Efficient and Secure Statistical Port Scan Detection Scheme

Hussein Majed[1], Hassan N. Noura[1], Ola Salman[2], Ali Chehab[2], Raphaël Couturier[3]

[1] Arab Open University, Department of Computer Sciences, Beirut, Lebanon
[2] Dept. of Electrical and Computer Engineering, American University of Beirut, Beirut 1107 2020, Lebanon
[3] Univ. Bourgogne Franche-Comté, FEMTO-ST Institute, CNRS, Belfort, France

Abstract. One of the most challenging problems in Cybersecurity is the identification and prevention of port scanning, which is the primary phase of further system or data exploitation. This paper proposes a new statistical method for port scan detection, in addition to preventive and corrective counter-measures. The suggested solution is intended to be implemented at the Internet Service Provider (ISP) side. The proposed solution consists of aggregating NetFlow statistics and using the Z-score and co-variance measures to detect port scan traffic as a deviation from normal traffic. The experimental results show that the proposed method achieves a high detection rate (up to 100%) within a time frame of 60 seconds.

Keywords: Port Scan; Intrusion Detection; Traffic Aggregation; Network Security.

1 Introduction

The Internet is drastically expanding in terms of number of users, number of applications, and number of connected devices [1]. This will result in a huge amount of generated data to be digitally stored and available online. Having a heterogeneous set of connected devices with different capabilities presents new security challenges and vulnerabilities. This will attract cyber criminals to exploit and control vulnerable Internet connected devices. Cyber attacks go through multiple phases before achieving a successful exploitation. The first phase is the reconnaissance of the target, which can be either active or passive reconnaissance. Port scanning is one active reconnaissance technique to probe a server or host for open ports. Ports are usually numbered from 0 to 65535. The "Well Known" ports are within the range of 0 to 1023 and have been assigned by the Internet Assigned Numbers Authority (IANA) to well-known protocols and applications [2].

During a port scan, attackers send a message to each port, one at a time. The received response from each port determines whether this port is opened or closed. The scan can be conducted on Transmission Control Protocol (TCP) or User Datagram Protocol (UDP). According to SANS, port scanning is the most popular technique used by attackers to discover vulnerable services/devices and to exploit them [3].

The most common tool used for port scanning is NMAP [4]. The attacker can perform scan on a single Internet Protocol (IP) address (host) or a subnet of IPs (hosts). The attacker can also choose to scan a specific port or a range of ports. NMAP can be used to perform many types of scanning (See Fig. 1). The major types are mainly known as follows:

– TCP Syn Scan: connects quickly to thousand of ports without completing TCP handshake.
– TCP Connect Scan: a full connect scan that completes the three-way handshake to the target.
– TCP ACK Scan: sends packets with ACK flag set. If the port is open, the target will send an RST packet in the reply. If the port is closed, the target will ignore the packet.
– Xmas Scan: sends a set of flags to create a nonsensical interaction. If an RST packet is received, it means the port is closed, otherwise, the port is considered open.
– UDP Scan: sends an empty UDP packet to every targeted port. If no response is received the port is considered open, otherwise, if ICMP type 3 (destination unreachable message) is received, it means the port is closed.

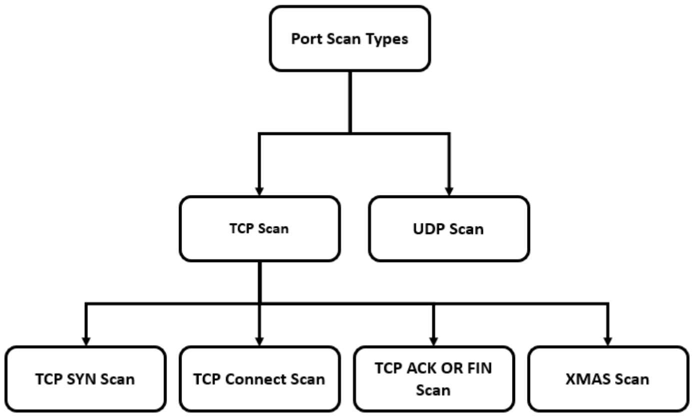


Fig. 1: Port Scan Taxonomy

In this paper, we propose a new port scan detection method based on NetFlow statistics. NetFlow is a network protocol developed by Cisco for collecting and monitoring IP network traffic. New measures are proposed based on flows statistics to differentiate between port scan and normal traffic based on the count of unique contacted port numbers and IP addresses. The experimental results conducted on an ISP collected data-set, containing port scan attack traffic, show that the proposed method achieves a high detection rate with low false positive rate.

This paper is organized as follows. In Section 2, we review the existing port scan detection techniques. In Section 3, we present our proposed detection solution. The experimental results are presented and discussed in Section 4. Finally, we conclude in Section 5.

2 Related Work

In this section, we review the most recent port scanning detection techniques. In [5], a mathematical model for detecting anomalies caused by port scan attacks has been proposed. The proposed solution relies on constructing a set of vectors based on the IP-address of the sender, and the packets' TCP flags. Based on assumption that normal traffic patterns are known, the model calculates the frequency of occurrence of TCP flags. The negative likelihood logarithm is used to define the anomaly packets index. In [6], Balram et al. considered the count of TCP control packets to detect TCP SYN scan by training a Neural Network model. The counts of TCP SYN, SYN-ACK, and FIN packets were used to train the model to differentiate between attack and normal behaviors. The authors simulated the TCP SYN scan with multiple probe delays from 5 to 300 seconds. They found out that they cannot rely on the RST flag for detection, instead the calculation of the difference between incoming SYN and outgoing SYN-ACK were the key features for detection.

