Computer Networks 57 (2013) 3826–3839

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

A systematic approach for detecting and clustering distributed cyber scanning

Elias Bou-Harb *, Mourad Debbabi, Chadi Assi

Concordia Institute for Information Systems Engineering, Concordia University, Montreal, Quebec, Canada

ARTICLE INFO

Article history:
Received 8 February 2013
Received in revised form 13 May 2013
Accepted 12 September 2013
Available online 29 September 2013

Keywords:
Cyber scanning detection
Detrended Fluctuation Analysis
Unsupervised data clustering

ABSTRACT

We present in this paper an approach that is composed of two techniques that respectively tackle the issues of detecting corporate cyber scanning and clustering distributed reconnaissance activity. The first employed technique is based on a non-attribution anomaly detection approach that focuses on *what* is being scanned rather than *who* is performing the scanning. The second technique adopts a statistical time series approach that is rendered by observing the correlation status of a traffic signal to perform the identification and clustering. To empirically validate both techniques, we utilize and examine two real network traffic datasets and implement two experimental environments. The first dataset comprises of unsolicited one-way telescope/darknet traffic while the second dataset has been captured in our lab through a customized setup. The results show, on one hand, that for a class C network with 250 active hosts and 5 monitored servers, the training period of the proposed detection technique required a stabilization time of less than 1 s and a state memory of 80 bytes. Moreover, in comparison with Snort's sfPortscan technique, it was able to detect 4215 unique scans and yielded zero false negative. On the other hand, the proposed clustering technique is able to correctly identify and cluster the scanning machines with high accuracy even in the presence of legitimate traffic. We further validate this clustering technique by formulating the presented scenario as a machine learning problem. Specifically, we compare our proposed technique with an unsupervised data clustering technique that adopts the *k*-means and the expectation maximization approach. The results authenticate our clustering technique rendering it feasible for adoption.

1. Introduction

The ever increasing population and embracing of cyberspace has been a great asset both socially and economically. However, recent events demonstrated that cyberspace could be subjected to amplified, debilitating and disrupting attacks that might lead to severe security issues with drastic consequences. In general, cyberspace could facilitate distributed denial of service attacks [1], advanced persistent threats [2], zero day exploits [3] and cyber terrorism/warfare [4,5]. Despite efforts to protect the cyberspace, the latest report from Ottawa's Auditor General highlighted that only limited progress has been made in improving the cyber security of crucial networks [6]. Cyber scanning, the task of probing enterprise networks or Internet wide services, searching for vulnerabilities or ways to infiltrate IT assets, has been a growing cyber security concern. The latter is due to the fact that cyber scanning is commonly the primary stage of an intrusion attempt that enables an attacker to remotely locate, target, and subsequently exploit vulnerable systems. It is basically a core technique and the main enabler of the above mentioned cyber attacks. Fig. 1 depicts a general anatomy of a cyber attack where cyber scanning plays a major role. Indeed, the capability to detect, identify and attribute such scanning activity and its components is a very important task to achieve as this would aid in preventing or mitigating the actual cyber attack from occurring.

* Corresponding author. Tel.: +1 5146495049.
E-mail address: e_bouh@encs.concordia.ca (E. Bou-Harb).

1389-1286/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.comnet.2013.09.008

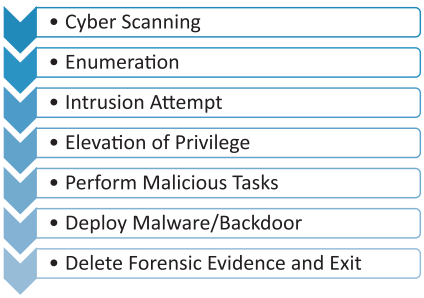


Fig. 1. A general anatomy of a cyber attack.

Motivated by such requirement, this paper contributes by:

- Employing a non-attribution (i.e., independent from the scanning source) anomaly detection approach that allows the detection of sophisticated reconnaissance activity with zero false negative and limited manageable false positive rates in addition to requiring minimalistic system state storage with a fast stabilization period.
- Proposing and adopting a new distributed scanning clustering approach based on a statistical time series analysis and observation method. The approach is able to identify and cluster the scanning machines with high accuracy even in the presence of legitimate traffic.
- Utilizing the Simple Network Management Protocol (SNMP) to manage the anomaly detection approach's training period and by applying the Detrended Fluctuation Analysis (DFA) technique to the problem of clustering distributed cyber scanning. These approaches have never been investigated before in such context.
- Validating and authenticating the DFA clustering approach by comparing it with the well established machine learning data clustering techniques, namely the k-means and the expectation maximization.
- Experimenting, to empirically validate both techniques, with two *real* network traffic data sets.

The remainder of this paper is organized as follows. Section 2 discusses current detection and clustering techniques and pinpoints the drawbacks of attribution-based approaches. Section 3 presents the non-attribution anomaly detection approach and provides a discussion related to the training and detection periods. Section 4 presents the clustering statistical-based approach by discussing the Detrended Fluctuation Analysis (DFA). Section 5 highlights the machine learning data clustering approach that is used to validate the DFA clustering technique. The evaluation environments coupled with the results are described in Section 6. Finally, Section 7 summarizes the paper and highlights the future work.

2. Related work

In this section, we discuss cyber scanning current detection and clustering techniques and subsequently pinpoint the drawbacks of attribution-based approaches.

Zhang et al. [7] proposed a scan detection method based on a distributed cooperative model. Their technique is composed of feature-based detection, scenario-based detection and statistic-based detection. Their proposed architecture is decomposed into five layers (sensors, event generators, event detection agents, a fusion center and a control center) that collaborate to achieve the intended task. The technique's statistic-based detection employs predefined thresholds that allows the detection of both scan and denial of service attacks. A positive aspect of this work is that the proposed technique is well suited to distributed large-scale environments. However, the presented work was based on an illustrated described scenario and the authors did not discuss its applicability on real data samples. In [8], Bhuyan et al. presented the adaptive outlier based approach for coordinated scan detection (AOCD). First, the authors used the principal component analysis feature reduction technique to identify the relevant feature set. Second, they employed a variant of the fuzzy c-means clustering algorithm to cluster information. The authors tested their algorithm using different real-life datasets and compared the results against other available literature techniques. Their approach assumes that the target of the scanning is a set of contiguous addresses, which is not always the case. In another work, Baldoni et al. [9] proposed a collaborative architecture where each target network deploys local sensors that send alarms to a collaborative layer. This, in turn, correlates this data with the aim of (1) identifying coordinated cyber scanning activity while (2) reducing false positive alarms and (3) correctly separating groups of attackers that act concurrently on overlapping targets. The soundness of the proposed approach was tested on real network traces. Their proposed system is designed to leverage information coming from various network domains to detect distributed scanning. Hence, the collaborative layer appears to be ineffective when the adversary is acting only against one network domain. In a more general work, Dainotti et al. [10] presented the measurement and analysis of a 12-day world-wide cyber scanning campaign targeting VoIP (SIP) servers. The authors used darknet/telescope data collected at the UCSD network telescope to exclusively focus on the analysis and reporting of that SIP scanning incident.

Most of the aforementioned detection and clustering techniques and other literature work [11–13] could be noted as being attribution-based; they detect and cluster distributed scanning based on the last perceived scanning source. Hence, they might encounter one of the following issues:

- Determining attribution is not always possible, which might decrease the effectiveness of such techniques.
- The scans may either be so slow or so broadly distributed that they exhaust the finite computational state of scanning detection systems or fail to exceed some predefined alert threshold.
- A significant amount of system state (i.e., memory, network topology information, storage) needs to be maintained by the monitoring system in order to perform effectively (reducing the detection time window to accommodate network traffic fluctuations might cause excessive false negatives and false positives).

In the next two sections, we present and elaborate on our approach that is composed of two techniques. Specifically, Section 3 presents the non-attribution anomaly detection technique while Section 4 describes the statistical time series clustering technique. In a nutshell, the first technique consists of two periods: (1) A training period and (2) an anomaly detection period. The outcome of this technique is detected scans with minimal false positive rate. The second technique takes as input the detected scans from the first technique and aims to cluster and identify the scanning machines even in the presence of legitimate traffic.

