Network Security and Forensics – EECS 4482
TCP Attacks
Prepared by: Dr. Ruba Al Omari
Fall 2024

Contents

Disclaimer

- This assignment is designed for the purpose of education and training, but not for any illegal activities including hacking. Beware to only use these exploits on hosts that you have written permission to hack.
- Creating or deploying malware, or engaging in any malicious activities is against ethical. guidelines and is illegal.
- Always ensure that you have proper authorization and are acting within the bounds of the law and ethical guidelines in a controlled environment.

Instructions

- **Uploading of course material, assignments, labs, sample solutions, or tests to online sites is prohibited. No reuse of this assignment is allowed in part or in full without my written permission.**
- This assignment should be completed individually.
- When a question asks for a screenshot, your screenshot must:
  - Include the full window (the application window, or the terminal window, etc... but not the whole display),
  - have the PROMPT setup as per the instructions, including your name, and the date

and time in the same format provided in the instructions. Screenshots without the prompt setup will receive zero credit,

- be clearly readable,
- include all the information required by the question, and
- **not** include extra commands, failed attempts, and/or error messages. Providing more than what is required will result in a penalty of:
    a. -5 Marks per extra screenshot.
    b. -5 Marks per extra command/answer/comment.
    c. -5 Mark per screenshot with error messages.

- You should type the commands below and not copy them from this document. Copying text from a .pdf or .docx file into a terminal doesn't always (almost NEVER) work as intended.
- The below instructions are for guidance, you are expected to search and troubleshoot any warnings or errors you run into while following lab instructions or working on your assignments.
- Sample screenshots (screenshots with the word SAMPLE) are for guidance only. You will/may need to run other commands that are not displayed in the sample screenshots.
- Operating Systems, vulnerable boxes, libraries, tools, and commands get updated frequently, if a command in this document is deprecated, find the current alternative command and use it. If a software in this document has a newer release you can use it.

Environment Setup

We will use the SEED Ubuntu 20.04 Virtual Machine available at https://seedsecuritylabs.org/.
Refer to the Environment Setup documents on eClass for more information.

1- Download the **TCP Attacks Lab** document available here:
https://seedsecuritylabs.org/Labs_20.04/Files/TCP_Attacks/TCP_Attacks.pdf
We will refer to this document as **TCP-SEED** document.
2- Download the **Labsetup** file available at
https://seedsecuritylabs.org/Labs_20.04/Networking/TCP_Attacks/ and save it to your SEED Ubuntu 20.04 Virtual Machine.
3- Read the TCP-SEED file, and use it to aid you in carrying out the below tasks which are mapped to the tasks in the TCP-SEED file.

Task 1 – SYN Flooding Attack

In this task, you will launch a SYN flood attack. SYN flood is a form of DoS attack in which attackers send many SYN requests to a victim's TCP port, but the attackers have no intention to finish the 3-way handshake procedure. Attackers either use spoofed IP addresses or do not continue the procedure.

Through this attack, attackers can flood the victim's queue that is used for half-opened connections. Half-opened connections are the connections that have finished SYN, SYN-ACK, but have not yet gotten a final ACK back. When this queue is full, the victim cannot take any more connection requests.

