Network Monitoring Tool to Identify Malware Infected Computers

Navpreet Singh
Principal Computer Engineer
Computer Centre, Indian Institute of Technology Kanpur, India
navi@iitk.ac.in

Megha Jain, Payas Gupta & Shikha Bansal
3rd Year UG Students of LNMIIT, Jaipur, India

ABSTRACT

These days most of the Organizational Networks are facing a critical problem. Lately there has been a lot of increase in Malwares such as worms, adwares, spywares etc., which get installed on the users PC and generate Network and Internet traffic without the user's knowledge i.e. in the background. As a result, the overall utilization of the network, specially the Internet link, gets drastically minimized due to this unwanted traffic.

This tool monitors the network traffic and identifies all the (active) computers on the network which are infected with any kind of Malware. It provides the IP Address, MAC Address and type of infection for the identified hosts. There may be some hosts for which may it not be able to provide information on the type of infection, but it is able to identify them.

1. Introduction

This is a freeware tool which works on Linux. It is a user-friendly shell script which detects the Malware infected machines on a local area network and informs the network administrator about them. The infections detected are in the form of worms, adwares etc. which block a large amount of the network bandwidth.

Tcpdump is used to take a trace of the network traffic and shell scripts are used to analyze the collected data. Signature packets in the collected traces are used to identify the type of infection. The monitoring needs to be done for only 5 minutes to capture all active infected machines. Several configuration parameters can be set as per requirement.

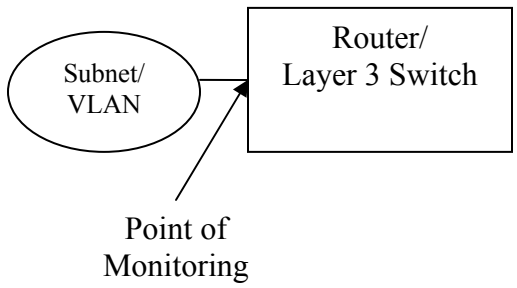
The tool can be installed on any Linux machine. To identify the infected hosts on any Subnet/VLAN, the traffic needs to be monitored on the mirror port of the Default Gateway for that Subnet/VLAN.

The ability of this tool to identify all the infected machines on the network as well as the infections in them sitting right on a network interface makes it a powerful and a distinguished Malware Detection Network Tool. This tool can also be set to run at regular intervals using cron and the reports generated can be sent to the administrator through email.

2. Description of the Tool

The tool is a Linux shell script which captures the network traffic using tcpdump [1] and analyzes the collected data after the capture is complete.

As shown in the figure, the tool can be applied at a mirror port of gateway of any Subnet or VLAN of a network. Monitoring at this port will identify all the infected machines in that Subnet or VLAN



The script runs 'tcpdump' for the amount of time which can be changed by the administrator (the default value is 5 minutes). It captures all the packets going through that particular point in that duration of time and writes them into a file which is created inside a directory of the current date and time as the name. Using this file several other files are made which contain packets of exclusively ARP, TCP & UDP protocols.

Now these files are studied for infections by identifying specific signatures and patterns in packet transmission as described in the next section.

3. Malicious Threats in Packets in a Network

There are untold billions of packets flying around the networks today. A great many of them are of malicious intent [2]. Following are the threats which the tool is able to detect.

3.1 WORMS [3]

3.1.1 W32.MYDOOM@MM

W32.Mydoom@mm is a mass-mailing worm that uses its own SMTP engine to send itself to the email addresses that it finds on an infected computer. The email contains a spoofed "From" address. The subject and message body vary, and the attachment has a .bat, .cmd, .exe, .pif, .scr, or .zip extension.

Systems Affected: Windows 2000, Windows 95, Windows 98, Windows Me, Windows NT, Windows Server 2003, Windows XP

Systems Not Affected: DOS, Linux, Macintosh, OS/2, UNIX, Windows 3.x

Detection Technique: Mydoom worm is detected by studying the data in UDP packets.

It attempts to guess the name of an SMTP server by prepending the following names to the domain names gathered from the local computer:

- gate.
- mail.
- mail1.
- mxs.
- mx.
- mx1.
- ns.
- relay.
- smtp.

