An Adaptive Log Based Realtime Network Scans in LAN

Puli Lakshmana Swami
Mtech (CSE)
Gudlavalleru Engineering College,
Gudlavalleru

Neela Surendrababu
Asst Professor
Gudlavalleru Engineering College,
Gudlavalleru

Abstract:

Network scans really are a common initial step within a network intrusion attempt. As a way to gain details about a possible network intrusion, it can be beneficial to analyze these network scans. Scanning activity is naturally a common activity over the internet today, representing malicious activity such as information gathering by the motivated adversary or automated tool on the lookout for vulnerable hosts (e.g., worms). Many scan detection techniques have been developed; however, their focus has been on smaller networks where packet-level information is available, or where internal characteristics of this very network have been observed. Existing approaches use scan detection technique as a way to identify of packet-level data between host pairs, and activities are identified by grouping sessions based on patterns inside the kinds of session, the IP addresses, plus the ports. For serious networks, which can include those of ISPs, large corporations or government organizations, people's information might not be available. Existing model offers a model of scans which can be used given only unidirectional flow data. Novel classification of scan detection methods based upon their network policy, since attackers usually takes benefit for such policies to evade detection. The intention of the proposed program is to analyze sample network traces to discover and classify properties of port scans using robust Probabilistic technique. Finally scan detection accuracy can possibly be analyzed by employing logistic regression approach that is used to model the post that informs a user if a scan is present. Experimental results will provides insight into Internet traffic by classifying known activities, giving visibility to threats to the network through scan detection, which correspondingly extending understanding of the activities occurring on the network.

Keywords –Attack, Scan,LAN.

I. INTRODUCTION

The process of scanning a network is normally performed to discover what exists throughout the network. Just for example, when an attacker is looking for exploitable web servers, then they would attempt to connect on TCP/UDP port 80 to each and every possible Internet protocol address in the next certain range. If there happen to be a web server using port 80 for any for these IP address, it certainly will probably respond. However, for addresses where there isn't anything, or where there is basically a computer that isn't operating a web server, you'll have no response. Detecting these scans is rather easy, but mining them for the required information regarding the attacker can possibly be relatively difficult.

An attack is capable of doing a number of things attempting to make this sort of scan anonymous; by way of example, run by the different source addresses or scanning destination addresses within the random order. The truth is, in fact it's even possible to do a scan in-directly by applying a fake source address so now the scan seems like in fact it's coming from a different computer. Denial of service and worm propagation attacks can also produce scan-like behavior, and since they do not need the target to respond, they often fake their source addresses also. The port number is likewise not sufficient for categorizing scans, because both malicious and benign scans might be appear on the very same port number. For example, both a web crawler plus a worm that targets webservers would target port 80. Therefore, a few other metric will have to be used for categorization purposes.

Experimental results have indicated that variations in arrival time of the scanning connections frequently have an extremely high correlation with particular sources. That is undoubtedly, the timing information will create digital 'fingerprint' that correlates to a particular source. In fact it's surmised the fact that correlation due to a combination of factors, that includes the connection application software employed, the supporting hardware plat- form, operating systems characteristics, and regular interference from different processes according to the source system that compete for these resources. Network location based factors such as number of hops and nature of intervening routers are certainly responsible for a level of the timing structure as well, and generate this a very interesting and challenging problem in network traffic forensics.

Clearly, as a result of the frequent changes among the address space usage of any network (e.g. dynamic IP assignments, laptops, hosts being replaced, etc), the longer all it takes is for an attacker to finish it the scan, the less truthful the data gathered in the beginning of this very scan is in comparison to the current state. Also, the topological information gathered from the attacker, could have a lifetime then becomes useless. For instance later on public announcement associated with a vulnerability, an attacker could possibly be interested in promptly locating all vulnerable hosts inside a protected network to attempt to compromise them. The time window for the attacker to perform the scan plus the following attack is the time needed by the system administrator to recognize the vulnerability, download a patch (if available) and mount it on several vulnerable hosts. Such policies are vital, because attackers try to scan stealthily and amnesty policies, if exploited maliciously, provides an approach for your attacker in order to make its behavior look normal. Consequently, amnesty policies certainly are a likely vector for evasive scan techniques and then we should recognize how these policies work and how they could be exploited. As a launching point, we propose a new classification of scan detection methods dependent on their amnesty policy. Pre intrusion activities are chosen to prepare for intruding into a network. Some of the most common are port scanning to discover a approach to getting into the network and IP spoofing to disguise the identity of one's attacker or intruder.