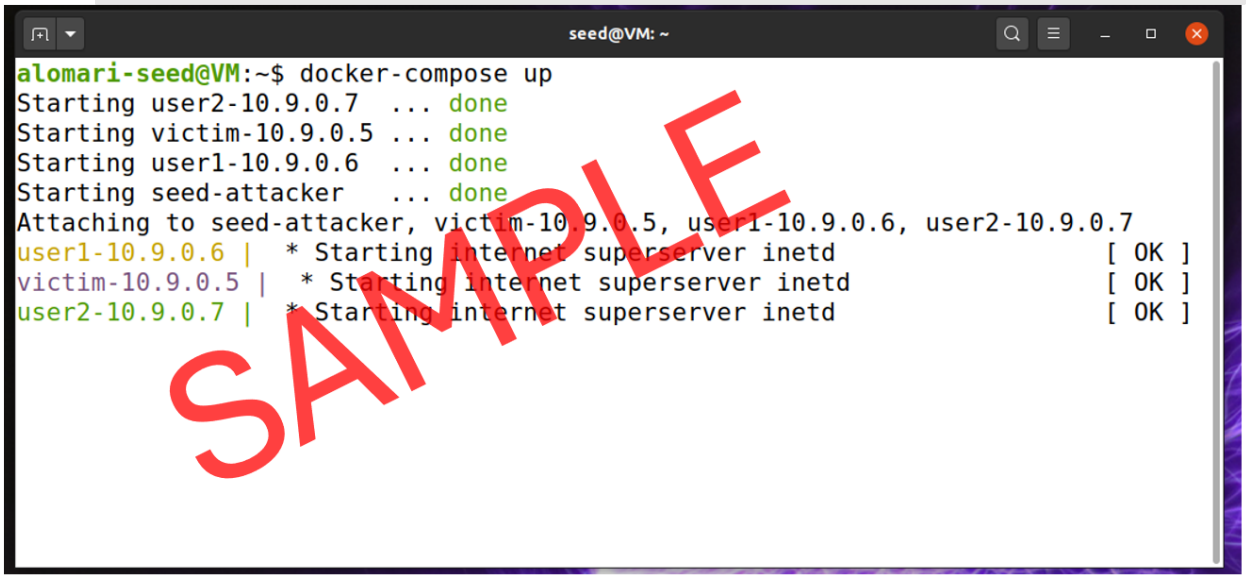
1. Start by changing the terminal prompt as shown in the command below:

```
$PS1='[`date "+%D"`] yourname [`date "+%r"`]-[~]'
```

Your terminal should look like the screenshot below.

2. Build and start the containers:

```
$ docker-compose build
$ docker-compose up
```



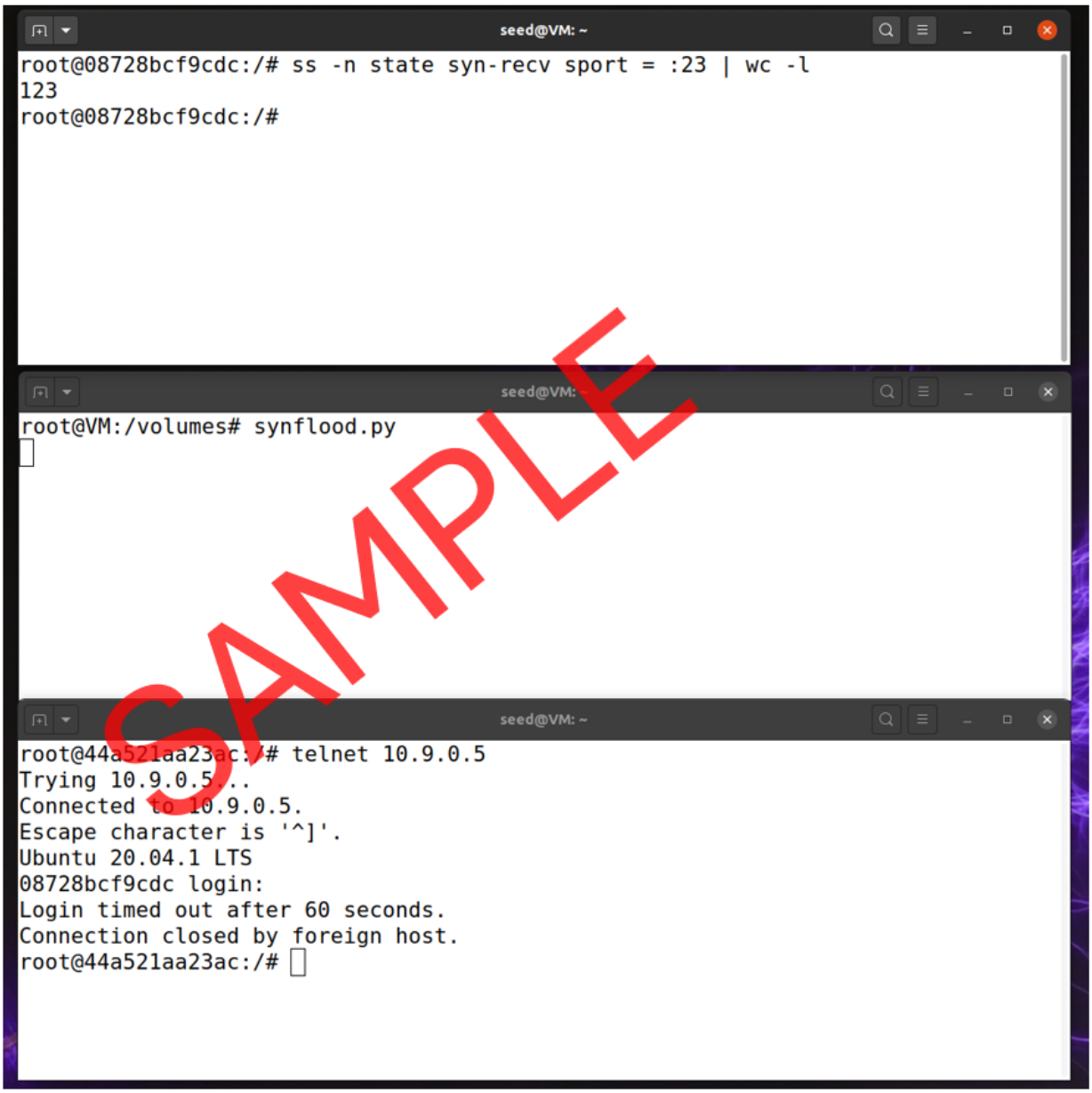
Important Note: For the remaining screenshots, my terminal prompt is set up with my name only without the date and time. But your terminal prompt should include the date, time, and format from the previous step.