3. The non-attribution anomaly detection technique

In this section, we present the non-attribution anomaly detection technique and provide a discussion related to its training and detection periods.

3.1. Idea rationale

The rationale behind the idea states that the available services that are provided by the hosts within an enterprise network represent the facade of that network; the offered services induce the possible leakage of information that could be retrieved by an attacker during a successful scan. Hence, the idea takes full advantage and solely of the network topology by constructing what we refer to as 'local host facade' (LHF) and 'enterprise network facade' (ENF). The former is the accessible services per host while the latter is the combination of all accessible services of all active hosts within the network.

3.2. ENF management

In the training phase of our technique, we leverage the SNMP [14] to manage the ENF. SNMP is an Internet-standard protocol for managing devices on IP networks. It consists of components for network management, including an application layer protocol, a database schema, and a set of data objects. The protocol's information exchange is performed between a management station and an agent (embedded in the managed entity) in the form of SNMP messages. For an in-depth review of SNMP, including its inner workings, we refer the readers to [15].

The idea is to exploit specific de facto SNMP procedures to manage the ENF. The latter task is divided into constructing the ENF by retrieving the list of listening ports on each host and maintaining (adding/deleting certain IPs/ports) the list in case of any change in accordance with a certain predefined update threshold. In the following, we briefly discuss the employed SNMP procedures and consequently elaborate on their roles in managing the ENF.

The `SNMP Receive-GetRequest` procedure [14] is issued by an SNMP management station in order to read or retrieve an object value from a managed entity. The managed SNMP entity responds to a `GetRequest` protocol data unit (PDU) with a `GetResponse` PDU. The `GetRequest` operation is atomic; either all the values are retrieved or none is. If the responding entity is able to provide values for all the variables listed in the incoming `VariableBindings` list, then the `GetResponse` PDU includes the `VariableBindings` field coupled with a value supplied for each variable. If at least one of the variable values cannot be supplied, then no values are returned [14].

In this current work, the procedure, namely, `SNMP Receive-GetRequest`, is used to construct the ENF by leveraging the following two request methods:

$$GetRequest(ipRouteDest, tcpNoPorts)$$

$$GetRequest(ipRouteDest, udpNoPorts)$$

On the other hand, the task of maintaining the ENF could be divided into two sub-tasks. The first is when we need to update the list of active IPs/hosts and the second is when we need to modify the list of listening TCP and UDP ports for a specific host. To accomplish this, another SNMP procedure is presented, namely `SNMP Receive-SetRequest` [14].

The procedure `SNMP Receive-SetRequest` is issued by an SNMP entity on behalf of a management station. It has the same PDU exchange pattern and the same format as the `GetRequest` PDU. However, the `SetRequest` is used to write an object value rather than reading or retrieving one.

In this work, we exploit `SNMP Receive-SetRequest` to update the ENF; based on a predefined update threshold, and whenever there is an update in the hosts (changing status from active to non-active or vice versa) or their corresponding listening ports, we issue a `SetRequest` PDU to reflect the changes. For instance, if we notice that an active (i.e., connected) host with an IP address of 10.0.0.1 is no longer active (i.e., disconnected), the following SNMP request [14] is issued to remove that host from the ENF:

$$SetRequest(ipRouteDest.10.0.0.1 = invalid)$$

The above two procedures provide methods to construct and maintain what we have defined as the enterprise network facade. Recall that this characterizes the *training period* of our proposed non-attribution detection technique. Since the management of the ENF is dependent solely on the enterprise network services and is totally decoupled from any external traffic, our approach is advantageous in two core areas. First, it requires almost negligible time to stabilize which renders its implementation very operationally feasible. Second, it relies on the observation and manipulation of a protocol (SNMP) found in every network, where its actual overhead on network bandwidth and hardware is minimal even in large network environments [16,17].

3.3. Using ENF for scan detection

Once the training period has completed and an ENF is constructed, the anomaly detection phase commences. Scan detection is performed by monitoring external incoming TCP or UDP connection attempts. The attempts could be destined to the following targets: (1) an unallocated IP address, (2) an allocated IP address with a port combination not found in the ENF, (3) an allocated IP address with a port combination found in the ENF and (4) an allocated or

an unallocated IP address outside of the monitored zone. In our approach, the detection occurs when we notice target 2 occurring, namely, an attempt to an allocated IP address with a port combination not found in the ENF. If the latter case occurs, we flag the connection attempt and log its corresponding details such as source and destination IP and port, protocol, and the timestamp. Target 1 is referred to as dark IPs [18] and their analysis is outside the scope of this work. Target 3 is as well excluded from the analysis. The exclusion of this target, at a first glance, seems to carry a limitation of our work in that scans to valid services (i.e., entries in the ENF) will not be detected. For instance, a DNS scan towards a naming (DNS) server is considered a valid activity and thus would not be considered a scan. However, this type of scan would indeed be detected using our approach as the same scan would almost certainly also occur against other hosts in the network not offering DNS. The scanning activity would not be detected if it were directed, although unlikely, solely at the naming server. However, we would consider the latter activity to be an actual attack (i.e., such as a denial of service attack) rather than a scan. Finally, target 4 depends on our monitored zone and intuitively we do not detect scans outside the monitored areas.

3.4. Discussion

In this part, we provide a discussion that is related to the technique's training and detection periods.

The training period is the period during which we first construct the ENF. Hence, as is the case with any technique that requires a training period, it is possible that malicious hosts activity may become part of the reference baseline. For example, if a trojan horse program [19] has been maliciously installed and has been running on one of the corporates's network servers, then the program would typically open up listening ports that are otherwise not supposed to be listening. To avoid this, we can match or verify the LHF with the enterprise network's security policy. Any inconsistencies are removed from the LHF to securely build the ENF. Moreover, our technique's training period is efficient as the ENF only needs to record and maintain the state of the network services. To further improve this, we can manipulate SNMP to gather and record information only about specific hosts within the enterprise network. For example, we can build a *custom ENF* that includes only some of the network servers and to exclude other servers and workstations (we refer to those selected servers as belonging to within the boundaries of the monitored zone).

On the other hand, our proposed anomaly scanning detection approach does not rely on the identification of the scanning source. Therefore, it can detect certain classes of sophisticated scanning techniques (such as distributed and slow scanning) that make determining the root cause of the scanning activity impractical. Furthermore, the detection technique requires only a single packet to flag an attempt as a scan event and requires minimalistic system state storage especially if used with a *shape custom ENF*. Additionally, our approach is transport protocol-independent and hence can detect both TCP and UPD scans.

Recall, that the outcome of this first technique is detected scans with minimal false positive rate. In the next section, we present a second technique that takes as input the detected scans from the first technique and aims to cluster and identify the scanning machines even in the presence of legitimate traffic.

4. The statistical time series clustering technique

In this section, we present the rationale and aim behind our proposed statistical time series clustering approach and subsequently describe the Detrended Fluctuation Analysis (DFA) method.

4.1. Idea rationale

Our approach is based on the observation that scanning machines that use the same technique to perform the scan will likely demonstrate temporal correlation and similarity. The idea is to capture such correlation in the traffic signal to perform the clustering. The approach aims at identifying and clustering the scanning machines even in the presence of legitimate traffic.

4.2. Detrended Fluctuation Analysis

To accomplish the above aim, we adopt the time series Detrended Fluctuation Analysis (DFA) method. DFA was first proposed in [20] and has since been used in many research areas to study signals correlation. Very limited work in the areas of cyber security and malicious traffic detection has utilized DFA [21,22], and to the best of our knowledge, no work has applied the DFA technique to the problem of clustering distributed cyber scanning. The DFA method is discussed next.

The DFA method of characterizing a non-stationary time series is based on the root mean square analysis of a random walk. DFA is advantageous in comparison with other methods such as spectral analysis [23] and Hurst analysis [24] since it permits the detection of long range correlations embedded in a seemingly non-stationary time series. It avoids as well the spurious detection of apparent long-range correlations that are an artifact of non-stationarity. Another advantage of DFA is that it produces results that are independent of the effect of the trend [25].

Given a traffic time series, the following steps need to be applied to implement DFA:

- Integrate the time series; The time series of length N is integrated by applying

$$y(k) = \sum_{i=1}^{k} [B(i) - B_{ave}] \tag{1}$$

where $B(i)$ is the ith interval and B_{ave} is the average interval.