Port scans: A program utilised by hackers to probe a system remotely and determine what TCP/UPD ports are open (and sensitive to attack) is termed a scanner. A scanner can find a vulnerable computer on the Internet, discover what services are running on this appliance, after which choose the weaknesses in them services. There are actually 65,535 TCP ports and an equal wide range of UDP ports. Stealth scanners use what is called an IP half scan, sending only initial or final packets as a substitute for establishing a linkedin connection, to refrain from detection.

IP spoofing: This is usually a way to changing the instructions among the headers regarding a packet to forge the original source IP address. Spoofing is made use of to impersonate an alternative machine that are caused by the one which actually sent information. You are able to do this to stay clear of detection and/and to target the machine to which is something spoofed address belongs. By spoofing an address that's a trusted port, the attacker could possibly get packets with a firewall. Port scans can be mainly classified into vertical scan and horizontal scan. An attacker does a horizontal scan when also has an exploit for a service and is actually interested by seeking a host that exposes the site . Vertical scan is when the attacker scans some or all ports using one host.

Other scan types include:

1. TCP connect() scan is where the attacker attempts a three way handshake upon the target. If it succeeds, the port is open. This scan type is not just regularly used seeing that it generates an accept() system call which may be logged
2. TCP SYN scan: In this sort of scan, the scanner generates just a first SYN packet. If he port is open, a SYN-ACK is received. Otherwise the target responds with a RST packet.
3. FIN and XMAS scans: FIN scan sends a packet onto the target host with FIN flag set. XMAS scan sends a packet with all flags set. Have responds with RST packet if the port closes.
4. UDP scans: In situations of UDP scans, host fails to respond if port is open. In case the port isn't open, te host sends back ICMP Destination Unreachable because Port Unreachable message. Wide range of addresses mandatory to complete the scan. The 2nd metric will be the large number of IP addresses that the attacker must complete the scan within the presence of the scan detection method, using a specific scan technique. Just for instance, for a Timeout-based method, offered the attacker has no time constrains, the attacker must have a single IP address to finish up the scan, simply because it just has got to try using a scan rate below the lowest detected through method.We call the evasive scan technique of probing for a rate below the minimum detected using a method, favour scan.

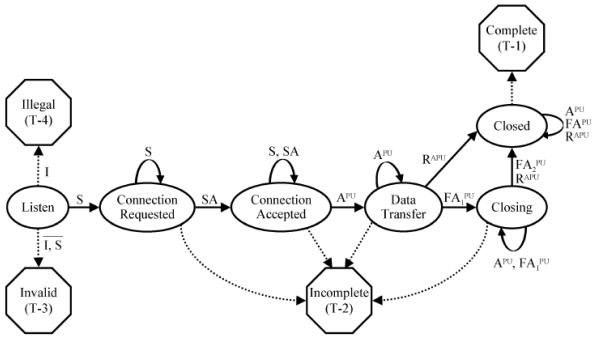
II. LITERATURE SURVEY

Several researchers have examined the detection of port scans, resulting in a variety of definitions. In general, a scan is a reconnaissance technique used to determine the existence of multiple targets, where the targets could be hosts or particular services on particular hosts. Scans can be used both by adversaries in order to determine what to attack and by system administrators to audit their network. Scans also occur from system mis-configurations, as well as from peer-to-peer applications searching for previously contacted peers. Scans can be a side effect of vulnerability searches, such as by vulnerability scanners or worms.

However, the threshold random walk (TRW) approach to detecting port scans developed by Jung *et al.* has become the gold standard for scan detection, and has been used for activities such as worm detection and quarantine . Their approach uses sequential hypothesis testing, where each new connection request from an external source is evaluated. If the destination exists, then there is more support for the source being benign (or, rather, not scanning). However, if the destination does not exist, then there is more support for the source to be scanning.