Thus if any machine on the network uses all the above prefixes with a single domain name to guess the name of an SMTP server then that machine is concluded to be infected by the mydoom worm.

The domain name along with the prefix is present in the UDP datagram.

3.1.12 W32.MYTOB.Q

W32.Mytob.Q@mm is a mass-mailing worm with back door capabilities that is infected with W32.Pinfi. The worm uses its own SMTP engine to send email to addresses that it gathers from the compromised computer.

Systems Affected**:** Windows 2000, Windows 95, Windows 98, Windows Me, Windows NT, Windows Server 2003, Windows XP

Detection Techniques: Mytob.q worm is also detected by studying the data in **UDP packets.** The functionality in the worm is designed to contact the following IRC server, join a specified channel, and wait for further instructions:

- 19.xxor.biz

It connects to the server 19.xxor.biz and accepts remote control commands, which will be performed on the affected computer.

Thus any machine having the above server name as domain name in its UDP packets is concluded to be infected by mytob.q worm.

3.1.3 W32.BLASTER

W32.Blaster Worm is a worm that exploits the DCOM RPC vulnerability [4] (first described in Microsoft Security Bulletin MS03-026) using TCP port 135.The worm targets only Windows 2000 and Windows XP machines. While Windows NT and Windows 2003 Server machines are vulnerable to the aforementioned exploit (if not properly patched), the worm is not coded to replicate to those systems. This worm attempts to download the msblast.exe file to the %WinDir%\system32 directory and then execute it.

Systems Affected**:** Windows 2000, Windows NT, Windows Server 2003, Windows XP

Systems Not Affected**:** Linux, Macintosh, OS/2, UNIX, Windows 95, Windows 98, Windows Me

Ports: TCP 135, TCP 4444, UDP 69

Target of Infection: Machines with vulnerable DCOM RPC Services running.

Detection Techniques: Generates an IP address and attempts to infect the computer that has that address. The IP address is generated according to the following algorithms

- For 40% of the time, the generated IP address is of the form A.B.C.0, where A and B are equal to the first two parts of the infected computer's IP address.

  C is also calculated by the third part of the infected system's IP address; however, for 40% of the time the worm checks whether C is greater than 20. If so, a random value less than 20 is subtracted from C. Once the IP address is calculated, the worm will attempt to find and exploit a computer with the IP address A.B.C.0.

  The worm will then increment the 0 part of the IP address by 1, attempting to find and exploit other computers based on the new IP address, until it reaches 254.
- With a probability of 60%, the generated IP address is completely random.

Sends data on TCP port 135 that may exploit the DCOM RPC vulnerability. The worm sends one of two types of data: either to exploit Windows XP or Windows 2000.

For 80% of the time, Windows XP data will be sent; and for 20% of the time, the Windows 2000 data will be sent.

Listens on UDP port 69. When the worm receives a request from a computer to which it was able to connect using the DCOM RPC exploit, it will send msblast.exe to that computer and tell it to execute the worm.

Thus if any machine sends tcp SYN request packets on tcp port 135 to increasing destination IPs continuously then it is infected by the w32.blaster worm.

3.1.4 W32.SASSER.WORM

W32.Sasser.Worm is a worm that attempts to exploit the vulnerability described in Microsoft Security Bulletin MS04-011. It spreads by scanning the randomly selected IP addresses for vulnerable systems.

Uses the AbortSystemShutdown API to hinder attempts to shut down or restart the computer.

Systems Affected**:** Windows 2000, Windows XP

Systems Not Affected: DOS, Linux, Macintosh, Novell Netware, OS/2, UNIX, Windows 3.x, Windows 95, Windows 98, Windows Me, Windows NT, Windows Server 2003

Ports: TCP 445, 5554, 9996

Target of Infection: Unpatched systems vulnerable to LSASS exploit – MS04-011.

Detection Techniques: It starts an FTP server on TCP port 5554. This server is used to spread the worm to other hosts.

Retrieves the IP addresses of the infected computer, using the Windows API, gethostbyname.

Generates another IP address, based on one of the IP addresses retrieved from the infected computer.

- 25% of the time, the last two octets of the IP address are changed to random numbers. For example, if A.B.C.D is the IP address retrieved, then C and D will be random.
- 23% of the time, the last three octets of the IP address are changed to random numbers. For example, if A.B.C.D is the IP address retrieved, then B, C, and D will be random.
- 52% of the time, the IP address is completely random.