- Divide the time series into "boxes" of length n.
- In each box, perform a least-squares polynomial fit of order p. The y coordinate of the straight line segments is denoted by $y_n(k)$.
- In each box, detrend the integrated time series, $y(k)$, by subtracting the local trend, $y_n(k)$. The root-mean-square fluctuation of this integrated and detrended time series is calculated by

$$F(n) = \sqrt{\frac{1}{N}\sum_{k=1}^{N}[y(k) - y_n(k)]^2} \quad (2)$$

- Repeat this procedure for different box sizes (i.e., time scales) n

The output of the above procedure is a relationship $F(n)$, the average fluctuation as a function of box size, and the box size n. Typically, $F(n)$ will increase with box size n. A linear relationship on a log–log graph indicates the presence of scaling; statistical self-affinity expressed as $F(n) \sim n^{\alpha}$. Under such conditions, the fluctuations can be characterized by a scaling exponent α, which is the slope of the line relating $\log F(n)$ to $log(n)$. The scaling exponent α can take the following values, disclosing the *correlation status* of the traffic time series:

- $\alpha < 0.5$: anti-correlated.
- $\alpha \approx 0.5$: uncorrelated or white noise.
- $\alpha > 0.5$: correlated.
- $\alpha \approx 1$: $1/f$-noise or pink noise.
- $\alpha > 1$: non-stationary, random walk like, unbounded
- $\alpha \approx 1.5$: Brownian noise.

Note that in our work, the traffic time series represent traffic originating from scanning and benign machines. Section 6.2 will present, clarify and evaluate this statistical time series clustering approach.

Recall that the proposed approach aims to achieve two goals; the detection and clustering of distributed cyber scanning. The approach is composed of two techniques (of Section 3 and this section) that respectively tackle those two issues. Hence, a single technique will only achieve one of those goals while a combination will achieve both. Further, we do not project that their combination will cause a decrease in overall approach efficiency. The latter statement is based on the fact that the two techniques operate independently, sequentially and are decoupled from each other. Moreover, they leverage techniques (such as SNMP and DFA) that are known to be efficient and do no require any synchronization or concurrency control.

In the next section, we present a machine learning unsupervised data clustering technique that adopts the k-means and the expectation maximization algorithm. The aim is to attempt to validate and authenticate the DFA clustering technique, that was proposed in this section, by comparing it the well established data clustering approach.

5. Technique validation using unsupervised learning

In this section, we present a method that is based on a machine learning approach, namely, unsupervised learning using data clustering.

Data clustering [26] has been playing an important role in various fields such as data mining [27], statistical data analysis [28] and data compression and reduction [29]. It tackles the problem of identifying k clusters given n observations or data points. It aims at maximizing the inter-cluster similarity while minimizing the intra-cluster resemblance of the data points. It is worthy to note, that in our work, the clusters represent the scanning and the benign machines and the observations are packet features extracted from the machines' corresponding traffic.

When the observations are not pre-labeled into defined numerical or categorical classes, as in our case, two standard widely deployed algorithms for data clustering using unsupervised learning could be employed. These are the k-means [30] and the Expectation Maximization (EM) [31] algorithms. The k-means algorithm finds k clusters by choosing n data points at random as initial cluster centers. Each data point is then assigned to the cluster with the center that is closest to that point. Each cluster center is then replaced by the mean of all the data points that have been assigned to that cluster. This process is iterated until no data point is reassigned to a different cluster. On the other hand, the EM algorithm views the data clustering problem as a framework for data density estimation using a probability density function. An effective representation of the probability density function is the *mixture model*, which asserts that the data is a combination of k individual component densities corresponding to k clusters. The EM problem can be summarized as follows: given a set of data observations, identify a set of k populations in the data and provide a density distribution model of each of the populations.

The EM algorithm is an effective and popular technique for estimating the mixture model parameters [32]. Note that, the k-means algorithm operates by minimizing the sum of squared Euclidean distances between data records in a cluster and the clusters mean vector. This assignment criterion implicitly assumes that the clusters are represented by Gaussian distributions located at the k cluster means. Moreover, since the k-means algorithm utilizes the Euclidean metric, it does not generalize to the problem of clustering discrete or categorical data. Furthermore, the k-means algorithm employs a membership function which assigns each data record to exactly one cluster. This harsh criteria does not allow for uncertainty in the membership of a data record within a cluster. On the other hand, the mixture model framework adopted by the EM relaxes these assumptions. Due to the probabilistic nature of the mixture model, clusters can be effectively represented by a suitable component density functions (i.e., Poission, non-spherical Gaussians, etc.). Additionally, categorical or discrete data is similarly handled by associating discrete data distribution over these attributes (i.e., Multinational, Binomial, etc.). In this work, we adopt both the k-means and the EM algorithm to validate the DFA approach that was previously proposed. The EM algorithm is thoroughly discussed next.

The mixture model approximates the data distribution by fitting k component density functions f_h, $h = 1,\ldots,k$, to a database D having m records and d attributes. In our work, the attributes represent packet features extracted from the scanning and the benign machines, the records represent the data instances related to those features and the database D is the complete set of data instances. Let x be a record from D, then the mixture model probability density function evaluated at x is given by:

$$p(x) = \sum_{h=1}^{k} w_h \cdot f_h(x \mid \Phi_h) \quad (3)$$

The weights w_h represent the fraction of database records belonging to cluster h and sum to one;

$$\sum_{h=1}^{k} w_h = 1, \quad w_h \geqslant 0$$

The functions $f_h(x|\Phi_h)$ $h = 1,\ldots,k$ are the clusters or component density functions modeling the records of the h^{th} cluster, and Φ_h represents the specific parameters used to compute the value of f_h (i.e., for a Gaussian component density function, Φ_h is the mean and the covariance matrix).

The mixture model also allows for overlapping clusters; data records may belong to all k clusters with different probabilities of membership. The probability of membership (i.e., weight) of data record x in cluster h is:

$$w_h(x) = \frac{w_h \cdot f_h(x \mid \Phi_h)}{\sum_i w_i \cdot f_i(x \mid \Phi_i)} \tag{4}$$

By making the assumption that the attributes of the database are independent over records within a given cluster, the component density functions can be decomposed as a product of density functions over each attribute $j = 1,\ldots,d$:

$$f_h(x \mid \Phi_h) = \prod_{j=1}^{d} f_{hj}(x_j \mid \Phi_h) \tag{5}$$

Although the framework we present in this work is general enough to address mixture model estimation having both continuous and discrete attributes, we focus on its application to continuous-valued data, modeled by multivariate Gaussians. In our current work, this choice of Gaussians for continuous-valued data is motivated by a result from density estimation theory stating that any distribution can be effectively approximated by a mixture of Gaussians [33].

Each population (i.e., cluster) is modeled by a d-dimensional Gaussian probability distribution. The multivariate Gaussian distribution for cluster $h = 1,\ldots,k$ is parameterized by the d-dimensional mean vector μ_h and $d \times d$ covariance matrix Σ_h:

$$f_h(x \mid \mu_h, \Sigma_h) = \frac{1}{\sqrt{(2\Pi)^d \mid \Sigma_h \mid}} exp\{\wp\} \tag{6}$$

where

$$\wp = -\frac{1}{2}(x - \mu_h)^T (\Sigma_h)^{-1}(x - \mu_h)$$

x and μ_h are column vectors, the superscript ^{T} indicates the transpose to a row vector, $|\Sigma_h|$ is the determinant of Σ_h and $(\Sigma_h)^{-1}$ is its matrix inverse. The Gaussian mixture model parameters consist of the means and covariance matrices for each cluster $h = 1,\ldots,k$ along with the weights w_h associated with each cluster. Let $\Phi = \{(w_h, \mu_h, \Sigma_h), h = 1,\ldots,k\}$ be the collection of mixture model parameters. The quality of a given set of parameters Φ is a measure of how well the corresponding mixture model fits the data. This is quantified by the log-likelihood of the data given the mixture model:

$$L(\Phi) = \sum_{x \in D} log\left\{\sum_{h=1}^{k} w_h \cdot f_h(x \mid \mu_h, \Sigma_h)\right\} \tag{7}$$

Table 1
Mathematical notations and definitions.

