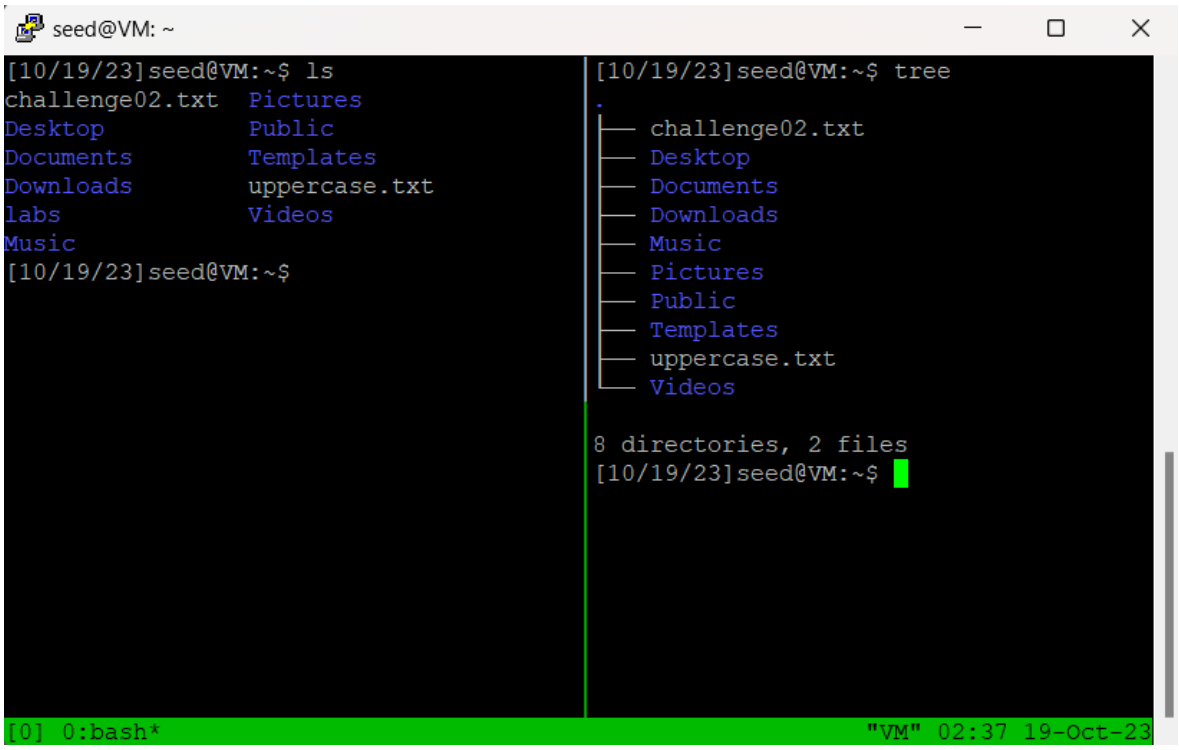
Nguyen Viet Hoang Nam

ITITIU19162

System and Network Security

Lab 1

Q1. Split your terminal vertically in tmux. Run ls on the left panel and tree on the right.



