ADRISYA: A FLOW BASED ANOMALY DETECTION SYSTEM FOR SLOW AND FAST SCAN

Muraleedharan N and Arun Parmar

Centre for Development of Advanced Computing (C-DAC)
Electronics City, Bangalore, India
{murali,parmar}@ncb.ernet.in

ABSTRACT

Attackers perform port scan to find reachability, liveness and running services in a system or network. Current day scanning tools provide different scanning options and capable of evading various security tools like firewall, IDS and IPS. So in order to detect and prevent attacks in the early stages, an accurate detection of scanning activity in real time is very much essential. In this paper we present a flow based protocol behaviour analysis system to detect TCP based slow and fast scan. This system provides scalable, accurate and generic solution to TCP based scanning by means of automatic behaviour analysis of the network traffic. Detection capability of proposed system is compared with SNORT and result proves the high detection rate of the system over SNORT.

KEYWORDS

Scan detection, Flow, IPFIX, Anomaly, Entropy

1. INTRODUCTION

Current day security attacks like malware, worm and botnet happens through multiple stages. In the initial stage, an attacker tries to understand the livness, reachability and running services in the system and vulnerabilities in it. Once attacker identifies these details, he can accurately plan the attack and get maximum benefit out of it with less probability of attack detection. So from security perspective, it is very important to detect the scanning attempt of a system or network accurately with the identity of attacker and victim. Since scanning is the first stage of an attack, if we can detect it properly in real time, multi stage attack prevention can be done through the scan detection. But, nowadays the sophistication of scanning tools are increasing and by using a single tool itself, an attacker can conduct different types of scanning on a network or system. Moreover, some of the scanning tools provide features for evading firewall rules or sneaking past intrusion detection or prevention systems [1].

In this paper we present a flow based port scan detection technique which provides a generic solution for different types of TCP scan and detects both slow and fast scan. Reasons for selecting TCP based scanning for our considerations are, firstly, TCP works in connection oriented mode, and therefore provides high accuracy in scan results and due to this advantage, attackers prefer TCP based scanning over UDP; Secondly all scanning tools provide TCP scanning as a defacto scan; Thirdly, compared to UDP based scanning, number of options available with TCP scanners are more, which can provide more information about the victim.

We are approaching port scan detection problem through transport layer protocol behaviour analysis. Scanning tools make use of the RFC definition of protocol for identifying the status of the port. So a generic approach to detect scan using different protocols (TCP and UDP) may generate false alarms. This system collects flow data as input for the analysis and defines flow records using IPFIX protocol [2]. The advantage of IPFIX over other flow definitions is that, the

DOI : 10.5121/ijnsa.2010.2417

user defined parameters can be incorporated in IPFIX flow definition. By using this method, system identify the type of scan, attacker, victim, attack time and other useful information.

The remainder of this paper is organized as follows. The related works are introduced in section 2. Section 3 explains the architecture and detection techniques of our system. In section 4, the experiment set-up details are described and results are explained in section 5. Result analysis is done in section 6. Section 7 concludes this paper.

2. RELATED WORK

Researchers have proposed different techniques for detecting scan activities in a network or a system. Allman et al [3] examine the scanning phenomenon in time dimension and describes a method for scan detection by means of connection and host classification. Their approach is based on the notion that connection attempts that do not result in established connection may be a possible scan. Our detection technique is also based on this concept, but parameters for the detection mechanism are very much different. Moreover for scan detection, we do not perform any classification based on connection or host. Jung et al [4] developed a technique known as Threshold Random Walk (TRW) based on sequential hypothesis for testing fast scan. This technique is limited to the detection of fast scan alone but our method addresses both fast and slow scan detection. A stealthy port scan detection mechanism is explained in [5] by storing the anomalous packets. This system works on packets and anomalous packets are stored for a long period for identifying slow scan, which may create processing overhead in current day high speed networks. Since proposed system works on flow data, compared to packets the volume of data for analysis is less and it is suitable for high speed network.