3. Leave the containers running in this terminal**.** Open a new terminal, change the prompt similar to what you did in step 1, and check the container IDs.

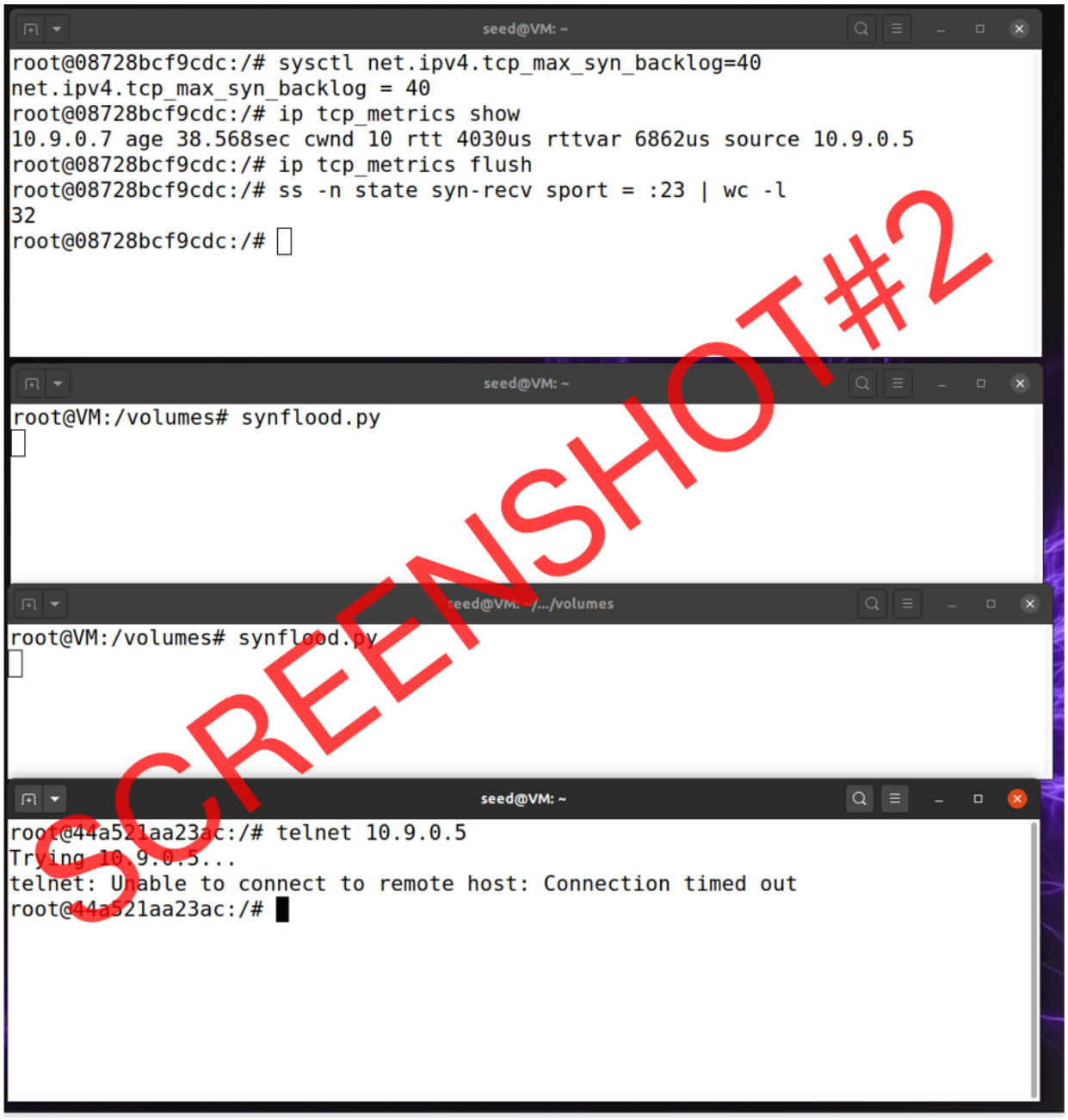
4. Take a screenshot of your containers' IDs similar to the one shown above, and place it under **Screenshot#1** in the answer file.
5. Connect to the victim machine, and ensure the **`syncookies`** flag is turned off (i.e., =0). If it is set to 1, ensure you set it to 0.