Connects to the generated IP address on TCP port 445 to determine if a remote computer is online.

Thus if any machine sends too many tcp packets on random IPs but on port 445 (microsoft-ds) and the destination IPs obtained per second is greater than a particular number then it is infected by sasser worm.

3.2 SUSPICIOUS PACKETS

Our tool has a feature of identifying the suspicious packets, packets which do not follow the standards for different protocols as defined in the RFCs [5].

- SYN FIN is probably the best known illegal combination. SYN is used to start a connection, while FIN is used to end an existing connection. It is nonsensical to perform both actions at the same time. Many scanning tools use SYN FIN packets, because many intrusion detection systems did not catch these in the past, although most do so now. You

can safely assume that any SYN FIN packets you see are malicious.

- SYN FIN PSH, SYN FIN RST, SYN FIN RST PSH, and other variants on SYN FIN also exist. These packets may be used by attackers who are aware that intrusion detection systems may be looking for packets with just the SYN and FIN bits set, not additional bits set. Again, these are clearly malicious.

- Packets should never contain just a FIN flag. FIN packets are frequently used for port scans, network mapping and other stealth activities.

- Some packets have absolutely no flags set at all; these are referred to as "null" packets. It is illegal to have a packet with no flags set.

Besides the six flag bits, TCP packets have two additional bits which are reserved for future use. These are commonly referred to as the "reserved bits". Any packet which has either or both of the reserved bits activated is almost certainly crafted.

There are several other characteristics of TCP traffic where abnormalities may be seen:

- Packets should never have a source or destination port set to 0.

- The acknowledgment number should never be set to 0 when the ACK flag is set.

- A SYN only packet, which should only occur when a new connection is being initiated, should not contain any data.

- Packets should not use a destination address that is a broadcast address, usually ending in .0 or .255 (.0 was an older style of broadcast). Broadcasts are normally not performed using TCP.

Many of the tools used by attackers to scan and probe our networks are based on the use of abnormal TCP packets. A large percentage of alerts detected by intrusion detection systems involve these types of packets, so it is critical to be able to identify them. Our tool alerts on all abnormal TCP packets.

Detection Techniques: Our tool analyses all the TCP packets for any of the above mentioned suspicions and lists source of these packets

3.3 SSH Dictionary Attack

Also known as a dictionary attack, which uses a list of known passwords, a program will connect to a remote SSH server and attempt to login using common user name/password combinations. The system administrators will notice large number of failed SSH login attempts in their log files. It is quite obvious that this is the work of an automated program as the user names used are attempted in alphabetical order. The time-stamps are also a dead give away, with connections only a few seconds apart. Such scripts are generally used by hackers for illegal intrusion.

A dangerous addition to these attacks is the attempts to break into “root” accounts.

Thus if any Linux machine attempts too many ssh attacks on any other machine then that machine is definitely suspicious.

Detection Techniques: If any linux machine connects normally to ssh port (port 22) of another machine then the number of tcp packets transmitted, starting from the time when the connection is established till it is closed (SYN till FIN) remains almost fixed. But if a machine is attempting a ssh dictionary attack on another then it

continuously sends tcp SYN requests on port 22. Thus this machine is infected.

3.4 ADWARES

Our tool uses TCP and UDP packets for adware detection purposes. Although detecting adwares is a bit difficult job but our tool tries to catch the infected machines in the best possible way. It uses the TCP payload and UDP data for this purpose. We have created a list of more than 5000 adware [6] and spyware [7][8] sites and the list is manually updated.

Detection Techniques: In the UDP data we simply compare the "NAME=" part in the packet with our list and if any of site matches then the machine with that particular packet is declared as infected.

In the TCP payload the situation is a bit different. First we capture all the TCP PUSH packets with destination port 80(http) or 443(https). Then the ngrep utility is run in order to separate the payload from the rest of the packet. Then we check the 'host' part of the payload and match it with our list. Here, we observe 2 cases-

Case1- Packets which do not use the proxy have the host name right there. Thus we directly grep the IP of infected machine (present in the first line of packet payload).