Leckie et al. [31] use probabilistic models of port connection information in order to detect scanning activity. These models require that each host in the local network be assigned a probability of how likely a given remote host will try and connect to it. In essence, the model assigns an access distribution probability for each host in the local network in terms of the number of remote hosts that have tried to make a connection to the

specific local host. This technique relies on the premise that remote scanning hosts will access local hosts with equal probability during a scan. Thus, the distribution of scanning systems and benign system (i.e non-scanners) should exhibit a strong modality that will allow scanning systems to be identified by their connection patterns. This technique has some limitations. Specifically, if the local network contains a small number of busy systems (e.g. servers), the probability distribution for less busy systems (e.g. clients) may be characterized too low and introduce a high rate of false positives. To remedy this, the method used to assign probabilities to individual systems should take this scenario into account. Additionally, as with any technique that relies on a training period, any ongoing scanning activity present in the training dataset will skew the probability distribution of the model for the local network incorrectly base-lining some scanning activity as legitimate connection behavior.



TCP session state machine. Dotted lines represent transition to a final session class. If a dotted line is unlabeled, the transition occurs after a specified interval of inactivity.

Symbol	Flag set	Event description
S	{S}	Request to open connection.
SA	{SA}	Agree to open connection.
A^{PU}	{A, PA, AU, PAU}	Acknowledgment of receipt.
FA^{PU}	{FA, FPA, FAU, FPAU}	Request to close connection.
R^{APU}	{R, RA, RP, RU, RPA, RPU, RAU, RPAU}	Tear down connection.
I	The set of all other flag combinations	Illegal.

Classes of TCP Session

T-1) Complete: A session that behaves according to expectations: opens with a three-way handshake, data is transferred, followed by a closing (graceful or otherwise).
T-2) Incomplete: A session that starts off properly, but either does not terminate, terminates too soon, or does not transfer data.
T-3) Invalid:Asession that begins with an invalid packet (i.e., not a SYN packet and not a packet with an illegal flag combination).
T-4) Illegal: A session that contains a flag combination that is illegal as per Table I, e.g., SF, FPU, null, or a session in which the flags arrive out of order, e.g., a SYN followed by a FIN.

Classes of UDP Session

Since UDP is a connectionless protocol , the state machine is very simple; only the directions of UDP packets in a session need to be tracked. There are two classes of UDP session.
U-1) Unidirectional: The packets flow in only one direction.
U-2) Bidirectional: The packets flow in both directions.

Classes of ICMP Session

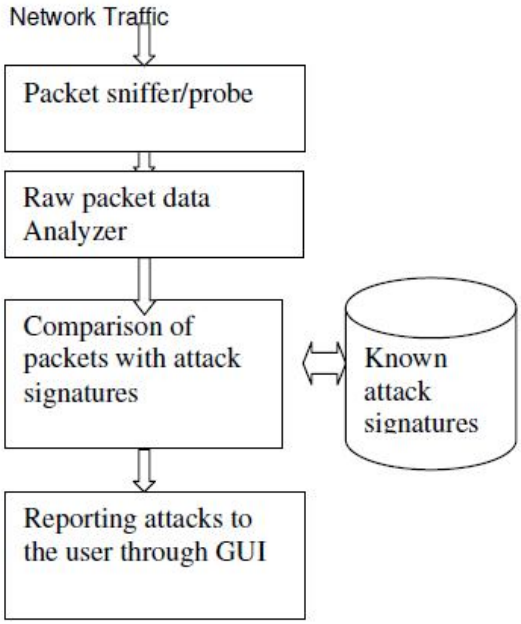
ICMP is used both to request information and to relay error messages . The concept of an ICMP session is different for the two cases. The state machine model is shown in Fig. 2. In the diagram, Q represents an ICMP query event, R represents an ICMP reply event, and E represents an ICMP error message that has no corresponding TCP or UDP session. All ICMP sessions start in a "Listen" state. The four classes of ICMP session are the following.
I-1) Request: An ICMP information request without a reply.
I-2) Pair: An ICMP information request and reply.
I-3) Lone reply: An ICMP information reply with no associated
request.
I-4) Lone error: An ICMP error message with no associated
session.