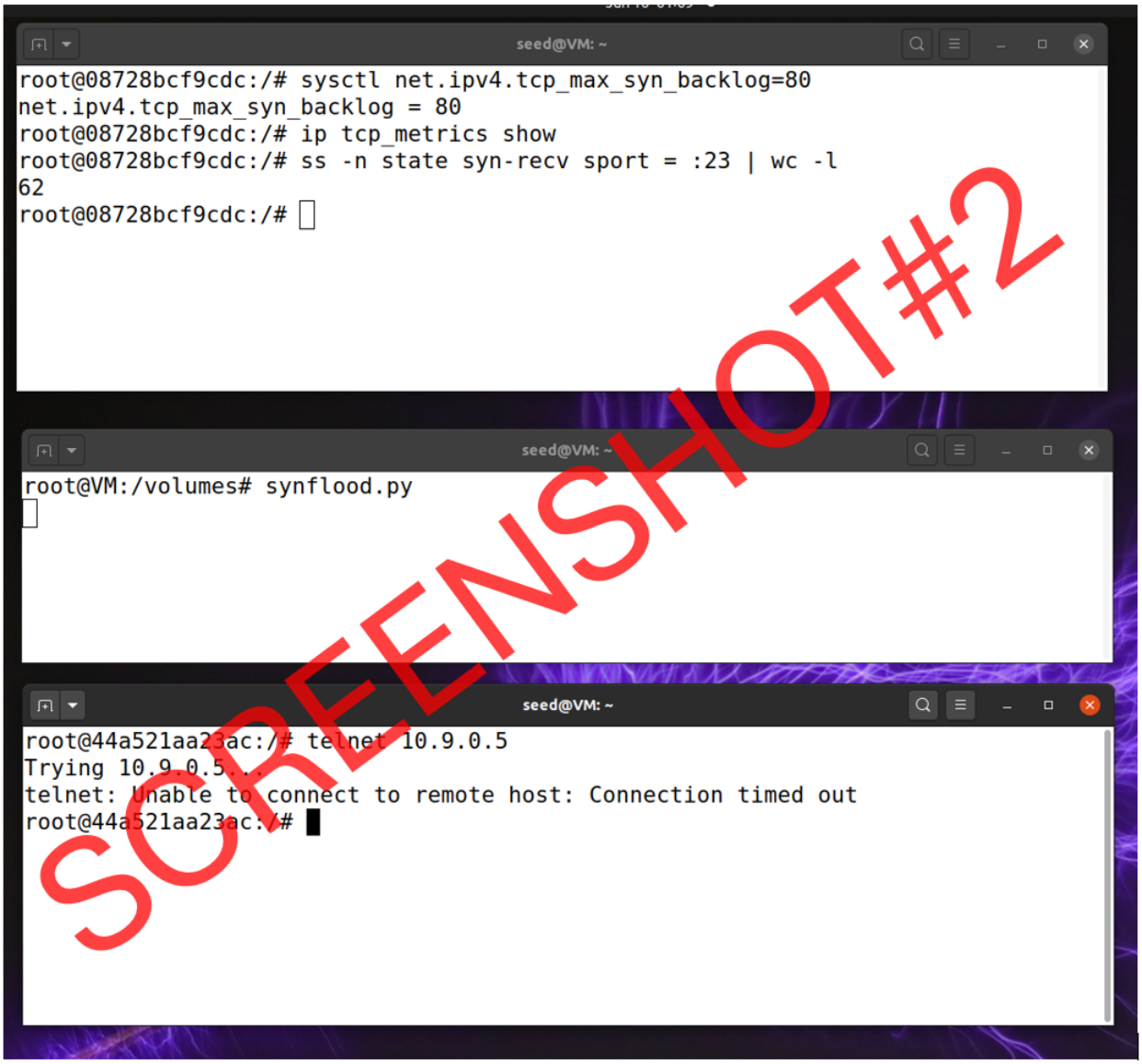
6. On the victim container, check the size of the **`tcp_max_syn_backlog`** queue.