Recently, instead of packet based analysis, flow based security analysis and attack detection are gaining attention from researchers. Quyen at al [6] explains a port scan technique by considering small volume flows. Myung-Sup Kim et al [7] also uses small sized flows for scan detection. But merely relying on small volume flows for scan detection can miss some scanning attacks if the attacker changes the size of the packets. Also some of the scanning tools provide option to change the size of scan packets. In [8], a flow based aggregation algorithm is used for identifying the distribution in the cluster. The algorithm relies on information-theoretic techniques and identifies the clusters of attack flows in real time and aggregates those large number of short attack flows into a few meta flows. Another work based on flow monitoring is explained in [9] which work on monitoring the four predefined metrics that capture the flow statistic of the network. This method is capable to detect UDP flood, ICMP flood and scanning, by using Holt-Winters Forecasting technique. This technique makes projection about future performance based on historical and current data of the network. The prediction which comes out by this technique may arise false alarms because the network behaviour is not static.

Myung et al [10] suggests that by aggregating packets of the identical flow, one can identify the abnormal traffic patterns that appear during an attack. They formalize detection function for attack detection, which are composed of several traffic parameters and constant values. Our system computes threshold by providing weight after considering the deviation in the values of parameters because of which results will be more accurate. Entropy estimation is a general technique but recently, the use of entropy has received a lot of attention and is suggested for fine-grained detection of abnormal behaviour [11, 12, 13]. Our system also uses entropy based techniques for slow scan detection. For detecting network scan, researchers also use probabilistic approaches [14]. However, Attackers can reduce the likelihood of detection by spreading the scan for long period.

3. PROPOSED SYSTEM

3.1. Objective

The design objectives of our system are diversity, scalability and accuracy.

Diversity In general, scanning attacks can be categorized into two types. Horizontal scan and Vertical scan. In horizontal scan, an attacker collects the details about different systems in the network and he may be interested in a specific service in the network like HTTP, SMTP, DNS, etc. In vertical scanning, attackers can target a critical system and try to identify all the services in that system. According to the delay between two consecutive scan packets, scanning can be classified into slow scan and fast scan. In slow scan the time difference between two consecutive scan packets will be high. Since slow scan does not create any deviation in the normal traffic, detection of this scan through anomaly and real time detection is very difficult. In fast scan an attacker will try to scan the entire port in a system or network in a short period. That creates changes in the normalcy of the traffic. One of the design objective of our system is to provide a generic solution to horizontal, vertical, slow and fast scan attacks.

Scalability Since the network traffic volume increases with time, the scan detection techniques also has to support high speed network. By means of parallel scanning techniques, scanning tools are capable of conducting very high number of port scan in short span of time. In this context, real time detection of scanning activity helps us to prevent the subsequent stages of an attack. But detection of scanning activity in a high speed network is a challenging task. To incorporate scalability, instead of conventional techniques like packet based analysis, we are considering flow data as input for scan detection.

Accuracy Another design goal of this system is to detect the scanning activity accurately. Most of the current day scanning tools support different scanning options using the same transport layer protocols. In this context, to identify the scanning attack accurately, we believe that, protocol based approach is the suitable one. Our system addresses the detection of TCP based scanning activities.

3.2. Architecture

Figure 1. Architecture Diagram

As a generic anomaly detection system, this system also works in a profile and detection mode. In profile time, the system identifies the normalcy of the traffic and derives a base line (threshold) for normalcy. In detection time, system calculates threshold using real time data and compares the calculated threshold with profile time threshold. Figure 1 illustrates the architecture of ADRISYA. Detailed description of each component is explained below.

Input: Network traffic flow plays an important role in network monitoring and security, both for anomaly detection and corresponding defense. Flow can be defined as uni-directional sequence of packets from source to destination for particular time duration with same protocol and port number [15]. Compared to packet based analysis, flow data have the advantage of less volume. So from performance perspective, flow data can provide faster responses and real time

analysis is also possible through flow data. Due to the lesser volume, flow based analysis is very much suitable for high speed network monitoring and analysis. Flow data can be used for different granular level of analysis like host, network, application and time. Another aspect of flow data is that, by means of protocols like IPFIX, customized flow definition can be possible. We uses flow data as the input of this system and to export the flow data from flow probe to collector IPFIX protocol is used.

