Midterm Solution

1 THE WALL (25%) ★★★

Subtask 1 - Illegal (10%)

Prepare

As hints we gave,

1. Goto `Diagnostics > Packet Capture`.
2. Set count to larger number such as 1000.
3. Record for a while.
4. Download the packet dump and use `wireshark` to open it. (We gave this hint during the exam.)

Find the packets –Solution 1

There are lots of packets between `10.0.5.1` and `10.0.8.1`. To check if all packets are transmitted between these two IPs, apply some filter such as `ip.dst != 10.0.8.1 and ip.src != 10.0.8.1`. Now, we have few packets left which connects to other IPs.

Find the packets –Solution 2

If you scroll through all the packets quickly, you may notice some parts of packet have different colors. (DNS packets colored as very distinctive cyan)

Bot behavior

If you capture both UDP and TCP, you can easily figure out hostnames that bot wants to connect in DNS packet. If you doesn' t capture DNS packets due to settings or some DNS cache issue, you have to find it out from HTTP Headers in the packets.

It turns out that the bot is trying to connect `yahoo`, `google`, `stackoverflow`, `thepiratebay`. The Pirate Bay is a well-known torrent search engine which contains lots of both legal and illegal contents. According to task descriptions, block traffics to this site to complete this task.

Subtask 2 - All in One (10%)

Solution 1 (preferable)

Scanning all ports of `10.0.8.1` to find out its response. You can either use some tools such as nmap, or write a small script by yourself. You will see port 42 replies `SSH-4VARTSmkwgjCwtrT` which contains our secret token.

Block the traffics to VLAN 8 at port 42 to complete this task. Be careful about the difference between `source` and `destination`.

Solution 2

Since our bot needs to check whether the server sends out our secret token, the token should appears in our packet dump (see Subtask 1 for capturing packets). Filtering with `frame contains "4VARTSmkwgjCwtrT"` and the only packet left is at port 42. (This filter is introduced in HW1)

Subtask 3 - SNMP (5%)

SNMP Daemon on pfSense

Goto **Services** > **SNMP**.

- Section **SNMP Daemon**
  - **Enable**: Check to enable SNMP Daemon.
- Section SNMP Daemon Settings
  - **Polling Port**: Leave it with the default port 161.
  - **System Location**: Anything meaningful. (This does not affect the functionality.
  - **System Contact**: Anything meaningful. (This does not affect the functionality.)
  - **Read Community Sting**: This is like the password to access the SNMP information on this host, so it is highly recommended to change it from default, e.g., set it to NASA2018.
- Section **SNMP Traps Enable**
  - **Enable**: For this subtask, SNMP trap doesn’t need to be enabled.
- Section **SNMP Modules**
  - Leave all options checked as default.
- Section **Interface Binding**
  - **Bind Interfaces**: Setting to All will do for this subtask. But in practice, it is better to have a management VLAN where your SNMP manager should be set up and setting this option to the management VLAN.

SNMP Manager

Assuming using the provided Ubuntu 16.04 ova.

1. Update package list first: `sudo apt update`
2. Install some required packages for this task: `sudo apt install snmp snmp-mibs-downloader`
3. Unzip and put the mibs under a directory.

```
wget https://www.csie.ntu.edu.tw/~vegetable/nasa_18spring/midterm/mibs.zip
unzip mibs.zip
mkdir ~/.snmp
mv mibs ~/.snmp
```

4. Edit **/etc/snmp/snmp.conf**.

   ```
   mibdirs +/home/nasa2018/.snmp/mibs
   defVersion 2c
   defCommunity NASA2018
   alias pfSense udp:[pfSense IP]:161
   ```

   - MIB files contain vender defined MIB tree. The first line includes the provided pfSense MIB files making the SNMP manager understand MIB names like `BEGEMOT-PF-MIB::pfInterfacesIfTable`.
   - The second line specifies the version that we are using, so we don' t have to type the `-v` flag everytime when using `snmp` related commands.
   - The third line set the community string(password) which should match the one set in pfSense. Adding this line, we can omit the `-c` flag when using `snmp` related commands.
   - The fourth line set an alias for pfSense, so we can use `alias:pfSense` to replace the host IP in the commands.

5. The configuration is now done and there should be no problem issuing the commands specified in the problem.

2 SSH Leak? (20%) ★★★★★

- If you observe your ssh connection after the connection is established, you' ll find out that each time you hit the keyboard it sends a packet of length 90 since the session is interactive.
- So you know the number of characters that the user entered is 28.
- You also know that the user entered:

  ```
  $ ./verify [password]
  $ exit
  ```

- That is `len("./verify ") + len(password) + len("\n") + len("exit\n") == 28 => 9 + len(password) + 1 + 5 + 1 == 28 => len(password) == 12`
  - Note that ssh connection sends one more packet of length 90 after exit, so there is a + 1 at the end.
- Once you find out the length, knowing that the password contains only 0 and 1, just brute force the 2^{12} possibilities.

Don't worry about this in real life scenario! Leaking the length is actually not a big deal. The tittle of the problem is just a click bait :)

BTW, the first version of **verify.c** was found 2 race condition bug. One could be used to bypass the limit of the number of tries, and the other could be used to compromise the whole program (You can get the message ”You got it!” without knowing the right password). I added the rule ”It is allowed to attack the server” since I am looking forward to find someone who could find another bug. You could even use `spectre` (the intel cpu bug which is still not fixed) or another privilege escalation exploit against the server. However, it seems like nobody has attempted to do these... So sad :(

3 Agent J (25%) ★★

1. `192.168.0.137`. because he is trying to brute force the password, and he POSTs a backdoor to the server.
2. The backdoor utilizes **ICMP tunnel** to encode the flag in `base64` format in the ICMP message. `NASA{u_R_a_Wiresh@rk_K1ng}`.
3. The destination IP of ICMP message is sent to `5.149.64.176`. IPv4 has geographic locality. Just google some ip lookup website and you'll find the IP is in Bosnia and Herzegovina(波士尼亞與赫塞哥維納).

