SEED Lab: Packet Sniffing and Spoofing Walkthrough

Spring 2025

Chengyi Qu

Background

- Packet sniffing and spoofing are two important concepts in network security; they are two major threats in network communication. Being able to understand these two threats is essential for understanding security measures in networking.
- There are many packet sniffing and spoofing tools, such as Wireshark, Tcpdump, Netwox, Scapy, etc. Some of these tools are widely used by security experts, as well as by attackers.
- Being able to use these tools is important for students, but what is more important for students in a network security course is to understand how these tools work, i.e., how packet sniffing and spoofing are implemented.

Lab Environment Setup

- Virtual Machine download (!!Use Ubuntu 20.04 VM, NOT cloud version):

  https://seedsecuritylabs.org/labsetup.html

- Virtual Machine setup manual:

  https://github.com/seed-labs/seed-labs/blob/master/manuals/vm/seedvm-manual.md

- Network Security Lab:

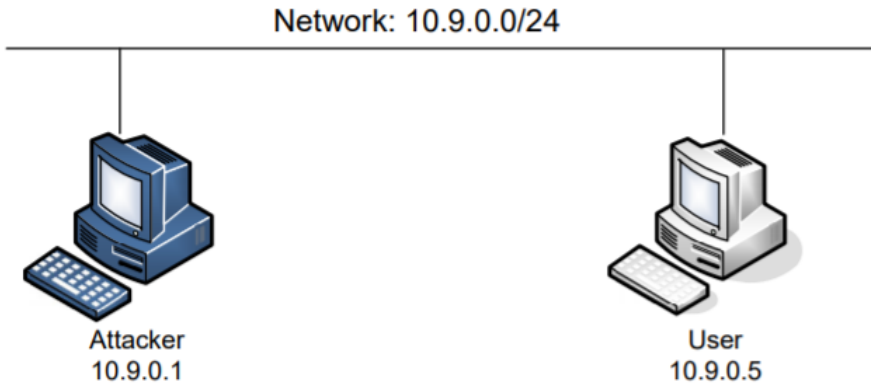
  https://seedsecuritylabs.org/Labs_20.04/Files/Sniffing_Spoofing/Sniffing_Spoofing.pdf

Lab objective

- Learn to use the tools and understanding the technologies underlying these tools.
- In the second objective, students will write simple sniffer and spoofing programs, and gain an in-depth understanding of the technical aspects of these programs. This lab covers the following topics:
    - Docker Containers
    - Scapy & Python
    - Sniffing
    - Spoofing

Setup the Docker Container

- Docker container can run whole application or a service.
- It helps in eliminating the usage of virtual machine images
- Download the “Labsetup.zip” from the manual (https://seedsecuritylabs.org/Labs_20.04/Networking/Sniffing_Spoofing/)
  - Enter the Labsetup folder and use the docker-compose.yml file to set up the lab environment.
- Some of the basic docker commands
  - *docker-compose up -d (to start the container)*
  - *dockps (to find out container ID)*
  - *docksh <id> (to log into the container)*



https://github.com/seed-labs/seed-labs/blob/master/manuals/docker/SEEDManual-Container.md

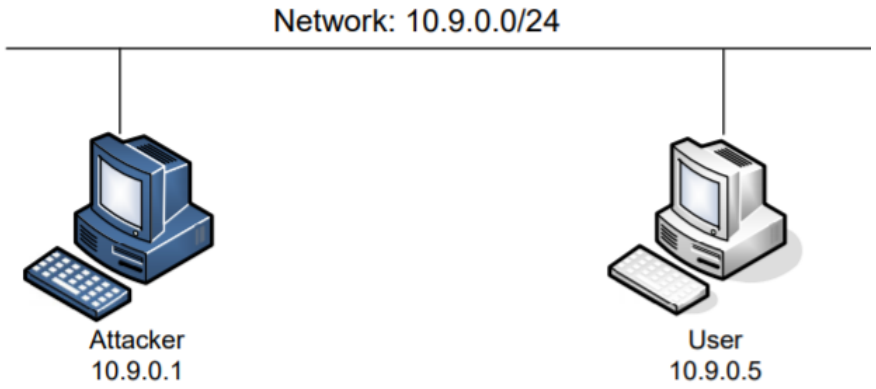
Tasks

- There are two sets of tasks in this lab. We will do ***the first set***.
- The first set focuses on using tools to conduct packet sniffing and spoofing. It only requires a little bit of Python programming (usually a few lines of code); students do not need to have a prior Python programming background. The set of tasks can be used by students with a much broader background.
- Follow step-by-step instructions in the below link (up to **Section 3.4**)

https://seedsecuritylabs.org/Labs_20.04/Files/Sniffing_Spoofing/Sniffing_Spoofing.pdf

How to get ready for the task?

- 1. Start up the docker container, and leave the CLI background.
- 2. Have three CLI windows, login to the three hosts (two hosts node and one hacker node)
- 3. Use the Hacker node to process the Tasks.
- 4. To generate traffic between two hosts, you may use ‘ping’.



https://github.com/seed-labs/seed-labs/blob/master/manuals/docker/SEEDManual-Container.md

Task Set 1: Using Tools to Sniff and Spoof Packets

- How Packets Are Received
  - ✓NIC (Network Interface Card) is a physical or logical link between a machine and a network
  - ✓Each NIC has a MAC address
  - ✓NIC checks the destination address for every packet, if the address matches the cards MAC address, it is further copied into a buffer in the kernel

Python + Scapy

- Powerful modules (or programs) for packet manipulation
- Packet parsing
- Packet sending and receiving
- Packet sniffing
- Packet spoofing
- Adding new protocols
- Many more applications …