Since this system addresses only TCP based scan detection, from the received flow data, using protocol field, it filters out TCP flows. Due to the unidirectional property of flow, for every TCP connection two flows are available, one from sender to receiver and another from receiver to sender. Every TCP flow consists of flow start time, end time, duration, source IP, destination IP, source port, destination port, number of packets, number of bytes and the cumulative sum of TCP flags available in the packets.

Profiler: Traffic profiler collects the flow data and identifies the normal traffic patterns of the traffic. In the initial stage, system will be in profile mode and after the profile interval (γ) the system moves into detection mode. The profile period duration should be long enough to capture the entire network traffic behaviour in a network. Profile interval is sub divided into profile periods (δ). In each profile period, the system calculates the threshold values and at the end of profile interval, average of those thresholds and standard deviation is calculated for fixing the final threshold. The profiler component has two subsections in it, Short Term Profiler (STP) and Long Term Profiler (LTP).

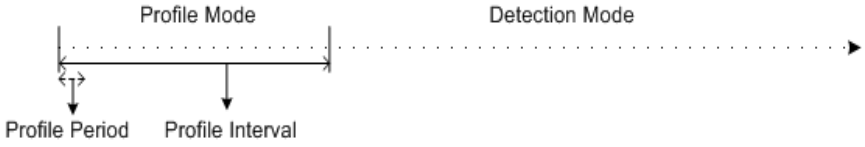

Figure 2. Short Term Profiler

Short Term Profiler (STP) STP profiles traffic behaviour to detect fast scan activities in a network or host. Fast scan attacks last for a short period of time and it makes deviation in the normal traffic pattern. STP works with flow duration, number of packets in a flow, average packet size, number of flows and count of single packet flows to fix baseline (T2) for fast scan. Short term profiler uses a time based profile technique to fix the threshold values for fast scan. Profile interval of STP is subdivided into profile period with fixed time duration. After every profile period, STP analyzes the dataset and calculate the threshold. In detection time, after every profile period, STP calculates the threshold for current dataset and compares it with profile time threshold.

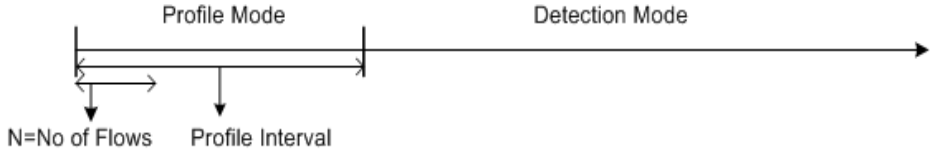

Figure 3. Long Term Profiler

Long Term Profiler (LTP) Since the slow scan attacks last for long duration, to detect it accurately long term traffic profiling is required. To identify the slow scan activities, LTP uses a count based profile technique. Independent of time, LTP waits for 'n' flows and if it receives 'n'

flows then checks for slow scan activities in those data set. The reason to select count based profile for long term profiler is that, firstly, network traffic is dynamic in nature. So if we set a fixed time interval for long term profiler, with in that time period, traffic behaviour can be different. Secondly, since we use entropy based method for detecting slow scan activities, if the number of records are fixed in the dataset, the maximum entropy value will also be same for all data set. Thirdly, long term profiler stores data for a long period to find the behaviour changes. So if it is a time based profile, due to the dynamic nature of network traffic, number of data records is different. Hence, required storage space is also variable. LTP takes source IP, destination IP, source port, destination port and packet size as parameters and applies an entropy based method for setting the baseline (T1) for slow scan. In our system, during detection time, STP check for fast scan and if it not finds any scan activities then only LTP considers those data set for updating.