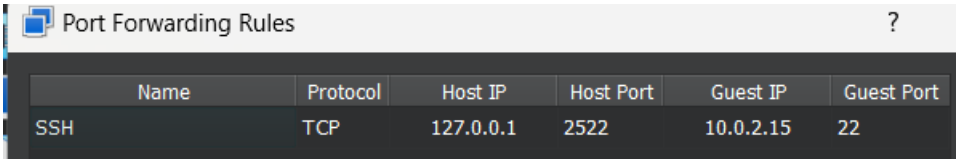
Q2. Run

```
02:45:42.218256 IP _gateway.50449 > VM.ssh: Flags [.], ack 20681232, win 65535, length 0
02:45:42.218279 IP _gateway.50449 > VM.ssh: Flags [.], ack 20682692, win 65535, length 0
02:45:42.218285 IP _gateway.50449 > VM.ssh: Flags [.], ack 20684152, win 65535, length 0
02:45:42.218289 IP _gateway.50449 > VM.ssh: Flags [.], ack 20684160, win 65535, length 0
02:45:42.218464 IP _gateway.50449 > VM.ssh: Flags [.], ack 20685620, win 65535, length 0
02:45:42.219602 IP VM.ssh > _gateway.50449: Flags [P.], seq 20688064:20691904, ack 109681, win
65535, length 3840
02:45:42.219916 IP _gateway.50449 > VM.ssh: Flags [.], ack 20689524, win 65535, length 0
02:45:42.219935 IP _gateway.50449 > VM.ssh: Flags [.], ack 20690984, win 65535, length 0
02:45:42.219941 IP _gateway.50449 > VM.ssh: Flags [.], ack 20691904, win 65535, length 0
02:45:42.221090 IP VM.ssh > _gateway.50449: Flags [P.], seq 20691904:20695808, ack 109681, win
65535, length 3904
02:45:42.221608 IP _gateway.50449 > VM.ssh: Flags [.], ack 20693364, win 65535, length 0
02:45:42.221633 IP _gateway.50449 > VM.ssh: Flags [.], ack 20694824, win 65535, length 0
02:45:42.221639 IP _gateway.50449 > VM.ssh: Flags [.], ack 20695808, win 65535, length 0
02:45:42.222761 IP VM.ssh > _gateway.50449: Flags [P.], seq 20695808:20699712, ack 109681, win
65535, length 3904
02:45:42.223197 IP _gateway.50449 > VM.ssh: Flags [.], ack 20697268, win 65535, length 0
02:45:42.223222 IP _gateway.50449 > VM.ssh: Flags [.], ack 20698728, win 65535, length 0
02:45:42.223229 IP _gateway.50449 > VM.ssh: Flags [.], ack 20699712, win 65535, length 0
02:45:42.224028 IP VM.ssh > _gateway.50449: Flags [P.], seq 20699712:20703616, ack 109681, win
65535, length 3904
02:45:42.224510 IP _gateway.50449 > VM.ssh: Flags [.], ack 20701172, win 65535, length 0
02:45:42.224525 IP _gateway.50449 > VM.ssh: Flags [.], ack 20702632, win 65535, length 0
02:45:42.224529 IP _gateway.50449 > VM.ssh: Flags [.], ack 20703616, win 65535, length 0
02:45:42.224988 IP VM.ssh > _gateway.50449: Flags [P.], seq 20703616:20707456, ack 109681, win
65535, length 3840
02:45:42.225320 IP _gateway.50449 > VM.ssh: Flags [.], ack 20705076, win 65535, length 0
02:45:42.225336 IP _gateway.50449 > VM.ssh: Flags [.], ack 20706536, win 65535, length 0
02:45:42.225340 IP _gateway.50449 > VM.ssh: Flags [.], ack 20707456, win 65535, length 0
02:45:42.225615 IP _gateway.50449 > VM.ssh: Flags [P.], seq 109681:109745, ack 20707456, win 65
535, length 64
02:45:42.225894 IP _gateway.50449 > VM.ssh: Flags [P.], seq 109745:109905, ack 20707456, win 65
535, length 160
02:45:42.226505 IP VM.ssh > _gateway.50449: Flags [.], ack 109905, win 65535, length 0
02:45:42.226599 IP VM.ssh > _gateway.50449: Flags [P.], seq 20707456:20707504, ack 109905, win
65535, length 48
02:45:42.226870 IP _gateway.50449 > VM.ssh: Flags [.], ack 20707504, win 65535, length 0
02:45:42.227494 IP VM.ssh > _gateway.50449: Flags [P.], seq 20707504:20711408, ack 109905, win
65535, length 3904
02:45:42.227796 IP _gateway.50449 > VM.ssh: Flags [.], ack 20708964, win 65535, length 0
02:45:42.227807 IP _gateway.50449 > VM.ssh: Flags [.], ack 20710424, win 65535, length 0
02:45:42.227810 IP _gateway.50449 > VM.ssh: Flags [.], ack 20711408, win 65535, length 0
^C
33285 packets captured
34008 packets received by filter
719 packets dropped by kernel
[10/19/23]seed@VM:~$
```