An adaptive threshold for detecting various types of port scans based on using time independent features set was proposed in [7]. The approach consists of updating the threshold by relying on a fuzzy-based detector. Tested on DARPA 98/99, the proposed solution presented a high false positive rate when traffic is destined to servers. Additional features should be added to the solution in order to address this issue. Another study relied also on the fuzzy logic to analyze various traffic parameters in order to detect port scan. The solution in [8] relied on the time average between received packets by destination/victim, the number of sent packets by source, and the number of received packets by the destination/victim. The results showed the effectiveness of the proposed method when multiple attackers are scanning a single target at the same time. Also, a recent port scan detection method that used the fuzzy rule interpolation was presented in [9].

In [10], a new port scan detection approach was proposed using time-based flow size distribution sequential hypothesis testing (TFDS). This solution can be applied in transit networks, where only unidirectional flows' information is available. The authors realized that the scanners produce small flows with equal byte size and thus, they adopted the Flow Size Distribution (FSD) in bytes for modeling the scanning activity to build the FSD entropy metric. The FSD for each source IP is used to build and update the likelihood ratio table. Repetitive likelihood measures were used to detect scanning activity. Profiling of IP addresses is another technique that was proposed for detecting TCP SYN scan in [11]. IP profiling consists of constructing profiles for each IP address participating in TCP connections. The whole theory is built on the assumption that a scanner leaves more initialised TCP handshakes half-open. If the attacker does not send ACK request within 30 seconds of receiving the SYN ACK, it will be flagged as an attacker, otherwise the traffic will be considered normal. Although this detection method can be easily bypassed by using full connection scanning techniques, yet it showed promising results when applied on several data-sets. Moreover, deep learning and Support Vector Machine (SVM) algorithms were used in [12] for detecting scan attempts. In the deep learning case, the features were extracted automatically while in SVM, a supervised machine learning, the researchers selected all fields as features. Deep learning performed much better than SVM, where 97.80%, and 69.79% detection rates were achieved, respectively. Also, a set of recent solutions that are based on deep learning were presented such as [13,14,15,16,17,18,14,19].

However, in this paper, we consider a lightweight scan detection technique built on top of NetFlow protocol. NetFlow is a network protocol introduced by Cisco to collect traffic/flows statistics at the router interfaces. Handling huge amount of traffic at the network core or the ISP side, NetFlow is a good candidate for extracting flows statistics without burdening the network with new APIs or add-ons to extract measures for port scan detection. In this context, this work considers the Z-score and co-variance measures derived from NetFlow statistics. Moreover, this work considers the combination of different features such as the count of destination IP addresses, the count of source and destination port numbers, and the packet size distribution. Thus, the contributions of this paper can be summarized as follows:

- Proposing a customized NetFlow aggregation targeting the two main scan strategies (single host port scan and sub-net port scan).
- Presenting a fast and reliable statistical approach to detect different types of port scan based on Z-score, co-variance and mean values.
- Proposing a mitigation process to block further communication between the attacker and victims, at the ISP level.

3 Proposed Port Scanning Detection

The proposed port scan detection method consists of 5 main steps as illustrated in Fig. 2, and detailed below.

3.1 Data Collection

At this stage, the incoming traffic statistics are collected for a regular interval of time such as one minute. This is possible by configuring the NetFlow exporter

Table 1: Table of notations

Symbol	Definition
μ	Mean
σ	Standard deviation
CV_{SP}	Coefficient variation of the number of source ports per group of source and destination IPs
CV_{DP}	Coefficient variation of the number of destination ports per group of source and destination IPs
CV_{DA}	Coefficient variation of the number of destination addresses per group of source addresses and destination sub-nets
ZSC_{SP}	Z-score of the number of source ports per group of source and destination IPs
ZSC_{DP}	Z-score of the number of destination ports per group of source and destination IPs
ZSC_{DA}	Z-score of the number of destination addresses per group of source addresses and destination sub-nets

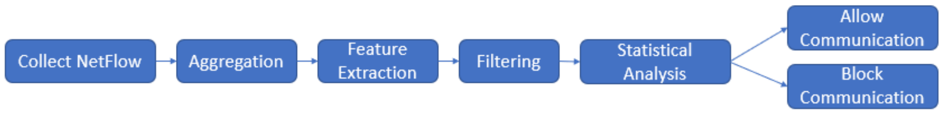


Fig. 2: Proposed port scan detection scheme

on the edge router to send all the NetFlow statistics to an external surveillance device (i.e. NetFlow collector). The port scan was performed on several targets from different sources and a time table was updated each time a scan is performed, with the type of attack, source IP and target. Based on the time table, the data-sets were manually labeled.