Anomaly Detector: Once the profile interval is over, system moves into the detection mode. In detection mode, for every δ time, using current traffic, system calculates the threshold values and compares it with profile time threshold. If the threshold value changes with profile threshold, anomaly detector analyzes the data and identifies the scanning activities. As mentioned in the profiler, anomaly detector is categorized in to two types, Short Term Anomaly Detector (STAD) and Long Term Anomaly Detector (LTAD). STAD process the data and after every δ time it will check for fast scan activities. LTAD detects slow scan activities by means of higher δ value and related parameters.

Since anomaly detector analyze the dataset only after identifying changes in the threshold, we consider only minimal parameters for anomaly detection. Once it detects changes in the threshold, more analysis is conducted on the data set which is collected for those time intervals.

Data storage: This component take care storage of the threshold values and intermediate result like profile interval threshold values and current flow data. The detailed analyses of flow data is required only if current threshold is higher than the profile threshold, so after every δ time we can purge the flow data. Since the δ value is high for LTAD, the storage space required for LTAD will be more but instead of storing all the flow records for long period, we are storing only the relevant flow records for slow scan detection. More details of this are explained in slow scan detection section.

3.3. Detection Methods

TCP protocol uses three-way handshake procedure to establish a connection [16]. Since data transfer can be possible only through the established connection, for any successful TCP communication need to have more than three packets. But in the scanning time, an attacker's intention is only to verify status of the port, and therefore most of the connection will terminate before the completion of three way handshake process [17]. Moreover, if three-way handshake does not happen, connection details cannot be identified by log analysis.

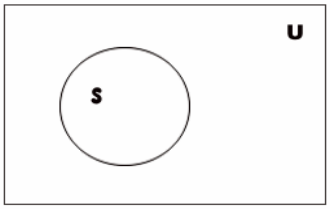

Figure 4. Single Packet Flow Vs Scan Flow

Due to the unidirectional property of the flow, every connection will create two flows and each flow will contains at least one packet in the connection establishment process itself. After the connection establishment, data transfer and/or connection release has to be done. So the total number of packets in a proper TCP flow will be more than one. But in scanning time, since proper handshake process is not done, each flow will have single packets only ie, for every scanning attempt it will create two flows with single packets. Single packet TCP flow can be generated due to different other reasons like inactive time of the flow and TCP keepalive feature [18]. Figure 4 represents these options. U is the universal set of single packet flows and S is the scan flows and $S \leq U$. In scanning time number of S will be high and $S \approx U$.

Fast Scan Detection: In our previous work, [19], we consider single packet flow as the main parameter for scan detection, because at the time of scanning single packet flow rate will be high. Since scan tools knock more ports on a system or more number of machines in a network, the number of flow increases in scan time. Other than these two parameters, we consider average packet size, average number of packets in a flow and average flow duration for scan detection. Due to the dynamic nature of network traffic, the number of flows can vary from time to time. So if we are considering the number of flow as a parameter for setting a common threshold for different time periods, it can lead to false positives. To avoid the possibilities of these false alarms we are considering a threshold setting method which is independent on the number of flows. In our approach we are considering the percentage of single packet flows in total flows.

Table 1. Fast scan detection Result

Parameter	**Behaviour**		**Mean**	**Std**	**Weight**	**Total**
	Normal	**ScanTime**				
% of Single packet Flow	Low	High	μ1	σ1	W1=.60	T1
Average packet size	High	Low	μ2	σ2	W2=.15	T2
Average no of packets in a flow	High	Low	μ3	σ3	W3=.15	T3
Average flow duration	High	Low	μ4	σ4	W4=.10	T4

The behavioural changes of different parameters on scan time and the impact on those in the detection is depicted in Table 1. For setting the threshold, we calculate the corresponding deviations in the parameters values. To reduce the possibility of occurrence of false alarm, we are providing different weight to parameters. During scan time, the percentage of single packet flow is very high compared to the normal traffic and because of that we have assigned 60 percent weight to that parameter. Weight distributions between other parameters are, 15 for average packet size, 15 for average number of packets in a flow and 10 for flow duration. Based on profile period data, the system learns the network behaviour and keeps track of the required parameters for setting the threshold. The formula for setting threshold for fast scan detection is given as