Q3. tcpdump continuously prints packet headers on the terminal. Can you identify their source?

```
03:48:18.963066 IP (tos 0x10, ttl 64, id 50947, offset 0, flags [DF], proto TCP (6), length 328)
    10.0.2.15.22 > 10.0.2.2.50449: Flags [P.], cksum 0x194b (incorrect -> 0x2a4a), seq 1880432:1880720, ack 4641, win 65535, length 288
03:48:18.963485 IP (tos 0x10, ttl 64, id 50948, offset 0, flags [DF], proto TCP (6), length 536)
    10.0.2.15.22 > 10.0.2.2.50449: Flags [P.], cksum 0x1a1b (incorrect -> 0x908f), seq 1880720:1881216, ack 4641, win 65535, length 496
03:48:18.963836 IP (tos 0x0, ttl 64, id 15991, offset 0, flags [none], proto TCP (6), length 40)
    10.0.2.2.50449 > 10.0.2.15.22: Flags [.], cksum 0xb024 (correct), ack 1880720, win 65535, length 0
```

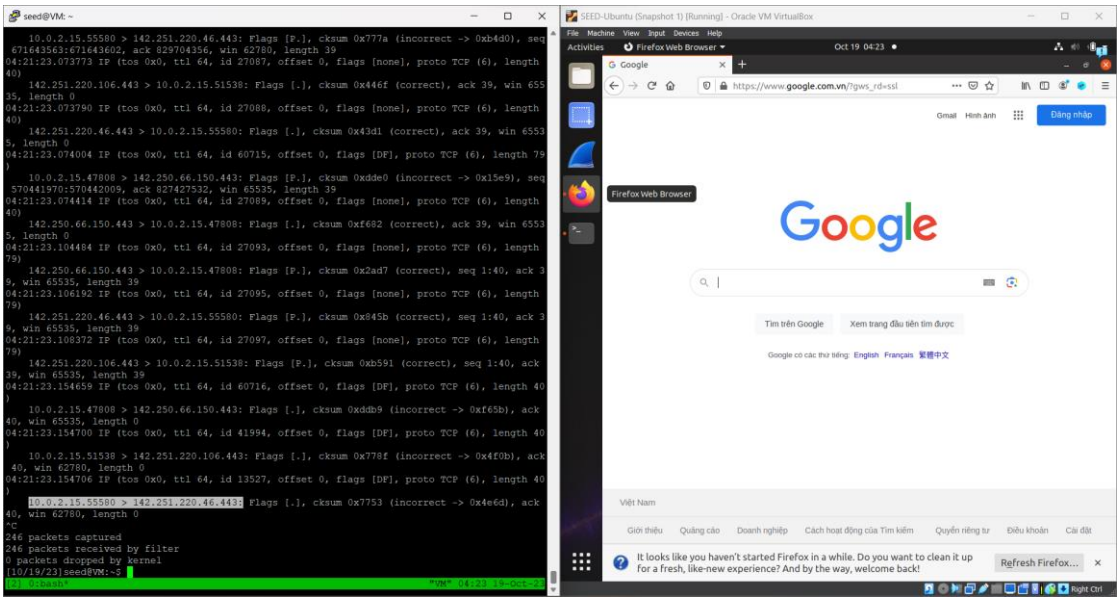
The first package in the first picture show that this is a package sent between a source IP of 10.0.2.15 with port 22 which is belong to the SEED virtual machine network and the destination IP is 10.0.2.2 with port 50449 which is a special alias of the host loopback interface – 127.0.0.1



Port Forwarding Rules

Name	Protocol	Host IP	Host Port	Guest IP	Guest Port
SSH	TCP	127.0.0.1	2522	10.0.2.15	22

Q4. To prevent those packets, include a filter in the command from the previous question.