Case2- If any machine uses proxy for sending these packets then first 'host' is checked for the destination site and then we grep the 'X-forwarded for' part of the payload to know the source IP which generates the adware packet.

3.5 Unidentified Infections

There are many infected machines which cannot be identified for the type of infection but the tool is able too identify machines which are possibly infected.

Detection Techniques: The tool identifies all the machines which sending packets at a rate greater than **n** Packets per second (the default value of n is 10). Many of these machines may be servers or may be machines which are doing heavy data transfer (e.g. ftp, nfs, windows sharing etc.). We maintain a list of server IPs (can be modified in the tool). So a list of IPs which are generating heavy traffic, leaving the server IPs and the IPs of machines detected to have an identified infection, is generated. Every time the tool is run, such list is generated. Since infected machine will be in the list most of the time whereas other machine will be there only when they are doing heavy data transfer, correlating the report generated in repeated runs, can be used to identify the infected machines.

4. Limitations of the Tool

There are several limitations of the tool. Some are:

- *Dynamic behavior of Malware signatures: A* worm never contains exactly the same string pattern (signatures) and can vary for different attacks making it difficult to recognize it entirely on the basis of the string. So it generates possibilities of misses as well as false alarms.
- *Appearance of a new threat each day:* Any new infection which is not in our library would pass unchecked into the network. So the tool should be updated regularly to cover new infections.
- *Problems related to encrypted data:* Certain worm/virus developers prefer to send the malicious code in the encrypted form making its detection tedious.
- *Intent of the suspicious packets still unknown:* Though the suspicious

packets can be detected but their intention remains a question.

- *False Alarm:* The identification of machines as possibly infected poses a high probability of false alarm. The extent of false alarms generated has not been studied yet.

5. Possible Enhancements in the Tool

Modifications can be done in the script and/or hardware on which it is implemented to cover more Malware infections for which it can do complete analysis, increase the efficiency & make it real time analysis tool. Specific improvements can be:

- *Improving the dictionary of worm signatures:* This would help us ensure that there are less number of misses and false alarms.
- *Making the tool work as a Prevention Tool also: S*pecific ports on switches to which infected machines which are reportedly generating large number of malicious or suspicious packets are connected can be blocked, thus removing the infected machines from the network. The MAC address of the machines identified as infected or suspicious can also be blocked, so that the infection doesn't spread in the network and also save the network bandwidth.
- *Reducing the time taken for complete analysis:* Improvements can be done in the script like inserting certain new functions or using some different algorithms or employing some new logic that can bring down the analysis time. Parallel programming can be one possible solution to make the code run faster.
- *Strengthening the Hardware:* Using more sophisticated hardware based on newer technologies to run this tool can help it perform better. For instance, using more RAM or multiple processor etc. will definitely lead to reduction in the processing time and improvement in the performance of the tool.
- *Implementing parallelization at the hardware level:* This requires the captured traffic itself to be divided and processed parallel on different machines (which are physically separated but logically connected in a cluster) and the result be compiled in the end. This brings down the analysis time to as close as the capture time making the tool more suitable for real time applications.

6. Conclusion

Lately there has been a lot of increase in Malware which get installed on the users' PC and connect to Internet without the users knowledge i.e. in the background. This tool helps Network Administrator to identify such computers and take necessary action.

Currently the tool is a basic tool which is very useful in identifying worms like sasser, mytob, mydoom, blaster etc., adwares, ssh dictionary attacks, and suspicious packets including port scans. Still there are many infected machines in the network which it is not able to identify with certainty. There are possibilities of misses and false alarms in case of identifying machines which are infected with unidentified Malware.

The tool can be enhanced to work as a realtime Malware Infection Detection System (MIDS), towards which we are working.

7. References

[1]http://www.tcpdump.org/tcpdump_man.html

[2]http://www.cromwell-intl.com/security/security-netaudit.html

[3]http://www.doshelp.com/Ports/Trojan_Ports.htm

[4]http://securityresponse.symantec.com/avcenter/venc/data/detecting.traffic.due.to.rpc.worms.html

[5]http://www.securityfocus.com/infocus/1223

[6]http://sarc.com/avcenter/security_risks/adware/

[7]http://www.spywaredetector.us/spywareweremove6.htm

[8]http://www.trojanguide.com/spywarelist.html