Mathematical notation	Definition
$p(x)$	Probability density function evaluated at a data record x
w_h	The probability of a data record belonging to a cluster h
$f_h(x\|\Phi_h)$	Component density functions
Φ_h	The parameters of the component density function
$\prod_{j=1}^{d} f_{h,j}(x_j \mid \Phi_h)$	The product of density functions over the attributes
μ_h	Mean vector
Σ_h	Covariance matrix
$L(\Phi)$	The mixture model

Table 2
Protocols distribution.

TCP	UDP	ICMP	Others
86.3%	11.7%	1.8%	0.2%

Table 3
IP class distribution.

Class	Usage (%)	
	Source	Destination
A	63.3	0.3
B	21.2	9.5
C	15.5	90.2

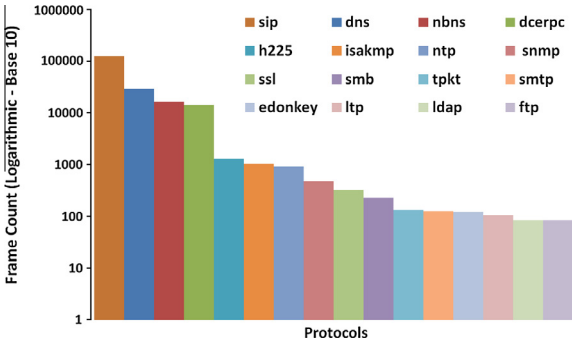


Fig. 2. Application layer protocols.

The EM algorithm begins with an initial estimation of Φ and iteratively updates it. The sequence of Φ-values is such that $L(\Phi)$ is non-decreasing at each iteration. We next outline the operation of the EM algorithm.

Given a database D with m records with d continuous-valued attributes, a stopping tolerance $\epsilon > 0$ and a mixture model parameters Φ^j at iteration j, compute Φ^{j+1} at iteration $j + 1$ as follows:

1. For each database record $x \in D$, compute the membership probability of x in each cluster $h = 1,\ldots,k$:

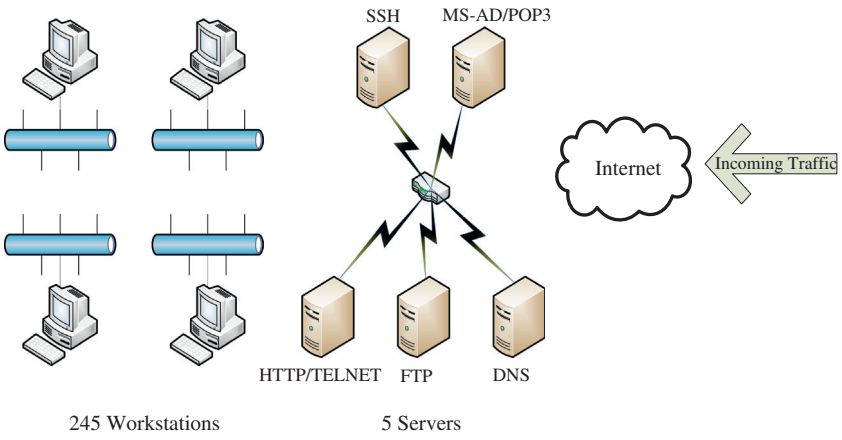


Fig. 3. Enterprise network.

$$w_h^j(x) = \frac{w_h^j \cdot f_h(x \mid \mu_h^j, \Sigma_h^j)}{\sum_i w_i^j \cdot f_i(x \mid \mu_i^j, \Sigma_i^j)}$$

2. Update the mixture model parameters: $w_h^{j+1} = \sum_{x \in D} w_h^j(x)$,

$$\mu_h^{j+1} = \frac{\sum_{x \in D} w_h^j(x) \cdot x}{\sum_{x \in D} w_h^j(x)} \Sigma_h^{j+1} = \frac{\sum_{x \in D} w_h^j(x) \left(x - \mu_h^{j+1}\right)\left(x - \mu_h^{j+1}\right)^T}{\sum_{x \in D} w_h^j(x)}$$

Stopping criteria: if $|L(\Phi^j) - L(\Phi^{j+1})| \leqslant \epsilon$, stop. Else set $j \leftarrow j + 1$ and go to 3. $L(\Phi)$ is given in 7.

Note that, steps 1 and 2 are respectively referred to as the estimation and maximization steps. Moreover, a full data scan is required at each iteration of the EM algorithm to compute the membership probability for each data record in each cluster. The number of iterations required before the stopping criteria is satisfied is dependent upon the initial parameter values and the actual data distribution. In general the number of iterations is arbitrary but the procedure is guaranteed to converge. Note that, all the mathematical notations that were presented in this section coupled with their corresponding definitions are summarized in Table 1.

Recall that the aim of this section is to attempt to validate and authenticate the DFA clustering technique that was proposed in Section 4 by comparing it with the presented data clustering approach.

6. Evaluation: datasets, methodologies and results

For the purpose of empirically validating our approach, which consists of the proposed two techniques, we utilized and examined two real network traffic datasets and implemented two experimental environments.

Table 4
ENF details.

Host	TCP ports	Description
Server 1	80, 23	HTTP/TELNET
Server 2	21	FTP
Server 3	53	DNS
Server 4	23	SSH
Server 5	445, 110	MS-active directory/POP3

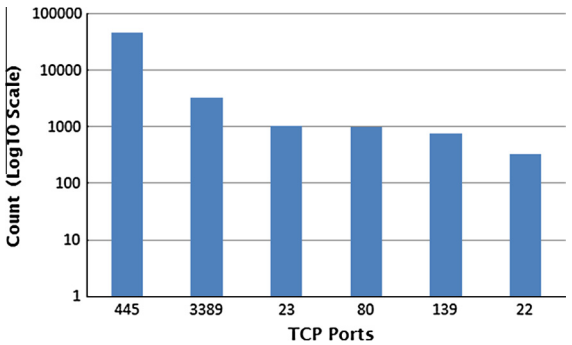


Fig. 4. Top 6 scanned TCP ports – one day sample.

6.1. Evaluating the non-attribution anomaly detection approach

We used a dataset that consists of unsolicited one-way telescope/darknet traffic [34] retrieved in real-time from a trusted third party framework. The traffic originates from the Internet and is destined to numerous /24 and /16 network sensors. The data were collected during the period of November 1, 2012 and December 1, 2012. Tables 2,3, and Fig. 2 show some network, transport and application level statistical information about the dataset.

We selected part of the traffic that is destined to a /24 network collected at the sensor. We assumed that an operational/corporate network, having the same IP configuration as the incoming traffic, exists behind the sensors.

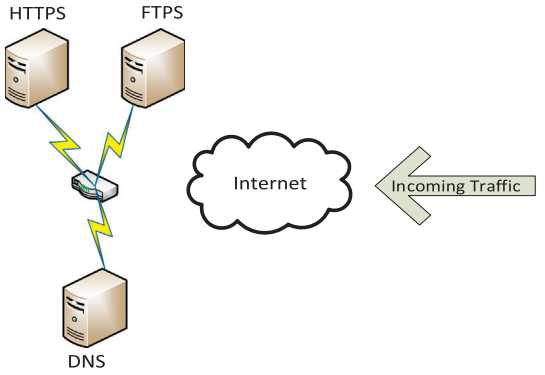


Fig. 5. The DMZ network.

Table 5
ENF details.

Host	TCP ports	Description
Server 1	443	HTTPS
Server 2	989, 990	FTPS
Server 3	53	DNS

Consequently, we built the network that is illustrated in Fig. 3. The network has a Classless Inter-Domain Routing (CIDR) address of 192.168.1.0/24 and is composed of 250 active hosts divided into 245 workstations and 5 servers. We as well took advantage of the SNMP procedures of Section 3.2 to develop an SNMP tool. The tool is based on the software components provided by eMarksoft SNMP [35].

We first used the developed tool to execute the training period of our proposed approach. The ENF was populated with 5 LHFs (the other workstations are not offering any services) as illustrated in Table 4. The task was completed in 0.32 s and required 80 bytes of state memory.

Table 6
Summary – detection capability.