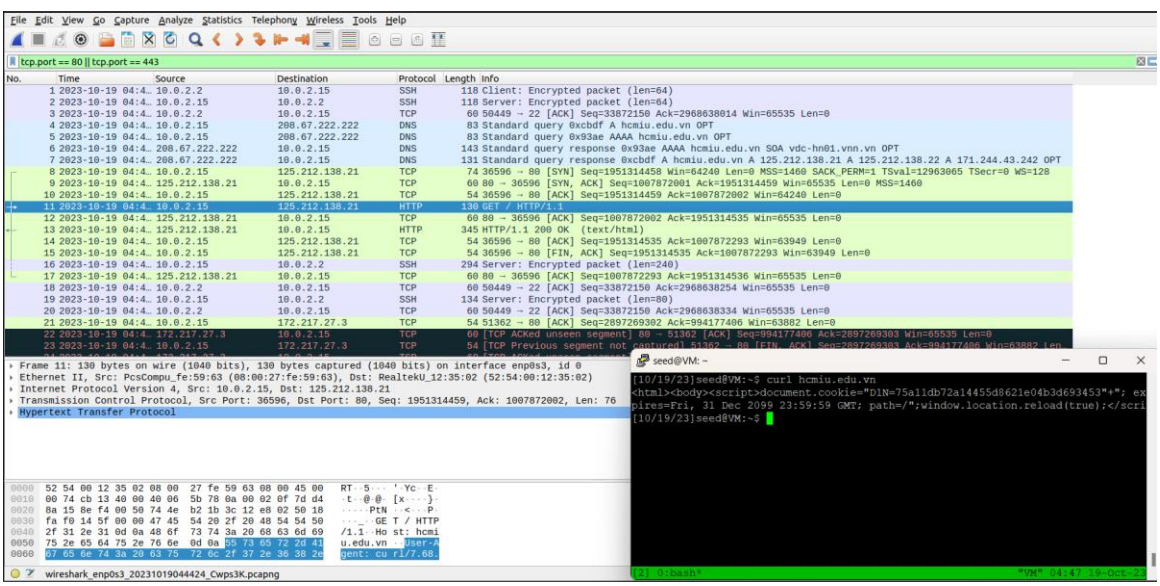
The command: “sudo tcpdump -i any port 443 -vn” which will only display the package that have port of 443 which will be the HTTPS connection.

And the result indicated that the package was sent from the local SEED to Google IP (142.251.220.46:443)

Q5. Use tcpdump to capture web traffic. Test with curl or wget.

Here I will use curl to http://google.com.vn and the other panel captured the filtered package with port of 80.

Q6. Use Wireshark to capture only web traffic. Test by downloading a website with curl or wget.



Filtering: tcp.port == 80 || tcp.port == 443

I curl the http://hcmiu.edu.vn and the Wireshark captured the GET package and returned a 200 OK code indicated that it was successful.

Q7. Capture the ARP traffic and analyze the source MAC addresses, source IP addresses, destination MAC addresses, destination IP addresses of the packets (ARP request/reply packets). Generate ARP packets by pinging an address in the same network.

```
23 2023-10-21 07:1… PcsCompu_fe:59:63    RealtekU_12:35:02   ARP   42 Who has 10.0.2.2? Tell 10.0.2.15
24 2023-10-21 07:1… RealtekU_12:35:02    PcsCompu_fe:59:63   ARP   60 10.0.2.2 is at 52:54:00:12:35:02

▸ Frame 23: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface enp0s3, id 0
▸ Ethernet II, Src: PcsCompu_fe:59:63 (08:00:27:fe:59:63), Dst: RealtekU_12:35:02 (52:54:00:12:35:02)
▾ Address Resolution Protocol (request)
    Hardware type: Ethernet (1)
    Protocol type: IPv4 (0x0800)
    Hardware size: 6
    Protocol size: 4
    Opcode: request (1)
    Sender MAC address: PcsCompu_fe:59:63 (08:00:27:fe:59:63)
    Sender IP address: 10.0.2.15
    Target MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00)
    Target IP address: 10.0.2.2
```

It showed the source MAC address which is SEED - 08:00:27:fe:59:63 and the source IP 10.0.2.15 when sending an ARP.

```
seed@VM: ~
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.2.15  netmask 255.255.255.0  broadcast 10.0.2.255
        inet6 fe80::48d5:f93f:c20c:f35e  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:fe:59:63  txqueuelen 1000  (Ethernet)
        RX packets 581  bytes 346971 (346.9 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 442  bytes 65169 (65.1 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 166  bytes 12736 (12.7 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 166  bytes 12736 (12.7 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

[10/21/23]seed@VM:~$
[0] 0:bash*                                          "VM" 07:06 21-Oct-23
```

Q8. In order to complete this task, you need to capture the DHCP packets (both request and reply) and then analyze the source MAC addresses, source IP addresses, destination MAC addresses, and destination IP addresses of those packets. To achieve this, you may need to disable and re-enable the interface using the following commands: ip link set dev enp0s3 down and ip link set dev enp0s3 up

