On Detecting and Clustering Distributed Cyber Scanning

Elias Bou-Harb
CIISE, Concordia University
Montreal, Quebec, Canada
e_bouh@encs.concordia.ca

Mourad Debbabi
CIISE, Concordia University
Montreal, Quebec, Canada
debbabi@ciise.concordia.ca

Chadi Assi
CIISE, Concordia University,
Montreal, Quebec, Canada
assi@ciise.concordia.ca

***Abstract*—This paper proposes 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 experiment with two real network traffic datasets and implement two proof-of-concept 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 proposed detection technique's training period required a stabilization time of less than 1 second 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.**

I. Introduction

The ever increase population and adoption 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 government officials 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. Indeed, the capability to detect, identify and attribute such scanning activity and its components is an important task to achieve as this would aid in preventing or mitigating the actual cyber attack from occurring.

978-1-4673-2480-9/13/$31.00

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 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.
- Experimenting, to empirically validate both techniques, with two *real* network traffic data sets.

The remainder of this paper is organized as follows. Section II discusses current detection and clustering techniques and pinpoints the drawbacks of attribution-based approaches. Section III presents the non-attribution anomaly detection approach and provides a discussion related to the training and detection periods. Section IV presents the clustering statistical-based approach by discussing the detrended fluctuation analysis. The evaluation environments coupled with the results are described in Section V. Finally, Section VI summarizes the paper and highlights the future work.

II. 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 5 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.

Most of the aforementioned detection and clustering techniques and other literature work [10–12] 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 III presents the non-attribution anomaly detection technique while Section IV 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.

III. 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.

A. 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.

B. ENF Management

In the training phase of our technique, we leverage the SNMP [13] 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 [14].

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 procedure `SNMP Receive-GetRequest` [13] 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 [13].

In the current work, this 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` [13] .

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 [13] 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 [15, 16].

C. 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 [17] 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.

D. 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 [18] 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 *custom ENF*. Additionally, our approach is transport protocol-independent and hence can detect both TCP and UPD scans.

IV. 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.

A. 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.

B. Detrended Fluctuation Analysis

To accomplish the above aim, we adopt the time series Detrended Fluctuation Analysis (DFA) method. DFA was first proposed in [19] 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 [20, 21], 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 [22] and Hurst analysis [23] 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 [24].

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

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

  where $B(i)$ is the i^{th} 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 $logF(n)$ 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.

V. Evaluation: Datasets, Methodologies and Results

For the purpose of empirically validating our approach, which consists of the proposed two techniques, we experimented with two real network traffic datasets and implemented two proof-of-concept environments.

A. Evaluating the non-attribution anomaly detection approach

We used a dataset that consists of unsolicited one-way telescope/darknet traffic [25] 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 was collected during the period of November 1, 2012 and December 1, 2012. Tables I and II, and Figure 1 show some network, transport and application level statistical information about the dataset.

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

TABLE I: Protocols Distribution

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

TABLE II: IP Class Distribution

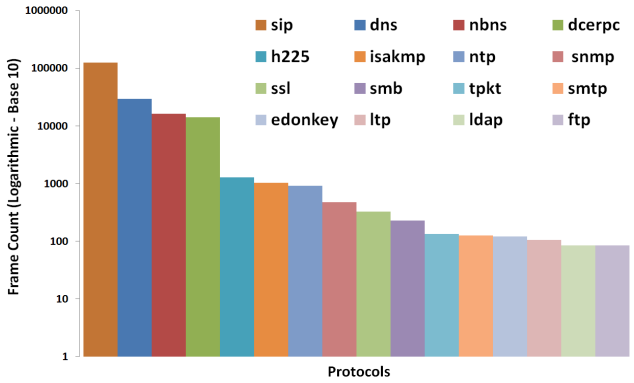


Fig. 1: Application Layer Protocols

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. Consequently, we built the network that is illustrated in Figure 2. 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 III-B to develop an SNMP tool. The tool is based on the software components provided by eMarksoft SNMP [26].

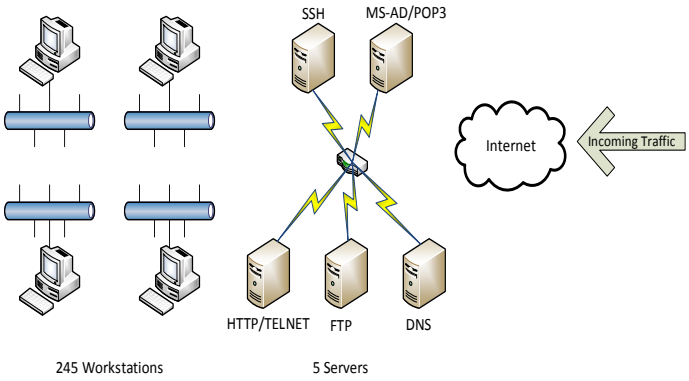


Fig. 2: Enterprise Network

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 III. The task was completed in 0.32 seconds and required 80 bytes of state memory.