Detected scans-proposed approach	Detected scans-Snort's sfPortscan
3421	3112

To validate the detection capabilities of our approach, we experimented with a one day sample traffic captured from our dataset. We also compared our approach with Snort's sfPortscan preprocessor using the same day sample. sfPortscan [36], a preprocessor plugin for the open source network intrusion and detection system Snort [37], provides the capability to detect TCP, UDP, and ICMP scanning. The sfPortscan preprocessor detects scans by counting RST packets from each perceived target during a predetermined timeout interval. Before declaring a scan, five events (i.e., RST packets) are required from a given target within a window. The sliding timeout window varies from 60 to 600 s by sensitivity level; at the highest level, an alert will be generated if the five events are observed within 600 s. We have chosen to compare our approach with Snort's sfPortscan preprocessor since Snort is one of the most broadly deployed intrusion detection/prevention technology worldwide and has become a de facto standard.

According to the results, using our approach with this specific data sample, we were able to detect 4215 unique scans (unique IP/port pairs). Moreover, Fig. 4 illustrates the top six scanned TCP ports. Scans towards those services could indicate that they are vulnerable to exploits.

To elaborate on the results, we subsequently present an analytical discussion on our technique's false negatives and false positives.

6.1.1. False negatives

We fed the same dataset as an input to Snort's sfPortscan. We relied on the output as a baseline for our comparison.

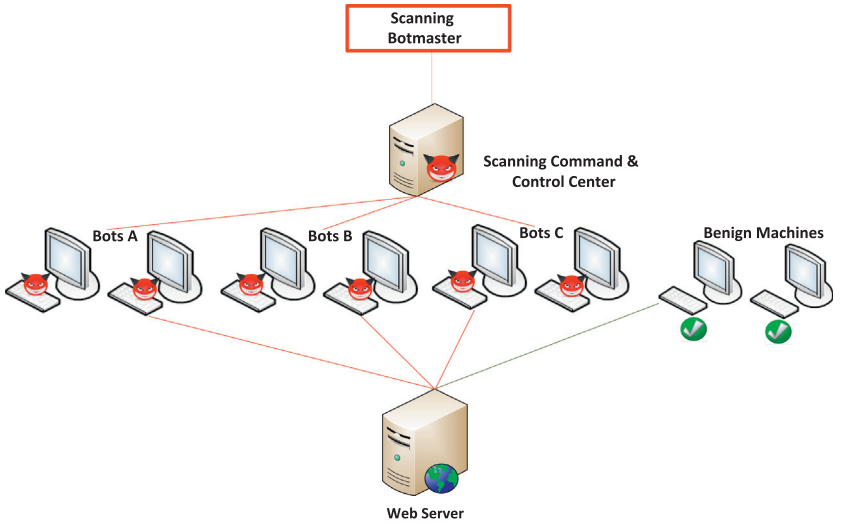


Fig. 6. Evaluating scenario.

Snort's sfPortscan technique detected 3690 unique scans. After a semi-automated analysis and comparison that was based on the logged scanning traffic flows (i.e., source and destination IP and port, protocol, and timestamp), we identified that all the 4215 scans that our approach detected include sfPortscan's 3690 scans. Therefore, relative to this technique and experimenting with this specific data set, we confirm that our approach yielded no false negative.

6.1.2. False positives

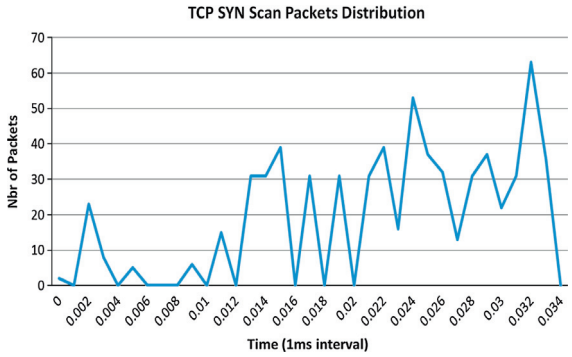
Our approach flags an attempt as a scan whenever a connection is made to a host or service not offered by the network. The following can exist as sources of false positive: (1) User error and network misconfiguration; the intent was not to perform a scan but rather to access a legitimate service that have failed. Since there exists no scientific way to judge the connection intention, we have to classify those attempts as scans. (2) Backscattered traffic [38] destined to the corporate network; such traffic commonly refers to unsolicited traffic that is the result of responses to denial of service attacks with spoofed source IP addresses. To avoid this false positive, we can investigate such traffic using the proposed method in [39], which uses flags in packet headers, such as TCP SYN + ACK, RST, RST + ACK, and ACK, to accomplish the filtering. (3) Attempts to newly available services that were not part of the training period; to reduce the occurrences of this, we can optimize the update threshold of an ENF to include the new services.

Moreover, our approach can detect certain types of scans that were not included at the time of the experiment, and by default, in Snort's sfPortscan definitions. These include scans from a single host to a single port on a single host, slow scans and a specific host scanning multiple ports on multiple hosts. In general, we claim that a certain limited, acceptable and a manageable number of false positives will occur. Although future manual packet inspection needs to be performed to get the exact number of false positives, we need as well to consider Snort's sfPortscan false negatives and the different types of scans that our approach was able to detect.

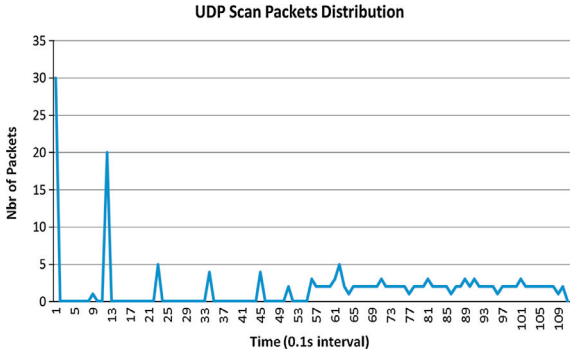
To further evaluate the proposed non-attribution anomaly detection approach that was presented in Section 3, we consider another network scenario. We selected part of the traffic that is destined to a /24 network collected at the sensor. We assumed that a De-Militarized Zone (DMZ) perimeter network, having the same IP configuration as the incoming traffic, exists behind the sensors. A DMZ network typically refers to a logical sub-network that contains and exposes an organization's external-facing (i.e., public) services to a larger untrusted network, such as the Internet. Consequently, we built the network that is illustrated in Fig. 5.

The DMZ network consists of three public servers. The corresponding ENF is summarized in Table 5. The procedure to build the ENF executed in 0.23 s and required 54 bytes of state memory.

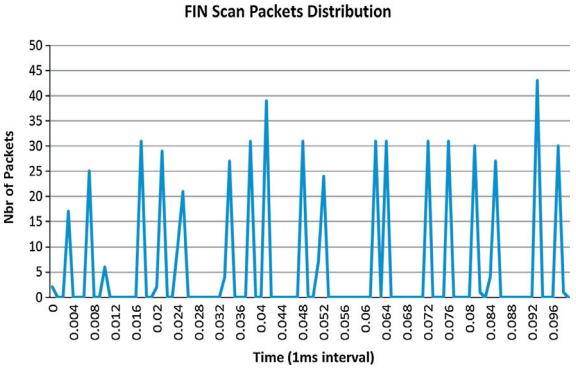
To validate the detection capability of our approach, we experimented with a one day sample traffic captured from our dataset. We also compared our approach with Snort's sfPortscan preprocessor using the same data sample. Table 6 summarizes the findings. After a semi-automated



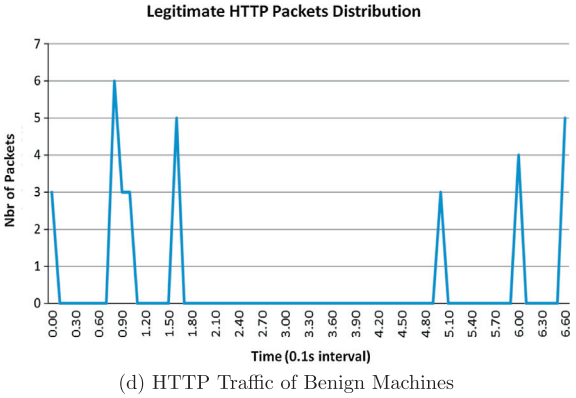
(a) TCP SYN Scanning Traffic of Bots A



(b) UDP Scanning Traffic of Bots B



(c) FIN Scanning Traffic of Bots C



(d) HTTP Traffic of Benign Machines

Fig. 7. Packets' distribution generated by the three bot groups and the legitimate machines.

analysis and comparison that was based on the logged scanning traffic flows (i.e., source and destination IP and port, protocol, and timestamp), we identified that all the 3421 scans that our approach detected include sfPortscan's 3112 scans. Therefore, relative to this technique and experimenting with this specific data set, we confirm, once again, that our approach yielded no false negative.