No.	Time	Source	Destination	Protocol	Length	Info
1	2023-10-21 07:2…	10.0.2.2	10.0.2.15	SSH	118	Client: Encrypted packet (len=64)
2	2023-10-21 07:2…	10.0.2.15	10.0.2.2	SSH	118	Server: Encrypted packet (len=64)
3	2023-10-21 07:2…	10.0.2.2	10.0.2.15	TCP	60	55488 → 22 [ACK] Seq=249225302 Ack=3554558549 Win=65535 Len=0
4	2023-10-21 07:2…	0.0.0.0	255.255.255.255	DHCP	342	DHCP Discover - Transaction ID 0xf8c9613a
5	2023-10-21 07:2…	10.0.2.2	10.0.2.15	DHCP	590	DHCP Offer - Transaction ID 0xf8c9613a
6	2023-10-21 07:2…	0.0.0.0	255.255.255.255	DHCP	342	DHCP Request - Transaction ID 0xf8c9613a
7	2023-10-21 07:2…	10.0.2.2	10.0.2.15	DHCP	590	DHCP ACK - Transaction ID 0xf8c9613a
8	2023-10-21 07:2…	10.0.2.15	10.0.2.2	SSH	134	Server: Encrypted packet (len=80)
9	2023-10-21 07:2…	10.0.2.15	10.0.2.2	SSH	118	Server: Encrypted packet (len=64)
10	2023-10-21 07:2…	10.0.2.2	10.0.2.15	TCP	60	55488 → 22 [ACK] Seq=249225302 Ack=3554558629 Win=65535 Len=0
11	2023-10-21 07:2…	10.0.2.2	10.0.2.15	TCP	60	55488 → 22 [ACK] Seq=249225302 Ack=3554558693 Win=65535 Len=0
12	2023-10-21 07:2…	10.0.2.15	10.0.2.2	SSH	150	Server: Encrypted packet (len=96)
13	2023-10-21 07:2…	10.0.2.15	10.0.2.2	SSH	118	Server: Encrypted packet (len=64)
14	2023-10-21 07:2…	10.0.2.2	10.0.2.15	TCP	60	55488 → 22 [ACK] Seq=249225302 Ack=3554558789 Win=65535 Len=0
15	2023-10-21 07:2…	10.0.2.2	10.0.2.15	TCP	60	55488 → 22 [ACK] Seq=249225302 Ack=3554558853 Win=65535 Len=0
16	2023-10-21 07:2…	10.0.2.15	10.0.2.2	SSH	150	Server: Encrypted packet (len=96)
17	2023-10-21 07:2…	10.0.2.2	10.0.2.15	TCP	60	55488 → 22 [ACK] Seq=249225302 Ack=3554558949 Win=65535 Len=0

```
▸ Frame 4: 342 bytes on wire (2736 bits), 342 bytes captured (2736 bits) on interface enp0s3, id 0
▸ Ethernet II, Src: PcsCompu_fe:59:63 (08:00:27:fe:59:63), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
▾ Internet Protocol Version 4, Src: 0.0.0.0, Dst: 255.255.255.255
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  ▸ Differentiated Services Field: 0x10 (DSCP: Unknown, ECN: Not-ECT)
    Total Length: 328
    Identification: 0x0000 (0)
  ▸ Flags: 0x0000
    Fragment offset: 0
    Time to live: 128
    Protocol: UDP (17)
    Header checksum: 0x3996 [validation disabled]
    [Header checksum status: Unverified]
    Source: 0.0.0.0
```

At first it would send a boardcast massage (to IP: 255.255.255.255, MAC: ff:ff:ff:ff:ff:ff) with its source MAC address - 08:00:27:fe:59:63 to “Discover” the DHCP then the DHCP “Offer” it the preferred IP address. After that, the host would “Request” that IP and got it.