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

TABLE III: ENF Details

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 [27], a preprocessor plugin for the open source network intrusion and detection system Snort [28], 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, 5 events (i.e., RST packets) are required from a given target within a window. The sliding timeout window varies from 60 to 600 seconds by sensitivity level; at the highest level, an alert will be generated if the 5 events are observed within 600 seconds. 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

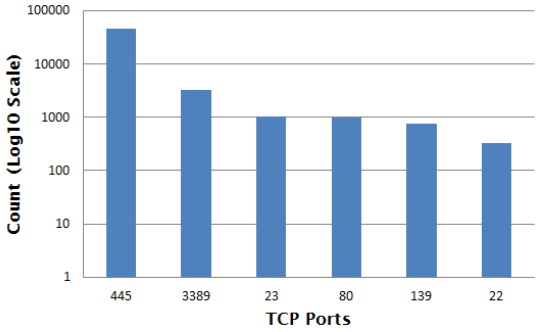


Fig. 3: Top 6 Scanned TCP Ports - One Day Sample

scans (unique IP/port pairs). Moreover, Figure 3 illustrates the top 6 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.

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. 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.

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 [29] 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 [30], 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

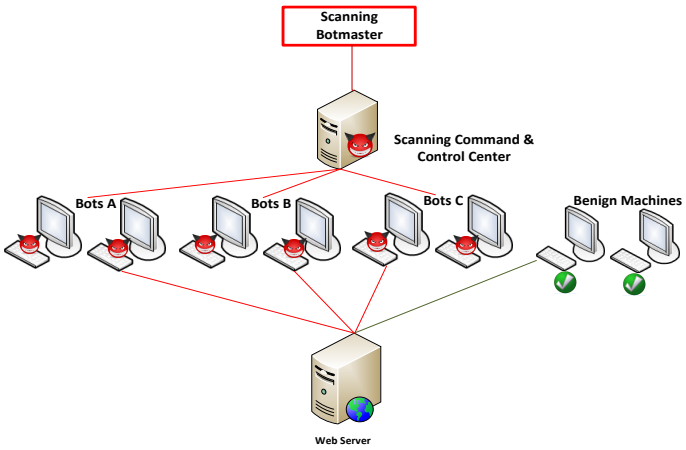


Fig. 4: Evaluating Scenario

scans that our approach was able to detect.

B. 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 Figure 4. 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 [31] 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 [32], an open source utility for network scanning and discovery. The bots, as shown in the scenario of Figure 4, 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 Figure 5.

To implement DFA, we have utilized the MATLAB code found in [33]. The output of applying the DFA method on the previous traffic time series distributions is shown in Figure 6 and the output of the scaling exponents α is summarized in Table IV.

Traffic Type	Scaling Exponent
TCP SYN Scanning	1.2
UDP Scanning	0.64
FIN Scanning	0.32
HTTP Traffic	0.95

TABLE IV: Summary of the scaling exponents α

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 IV, 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 ($\alpha \approx 1$).