(1)

$$\text{Sum of averaged parameters } AVG = \sum_{i=1}^{4} \mu i$$

$$\text{Sum of standard deviation of parameters } STD = \sum_{i=1}^{4} \sigma i$$

$$Ci = \frac{\mu i + \sigma i}{AVG + STD}$$

$$Ti = Ci \times Wi$$

$$Threshold = T2 + T3 + T4 - T1$$

In detection time, the system calculates thresholds using current time flow data set. Equation 2 describes the steps to calculate the threshold values at detection time.

(2)

$$AVG = \sum_{1=1}^{4} \mu i$$

$$Ci = \frac{\mu i}{AVG}$$

$$Ti = Ci \times Wi$$

$$Threshold = T2 + T3 + T4 - T1$$

In scan traffic, the calculated threshold value will be less than the profile threshold because of the increase of single packet flow ratio (T1) will reduce the thresholds value. In normal traffic the detection time threshold will be greater than the profile threshold. Once we detect scan activities using the above formula, detailed inspection of flow dataset can be obtained for identifying type of scan, attacker and victim. This can be done in following way.

Step1. Filter out all flows which contain single packets.

Step2. Sort the flows based on TCP flag values in ascending order. This provides the scan type.

Step3. Sort the flows based on source IP field in ascending order. This provides the attacker IP if attack is done from single attacker.

Step4. Sort the flows based on Destination IP field in ascending order. This provides the victim IP of Vertical scan.

Step5. Sort the flows based on Destination Port field in ascending order. This provides the victim service of Horizontal scan

Slow Scan Detection In slow scan, the duration of scan process spans over a long period, so the data for detecting and analyzing the scan has to be preserved for long period. Our system stores only optimal data for detection and analysis of scan. As we shown in the Figure 4, scan traffic flows are subset of single packets flows. But scan type like connect scan, completes the three-way hand shake process as part of the scan. Even this type of scan can be identified using single packet flow data, because if it just establishes the handshake process, three packets are distributed into two flows so one of the flow have only single packet and other flow consists of two packets. This will cause an increase in the number of flows with single packet.

Long Term Profiler (LTP) identifies and stores all single packet flows for a profile period. Since the profile period value of LTP is more, it has a long term details of single packet flows. Since scan flow also have single packet in it, through single packet flow profiling, we can collect the scan traffic and number of single packet flows are comparatively very less in entire traffic flow. Hence it requires less storage space and less processing power. Another feature of this system is that once the fast scan detector detects scanning using Short Term Profiler (STP), flow records in that profile period will not be updated by Long Term Profiler. This can be done because the duration of the profile period of STP is less than that of LTP.

We are using entropy based detection techniques for identifying slow scan activities. In information theory, entropy is a measure of the uncertainty associated with a random variable and entropy of a random variable X is defined as $H(X) = -\sum_{i=1}^{N} p(x_i)\log_2(p(x_i))$ where $p(x_i)$ is the probability that X takes the value i. For standardizing the value of entropy between 0 and 1, we have taken normalized entropy using $\frac{H}{\log(N_o)}$ where N_o is the number of distinct x_i values present in the data set. To detect slow scan activities, we extract source address, destination address, source port, destination port and flow size from the single packet flow data set. After that we calculate the entropy values of those parameters.

4. EXPERIMENT

4.1. Test Set-Up

For this experiment, we collected data from a live network which has around 250 machines with different operating systems and application. This network is connected through a 2 Mbps link and Internet is accessed through a proxy machine. We have collected flow data from a gateway machine which is connected to the mirrored port of the switch. So, all incoming, outgoing and internal traffic can be accessed by the probe for flow creation. Once the probe creates flow, it export flow records to a collector machine. Collector machine keep track of flow data and analyze it for scanning activity. Proposed system and Snort [20] intrusion detection system are deployed in the port mirrored switch through a hub. So both the systems can access the same traffic which includes incoming, outgoing and internal traffic. The 'sfportscan' preprocessor of snort is enabled and configured to detect the scan activities.