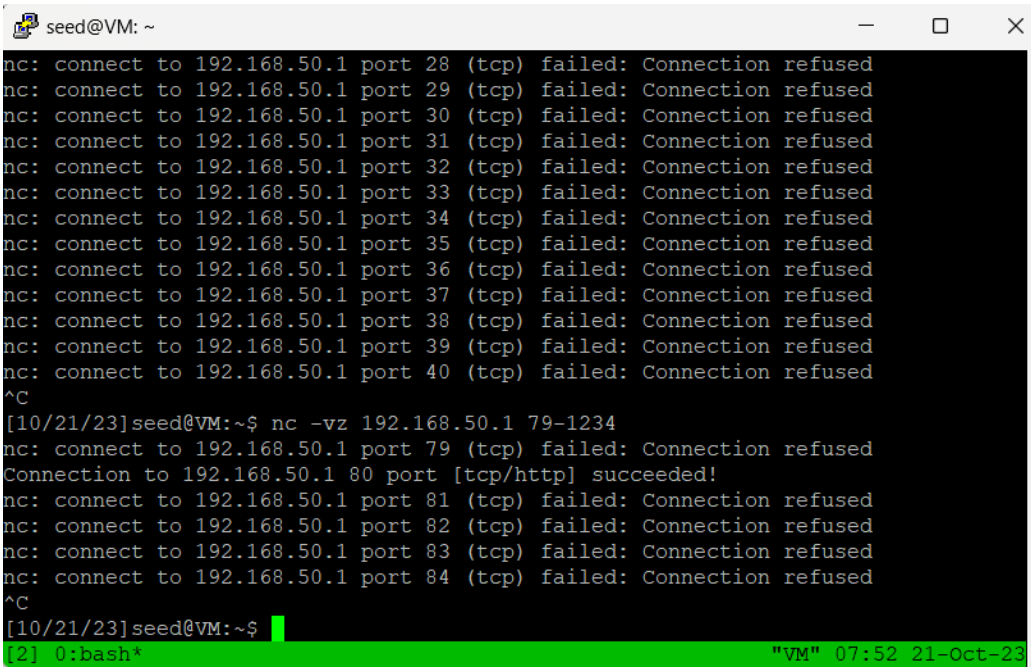
Q9. Capture the DNS packets when you resolve a domain name with host command to identify port, transport layer type, and DNS record type.

No.	Time	Source	Destination	Protocol	Length	Info
1	2023-10-21 07:3…	10.0.2.2	10.0.2.15	SSH	118	Client: Encrypted packet (len=64)
2	2023-10-21 07:3…	10.0.2.15	10.0.2.2	SSH	118	Server: Encrypted packet (len=64)
3	2023-10-21 07:3…	10.0.2.2	10.0.2.15	TCP	60	55488 → 22 [ACK] Seq=249227222 Ack=3554561573 Win=65535 Len=0
4	2023-10-21 07:3…	10.0.2.15	192.168.50.1	DNS	83	Standard query 0x7ea7 AAAA hcmiu.edu.vn OPT
5	2023-10-21 07:3…	10.0.2.15	10.0.2.2	SSH	230	Server: Encrypted packet (len=176)
6	2023-10-21 07:3…	10.0.2.2	10.0.2.15	TCP	60	55488 → 22 [ACK] Seq=249227222 Ack=3554561749 Win=65535 Len=0
7	2023-10-21 07:3…	192.168.50.1	10.0.2.15	DNS	143	Standard query response 0x7ea7 AAAA hcmiu.edu.vn SOA vdc-hn01.vnn.vn OPT
8	2023-10-21 07:3…	10.0.2.15	10.0.2.2	SSH	406	Server: Encrypted packet (len=352)
9	2023-10-21 07:3…	10.0.2.2	10.0.2.15	TCP	60	55488 → 22 [ACK] Seq=249227222 Ack=3554562101 Win=65535 Len=0
10	2023-10-21 07:3…	10.0.2.15	10.0.2.2	SSH	150	Server: Encrypted packet (len=96)
11	2023-10-21 07:3…	10.0.2.2	10.0.2.15	TCP	60	55488 → 22 [ACK] Seq=249227222 Ack=3554562197 Win=65535 Len=0

```
    Fragment offset: 0
    Time to live: 64
    Protocol: UDP (17)
    Header checksum: 0xb345 [validation disabled]
    [Header checksum status: Unverified]
    Source: 10.0.2.15
    Destination: 192.168.50.1
▾ User Datagram Protocol, Src Port: 60916, Dst Port: 53
    Source Port: 60916
    Destination Port: 53
    Length: 49
    Checksum: 0xfefa [unverified]
    [Checksum Status: Unverified]
    [Stream index: 0]
  ▸ [Timestamps]
▾ Domain Name System (query)
    Transaction ID: 0x7ea7
  ▸ Flags: 0x0100 Standard query
    Questions: 1
    Answer RRs: 0
    Authority RRs: 0
    Additional RRs: 1
  ▾ Queries
    ▸ hcmiu.edu.vn: type AAAA, class IN
```