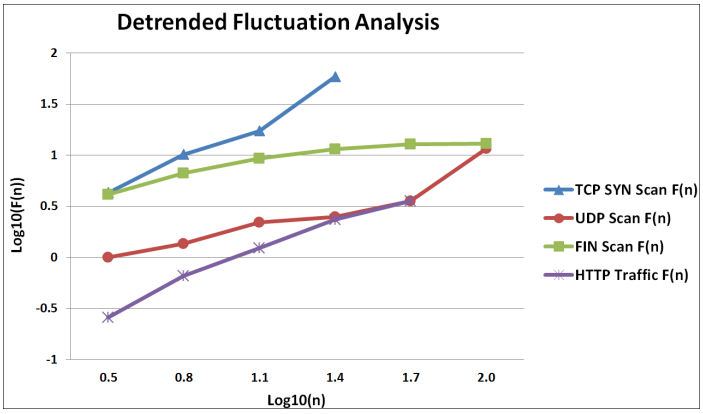
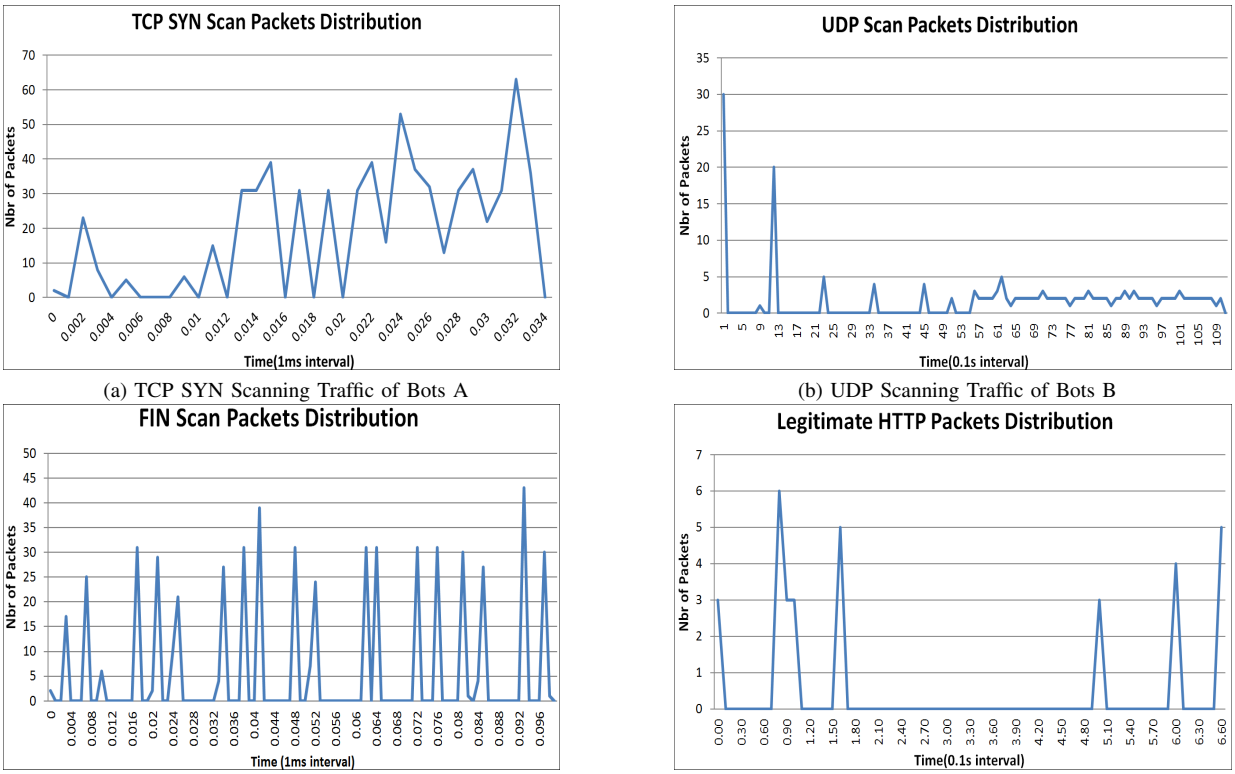


Fig. 6: The application of DFA on the scanning and benign traffic

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



(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. 5: Packets' Distribution generated by the three bot groups and the legitimate machines

flow, we applied the DFA method and then cross-matched it with the correlation status information that we have in Table IV. By accomplishing this, we showed that by only using the traffic signals self similarity feature, our proposed technique's clustering mechanism is able to correctly identify and cluster the scanning bots with high accuracy even in the presence of legitimate traffic.

VI. 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 5 monitored servers, the proposed technique's training period required a stabilization time of less than 1 second 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 exhibited unique temporal correlation. Such uniqueness allowed the successful identification and clustering of the bots and the benign machines.

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 thoroughly validating it by comparing it with well established

methods such as machine learning classifiers.

REFERENCES

[1] Yoo Chung. Distributed denial of service is a scalability problem. *SIGCOMM Comput. Commun. Rev.*, 42(1):69–71, January 2012.

[2] M.K. Daly. Advanced persistent threat. *Usenix, Nov*, 4, 2009.

[3] Leyla Bilge and 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, pages 833–844, New York, NY, USA, 2012. ACM.

[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, and X. Fu. Scan attack detection based on distributed cooperative model. In *Computer Supported Cooperative Work in Design, 2008. CSCWD 2008. 12th International Conference on*, pages 743–748. IEEE, 2008.

[8] M.H. Bhuyan, D.K. Bhattacharyya, and J.K. Kalita. Aocd: An adaptive outlier based coordinated scan detection approach. *International Journal of Network Security*, 14(6):339–351, 2012.

[9] R. Baldoni, G. Di Luna, and L. Querzoni. Collaborative Detection of Coordinated Port Scans. Technical report, 2012. http://www.dis.uniroma1.it/~midlab; Last accessed: 27/10/2012.

