 239–48.

[34] J. R. Johnson and E. A. Hogan, "A graph analytic metric for mitigating advanced persistent threat," in *Proceedings of the 2013 IEEE International Conference on Intelligence and Security Informatics*, 2013, pp. 129–33.

[35] G. Gu, R. Perdisci, and J. Zhang et al., "Botminer: Clustering analysis of network traffic for protocol-and structure-independent botnet detection," in *USENIX Security Symposium*, vol. 5, no. 2, 2008, pp. 139–54.

[36] M. Tylutki and K. Levitt, "A network-based response framework and implementation," in *Proceedings of the IFIP International Working Conference on Active Networks*, 2005, pp. 65–82.

[37] M. A. Noureddine, A. Fawaz, W. H. Sanders, and T. Başar, "A game-theoretic approach to respond to attacker lateral movement," in *Proceedings of the 7th International Conference on Decision and Game Theory for Security*, 2016, pp. 294–313.