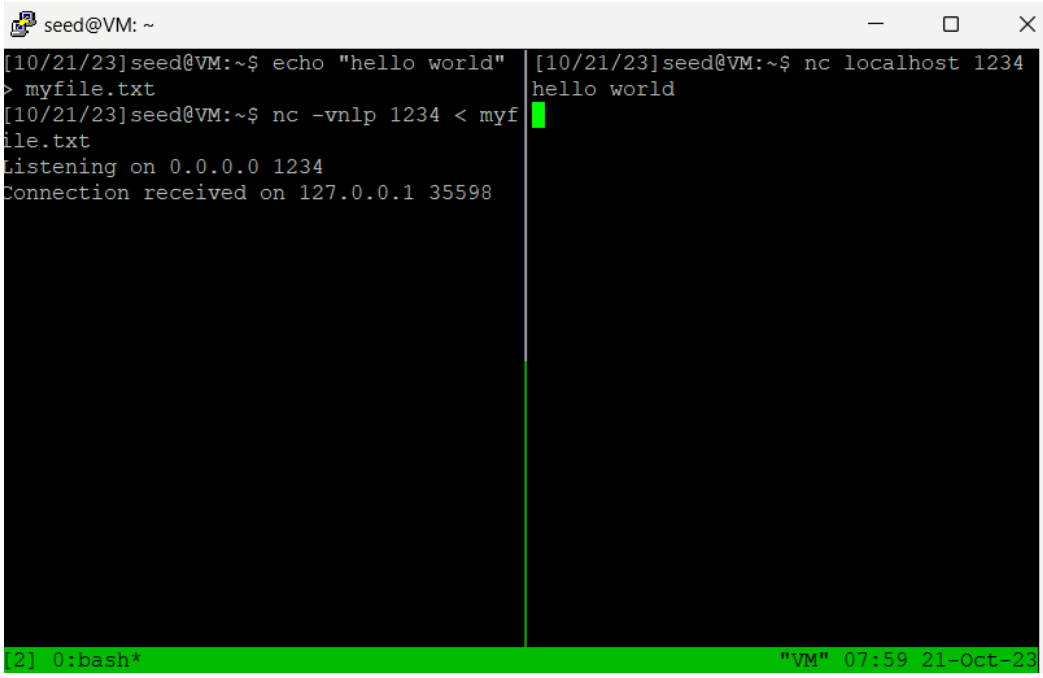
In the initial DNS package it was send from the SEED with source port of 60916 to port 53 and was using transport layer of UDP, record type of AAAA.

Q10. Write a script to scan well-known ports (0 - 1023).