7. Modify the `synflood.py` code provided in the TCP-SEED file as needed to launch a SYN flood attack from the **seed-attacker** container to the **victim** container. While the attack is running try to telnet to the victim machine from one of the users' containers. As you may have noticed the attack is unlikely to succeed from the first attempt (as shown in the sample screenshot below, where the DoS on telnet failed).



8. You will need to go through the troubleshooting tips provided in the TCP-SEED file until the attack succeeds. Below are 2 sample screenshots that show a successful SYN flood attack, causing a DoS on the victim's telnet service. Your screenshot may have different settings.


seed@VM: ~
root@08728bcf9cdc:/# sysctl net.ipv4.tcp_max_syn_backlog=40
net.ipv4.tcp_max_syn_backlog = 40
root@08728bcf9cdc:/# ip tcp_metrics show
10.9.0.7 age 38.568sec cwnd 10 rtt 4030us rttvar 6862us source 10.9.0.5
root@08728bcf9cdc:/# ip tcp_metrics flush
root@08728bcf9cdc:/# ss -n state syn-recv sport = :23 | wc -l
32
root@08728bcf9cdc:/#
seed@VM: ~
root@VM:/volumes# synflood.py
seed@VM: ~/.../volumes
root@VM:/volumes# synflood.py
seed@VM: ~
root@44a521aa23ac:/# telnet 10.9.0.5
Trying 10.9.0.5...
telnet: Unable to connect to remote host: Connection timed out
root@44a521aa23ac:/#
SCREENSHOT#2



9. Take a screenshot similar to ONE of the 2 screenshots above, and place it under **Screenshot#2** in the answer file. Take one screenshot only, whichever works with you.

10. **Question#1**: Why did you have to flush the cache at the victim machine?