6.2. Evaluating the statistical time series clustering approach

We now presume that the output of the previous proposed technique generated real scans towards the corporate web server. Hence, to evaluate the proposed DFA approach, which aims at identifying and clustering the scanning machines, we created in a lab environment a customized setup as illustrated in Fig. 6. The setup consists of a scanning command and control server, six scanning machines, two benign/legitimate machines and the corporate webserver. The scenario discloses that a scanning botmaster operating the command and control center has compromised the machines into his botnet and aims to scan the webserver. At the same time, the webserver is still servicing the requests of the legitimate machines.

We have setup a TCPDUMP [40] sink on the webserver to collect the network traffic data originating from the bots and the benign machines. To emulate the effect of the scanning bots, we have utilized nmap [41], an open source utility for network scanning and discovery. The bots, as shown in the scenario of Fig. 6, are divided into three groups, namely, Bots A, B and C, where each group uses a certain scanning technique. Bot groups A, B and C uses the TCP SYN scan (nmap -sS), the UDP scan (nmap -sU) and FIN scan (nmap -sF) respectively. Although typically, one botnet campaign might utilize one scanning technique to perform its scan, we thought it would be more representative and challenging if we have a scenario with various scanning techniques. We can think of the three different scanning techniques as if there exist three different botnets or one botnet utilizing various techniques. Regardless of the scenario, recall that the aim is to correctly identify the compromised machines (i.e., the scanning machines) from the non-compromised in addition to clustering the bots that belong to the same botnet. In the case of the non-compromised machines, we issued HTTP requests using the `wget` command. Note that, the bots, the benign machines and the webserver are configured as virtual machines running Ubuntu Linux 11.04 where they are connected using a LAN isolated from any external/Internet network activity. After finalizing the aforementioned setup, we concurrently ran the above procedure and collected the dataset in pcap format.

Using the source IPs of the bots and the legitimate machines, we extracted their corresponding traffic from the dataset that we had previously collected. The packets' distribution of the scanning traffic generated by the three bot groups in addition to the benign HTTP traffic generated by the legitimate machines is illustrated in Fig. 7.

To implement DFA, we have utilized the MATLAB code found in [42]. The output of applying the DFA method on the previous traffic time series distributions is shown in Fig. 8 and the output of the scaling exponents α is summarized in Table 7.

The results concur our observation that scanning machines that use the same technique to perform the scan will likely demonstrate a *unique* temporal correlation and similarity. This is indeed demonstrated in Table 7, where TCP SYN scanning that was generated from Bots A, according to the DFA results, showed that their traffic signals possessed a distinguished fingerprint where the traffic is similar to a non-stationary signal ($\alpha > 1$). On the other hand, UDP scanning from Bots B revealed that their signal's traffic is correlated ($\alpha > 0.5$) while the signal's traffic from the FIN scanning of Bots C showed an anti-correlated signal ($\alpha < 0.5$). Further, the benign HTTP traffic that was generated from the legitimate machines was similar to noise ($\alpha \approx 1$).

Having the above significant traffic fingerprinting information, it is now straightforward to identify and cluster the machines. For that purpose, we went back to our collected dataset and extracted 8 traffic flows, where each flow is identified by a source and a destination IP and port, and a transport layer protocol (TCP, UDP, ICMP). For each traffic flow, we applied the DFA method and then cross-matched it with the correlation status information that we have in Table 7. By accomplishing this, we showed that by only using the traffic signals self similarity feature, our proposed technique's clustering mechanism is able to

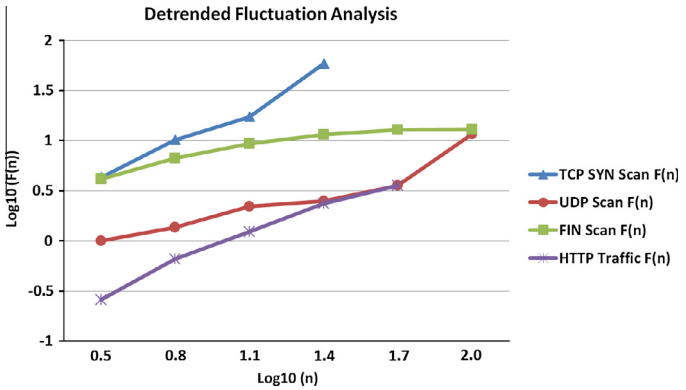


Fig. 8. The application of DFA on the scanning and benign traffic.

Table 7
Summary of the scaling exponents α.

Traffic type	Scaling exponent
TCP SYN Scanning	1.2
UDP Scanning	0.64
FIN Scanning	0.32
HTTP Traffic	0.95

Table 8
Summary of the scaling exponents α.

Traffic type	Scaling exponent
RPC Scanning	1.32
TCP Connect() Scanning	0.69
ACK Scanning	0.29

Table 9
Features description.

Features		
Data link features	1	Delta time with previous capture packet
	2	Packet length
	3	Frame length
	4	Capture length
	5	The flag 'frame' is marked
Network layer features	6	IP header length
	7	IP Flags.
	8	IP Flags: reversed bit
	9	IP Flags: do not fragment bit
	10	IP Flags: more fragments bit
	11	IP Fragment offset
	12	IP Time to live
	13	IP Protocol
Transport layer features	14	TCP segment length
	15	TCP sequence number
	16	TCP next sequence number
	17	TCP acknowledgement number
	18	TCP header length
	19	TCP flags
	20	TCP flags: congestion window reduced
	21	TCP flags: ECN-Echo
	22	TCP flags: urgent
	23	TCP flags: acknowledgement
	24	TCP flags: push
	25	TCP flags: reset
	26	TCP flags: syn
	27	TCP flags: fin
	28	TCP window size
	29	UDP length

correctly identify and cluster the scanning bots with high accuracy even in the presence of legitimate traffic.

Please note that we have also experimented with three different scanning techniques, namely, RPC, TCP Connect() and ACK scanning. The output of their corresponding DFA scaling exponents α is summarized in Table 8. Such results, once again, concur our observation that scanning machines that use the same technique to perform the scan will likely demonstrate a unique temporal correlation and similarity. Hence, the proposed approach seems also to correctly operate on these scanning techniques.

Table 10
Distribution of clustered instances.

Cluster 1	Cluster 2	Cluster 3	Cluster 4
19%	37%	29%	15%

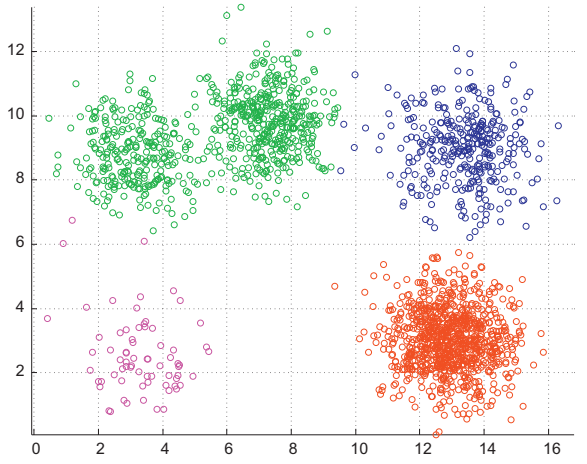
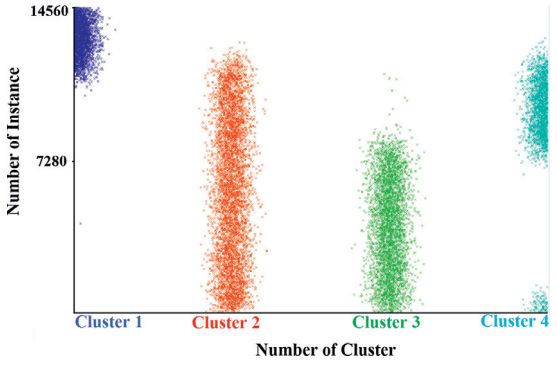


Fig. 9. *k*-Means output-cluster formation.