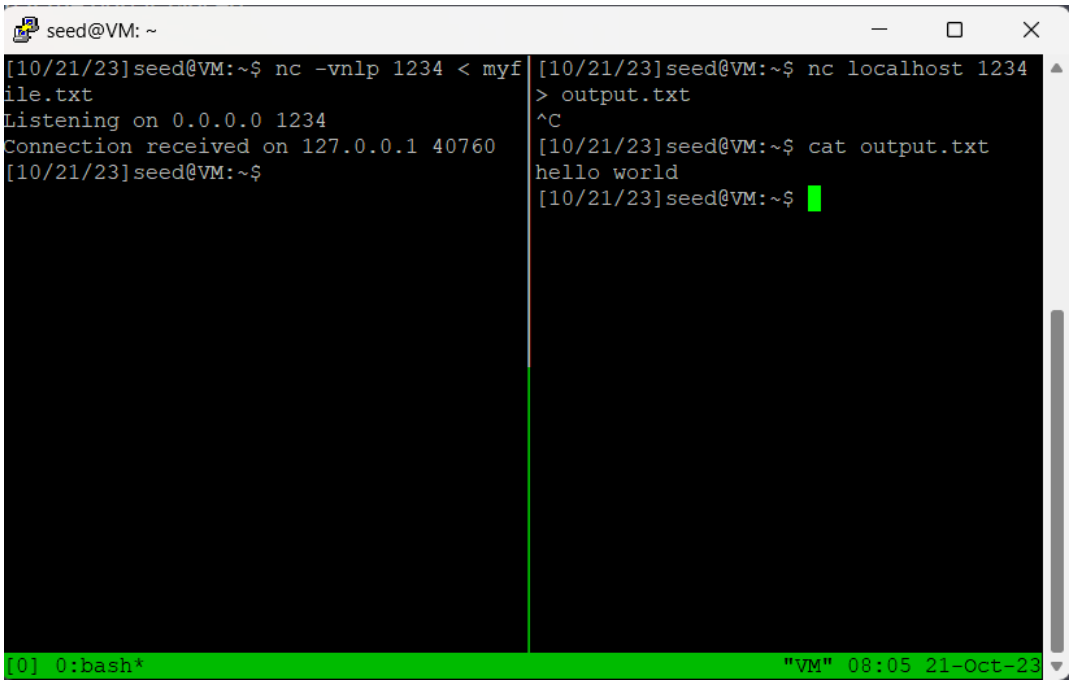


I used: nc -vz 192.168.50.1 1-1234

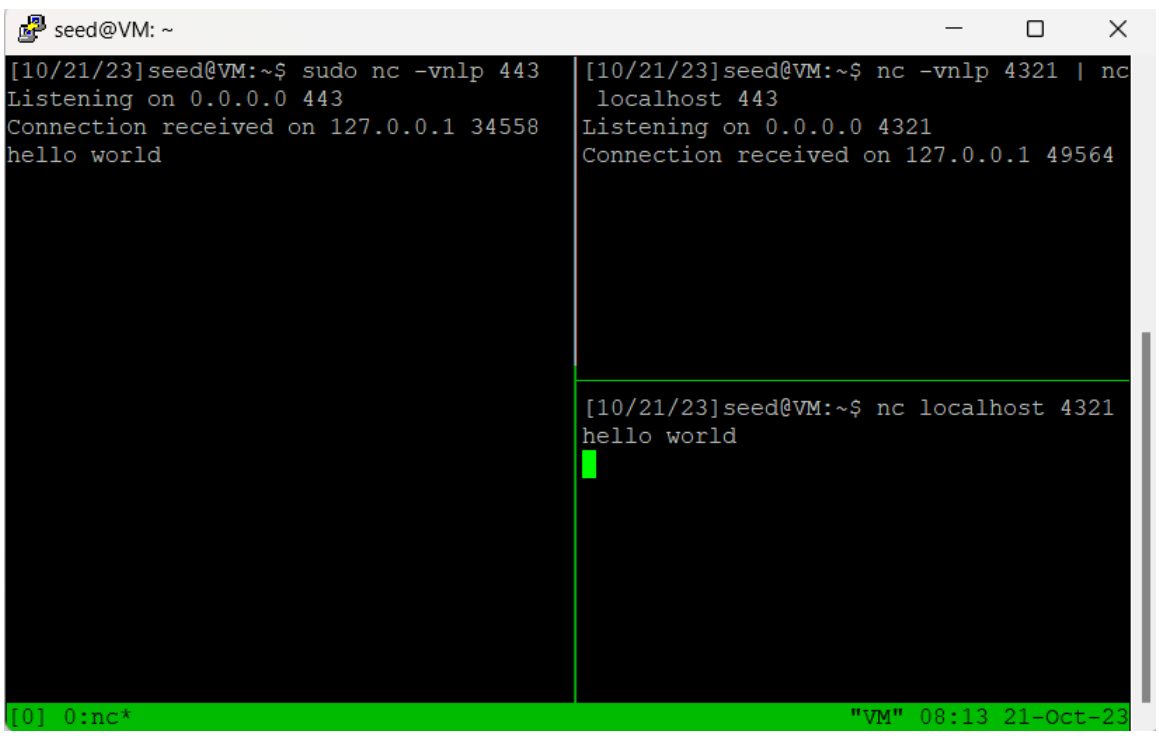
Q11. Send contents of a file from client to server.



Q12. Save the contents received to a file. Hint: use the redirection .



Q13. Execute server and client commands to demonstrate that the tunneling is functioning correctly.



The server listenning on port 443, the tunneling machine will listen from port 4321 and redirect the traffic to port 443. The client will connect to the tunneling machine port 4321 and get access to the server.

Q14. What does command cat do without any argument?

The command “cat” alone will open up a blank line, wait for the user to type in their input and when hit Enter, it will display what they have just entered above in the next line.

Q15. Clarify how the two commands used to create a backdoor function. Additionally, explain how these commands work in detail?

The first command will “remove” the file located at “/tmp/f” with the -f flag to force it to delete without any confirmation. Then, it continue to create a named pipe (FIFO) at “/tmp/f” which will be used to process the communication between the attacker and user.

The second command will wait for the input from the user to “/tmp/f”. Then, it will start a new instance of the shell in the interacting mode for the attacker to do shell prompt. After that, it will open up a sever on localhost port 1234 and redirect the output of netcat back to “/tmp/f” which will go back to the named pipe for the attacker to prompt again.