3.2 Data Aggregation and Features Extraction

In this step, the collected traffic NetFlow statistics are aggregated. Port scanning can target a single address or an IP sub-net. For this reason, we propose two data aggregation methods as follows:

- For sub-net scanning, we focus on knowing if a source IP communicated with different destination IPs in the same sub-net on different destination ports; that is why the traffic is grouped by source address, destination sub-net, destination port, and transport protocol. After grouping the flows, the number of distinct destination IPs is counted.
- For single device scanning, we focus on knowing if a source IP communicated with a single destination IP on multiple distinct ports; that is why the traffic is grouped by source and destination IP addresses, and the count of unique source and destination port numbers is calculated for each group.

3.3 Filtering Process

Filtering is a very essential phase to obtain correct results. In normal conditions, the flow of traffic contains hundreds or thousands of connections from one single IP address to another single IP address. If we keep these records, it will definitely affect the standard deviation and mean values during the statistical analysis. Thus, we keep only the flows that serve the purpose of each conducted test. Concerning the sub-net scan, the filtering process should eliminate all the flow records, if the source IP address communicates with less than 4 destination IP addresses, and also filter and remove packets that do not share the same size. During sub-net scan, the attacker communicates with multiple IP addresses within the same sub-net. In this case, eliminating traffic from one source IP address to less than 4 destination IP addresses will only keep potential sub-net scan flows. While in case of single host port scan, the filtering process should eliminate all the flow records if the source IP address communicates with less than 4 port numbers on the same destination IP address. During single host port scan, the attacker communicates with a range of port numbers on a single host. In this case, eliminating traffic from one source IP to destination IP, if the number of distinct destination port numbers is less than 4 will only keep the potential single host port scan traffic.

3.4 Statistical Analysis

At this phase, the aggregated filtered statistics are analyzed using the Z-score metric (see Algorithm 1. The basic Z-score formula for a feature x is:

$$z = \frac{(x - \mu)}{\sigma} \tag{1}$$

where μ and σ represent the mean and the standard deviation of each feature in the aggregated traffic, respectively.

Sub-net Port Scan The NetFlow is grouped per source address, destination sub-net, destination port number, and protocol. For each group, the count of unique destination addresses is calculated. During TCP scan, the attacker tries to probe the open ports by sending packets containing TCP flags and making assumption based on the response. These packets have the same size and are destined to different IPs within the same sub-net. The count of distinct TCP packets sizes is also computed. The Z-score of the count of unique destination IP addresses for each group is calculated.

Single Host Port scan The NetFlow statistics are grouped per source and destination IP addresses. For each group, the count of unique destination port numbers is calculated. In addition, the count of unique source port numbers is calculated for TCP packets to avoid false positive alarms in case of communication between a client and a server. While the count of source ports is calculated for UDP, because for example, NMAP uses the same source port for scanning. The Z-score of the count of unique destination ports and the Z-score of the count of source ports for each group are calculated.

These values are used as input to the next step. The aggregation can be seen as a form of non-reversible compression. The advantage of this step is to reduce the required processing complexity and storage overhead. Thus, the following detection features are selected and used for each scan method:

- TCP based sub-net port scan: the count of unique destination addresses and the count of distinct packet sizes.
- UDP based sub-net port scan: the count of unique destination addresses.
- TCP based host port scan: the count of unique source port numbers and the count of unique destination port numbers.
- UDP based host port scan: the count of source port numbers and the count of unique destination port numbers.

3.5 Detection Process

For each aggregated filtered flow, the obtained Z-scores of the selected features are compared to the corresponding threshold values that are based on the coefficient of variation (CV). CV represents a statistical indicator for the dispersion of data points in a data set (series) around the mean. It is the ratio of the standard deviation to the mean, and it is a useful statistical metric for comparing the degree of variation from one data series to another, even if the means are different. CV is computed as illustrated in the following equation:

$$\mathrm{CV} = \frac{\sigma}{\mu} \tag{2}$$

Accordingly, a port scan is detected if the obtained Z-score values of these features are greater or equal to the corresponding thresholds (see lines 7, or 10 of Algorithm 1 & line 9 of Algorithm 2). These thresholds are set based on a training step (initial step), and depend on the ISP profile (traffic and network characteristics). Using static thresholds is not practical because they need to be updated manually each time a bandwidth upgrade takes place. In addition, they might lead to false positives and false negatives. To make the proposed solution based on a dynamic threshold, the Z-score value of each feature is compared to its CV.

Sub-net Port Scan In case of sub-net port scan, the attacker tries to connect to a single or multiple ports for each destination address in the same sub-net. The scan will increase the mean value and standard deviation of the count of unique destination IPs for the same sub-net in the data-set, which leads to a high CV of the count of unique destination IPs. By Calculating the Z-score of each feature for each group, the group of source IP address, destination sub-net, and protocol will have a high Z-score value in relation to the count of unique destination address. Since we have the CV value of the count of unique destination IP addresses, we can compare it to the Z-score of count of unique destination IP addresses of each data point; there is a slight difference between the behavior of TCP scan and UDP scan. While studying the results of the approach, we realized that the TCP scan traffic always produces a CV of the count of unique destination addresses higher than 1, while UDP scan always produce Z-score higher than the mean value of the count of unique destination addresses.

- In case of TCP, If the Z-score of unique destination addresses is higher than the CV, and the CV of the count of unique destination addresses is higher than 1, the flow is considered an outlier compared to the rest of the flows (see line 7 of Algorithm 1).
- In case of UDP, If the Z-score of unique destination addresses is higher than the CV, and the count of unique destination addresses is higher than the mean, the flow is also considered an outlier compared to the rest of the flows (see line 10 of Algorithm 1).