We have identified two machines inside our network, one to initiate scan and other as victim of the scan. Time synchronization of the attacker machine and flow analyzer is done through the 'ntp' service. Using 'nmap' tool, we have done different TCP based scanning from attacker machine to victim machine. Once the flow analyzer detects the scanning, it generates an alert and after verification of the scanning activity, we calculate the detection delay. Similarly, we verified the scan detection capability of Snort using the generated alerts.

4.2. Tools Used

For generating flow data we have used an IPFIX library called 'libIPFIX'. This library consists of both flow probe and flow collector. We have written a program for the analysis of flow data which takes the input from flow collector and produces the analysis results. To conduct scanning activities, we used 'nmap' [21] scanning tools.

5. RESULT

Table 2. Fast scan detection Result

No	Scan Type	Scan Duration(sec)	Detection Delay(sec)	
			Proposed System	Snort
1	SYN	3.908	44	1
2	Connect	3.670	48	1
3	ACK	1.919	48	Not Detected
4	NULL	2.038	46	Not Detected
5	FIN	2.167	40	Not Detected
6	XMAS	1.979	50	Not Detected
7	OS Finger printing	5.849	48	1
8	Maimon	2.138	50	Not Detected
9	Window	1.966	45	Not Detected
10	Fast Scan	2.343	45	1
11	Data Length	6.000	44	Not Detected
12	Version Detection	109.642	49	1
13	Polite scan	670.304	50	Not Detected

To test detection capabilities of the system, we have configured profile interval of STP as one day and profile time period as 10 seconds. Flow probe exports flow records for every 10 seconds and flow analyzer checks for scan activity. Table 2 shows the summary of fast scan results. Using nmap tool, 13 different types of scan activities have done in a victim machine from the attacker machine. Scan type shows different types of scan details. Third column shows the time taken for scan activities in terms of seconds. Detection delay indicates difference between detection time and start time of the scan and last two columns shows the detection delay of proposed and snort respectively.

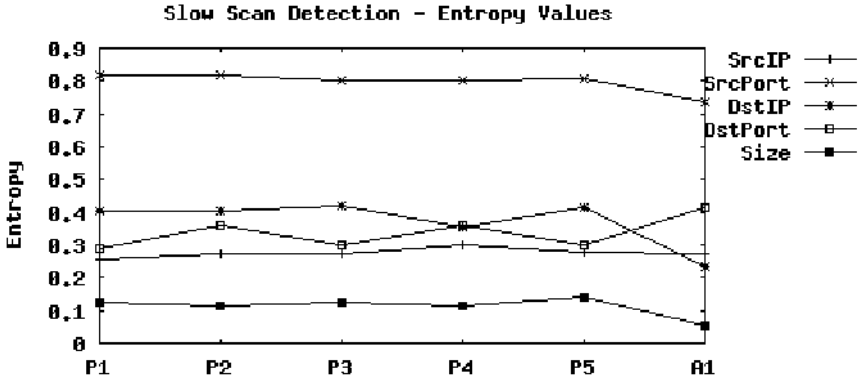

Figure 5. Slow scan detection

We configured the profile interval for slow scan profiler (LTP) for one day. Instead of setting profile period based on time, we have taken 20000 single packet flow records for a single profile period. In profile time, LTP collects a group of 20000 records and calculate the entropy

values for source IP, Destination IP, source port, destination port and packet size. Using slow scan option (-T sneak) in 'nmap' we conducts a slow scan and collects those scan data details form LTP. nmap sneak scan has taken 27529.794 seconds (7.64 Hours) to complete the scan in a single host. We have calculated the entropy values of the parameters from 20000 records of those scan data. Figure 5 summarizes the slow scan result of the proposed system. X axis shows the different data set of 20000 flow records where P1 to P5 shows the profile time data and A1 indicates the attack time data set.