[10] G. Conti and K. Abdullah. Passive visual fingerprinting of network attack tools. In *Proceedings of the 2004 ACM workshop on Visualization and data mining for computer security*, pages 45–54. ACM, 2004.

[11] J. Treurniet. A network activity classification schema and its application to scan detection. *Networking, IEEE/ACM Transactions on*, 19(5):1396–1404, 2011.

[12] S. Staniford, J.A. Hoagland, and J.M. McAlerney. Practical automated detection of stealthy portscans. *Journal of Computer Security*, 10(1/2):105–136, 2002.

[13] Internet Engineering Task Force (IETF). A Simple Network Management Protocol (SNMP), 1990. http://www.ietf.org/rfc/rfc1157.txt; Last accessed: 23/08/2012.

[14] William Stallings. *SNMP,SNMPV2,Snmpv3,and RMON 1 and 2*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 3rd edition, 1998.

[15] 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.

[16] L. Andrey, O. Festor, A. Lahmadi, A. Pras, and J. Schönwälder. Survey of snmp performance analysis studies. *International Journal of Network Management*, 19(6):527–548, 2009.

[17] Claude Fachkha, Elias Bou-Harb, Amine Boukhtouta, Son Dinh, Farkhund Iqbal, and Mourad Debbabi. Investigating the dark cyberspace: Profiling, threat-based analysis and correlation. *2012 7th International Conference on Risks and Security of Internet and Systems (CRiSIS)*, 0:1–8, 2012.

[18] Symantec. Trojan Horse. http://www.symantec.com/security_response/writeup.jsp?docid=2004-021914-2822-99; Last accessed: 1/10/2013.

[19] C.-K. Peng, S. V. Buldyrev, S. Havlin, M. Simons, H. E. Stanley, and A. L. Goldberger. Mosaic organization of dna nucleotides. *Phys. Rev. E*, 49:1685–1689, Feb 1994.

[20] U. Harder, M.W. Johnson, J.T. Bradley, and W.J. Knottenbelt. Observing internet worm and virus attacks with a small network telescope. *Electronic Notes in Theoretical Computer Science*, 151(3):47–59, 2006.

[21] K. Fukuda, T. Hirotsu, O. Akashi, and T. Sugawara. Correlation among piecewise unwanted traffic time series. In *Global Telecommunications Conference, 2008. IEEE GLOBECOM 2008. IEEE*, pages 1–5. IEEE, 2008.

[22] M.B. Priestley. Spectral analysis and time series. 1981.

[23] J.A.O. Matos, S. Gama, H.J. Ruskin, A.A. Sharkasi, and M. Crane. Time and scale hurst exponent analysis for financial markets. *Physica A: Statistical Mechanics and its Applications*, 387(15):3910–3915, 2008.

[24] K. Hu, P.C. Ivanov, Z. Chen, P. Carpena, and H.E. Stanley. Effect of trends on detrended fluctuation analysis. *Physical Review E*, 64(1):011114, 2001.

[25] Eric Wustrow, Manish Karir, Michael Bailey, Farnam Jahanian, and Geoff Huston. Internet background radiation revisited. In *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement*, IMC '10, pages 62–74, New York, NY, USA, 2010. ACM.

[26] eMarkSof Inc. eMarksoft SNMP Component, 2002-2012. http://www.emarksoft.com/mib-snmp-component.htm; Last accessed: 6/09/2012.

[27] Daniel Roelker, Marc Norton and Jeremy Hewlett. sfportscan, 2004. http://projects.cs.luc.edu/comp412/dredd/docs/software/readmes/sfportscan; Last accessed: 21/08/2012.

[28] Snort. Available at: http://www.snort.org.

[29] D. Moore, G.M. Voelker, and S. Savage. Inferring internet denial-of-service activity. Technical report, DTIC Document, 2001.

[30] E. Wustrow, M. Karir, M. Bailey, F. Jahanian, and G. Huston. Internet background radiation revisited. In *Proceedings of the 10th annual conference on Internet measurement*, pages 62–74. ACM, 2010.

[31] V. Jacobson, C. Leres, and S. McCanne. The tcpdump manual page. *Lawrence Berkeley Laboratory, Berkeley, CA*, 1989.

[32] G.F. Lyon. Nmap network scanning: The official nmap project guide to network discovery and security scanning author: Gordon fyodor l. 2009.

[33] M. Little, P. McSharry, I. Moroz, and S. Roberts. Nonlinear, biophysically-informed speech pathology detection. In *Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings.*, volume 2, page II.