Host Port Scan In case of host port scan, the attacker will try to connect to multiple ports on the scanned host. The scan will increase the mean value and standard deviation of the count of source ports and the count of unique destination ports of the affected group of communication between one IP and another in the data-set, which leads to a high CV for each affected feature. By Calculating the Z-score of each feature for each group, the group of source addresses, destination addresses, and protocol will have a high Z-score value for each selected feature (count of source ports and destination ports). Since we have the CV value of each feature, we can compare it to the Z-score of each corresponding feature. If the Z-score of source port count and Z-score of destination port count are higher than the CV of the corresponding feature, it is considered an outlier compared to the rest of the flows (see line 9 of Algorithm 2).

It should be indicated that the Z-score is a well known method for the detection of outliers. Applying the Z-score on each feature for the aggregated flows clearly reveals the outliers, but each type of scan has its own characteristics.

4 Experimental Setup and Results

In this section, we explain the implementation of the proposed port scan detection scheme in details.

4.1 NetFlow Collection

For data collection, the set up was implemented in an ISP network taking into consideration the full traffic visibility. NetFlow is a network protocol introduced by Cisco which collects IP network traffic statistics as the packets flow in or out of an interface. A virtual machine based on Centos 7.0 was deployed on Vmware ESXi 7.0 as a NetFlow Collector. The NetFlow capture daemon (nfcapd), a part

Algorithm 1 Subnet port scan detection algorithm

```
    Input: Aggregated Flow $AF$
    Output: $R$
 1: procedure $R = \mathsf{Detection}(AF)$
 2:     $NIP \leftarrow NumberofRows(AF)$
 3:     $CV_{DA} \leftarrow \frac{std(C_{DA})}{mean(C_{DA})}$
 4:     for $i \leftarrow 0$ to $NIP$ do
 5:         $ZSC_{DA} \leftarrow Zscore(C_{DA})$
 6:         $Protocol \leftarrow Value(C_{PR})$
 7:         if $|ZSC_{DA}| \geq CV_{DA}\&\&|Protocol| = TCP\&\&|CV_{DA}| \geq 1$ then
 8:             TCP subnet scan detected on DA subnet
 9:         end if
10:         if $|ZSC_{DA}| \geq CV_{DA}\&\&|Protocol| = UDP\&\&C_DA > mean(C_{DA})\&\&CV_{DA} \geq 1$ then
11:             UDP subnet scan detected on DA subnet
12:         end if
13:     end for
14: end procedure
```

Algorithm 2 Single host port scan detection algorithm

```
    Input: Aggregated Flow $AF$
    Output: $R$
 1: procedure $R = \mathsf{Detection}(AF)$
 2:     $NIP \leftarrow NumberofRows(AF)$
 3:     $CV_{DP} \leftarrow \frac{std(C_{DP})}{mean(C_{DP})}$
 4:     $CV_{SP} \leftarrow \frac{std(C_{SP})}{mean(C_{SP})}$
 5:     for $i \leftarrow 0$ to $NIP$ do
 6:         $ZSC_{DP} \leftarrow Zscore(C_{DP})$
 7:         $ZSC_{SP} \leftarrow Zscore(C_{SP})$
 8:         $Protocol \leftarrow Value(C_{PR})$
 9:         if $|ZSC_{DP}| \geq CV_{DP}\&\&|ZSC_{SP}| \geq CV_{SP}$ then
10:             Single host port scan detected on DA
11:         end if
12:     end for
13: end procedure
```

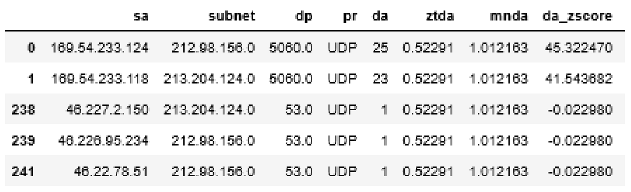
of the nfdump tool, was installed on the NetFlow Collector virtual machine to capture all the incoming flows. The NetFlow exporters of two Cisco edge routers (ASR 9K) have been configured to send the traffic to the NetFlow Collector; each router was configured to send the NetFlow statistics to a different port on the traffic Collector. Two instances of the ”nfcapd” were launched to listen on two separate ports, each port is dedicated for one router. The ”nfcapd” was configured to save the captured flows of each router in a separate file, every one minute.

nfcapd -w -D -l /flow__base__dir/router1 -p 12345
nfcapd -w -D -l /flow__base__dir/router2 -p 12346

The traffic statistics from both routers were collected for one month. Multiple port scans were performed during the mentioned period and a time table was updated after each performed scan to differentiate the normal traffic from the scan traffic. The nfcapd saves the files in raw format. The saved files are converted, by the "nfdump" tool, to the Comma-Separated Values (CSV) format.