QueSO	Class	Nmap	Class
S	T-2	S12 (open port)	T-2
SA	T-3	null(open port)	T-4
F	T-4	SFPU (open port)	T-4
FA	T-3	A (open port)	T-3
SF	T-4	S (closed port)	T-2
P	T-4	A (closed port)	T-3
S12	T-2	FPU (closed port)	T-4

The pcap data analyzed in this work was sniffed from outside of the border of a quadruple-class-B network. It spans 24 hs and contains 82 476 239 TCP, UDP, and ICMP packets.

III PROPOSED ALGORITHMS

IMPROVED TCP UDP AND ICMP CLASSIFICATION steps:



For the TCP protocol, we identified 21 different variables as being possible indicators of whether the event contained scanning activity. These variables are:
1. maximum /24 subnet run length,
2. ratio of flows that do not have the ACK bit set to all flows,
3. ratio of flows to known malware ports to all flows,
4. ratio of flows with fewer than 3 packets to all flows,
5. maximum run length of IP addresses in any one /24 subnet,
6. maximum number of IP addresses contacted in any one /24 subnet,
7. maximum number of high destination ports contacted on any one host,
8. maximum number of low destination ports contacted on any one host,
9. maximum number of consecutive high destination ports contacted on any one host,
10. maximum number of consecutive low destination ports contacted on any one host,
11. number of unique destination IP addresses,
12. number of unique source ports,
13. average number of source ports per destination IP address,
14. ratio of flows with "standard" flag combinations (SYN and ACK set, along with either the FIN or RST bit set) to all flows,
15. ratio of the number of flows with the average bytes/packet > 60 to all flows,
16. median value of packets per destination IP address,
17. ratio of flows with "standard" combination (standard flag combination and at least three packets and at least 60 bytes/packet on average) to all flows ,
18. ratio of flows with backscatter combination (RST, RST-ACK, or SYN-ACK for the flag combination and the average number of bytes/packet is $\leq$ 60 and the number of packets per flow is $\leq$2) to all flows,
19. ratio of unique destination IP addresses to number of flows,
20. ratio of unique source ports to number of flows, and
21. ratio of flows with backscatter flag combinations (SA—RA—R) to all flows.

$$\hat{P}(\text{event contains a scan}) = \frac{e^{\hat{y}}}{1+e^{\hat{y}}}$$

where

$$\hat{y} = -2.83835 + 3.30902x_2 - 0.15705x_4 - 0.00232x_{13} - 1.04741x_{15} + 3.16302x_{19} - 3.26027x_{21}$$

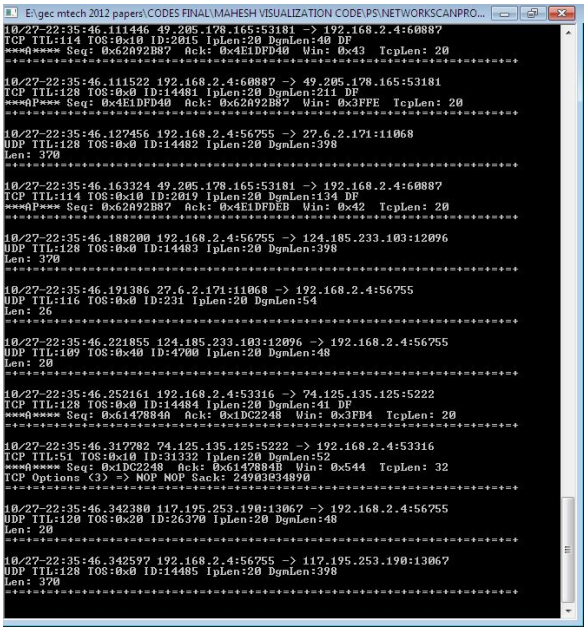
where x_2 is the ratio of flows with no ACK bit set to all flows, x_4 is the ratio of flows with fewer than three packets to all flows, x_{13} is the average number of source ports per destination IP address, x_{15} is the ratio of the number of flows that have an average of 60 bytes/packet or greater to