Simple Scapy program

- Here is sample code on how scapy works

```
$ view mycode.py
#!/bin/bin/python3
from scapy.all import *
a = IP()
a.show()

$ sudo python3 mycode.py
###[ IP ]### version = 4
            ihl         = None
...
```

Interactive mode of Python

- We can run the program one line at a time at the Python prompt.

```
$ sudo python3
>>> from scapy.all import *
>>> a = IP()
>>> a.show()
###[ IP ]###
version = 4
ihl = None
...
```

Lab Tasks

Lab Task Set I

- Task 1.1A Sniffing packets (30 points)
- Task 1.1B Scapy's filtering (10 Extra Points!)
- Task 1.2 Spoofing ICMP packets (30 points)
- Task 1.3 Traceroute (40 points)
- Task 1.4 Sniffing and-then Spoofing (10 Extra Points!)

Note: We will write our code in the ./volumes folder (on the VM), so they can be used inside the containers.

Task 1.1A Sniffing Packets (30 Points)

- Sniffing program in Scapy
- You need to find your own iface value

using the below command

- “ifconfig” [Refer 2.2]

```
#!/usr/bin/python3
from scapy.all import *

def print_pkt(pkt):
pkt.show()
pkt = sniff(iface='br-
c93733e9f913',filter='icmp',prn=print_pkt)
```

Task 1.1A. The above program sniffs packets. For each captured packet, the callback function print_pkt() will be invoked; this function will print out some of the information about the packet. Run the program with the root privilege and demonstrate that you can indeed capture packets. After that, run the program again, but without using the root privilege; describe and explain your observations.

Task 1.1B Scapy's filtering (Extra points!)

- Understand the syntax and usage of BPF (Berkeley Packet Filter)
- Capture only the ICMP packet (*3 extra points*)
- Capture any TCP packet that comes from a particular IP and with a destination port number 23. (*3 extra points*)
- Capture packets comes from or to go to a particular subnet. You can pick any subnet, such as 128.230.0.0/16; you should not pick the subnet that your VM is attached to. (*4 extra points*)

- To obtain extra credits, for each subtask, please attach both the code you modified, and the Scapy's output details.

Packet Spoofing

- When some critical information in the packet is forged, we refer to it as packet spoofing.
- Many network attacks rely on packet spoofing.
- Let’s see how to send packets with spoofing.

Task 1.2 Spoofing ICMP Packets (30 points)

- How to spoof an ICMP packets

```
>>> from scapy.all import *
>>> a = IP()
>>> a.dst = ’10.0.2.3’
>>> b = ICMP()
>>> p = a/b
>>> send(p)
. Sent 1 packets.
```

- Make any necessary change to the sample code, and then demonstrate that you can spoof an ICMP echo request packet with an arbitrary source IP address.
- Use Wireshark to observe whether our request will be accepted by the receiver.
- ***Submit code details, output from code and Wireshark screenshot to get full points***

Task 1.3 Traceroute (40 points)

- Time-To-Live (TTL)
- Uses Scapy to estimate the distance, in terms of number of routers, between your VM and a selected destination
- Example python program:

```
>>> a = IP()
>>> a.dst = ’1.2.3.4’
>>> a.ttl = 3
>>> b = ICMP()
>>> send(a/b)
```

Task: Write your tool to perform the entire procedure automatically (Send+ Receive)

HINT

- Use `sr1`() in Scapy to receive response from the destination i.e., to check if the IP packet has reached the destination or not
- Check for ICMP type in the response packet. If the packet reaches the destination, type should be 0
  https://networkdirection.net/articles/network-theory/icmptypes/

Sniffing and-then Spoofing

- In many situations, we need to capture packets first, and then spoof a response based on the captured packets.
- Procedure (using ICMP as example)
  - Use PCAP API to capture the packets of interests
  - Make a copy from the captured packet
  - Replace the data field with a new message and swap the source and destination fields
  - Send out the spoofed reply

Task 1.4: Sniffing and Then Spoofing Using Scapy (Extra points!)

- ✓ In this task, you will combine the sniffing and spoofing techniques to implement the following sniff-and then-spoof program.
- ✓ You need two machines on the same LAN: the VM and the user container. From the user container, you ping an IP X. This will generate an ICMP echo request packet. If X is alive, the ping program will receive an echo reply, and print out the response.
- ✓ Your sniff-and-then-spoof program runs on the VM, which monitors the LAN through packet sniffing. Whenever it sees an ICMP echo request, regardless of what the target IP address is, your program should immediately send out an echo reply using the packet spoofing technique.
- ✓ You need to use Scapy to do this task. In your report, you need to provide evidence to demonstrate that your technique works.

- ✓ Show original code and output for each IP you check (i) a non-existing host on the Internet (3 points), ii) a non-existing host on the LAN (3 points), and iii) an existing host on the Internet (4 points).

What you need to submit

- ✓ A detailed **lab report** that should
  - Describe what you have done, what you have observed, and how you interpret the results.
  - Include evidences to support the observations. Evidences include packet traces, screenshots, etc.
  - Reports should also list the important code snippets with explanations. Simply attaching code without any explanation will not receive credits.

References

- https://seedsecuritylabs.org/labsetup.html
- https://github.com/seed-labs/seed-labs/blob/master/manuals/vm/seedvm-manual.md
- https://github.com/seed-labs/seed-labs/blob/master/manuals/docker/SEEDManual-Container.md
- https://seedsecuritylabs.org/Labs_20.04/Files/Sniffing_Spoofing/Sniffing_Spoofing.pdf