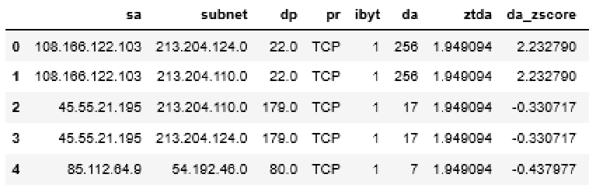
4.2 Data Aggregation

A python script was written to customize the aggregation of NetFlow statistics and to visualize the captured flows. The script is based on the concept of grouping the flows by two different methods in order to capture the different scan strategies. The first aggregation method is based on aggregating the flows with the same source address, destination sub-net, and same destination addresses (See Fig. 3a & Fig. 3b). The second method is based on aggregating the flows with the same source address, destination address, count of source ports, and count of unique destination ports (See Fig. 4a & Fig. 4b).



	sa	subnet	dp	pr	da	ztda	mnda	da_zscore
0	169.54.233.124	212.98.156.0	5060.0	UDP	25	0.52291	1.012163	45.322470
1	169.54.233.118	213.204.124.0	5060.0	UDP	23	0.52291	1.012163	41.543682
238	46.227.2.150	213.204.124.0	53.0	UDP	1	0.52291	1.012163	-0.022980
239	46.226.95.234	212.98.156.0	53.0	UDP	1	0.52291	1.012163	-0.022980
241	46.22.78.51	212.98.156.0	53.0	UDP	1	0.52291	1.012163	-0.022980

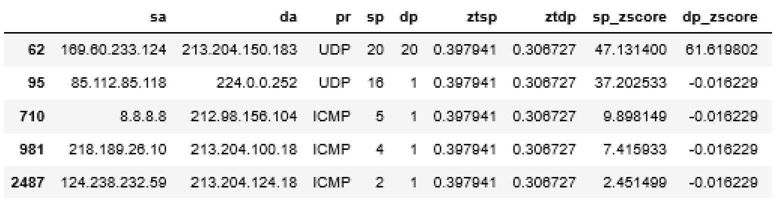
(a) UDP based sub-net port scan



	sa	subnet	dp	pr	ibyt	da	ztda	da_zscore
0	108.166.122.103	213.204.124.0	22.0	TCP	1	256	1.949094	2.232790
1	108.166.122.103	213.204.110.0	22.0	TCP	1	256	1.949094	2.232790
2	45.55.21.195	213.204.110.0	179.0	TCP	1	17	1.949094	-0.330717
3	45.55.21.195	213.204.124.0	179.0	TCP	1	17	1.949094	-0.330717
4	85.112.64.9	54.192.46.0	80.0	TCP	1	7	1.949094	-0.437977

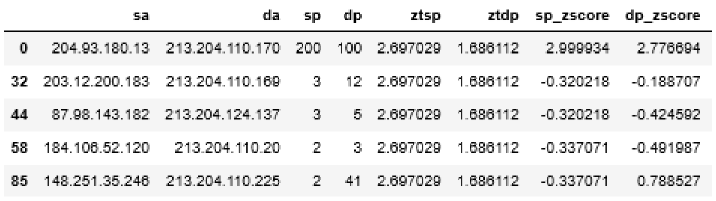
(b) TCP based sub-net port scan

Fig. 3: An example of grouped and aggregated NetFlow of UDP and TCP based sub-net port scan



	sa	da	pr	sp	dp	ztsp	ztdp	sp_zscore	dp_zscore
62	169.60.233.124	213.204.150.183	UDP	20	20	0.397941	0.306727	47.131400	61.619802
95	85.112.85.118	224.0.0.252	UDP	16	1	0.397941	0.306727	37.202533	-0.016229
710	8.8.8.8	212.98.156.104	ICMP	5	1	0.397941	0.306727	9.898149	-0.016229
981	218.189.26.10	213.204.100.18	ICMP	4	1	0.397941	0.306727	7.415933	-0.016229
2487	124.238.232.59	213.204.124.18	ICMP	2	1	0.397941	0.306727	2.451499	-0.016229

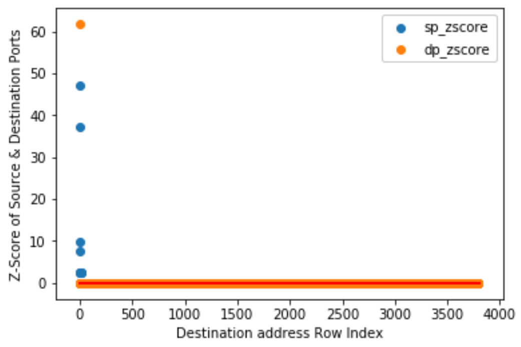
(a) UDP based Host Port Scan



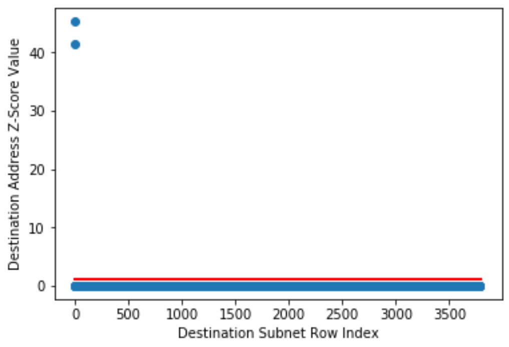
	sa	da	sp	dp	ztsp	ztdp	sp_zscore	dp_zscore
0	204.93.180.13	213.204.110.170	200	100	2.697029	1.686112	2.999934	2.776694
32	203.12.200.183	213.204.110.169	3	12	2.697029	1.686112	-0.320218	-0.188707
44	87.98.143.182	213.204.124.137	3	5	2.697029	1.686112	-0.320218	-0.424592
58	184.106.52.120	213.204.110.20	2	3	2.697029	1.686112	-0.337071	-0.491987
85	148.251.35.246	213.204.110.225	2	41	2.697029	1.686112	-0.337071	0.788527