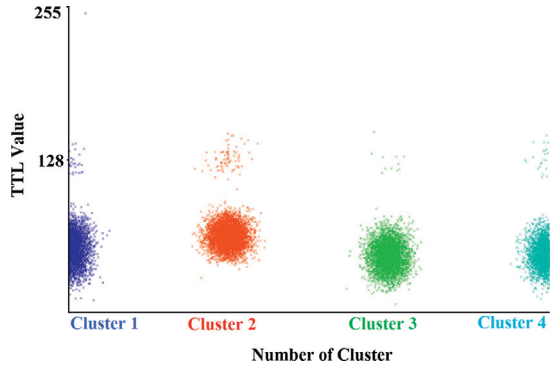
6.3. Validating the statistical time series clustering approach

To validate and authenticate the proposed DFA clustering technique that was previously proposed and demonstrated, we compare it with the well established machine learning clustering algorithms that were previously discussed in Section 5. To achieve this, we went back to our collected dataset of Section 6.2. Subsequently, we extracted from it a total of 29 data link, network and transport layer packet features as summarized in Table 9. To apply the *k*-means and the EM clustering algorithms, we compiled the extracted features into a coherent unified data file of 14,560 data instances and then fed the file into the WEKA data mining tool [43]. Consequently, we used the tool to run the clustering algorithms on the data file. The output of this operation is depicted in Table 10 and Figs. 9 and 10.

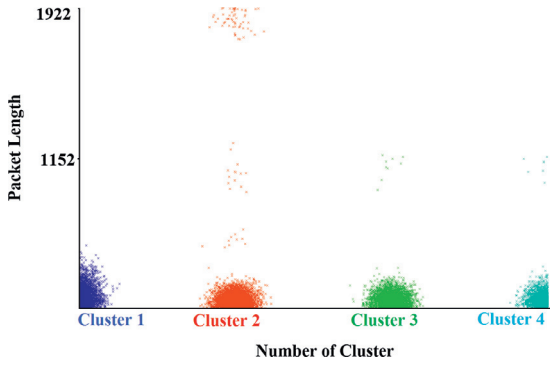
Table 10 demonstrates that the algorithms successfully clustered the traffic into four distinct classes (referring to the three scanning traffic and the legitimate traffic). The percentages represent the distribution of clustered instances upon the four classes. Fig. 9 clearly shows the formation of four clusters using the *k*-means. This result provides evidence that the traffic originates from four different classes. Further, Fig. 10a advocates such result by illustrating the four clusters that were formed by using the EM algorithm. Moreover, Fig. 10b and c shows two features, namely, the Time-To-Live (TTL) and the packet length features, that were used by the EM algorithm to perform the clustering. Note that, these 2 discriminating features are out of 12 other discriminating features. The clustering results are coherent with the scenario of Fig. 6. Moreover, it validates the DFA approach that was demonstrated in Section 6.2; the previously presented DFA



(a) Cluster Formation



(b) Cluster Formation using the TTL Feature



(c) Cluster Formation using the Packet Length Feature

Fig. 10. EM output-cluster formation and two discriminating features.

approach proved the hypothesis that states that it is feasible to cluster the scanning and the benign machines by investigating the traffic self similarity/correlation characteristics. Moreover, our proposed DFA clustering technique, in the context of cyber scanning, possesses two advantages over the classical machine learning approach. First, it can distinguish between scanning and benign machines and second it can infer which specific type/technique of scanning was employed by the scanning machines. On the other hand, the *k*-means and the EM approach is only able to detect that the traffic originates from four distinct classes.

7. Conclusion

This paper discussed an approach that is composed of two techniques that respectively tackle the issues of detecting corporate cyber scanning and clustering distributed reconnaissance activity.

First, the paper proposed a non-attribution anomaly detection technique. Motivated by the shortcomings of attribution-based approaches to cyber scan detection, this technique presented an alternative view of the problem/solution.

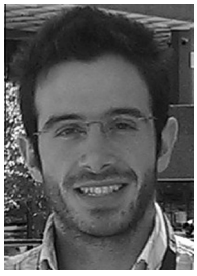
The idea is to focus on what is being offered by the network and hence on what is being scanned rather than who is performing the scanning. To characterize this, we introduced and elaborated on the notion of *enterprise network facade*. To construct and maintain the ENF, we leveraged the SNMP by presenting certain management procedures. The approach's training period is decoupled from any external traffic which makes its implementation very operationally feasible, in addition to having fast stabilization time yet requiring minimalistic system state storage. The technique's detection period is attribution-independent, which allows the detection of sophisticated reconnaissance activity, requires only a single packet to detect a scan and allows the detection of both TCP and UDP scans. To evaluate our technique, we experimented using a real network traffic dataset and implemented a proof-of-concept environment. The results demonstrated that for a class C network with 250 active hosts and five monitored servers, the proposed technique's training period required a stabilization time of less than 1 s and a state memory of 80 bytes. Moreover, in comparison with Snort's sfPortscan technique, it was able to detect 4215 unique scans and yielded zero false negative.

Second, the paper proposed a statistical time series clustering Technique. This approach was motivated by the observation that scanning machines that use the same technique to perform the scan will likely demonstrate temporal correlation and similarity. The idea is to capture such correlation in the traffic's signal, by utilizing the Detrended Fluctuation Analysis method. The aim is to correctly identify the scanning machines from the legitimate machines in addition to clustering those that utilize the same scanning technique. Using a customized evaluation scenario and setup, we found out that different scanning traffic originating from different bot groups indeed exhibited unique temporal correlation. Such uniqueness allowed the successful identification and clustering of the bots and the benign machines. We further validated our DFA approach by comparing it with the well established machine learning *k*-means and EM clustering techniques.

For future work, the next step would be to incorporate both techniques into a coherent system and perform its validation. Concerning the anomaly detection technique, we intend to leverage it by building efficient and effective heuristics for the detection of slow scans. On the other hand, concerning the proposed statistical time series clustering technique, we intend to experiment with real scanning botnet traffic in addition to fingerprinting other scanning techniques.