6. RESULT ANALYSIS

6.1. Fast scan detection

From table 2, it is clear that the system capable to detect conventional TCP based scan like SYN, Connect and ACK scan. Another feature of this system is that, it detects stealthy scans like NULL, FIN and XMAS accurately even though those scans are capable to sneak through certain non-stateful firewalls and packet filtering routers. Since we are not depend on the packet size for scan detection, our system can detects scan using nonstandard size TCP packets , those are capable to evade security systems like firewall and IDS, like '–data length' option in nmap scan. 'nmap' tools provides a fine-grained timing control for scanning under the '-T' option. Polite scan is one scan available in that category which waits for 0.4 seconds between scan probes moreover, it is slower than default nmap scan. The result reveals that the proposed system has the capability to detect different types of scan over snort. Out of conducted 13 different scan, snort detects only 5 of them (detection rate 38.46). Compared to Snort, the proposed system has an advantage on stealth scan like XMAS, NULL, FIN. Scan using ACK flags in TCP header can be generally detected using state-full mechanism but proposed system is able to detect ACK scan.

6.2. Slow scan detection

Figure 5 shows the standardized entropy values of different parameters selected for slow scan detection. Except the source IP, all the other parameters clearly show the difference in entropy values during scan. Table 3 summarizes the impact of scan in entropy values to different parameters. Column 2 indicates the average standard entropy values of different parameters in normal traffic and column 3 shows the standard deviation. Fourth column shows the entropy values at scan time and last column indicates the difference in entropy values in normal and scan time. In scan time, by default, nmap uses same source port to send different packets. Since most of the flows have same source port, the entropy value of that parameter is less than that of normal traffic. During vertical scan, all the scan traffic goes to the same destination. So the entropy value of destination IP at vertical scan time should be less than normal traffic entropy value. Since we have done a vertical scan, scan traffic entropy value of destination IP is less. Destination port is another parameter which has a change in the entropy value during scan.

Table 3. Slow scan detection Result

Parameter	**Normal Traffic entropy**		**Scan Traffic**	**Difference**
	Average	**STD**		
Source IP	0.27480	0.01370	0.27012	-0.00468
Source Port	0.81019	0.00761	0.73403	-0.07616
Destination IP	0.39932	0.02310	0.23494	-0.16438
Destination Port	0.32155	0.03148	0.41293	0.09138
Packet Size	0.12361	0.01043	0.05446	-0.06914

In vertical scan, attacker probes different ports of same machine and it creates different flows with different destination port. So the entropy value of destination port is high in scan time. In normal data set, the entropy values of packet size are almost similar and it is more than 0.1. But in scan time, it is reduced in to 0.05446. Scan tools uses same sized packets for probing different port, hence it reduces the entropy value of packet size during scan time. Since we have done the scan from same machine, minor change in source IP is visible in the scan data. If it is a distributed scan, the source IP entropy also indicates the difference. Similarly, in horizontal scan, if attacker targets a specific service on the network, destination port entropy value will be less and destination IP entropy value will be more. From this result, we can conclude that, even in slow scan there are changes in the entropy values of destination port, packet size, destination IP and source port. By means of a proper profiling and threshold setting, we can identify the slow scan activities.

7. CONCLUSION

In this paper, we have presented a system for detecting TCP based scan activities using IPFIX flow. This system provides a generic, scalable and accurate method to detect TCP based fast and slow scan. Since the scanning behaviour is different for slow and fast scan, we have selected different methods and parameters for detecting slow and fast scan. Using the properties of flow definition, we have developed a method for identifying fast scan, and uses entropy based approach to detect slow scan activity. The experimental result shows that the system effectively detects different types of fast scan. Regarding slow scan, the system takes minimal storage and resources for detection and detects scan even the scan activity last for very long period.

ACKNOWLEDGEMENTS

This work is supported in part by a special grant from Department of Information Technology, Ministry of Communications and Information Technology, Government of India.