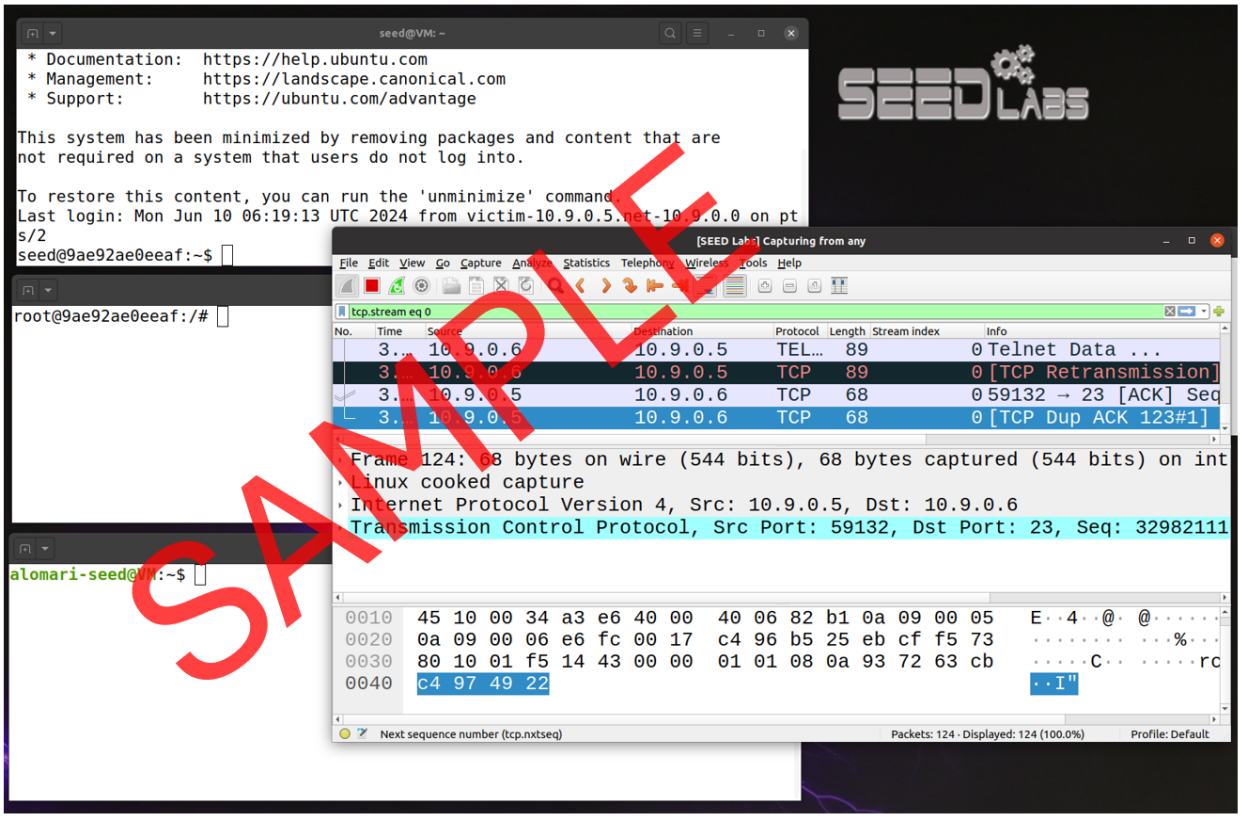
11. **Question#2**: Enable the SYN cookie mechanism, and run your attack again. Was the attack successful?

