Chance Currie
William Mahoney
Joe Selvera
David Abbot

CSCE 3560 Final Paper

1. Abstract

In this work, we have created an Intrusion Detection System (IDS) for a Linux Virtual Machine (VM) to detect three kinds of Denial-of-Service (DoS) attacks: TCP SYN Floods, Ping Floods, and Fork Bombs. Using the C programming language, we have implemented our own solutions to detect and mitigate the effects of these DoS attacks. The IDS that we have implemented is able to successfully monitor, detect, and mitigate these three attacks

2. Introduction

An IDS is an extremely useful tool to have in an environment that deals with system security. They can monitor, detect, and mitigate attacks that are extremely harmful to a system if the attacks were to be run successfully. In this project, we will build our own IDS that can monitor, detect, and mitigate the specified attacks in order to protect a Linux system. To do this, we coded the IDS in C and ran it inside of a Linux environment, and tested it by running three specific attacks against the Linux environment to see if the IDS can monitor, detect, and mitigate these attacks quickly and efficiently.

3. Related Works

A Slowloris attack is a type of slow DoS attack, meaning that it seeks to consume resources on the server rather than overwhelm the network directly. An advantage of a slow DoS attack is that it can look similar to legitimate network traffic and server requests. [1] In a study on detecting and measuring the behavior of a Slowloris attack on Wireless Mesh Networks (WMNs), the researchers were able to develop a tool called SDToW that was able to prevent the attack without affecting the number of connections able to be made to the server. The detection is performed by filtering the Slowloris connections from the legitimate connections, noting that the legitimate connection GET packets are different in size from Slowloris packets and that the packets sent from a Slowloris attack are reassembled protocol data units (PDU) while the legitimate packets are not reassembled. [2] The method we will use for the mitigation of this attack will simply look at the number of connections from an IP address.

A Ping Flood attack is a more volume based DoS attack. While volume attacks can also consume the server's resources, they have a larger impact on the network's resources than a slow DoS attack. As it is meant to consume bandwidth, the total bandwidth of the attacker has to be able to eclipse the bandwidth of the victim's network. [3] One method of mitigation proposed is to simply limit the overall amount of ICMP traffic that is allowed to arrive at the server in the firewall either by limiting the number of packets or the amount of data from these packets over a particular period of time. Any extra packets that would surpass either version of this threshold would be dropped by the firewall. [4] Our method is similar in that it looks for a burst of ping packets.

A Fork Bomb attack is a host based DoS attack as it is entirely meant to overwhelm and disable the host machine rather than have any real impact on the network itself. A fork bomb is a simple program that uses looped fork calls to overwhelm the host with the resources the machine allocates to each resulting process. One mitigation technique proposed in 2016 limits the resources of a process that is creating processes more rapidly than a set limit. The benefit of this technique is that the limited process would still be able to potentially finish in case of a false positive detection. [5] Our method will not employ this technique, but it as an option should be noted.

4. Solutions

Our solution to the following attacks is to create one IDS with multiple functioning parts. These multiple functioning parts will each monitor for a specific attack, detect a specific attack, and mitigate the attack.

The first attack is a Slowloris attack. This attack targets an HTTP web server and conducts a DDoS attack, specifically a TCP SYN flood, on it. It will create a huge amount of sockets that will try to connect to the web server, and to keep each socket connection alive, it will send a TCP SYN request from each connected socket. By sending these TCP SYN requests every so often, the connection stays open, and Slowloris will try to keep the connection open for as long as possible. By doing so, the web server is wasting resources by trying to respond to all of the TCP requests that are being sent.

To monitor for this attack, we first need to see how many connections are being made to the host machine's IP through a netstat command that is displayed in the Linux terminal. The command "netstat -ntu | awk '{print $5}' | cut -d: -f1 | sort | uniq -c | sort -nr" will display a list of IP addresses that are currently connected to the host machine, and it will also show how many connections each IP is making. This will continuously run until the user stops it in order to monitor for any potential Slowloris attacks.

To detect this attack, there is a threshold that is set, and this threshold determines the max connections to the web server that a single IP can have. This threshold can be adjusted accordingly, but the threshold is currently set to 30 max connections that one IP can have. By reading in the IP addresses that are currently connected to the web server and the amount of connections that are being made to it from each IP, we can use the set threshold to determine if the number of connections being made by one IP is greater than the maximum number of connections that one IP is allowed to have. If a particular IP does have more connections made than the set threshold, it is flagged as a potential Slowloris attack.