REFERENCES

[1] Firewall/IDS Evasion and Spoofing, Nmap Reference Guide, http://nmap.org/book/man-bypass-firewalls-ids.html

[2] RFC 5101, Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of IP Traffic Flow Information

[3] Allman, M., Paxson, V., Terrel, J.: A Brief History of Scanning. In: IMC 2007 Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement. ACM, New York (2007)

[4] Jung, J., Paxson, V., Berger, A.W., Balakrishnan, H.: Fast Port scan detection using sequential hypothesis testing. In: Proceedings of the IEEE Symposium on Security Privacy (May 2004)

[5] Staniford, S., Hoagland, J.A., McAlerney, J.M.: Practical automated detection of stealthy portscans. Journal of Computer Security 10 (2002)

[6] Quyen, L.T., Zhanikeev, M., Tanaka, Y.: Anomaly identification based on flow analysis. In: 2006 IEEE Region 10 Conference on TENCON 2006 (November 2006)

[7] Kim, M.-S., Kong, H.-J., Hong, S.-C., Chung, S.-H., Hong, J.W.: A flow-based method for abnormal network traffic detection. In: Network Operations and Management Symposium, NOMS 2004 (2004)

[8] Hu, Y., Chiu, D.-M., Lui, J.C.S.: Entropy Based Adaptive Flow Aggregation. In: IEEE/ACM (December 2007)

[9] Nguyen, H.A., Van Nguyen, T., Kim, D.I., Choi, D.: Network traffic anomalies detection and identification with flow monitoring. In: WCON 2008 (May 2008)

[10] Kim, M.-S., Kang, H.-J., Hong, S.-C., Chung, S.-H., Hong, J.W.: A Flow-based Method for Abnormal Network Traffic Detection. In: IEEE/IFIP Network Operations and Management Symposium (2004).

[11] Gu, Y., McCallum, A., Towsley, D.: Detecting anomalies in network traffic using maximum entropy estimation. In: Proc. IM 2005 (2005)

[12] Nychis, G., Sekar, V., Andersen, D.G., Kim, H., Zhang, H.: An empirical evaluation of entropy-based traffic anomaly detection. In: 8th ACM SIGCOMM Conference on Internet (2008)

[13] Wagner, A., Plattner, B.: Entropy based worm and anomaly detection in fast IP networks. In: 14th IEEE International Workshops on Enabling Technologies 2005(2005)

[14] Leckie, C., Kotiagiri, R.: A probabilistic approach to detecting network scans. In: 2002 IEEE/IFIP Network Operations and Management Symposium (2002)

[15] Introduction to cisco IOS netflow - a technical overview, http://www.cisco.com/en/US/prod/collateral/iosswrel/ps6537/ps6555/ps6601/prodwhite paper0900aecd80406232.html

[16] RFC793 - Transmission Control Protocol (September 1981)

[17] de Vivo, M., Carrasco, E., Isern, G., de Vivo, G.O.: A review of port scanning techniques. ACM SIGCOMM Computer Communication Review (April 1999)

[18] RFC1122 - Requirements for Internet Hosts – Communication Layers (October1989)

[19] Muraleedharan, N.: Analysis of TCP Flow data for Traffic Anomaly and Scan Detection. In: 16th IEEE International Conference on Networks (2008)

[20] Snort manual, http://www.snort.org

[21] nmap Reference guide, http://nmap.org.book/man.html

Authors

Muraleedharan N
Presently working as a senior staff scientist in Centre for Development of Advanced Computing (C-DAC), Bangalore, Electronics City. He received his Master of computer application degree from National Institute of Technology calicut. Past five years he has been engaged in teaching and research in the area of network security and current research area is flow based traffic analysis and anomaly detection.

Arun Parmar
Presently working as a project Associate in Centre for Development of Advanced Computing (C-DAC), Bangalore, Electronics City. He received his Master of science degree from Karnataka state open university. Past two years he has been working in network security and anomaly detection.