Task 2 – TCP RST Attacks on Telnet Connections

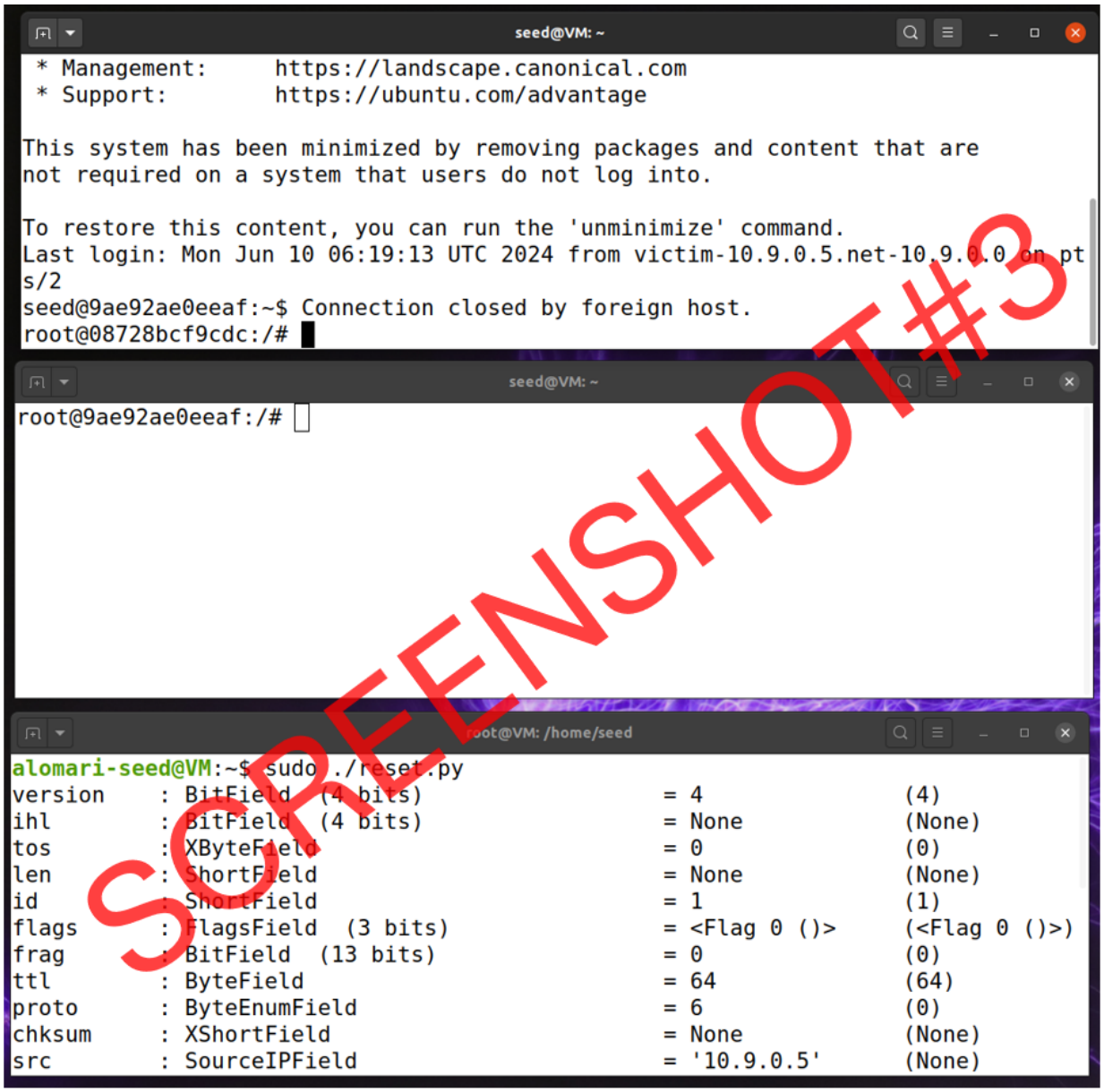
The TCP RST Attack can terminate an established TCP connection between two victims.

In this task, you will launch a TCP RST attack from the SEED VM to break an existing telnet connection between 2 users (the victim container @10.9.0.5 and user1 container @10.9.0.6).

1. Telnet from 10.9.0.5 to 10.9.0.6, while using Wireshark to sniff the telnet connection.



2. Using the skeleton code provided in the TCP-SEED file, and the info you captured in Wireshark, modify the code to carry out the TCP-RST attack against the telnet connection. Take a screenshot similar to the one below and place it under **Screenshot#3**.