To mitigate this attack, we first need to block the IP that is conducting the Slowloris attack. To do this, we add a UFW firewall rule as follows: "ufw deny from (ip)", and this rule will block that IP from attempting to make a connection through any of the ports. Before it is detected, Slowloris will have some connections that made it through before the IP was blocked, and it will keep those connections open, so we need to kill all of the current connections that are still being made by the Slowloris IP. Once that's done, the IP is currently blocked and the attack has been mitigated.

The second attack we will be protecting against is the Ping Flood attack. Pings are a common mechanism used to determine if there is connection between two devices connected to the internet. A ping command works by the sender sending a Internet Control Message Protocol (ICMP) echo request packet to a specified IP address, and upon a successful connection, the specified device will respond back with an echo reply. If the connection was unsuccessful, then there is no echo reply back. A Ping Flood works by an attacker's computer sending out a rapid spam of ping echo requests to the victim, and due to how the targeted computer replies to an echo request, if the attacker has a higher bandwidth than the victim, the victims network can become flooded, thus leading to a denial of service occurring.

To monitor for this attack, we grab incoming ICMP Ping packets that are targeted to our VM, and determine the passage of time between packets incoming to our system along with the count. A normal ping request from a system sends out a one packet every second, and while a user can change the time intervals, there is a minimal time the ping command allows someone to send a ping packet to a device, which is around 1 decisecond (.1s). Our IDS checks if we get ping packets incoming in intervals faster than 1 decisecond, and if there's a burst of packets occuring, which would indicate that there is a Ping Flood occurring to our system.

When the Ping Flood is detected, the system will mitigate the attack by temporarily blocking the attacker's IP address, so the VM will not acknowledge any echo requests incoming from that IP. To achieve this, the IDS will grab the IP address from the incoming ping packets, and with the help of ufw, create two rules: "ufw deny from {IP} to any", and update the before.rules file with "-A ufw-before-input -s {IP} -j DROP". This second rule needs to be implemented since ufw does not allow us to directly specify the protocol for ICMP. Once these rules are created, the ufw firewall is reloaded so our rules can take effect, and the attacker who is Ping Flooding our system will have all communications blocked, and a log file of the event with the time stamp will be written and updated. Since our IDS will temporarily block the IP address, our program will keep track of all the IP addresses it has banned from Ping Flood attacks, along with a timestamp of when the attack occurred. Once the specified amount of time (in seconds) has passed, the IDS will remove the two rules we have created, reload the firewall to unblock the IP address, update the log file of the IP address and time it was unblocked, and delete the IP address and time stored within the programs memory.

The final attack performed on the system is the Fork Bomb. The attack will utilize a C program that will run in an infinite loop calling fork(). This will cause the program to spawn many processes very quickly. To monitor for this attack the program will call the ps command and capture the output. The ps command will list the names of running processes. The “-e” option will list all processes. The “-o comm=” option will display only the command/process name column in the output. This creates an easy list for the program to parse through without needing all other common “ps” information. The program is then able to parse through this list and add a count for each unique name of each process. When the count of a process reaches 10 or more it will be deemed as a potential fork bomb attack. The detection of this potential attack is supported by the fact that most processes on a small scale system as this would not typically have greater than 10 unique process names running at the same time. To mitigate the attack, the program will then call the killall system call for that process name. This will terminate the initial C Fork Bomb attack and all the forked processes that it had created.

5. Experiments

For the Slowloris portion of the IDS, the experimentation for it was much more straight-forward than originally expected. We experimented by using two separate VMs that both had Kali Linux installed and running on them. The attacker VM had Slowloris on it and was able to run through a simple terminal command, and the victim VM had a very basic python3 HTTP web server running on it, as well as the Slowloris IDS code to monitor, detect, and mitigate the Slowloris attack once conducted. Due to the experimentation being surprisingly straight-forward, the implementation for it was also straight-forward, meaning that it is able to determine how many current connections one IP is making to a machine that is hosting a web server, and if the amount of connections being made is over a specified threshold, it will flag and block that IP. This portion of the IDS is accurate in detecting a Slowloris attack, and the threshold can also be easily adjusted if one feels that the current threshold is too high or too low based on the particular needs at that time.