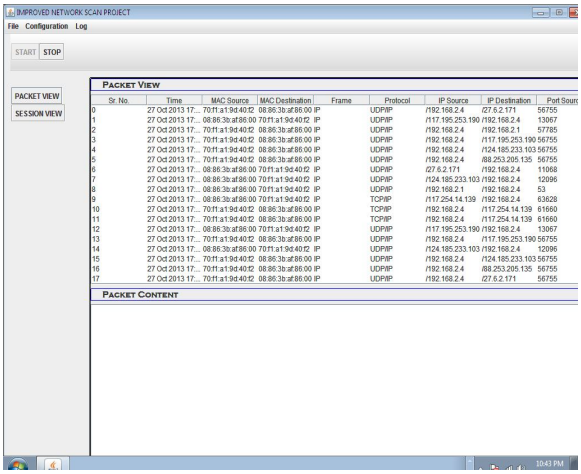
Let D={d1,d2…dn} denote the set of destinations in our target network, such that each destination has been accessed by an external source. In addition, let Di denote the subset of destinations that have been accessed by the source Si. Similarly P={p1,p2…pn} and Pi represent the corresponding sets of ports that have been accessed in the target network. For each source Si, our aim is to quantify a measure of how suspicious are the accesses from that source. We consider two main factors: (1) how unusual is each individual destination or port that has been accessed by Si, and (2) how many destinations or ports have been accessed by Si

1: if {(src_ip) is a local network address}
2: if {(dest_ip) not in scan_sources}
3: and {dest_port not in triggered_outbound_services}
4: and {state <> OTH}
5: add (dest ip) to friendly remotes
6: return false
7: endif
8: endif
9: if {(src_ip) in scan_sources}
10: return true
11: endif
12: if {state is one of S0, REJ, OTH, RST0}
13: established := false
14 endif
15: if {not established}
16: if {(src_ip, dest_ip) not in failed_locals}
17: flag := 1
18: add (scr ip, dest ip) to failed locals
19: ratio := (1- theta_one)/(1- theta_zero)
20: endif
21: elseif {(scr_ip, dest_ip) not in successful_locals}
22: flag := -1
23: add (src ip, dest ip) to successful locals
24: ratio := theta_one/theta_zero
25: endif
26: if {flag = 0}
27: return false

28: endif
29: updated_lambda[src_ip] := lambda[src_ip] * ratio
30: if {updated_lambda[src_ip] > eta_one}
31: add (src ip) to scan sources
32: raise alert scan detected
33: return true
34: endif
35: if {updated_lambda[src_ip] < eta_zero}
36: add (src ip) to benign sources
37: return false
38: endif

VI. RESULTS





Client Ip :/173.231.152.115
Server IP :/192.168.2.4
Client Port :80
Server Port :61699
StatusEstablished
DATA:*HTTP/1.1 200 OK
Content-Length: 287
Content-Type: application/json; charset=utf-8
Server: Microsoft-HTTPAPI/2.0
Date: Sun, 27 Oct 2013 17:13:39 GMT

{"GetUserResult":{"LicenseInfo":{"__type":"LicenseInfo:#ObjectModel","ApprovalsAllowedPerWeek":20,"DaysLeftToExpiry":9911,"GracePeriodInDays":2,"LicensePolicy":"{feature:accepts, limit:50,duration:30, expiration:none}","ProductType":0},"UserName":"0ffc66f7-b1d3-4955-9d17-59bc7bbcb708"}}----------

---

Client Ip :/192.168.2.4
Server IP :/173.231.152.115
Client Port :61699
Server Port :80
StatusEstablished
DATA:*----------

Client Ip :/173.231.152.115
Server IP :/192.168.2.4
Client Port :80
Server Port :61699
StatusEstablished
DATA:*HTTP/1.1 200 OK
Content-Length: 287
Content-Type: application/json; charset=utf-8
Server: Microsoft-HTTPAPI/2.0
Date: Sun, 27 Oct 2013 17:13:39 GMT