(b) TCP based Host Port Scan

Fig. 4: An example of grouped and aggregated NetFlow of UDP and TCP based host port scan



(a) UDP based host port scan Z-score results



(b) UDP based subnet port scan Z-score results

Fig. 5: An example of grouped and aggregated NetFlow of UDP and TCP based host port scan Z-score

4.3 Port Scan Detection

The approach was tested on the collected data-set, the Z-score of each feature was calculated and compared to the threshold, which is in our case the CV for each feature. The results indicate that records with positive Z-score are suspicious and could indicate that port scan is being conducted on the targets. But with the comparison of CV to each feature's Z-score value, we can validate the approach.

The results shown in Fig. 3b for TCP based sub-net scan and in Fig. 3a for UDP based sub-net scan, clearly show that using the coefficient of variation as a dynamic threshold is efficient for correctly detecting sub-net port scan. The results shown in Fig. 6 indicate that the sub-nets 213.204.124.0/24 &

Table 2: Tested data-sets

Data-set Name	Scan Type	Protocol	Target	Ports
Normal1	None	-	-	-
Normal2	None	-	-	-
Normal3	None	-	-	-
Normal4	None	-	-	-
TCP1	Full connect	TCP	212.98.156.0/24	21,22,80,8080
TCP2	Full connect	TCP	213.204.124.0/24 and 213.204.110.0/24	22
TCP3	Syn scan	TCP	212.98.156.0/24	22
TCP4	Syn scan	TCP	213.204.110.170	Well Known Range
UDP1	UDP scan	UDP	212.98.156.0/24 and 213.204.124.0/24	5060
UDP2	UDP scan	UDP	213.204.124.0/24	17185
UDP3	UDP scan	UDP	213.204.150.183	Multiple ports

213.204.110.0/24 have received connections from a single source address 108.166.122.103 on 250 different destination addresses in each sub-net.

Fig. 7 shows that the score value became higher and above the threshold of CV, which is also higher than 1, when sub-net scan is detected. By reviewing Fig. 3b, we can conclude that the sub-net port scan was on port number 22. The CV threshold was 1.94 while the Z-score for the two sub-nets is 2.23 taking into consideration that the next group of communication in Fig. 3b (third row) between the source IP 45.55.21.196 and the sub-net 213.204.110.0/24 has a Z-score value of -0.33. This clearly indicates the abnormality on the first two sub-nets, 213.204.124.0/24 & 213.204.110.0/24 from the attacker source IP 108.166.122.103.

Another sub-net port scan test was performed on UDP ports. Fig. 3a shows that the sub-net 212.98.156.0/24 received 23 requests on port 5060 from the source IP 169.54.223.124 and the subnet 212.98.124.0/24 received 25 requests on port 5060 from the source IP 169.54.223.118, while the next group in row 212.98.124.0/24 received only one request from 46.227.2.150. The Z-score values of the scanned sub-nets are above 40, while the rest are below zero and the mean value of destination address count is more than 1. Reviewing Fig. 5b shows that threshold was 1 and all the normal traffic has Z-score values below 0, while the two scanned sub-nets 212.98.156.0/24 & 212.98.124.0/24 have a Z-score value higher than 40 for destination address count.

For TCP based host port scan, the results shown in Fig. 8 indicate that a single host received connections form a single source address on 100 unique destination ports. By reviewing Fig. 9, it is clear that when host port scan is detected, the Z-score values of source ports and destination ports counts were relatively higher and above the threshold line. The CV of source port is 2.69 while the Z-score of source port is 2.99 and the CV of destination port is 1.68

while the Z-score of the destination port is 2.77, and the next group in row has negative Z-score values in both source and destination ports.

The same concept was applied to UDP based host port scan as shown in Fig. 4a. The scanned host received 20 connections on 20 different ports. In the UDP host scan, there are two thresholds, the source port count and destination port count, and in order to consider that a port scan was performed, the Z-score of both values should be above the threshold (see line 9 of Algorithm 2). Reviewing Fig. 5a, we see that there are multiple points crossing the threshold regarding the source port, but only one point has the value of 61 crossing the threshold of destination port, while the next in row is negative.

The same procedure was applied on all data-sets listed in table 2, and it showed 100% true positive and 0% false negative alerts. Some of the tested data-sets did not contain any port scan (Normal1, 2, 3, and 4) while the others contained multiple types of port scan targeting a single host and sub-nets.