For the Ping Flood portion of the IDS, the implementation is able to determine when there is a normal amount of incoming ping packets from a source, and when there is an influx of ping packets within a small span of time. Due to this, the IDS is able to properly distinguish between legitimate ping requests and illegitimate ping requests. When an attack occurs, the system is able to quickly identify the attack is occuring, and is able to block the attack a few moments later. When the IDS blocks the source of the attack, all requests the attacker is making are dropped, so the IDS is able to mitigate the effects of this DoS attack.

Experimentation for the Fork Bomb attack was quite difficult. Initially the attack was run without any detection or mitigation techniques in place to see how the attack worked. Once it was known that the attack was able to quickly spawn many processes within seconds and crash the computer, the attack was run slower. This was done by placing a sleep after each fork() iteration. This allowed time to see how the fork processes will grow exponentially. During the attack it was determined that the task manager would display these processes being spawned and it was noticed that they would all have the same name. This led to developing the program in such a way that it could monitor these processes as they were growing. Manually viewing the task manager showed that each process name was either unique or had very few duplicates. To start out with experimentation the program would just call “ps” to get the list of programs. Once that was successful, an iteration of this list was developed. Then when it was found out that the list could be parsed and have a count attached to each process it was very easy to pinpoint processes that were running abnormally.

6. Discussion

One of the bigger issues that was realized a bit late regarding the implementation of the Slowloris portion of the IDS is potential false positives. While the IDS does detect these attacks accurately when a Slowloris attack is conducted, Slowloris makes a large amount of connections extremely quickly. This means that it opens up a bunch of connections in a very short amount of time, so while the amount of connections were dealt with, a variable involving the time in which those connections were made between each other could have been implemented if this was noticed sooner. The number of connections being made can go up just by joining the web server and refreshing the page too many times, even if each refresh had a time of one minute in between each connection. However, the threshold that determines the maximum number of connections one IP is allowed to have can be adjusted to account for these forgotten factors.

The biggest issue regarding the implementation of the Ping Flood portion of the IDS is the usage of the ufw firewall. This firewall is not able to properly set rules for the protocol ICMP like it can for TCP and UDP, so we have to edit the before.rules file to have the firewall be able to properly block only that specific IP address for ICMP packets.

There were many issues in the development of the Fork Bomb attack. For one, the attack is almost impossible to prevent once it has started unless there is a hard process limit put on the user. This limit could be bypassed if the user that calls the Fork Bomb is a root user. This made experimenting with the attack difficult to manage. The computer facing the attack had to be restarted every time the attack was implemented until the mitigation was successful due to the sheer overwhelming of the CPU and RAM of the computer. A systematic and scientific approach was required to develop the mitigation technique because if too many variables were changed it was difficult to determine what caused the failure. Each line of code was added one by one until it became successful.

7. Conclusion

While the techniques implemented in our IDS program are far from the most sophisticated available, each attack was able to be detected and eliminated during our experimentation. The algorithms in use however might not be suitable for a commercial implementation. The methods used against Ping Flood and Slowloris attacks might block IPs from clients who need to access the servers. The Fork Bomb defense method used may not be able to shut down the processes quickly enough if the Fork Bomb attack is allowed to be run without speed limitations on its process creation. However, for the purposes of this project, we were able to come up with adequate solutions to show working methods that could substantially mitigate against versions of these attacks.

8. References

[1] Duravkin, Ievgen, Anastasiya Loktionova, and Anders Carlsson. "Method of slow-attack detection." *2014 First International Scientific-Practical Conference Problems of Infocommunications Science and Technology*. IEEE, 2014. https://ieeexplore.ieee.org/abstract/document/6992341

[2] V. S. Faria, Vinicius da Silva, et al. "SDToW: A slowloris detecting tool for WMNs." *Information* 11.12 (2020): 544. https://www.mdpi.com/2078-2489/11/12/544

[3] Harshita. "Detection and Prevention of ICMP Flood DDOS Attack." *International Journal of New Technology and Research* 3.3 (2017): 263333. https://www.neliti.com/publications/263333/detection-and-prevention-of-icmp-flood-ddos-attack

[4] Bogdanoski, Mitko, and Aleksandar Risteski. "Wireless Network Behavior under ICMP Ping Flood Dos Attack and Mitigation Techniques." *International Journal of Communication Networks and Information Security (IJCNIS)* 3.1 (2011).

[5] Nakagawa, Gaku, and Shuichi Oikawa. "Fork bomb attack mitigation by process resource quarantine." *2016 Fourth International Symposium on Computing and Networking (CANDAR)*. IEEE, 2016. https://ieeexplore.ieee.org/abstract/document/7818694