References

[1] Yoo Chung, Distributed denial of service is a scalability problem, SIGCOMM Comput. Commun. Rev. 42 (1) (2012) 69–71.
[2] M.K. Daly, Advanced persistent threat, Usenix, November 4, 2009 .
[3] Leyla Bilge, Tudor Dumitras, Before we knew it: an empirical study of zero-day attacks in the real world, in: Proceedings of the 2012 ACM Conference on Computer and Communications Security, CCS '12, ACM, New York, NY, USA, 2012, pp. 833–844.
[4] Symantec, W32.Stuxnet Dossier, 2012. <http://tinyurl.com/36y7jzb> (last accessed 25.10.2012).
[5] DefenseTech, Cyber War 2.0, Russia v. Georgia, 2012. <http://tinyurl.com/8l7cvm8> (last accessed 25.10.2012).
[6] The Globe and Mail, Ottawa Needs to Improve Cyber Security: Auditor General, 2012. <http://tinyurl.com/8n5sl7p> (last accessed: 25.10.2012).
[7] W. Zhang, S. Teng, X. Fu, Scan attack detection based on distributed cooperative model, in: Computer Supported Cooperative Work in Design, 2008, 12th International Conference on CSCWD 2008, IEEE, 2008, pp. 743–748.
[8] M.H. Bhuyan, D.K. Bhattacharyya, J.K. Kalita, Aocd: an adaptive outlier based coordinated scan detection approach, Int. J. Network Sec. 14 (6) (2012) 339–351.
[9] R. Baldoni, G. Di Luna, L. Querzoni, Collaborative Detection of Coordinated Port Scans. Technical Report, 2012, <http://www.dis.uniroma1.it/~midlab> (last accessed 27.10.2012).
[10] A. Dainotti, A. King, K. Claffy, F. Papale, A. Pescap, Analysis of a/0 stealth scan from a botnet, In: Internet Measurement Conference (IMC), November 2012.
[11] G. Conti, K. Abdullah, Passive visual fingerprinting of network attack tools, in: Proceedings of the 2004 ACM Workshop on Visualization and Data Mining for Computer Security, ACM, 2004, pp. 45–54.
[12] J. Treurniet, A network activity classification schema and its application to scan detection, IEEE/ACM Trans. Netw. 19 (5) (2011) 1396–1404.
[13] S. Staniford, J.A. Hoagland, J.M. McAlerney, Practical automated detection of stealthy portscans, J. Comput. Sec. 10 (1/2) (2002) 105–136.
[14] Internet Engineering Task Force (IETF), A Simple Network Management Protocol (SNMP), 1990. <http://www.ietf.org/rfc/rfc1157.txt> (last accessed 23.08.2012).
[15] William Stallings, SNMP, SNMPV2, Snmpv3, and RMON 1 and 2, third ed., Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1998.
[16] Internet Engineering Task Force (IETF), SNMP Overhead and Performance Impact, 2003. <http://tools.ietf.org/html/draft-breit-snmp-overhead-00> (last accessed 3.09.2012).
[17] L. Andrey, O. Festor, A. Lahmadi, A. Pras, J. Schönwälder, Survey of snmp performance analysis studies, Int. J. Network Manage. 19 (6) (2009) 527–548.
[18] Claude Fachkha, Elias Bou-Harb, Amine Boukhtouta, Son Dinh, Farkhund Iqbal, Mourad Debbabi, Investigating the dark cyberspace: profiling, threat-based analysis and correlation, in: 2012 7th International Conference on Risks and Security of Internet and Systems (CRiSIS), 2012, pp. 1–8.
[19] Symantec, Trojan Horse. <http://www.symantec.com/security_response/writeup.jsp?docid=2004-021914-2822-99> (last accessed 1.10.2013).
[20] C.-K. Peng, S.V. Buldyrev, S. Havlin, M. Simons, H.E. Stanley, A.L. Goldberger, Mosaic organization of dna nucleotides, Phys. Rev. E 49 (1994) 1685–1689.
[21] U. Harder, M.W. Johnson, J.T. Bradley, W.J. Knottenbelt, Observing internet worm and virus attacks with a small network telescope, Electron. Notes Theor. Comput. Sci. 151 (3) (2006) 47–59.
[22] K. Fukuda, T. Hirotsu, O. Akashi, T. Sugawara, Correlation among piecewise unwanted traffic time series, in: Global Telecommunications Conference, IEEE GLOBECOM, IEEE, 2008, pp. 1–5.
[23] M.B. Priestley, Spectral Analysis and Time Series, Academic Press, London, 1981.
[24] J.A.O. Matos, S. Gama, H.J. Ruskin, A.A. Sharkasi, M. Crane, Time and scale hurst exponent analysis for financial markets, Phys. A: Statist. Mech. Appl. 387 (15) (2008) 3910–3915.
[25] K. Hu, P.C. Ivanov, Z. Chen, P. Carpena, H.E. Stanley, Effect of trends on detrended fluctuation analysis, Phys. Rev. E 64 (1) (2001) 011114.
[26] A.K. Jain, M.N. Murty, P.J. Flynn, Data clustering: a review, ACM Comput. Surv. (CSUR) 31 (3) (1999) 264–323.
[27] M. Kantardzic, Data Mining: Concepts, Models, Methods, and Algorithms, Wiley-IEEE Press, 2011.
[28] R. Tibshirani, G. Walther, T. Hastie, Estimating the number of clusters in a data set via the gap statistic, J. Roy. Statist. Soc.: Ser. B (Statist. Methodol.) 63 (2) (2001) 411–423.
[29] R. Cilibrasi, P.M.B. Vitányi, Clustering by compression, IEEE Trans. Inform. Theory 51 (4) (2005) 1523–1545.
[30] J. MacQueen et al, Some methods for classification and analysis of multivariate observations, in: Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, vol. 1, California, USA, 1967, p. 14.
[31] A.P. Dempster, N.M. Laird, D.B. Rubin, Maximum likelihood from incomplete data via the em algorithm, J. Roy. Statist. Soc.: Ser. B (Methodol.) (1977) 1–38.
[32] R.M. Neal, G.E. Hinton, A view of the em algorithm that justifies incremental, sparse, and other variants, Nato ASI Ser. D: Behav. Social Sci. 89 (1998) 355–370.
[33] D.W. Scott, Multivariate Density Estimation, Wiley, New York, 1992.
[34] Eric Wustrow, Manish Karir, Michael Bailey, Farnam Jahanian, Geoff Huston, Internet background radiation revisited, in: Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement, IMC '10, ACM, New York, NY, USA, 2010, pp. 62–74.
[35] eMarkSof Inc., eMarksoft SNMP Component, 2002–2012. <http://www.emarksoft.com/mib-snmp-component.htm> (last accessed 6.09.2012).
[36] Daniel Roelker, Marc Norton, Jeremy Hewlett, sfportscan, 2004. <http://projects.cs.luc.edu/comp412/dredd/docs/software/readmes/sfportscan> (last accessed 21.08.2012).
[37] Snort, <http://www.snort.org>.
[38] D. Moore, G.M. Voelker, S. Savage, Inferring Internet Denial-of-Service Activity, Technical Report, DTIC Document, 2001.
[39] E. Wustrow, M. Karir, M. Bailey, F. Jahanian, G. Huston, Internet background radiation revisited, in: Proceedings of the 10th Annual Conference on Internet Measurement, ACM, 2010, pp. 62–74.
[40] V. Jacobson, C. Leres, S. McCanne, The Tcpdump Manual Page, Lawrence Berkeley Laboratory, Berkeley, CA, 1989.
[41] G.F. Lyon. Nmap Network Scanning: The Official nmap Project Guide to Network Discovery and Security Scanning Author: Gordon Fyodor l, 2009.
[42] M. Little, P. McSharry, I. Moroz, S. Roberts, Nonlinear, biophysically-informed speech pathology detection, in: Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings, vol. 2, p. II.
[43] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, I.H. Witten, The weka data mining software: an update, ACM SIGKDD Explor. Newsl. 11 (1) (2009) 10–18.



Elias Bou-Harb is a network security researcher pursuing his Ph.D. in Computer Science at Concordia University, Montreal, Canada. Previously, he has completed his M.A.Sc. degree in Information Systems Security at the Concordia Institute for Information Systems Engineering. He is also a member of the National Cyber Forensic and Training Alliance (NCFTA), Canada. His research interests focus on the broad area of cyber security, including operational cyber security for critical infrastructure, LTE 4G mobile network security, VoIP attacks and countermeasures and cyber scanning campaigns. He is supported by the prestigious Alexander Graham Bell Canada Graduate Scholarship (CGS) from the Natural Sciences and Engineering Research Council of Canada (NSERC).

Mourad Debbabi holds Ph.D. and M.Sc. degrees in computer science from Paris-XI Orsay University, France. He has published more than 70 research papers in international journals and conferences on computer security, formal semantics, mobile and embedded platforms, Java technology security and acceleration, cryptographic protocol specification, design, and analysis, malicious code detection, programming languages, type theory, and specification and verification of safety–critical systems. He is a full professor and the director of the Concordia Institute for Information Systems Engineering, Concordia University, Montreal, Quebec, Canada. He has served as a senior scientist at the Panasonic Information and Network Technologies Laboratory, Princeton, New Jersey; associate professor at the Computer Science Department of Laval University, Quebec, Canada; senior scientist at General Electric Research Center, New York; research associate at the Computer Science Department of Stanford University, Palo Alto, California; and permanent researcher at the Bull Corporate Research Center, Paris, France.

Chadi Assi received his B.Eng. degree from the Lebanese University, Beirut, Lebanon, in 1997 and his Ph.D. degree from the City University of New York (CUNY) in April 2003. He is currently a full professor with the Concordia Institute for Information Systems Engineering, Concordia University. Before joining Concordia University in August 2003 as an assistant professor, he was a visiting researcher with Nokia Research Center, Boston, Massachusetts, where he worked on quality of service in passive optical access networks. His research interests are in the areas of networks and network design and optimization. He received the prestigious Mina Rees Dissertation Award from CUNY in August 2002 for his research on wavelength-division multiplexing optical networks. He is on the Editorial Board of IEEE Communications Surveys & Tutorials, IEEE Transactions on Communications, and IEEE Transactions on Vehicular Technologies.