Table 3: Execution Time

Data-set Name	Size in MB	Number of Rows	Loading Time	Detection Time
TCP1	8	28621	0.453	0.125
TCP2	10.3	29011	0.582	0.234
TCP3	10.2	28740	0.469	0.187
TCP4	9.8	31711	0.4056	0.187
UDP1	11.6	32670	0.5309	0.11
UDP2	9.6	31050	0.406	0.156
UDP3	9.5	30781	0.468	0.078

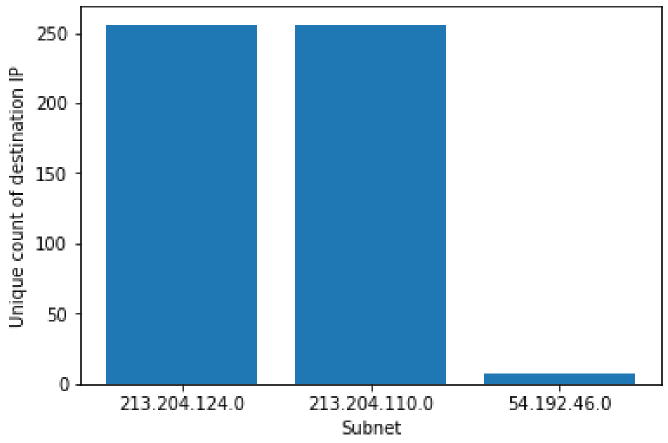


Fig. 6: Sub-net port scan results

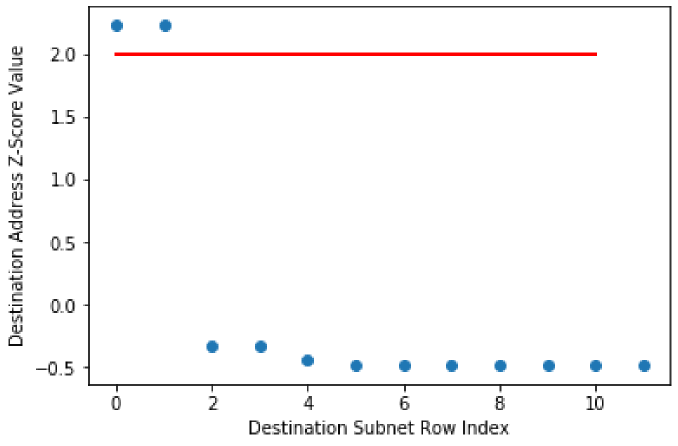


Fig. 7: Sub-net port scan Z-score results

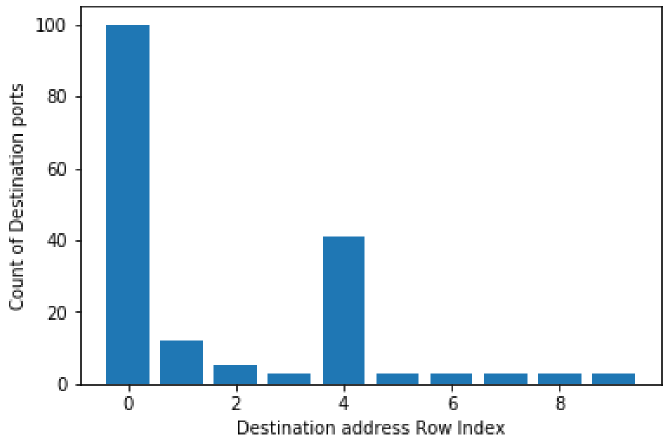


Fig. 8: Host port scan results

Execution Time In this part, the required execution loading and detection time of each data-set are computed. The loading time represents the time taken for each data-set to be loaded in the memory, and the detection time represents the computing time needed for flow aggregation, filtering, and detection of port scan presence. Table 3 shows these results. It indicated that the loading and detection time varies according to the collected flow size for 60 seconds. The average time for loading data-set is 0.47 second, the average time taken for detection is 0.15 second. Consequently, this indicates clearly that the proposed

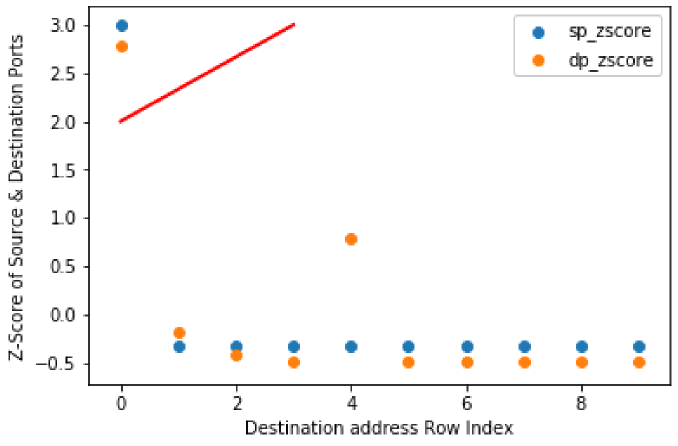


Fig. 9: Host port scan Z-score results

solution requires low latency to detect any listed port scan attack variant and especially in the context of huge volume traffic (ISP).

4.4 Proposed Mitigation Process

Based on the statistical results obtained from the previous phase, the ISP may update the Access Control List (ACL), on the edge routers, to block the communication between the attacker and the destination sub-nets or addresses to avoid any further communication between them for a specific period of time. To perform this action, a Secure Shell (SSH) user should be added to the edge router with privileges to create and remove ACL. When port scan is detected, the sensor should connect to the edge router via SSH and append the access list based on the detected scan as follows:

– HOST based: access-list 101 dynamic testlist timeout 10 deny ip host #AttackerIP# host #ScannedIP#
– Subnet Based: access-list 101 dynamic testlist timeout 10 deny ip host #AttackerIP# #Scanned Subnet# 0.0.0.255

The above commands will add a dynamic access list to the edge router in order to block the attacker for 10 minutes, whenever a scan is detected. The sensor will send the appropriate access-list based on the scan type.