{"GetUserResult":{"LicenseInfo":{"__type":"LicenseInfo:#ObjectModel","ApprovalsAllowedPerWeek":20,"DaysLeftToExpiry":9911,"GracePeriodInDays":2,"LicensePolicy":"{feature:accepts, limit:50,duration:30, expiration:none}","ProductType":0},"UserName":"0ffc66f7-b1d3-4955-9d17-59bc7bbcb708"}}----------

---

Client Ip :/192.168.2.4

Server IP :/173.231.152.115

Client Port :61699

Server Port :80

StatusEstablished

DATA:*----------

Client Ip :/192.168.2.1

Server IP :/192.168.2.4

Client Port :80

Server Port :61774

StatusEstablished

DATA:*HTTP/1.0 200 OK

Server: F7D1301-v1/1.0 UPnP/1.0

Date: Sun, 27 Oct 2013 17:13:40 GMT

Connection: close

SID: uuid:c3f840de-1de4-11b2-b951-08863baf8600

Timeout: Second-1800

V. CONCLUSION AND FUTURE SCOPE

The detection mechanism investigated is seen to accurately detect port scans. The system uses counts of control packets to estimate the posterior probabilities. So the behavior of the detection system is stable with respect to data from various machines and levels of user activity. The system is also computationally faster since only probability estimates need to be made for both classes. The learning phase is faster since learning involves the calculation of prior probabilities. Future work involves a comparison of the detection technique with existing detection algorithms. The detection system considers only SYN scanning. Other scan types can also be detected based on this technique.

REFERENCES

[1] A Network Activity Classification Schema and Its Application to Scan Detection Joanne Treurniet , IEEE/ACM TRANSACTIONS ON NETWORKING, VOL. 19, NO. 5, OCTOBER 2011

[2] Stefan Axelsson. The base-rate fallacy and the difficulty of intrusion detection. ACM Transactions on Information and System Security, 3(3):186 – 205, 2000.

[3] G. Elfving. Optimum allocation in linear regression theory. The Annals of Mathematical Statistics, 23(2):255 – 262, 1952.

[4] Andrew Gelman, John B. Carlin, Hal S. Stern, and Donald B. Rubin. Bayesian Data Analysis. Chapman and Hall, 1995. ISBN 0-412-03991-5.

[5] Simon Jackman. Estimation and inference via bayesian simulation: An introduction to markov chain monte carlo. American Journal of Political Science, 44(2):375 – 404, 2000.

[6] Jaeyeon Jung, Vern Paxson, Arthur W. Berger, and Hari Balakrishnan. Fast portscan detection using sequential hypothesis testing. In Proceedings of the 2004 IEEE Symposium on Security and Privacy, pages 211 – 225, Oakland, California, USA, 2004. IEEE Computer Society. May 9-12, 2004.

[7] C. Leckie and R. Kotagiri. A probabilistic approach to detecting network scans. In Proceedings of the 2002 IEEE Network Operations and Management Symposium, pages 359 – 372, Florence, Italy, 2002. April 15-19, 2002.

[8] Chris Muelder, Kwan-Liu Ma, and Tony Bartoletti. Interactive visualization for network and port scan detection. In Proceedings of 2005 Recent Advances in Intrusion Detection, 2005. September 7-9, 2005.

[9] Susmit Panjwani, Stephanie Tan, Keith M. Jarrin, and Michel Cukier. An experimental evaluation to determine if port scans are precursors to an attack. In Proceedings of the 2005 International Conference on Dependable Systems and Networks, pages 602 – 611, Yokohama, Japan, 2005. June 28-July 1, 2005.

[10] Vern Paxson. Bro: A system for detecting network intruders in real-time. In Proceedings of the 7th USENIX Security Symposium, 1998. San Antonio, Texas. January 26-29.

[11] Seth Robertson, Eric V. Siegel, Matt Miller, and Salvatore J. Stolfo. Surveillance detection in high bandwidth environments. In Proceedings of the 2003 DARPA DISCEX III Conference, pages 130 – 139, Washington, DC, 2003. IEEE Press. 22-24 April 2003.