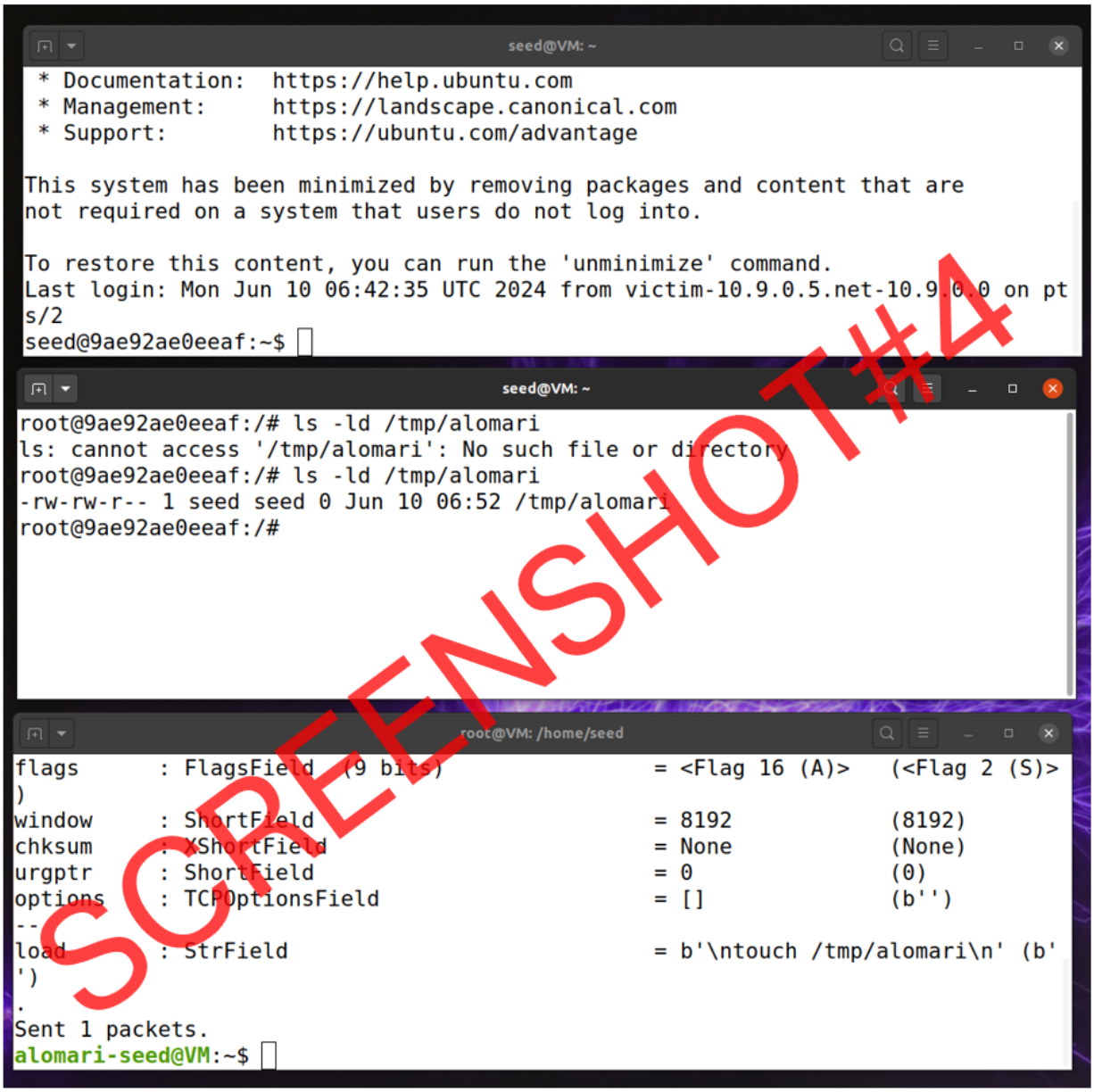
Task 3 –TCP Session Hijacking

In this task, you need to demonstrate how you can hijack a telnet session between two computers. Your goal is to get the telnet server to run a malicious command from you.

1- Telnet from the victim's container 10.9.0.5 to user1's container 10.9.0.6, while using Wireshark to sniff the telnet connection.

2- Using the skeleton code provided in TCP-SEED file, edit the code to perform TCP session hijacking from the SEED VM, and create a file with your name as shown below.

3- Take a screenshot of your TCP hijacking attack, similar to the one below, and place it under **Screenshot#4**.



Task 4 – Creating Reverse Shell using TCP Session Hijacking

When attackers can inject a command into the victim's machine using TCP session hijacking, they are interested in running a reverse shell.

1- Connect to the seed-attacker container through 2 terminals. In one terminal run a netcat listener.
2- Telnet from the victim's container 10.9.0.5 to user1' container 10.9.0.6, while using Wireshark to sniff the telnet connection.
3- Use the code from the previous task, but instead of touching a file, create a reverse shell.

4- Run the code on the second terminal of the seed-attacker container.
5- As you can see, the seed-attacker container has gained a reverse shell access to user1's container shown in the terminal where there was a listener running.
6- Take a screenshot similar to the one below and place it under **Screenshot#5**.