5 Conclusion

Port scanning techniques are real threats to network security. They should be detected and avoided in current and future networks. In this paper, detective, preventive and corrective measures against various possible techniques of port scanning are proposed. The solution is designed to be applied at the ISP level, and it is based on a statistical lightweight scheme. Technically, this method aggregates and filters NetFlow statistics, and then statistical analysis is performed to detect port scan traffic by using the Z-score and co-variation metrics. To guarantee the detection of a port scan, the proposed method requires 60 seconds. The experimental results showed that high detection rate (up to 100%) can be achieved using simple measures with a low false positive rate (0%).

Acknowlegments

This paper was partially supported by funds from the Maroun Semaan Faculty of Engineering and Architecture at the American University of Beirut and by the EIPHI Graduate School (contract ”ANR-17-EURE-0002”).

References

1. Arunan Sivanathan, Hassan Habibi Gharakheili, and Vijay Sivaraman. Can we classify an iot device using tcp port scan? In *2018 IEEE International Conference on Information and Automation for Sustainability (ICIAfS)*, pages 1–4. IEEE, 2018.
2. Michelle Cotton, Lars Eggert, Joe Touch, Magnus Westerlund, and Stuart Cheshire. Internet assigned numbers authority (iana) procedures for the management of the service name and transport protocol port number registry. *RFC*, 6335:1–33, 2011.
3. Roger Christopher. Port scanning techniques and the defense against them. *SANS Institute*, 2001.
4. Gordon Fyodor Lyon. *Nmap network scanning: The official Nmap project guide to network discovery and security scanning.* Insecure, 2009.
5. E. V. Ananin, A. V. Nikishova, and I. S. Kozhevnikova. Port scanning detection based on anomalies. In *2017 Dynamics of Systems, Mechanisms and Machines (Dynamics)*, pages 1–5, 2017.
6. S. Balram and M. Wiscy. Detection of tcp syn scanning using packet counts and neural network. In *2008 IEEE International Conference on Signal Image Technology and Internet Based Systems*, pages 646–649, 2008.
7. H. U. Baig, F. Kamran, and M. A. Sheikh. An adaptive fuzzy based scan detection technique using time independent feature set. In *2009 IEEE International Conference on Intelligent Computing and Intelligent Systems*, volume 3, pages 123–127, 2009.
8. Wassim El-Hajj, Fadi Aloul, Zouheir Trabelsi, and Nazar Zaki. On detecting port scanning using fuzzy based intrusion detection system. In *2008 International Wireless Communications and Mobile Computing Conference*, pages 105–110. IEEE, 2008.
9. Mohammad Almseidin, Mouhammd Al-Kasassbeh, and Szilveszter Kovacs. Detecting slow port scan using fuzzy rule interpolation. In *2019 2nd International Conference on new Trends in Computing Sciences (ICTCS)*, pages 1–6. IEEE, 2019.
10. Y. Zhang and B. Fang. A novel approach to scan detection on the backbone. In *2009 Sixth International Conference on Information Technology: New Generations*, pages 16–21, 2009.

11. Katalin Hajdú-Szücs, Sándor Laki, and Attila Kiss. A profile-based fast port scan detection method. In Ngoc Thanh Nguyen, George A. Papadopoulos, Piotr Jkedrzejowicz, Bogdan Trawiński, and Gottfried Vossen, editors, *Computational Collective Intelligence*, pages 401–410, Cham, 2017. Springer International Publishing.
12. D. Aksu and M. Ali Aydin. Detecting port scan attempts with comparative analysis of deep learning and support vector machine algorithms. In *2018 International Congress on Big Data, Deep Learning and Fighting Cyber Terrorism (IBIGDELFT)*, pages 77–80, 2018.
13. Bruce Hartpence and Andres Kwasinski. Combating tcp port scan attacks using sequential neural networks. In *2020 International Conference on Computing, Networking and Communications (ICNC)*, pages 256–260. IEEE, 2020.
14. Yulong Wang and Jiuchao Zhang. Deepport: Detect low speed port scan using convolutional neural network. In *International Conference on Bio-Inspired Computing: Theories and Applications*, pages 368–379. Springer, 2018.
15. Soman KP, Mamoun Alazab, et al. A comprehensive tutorial and survey of applications of deep learning for cyber security. 2020.
16. Hung Nguyen Viet, Quan Nguyen Van, Linh Le Thi Trang, and Shone Nathan. Using deep learning model for network scanning detection. In *Proceedings of the 4th International Conference on Frontiers of Educational Technologies*, pages 117–121, 2018.
17. Navaporn Chockwanich and Vasaka Visoottiviseth. Intrusion detection by deep learning with tensorflow. In *2019 21st International Conference on Advanced Communication Technology (ICACT)*, pages 654–659. IEEE, 2019.
18. Razan Abdulhammed, Miad Faezipour, Abdelshakour Abuzneid, and Arafat AbuMallouh. Deep and machine learning approaches for anomaly-based intrusion detection of imbalanced network traffic. *IEEE sensors letters*, 3(1):1–4, 2018.
19. Gabriel C Fernández and Shouhuai Xu. A case study on using deep learning for network intrusion detection. In *MILCOM 2019-2019 IEEE Military Communications Conference (MILCOM)*, pages 1–6. IEEE, 2019.