4 Cisco packet tracer (25%) ★★

On Core:

```
Switch> en
Switch# conf t
Switch(config)# int port-channel 1
Switch(config-if)# switchport mode trunk
Switch(config-if)# exit
Switch(config)# int range gi0/1-2
Switch(config-if)# switchport mode trunk
Switch(config-if)# channel-group 1 mode active
Switch(config-if)# exit
Switch(config)# int vlan 99
Switch(config-if)# ip address 192.168.99.10 255.255.255.0
Switch(config-if)# exit
Switch(config)# int fa0/1
Switch(config-if)# switchport mode access
Switch(config-if)# switchport access vlan 99
Switch(config-if)# exit
Switch(config)# hostname Core
Core(config)# enable secret cisco
Core(config)# username admin password nimda
Core(config)# line vty 0
Core(config-line)# login local
Core(config-line)# exit
```

On Switch0:

```
Switch> en
Switch# conf t
Switch(config)# int port-channel 1
Switch(config-if)# switchport mode trunk
Switch(config-if)# exit
Switch(config)# int port-channel 2
Switch(config-if)# switchport mode trunk
Switch(config-if)# exit
Switch(config)# int range gi0/1-2
Switch(config-if)# switchport mode trunk
Switch(config-if)# channel-group 1 mode active
Switch(config-if)# exit
Switch(config)# int range fa0/1-2
Switch(config-if)# switchport mode trunk
Switch(config-if)# channel-group 2 mode active
Switch(config-if)# exit
Switch(config)# int vlan 8
Switch(config-if)# exit
Switch(config)# int vlan 30
Switch(config-if)# exit
Switch(config)# int range fa0/3-4
Switch(config-if)# switchport mode access
Switch(config-if)# switchport access vlan 30
Switch(config-if)# exit
Switch(config)# int fa0/5
Switch(config-if)# switchport mode access
Switch(config-if)# switchport access vlan 8
Switch(config-if)# exit
```

On Switch1:

```
Switch> en
Switch# conf t
Switch(config)# int port-channel 1
Switch(config-if)# switchport mode trunk
Switch(config-if)# exit
Switch(config)# int range fa0/1-2
Switch(config-if)# switchport mode trunk
Switch(config-if)# channel-group 1 mode active
Switch(config-if)# exit
Switch(config)# int vlan 8
Switch(config-if)# exit
Switch(config)# int vlan 30
Switch(config-if)# exit
Switch(config)# int range fa0/3-4
Switch(config-if)# switchport mode access
Switch(config-if)# switchport access vlan 8
Switch(config-if)# exit
Switch(config)# int fa0/
5Switch(config-if)# switchport mode access
Switch(config-if)# switchport access vlan 30
Switch(config-if)# exit
```

5 Simple CSV Viewer (20%) ★★★★★

Please refer to the sample answer here.

6 Matryoshka Doll (20%) ★★★★

Please find the sample `doll.sh` here.

7 I Get My Job! (20%) ★★★

Please find the `monitor.sh` and its included scripts here.

8 Матрёшка (20%) ★★★★

Note that, some of the urls provided here are shortened, please find the completed version in question.

On the VM host

Set up bridge with `nmtui`.

```
yum install libvirt virt-install qemu-kvm
systemctl start libvirtd
```

Download anaconda kickstart script from link provided in HW4

```
curl "https://gist.githubusercontent.com/.../sa-hw4-anaconda.cfg" > vm.cfg
```

and fill the two blanks.

```
network --bootproto=dhcp
services --enabled="chronyd, serial-getty@ttyS0"
```

Prepare virtual disk and installation image.

```
cd /var/lib/libvirt/images/
curl "http://ftp.twaren.net/Linux/CentOS/7/isos/x86_64/CentOS-7..." > \
     centos7.iso
qemu-img create -f qcow2 vm1.qcow2 16G
```

Create the first VM.

```
virt-install \
    --name=vm1 \
    --disk path=/var/lib/libvirt/images/vm1.qcow2 \
    --graphics spice \
    --vcpus=2 --ram=1024 \
    --location=/var/lib/libvirt/images/centos7.iso \
    --network bridge=nm-bridge,model=virtio \
    --initrd-inject=/root/vm.cfg --extra-args "ks=file:/vm.cfg"
```

Congrats, you have got 40%!

In the VM

Stop firewalld

```
systemctl stop firewalld
```

Copy and paste from lab slide:

```
yum install yum-utils device-mapper-persistent-data
yum-config-manager --add-repo \
     https://download.docker.com/linux/centos/docker-ce.repo
yum install docker-ce
systemctl start docker
systemctl enable docker
```

Pull CentOS image

```
docker pull centos
```

or download and load it from provided link

```
curl http://linux1.csie.ntu.edu.tw:2018/centos.tar > centos.tar
docker load -i centos.tar
```

Then create docker image

```
mkdir ws
```

Create and edit Dockerfile

```
FROM centos
RUN yum install -y openssh-server
RUN ssh-keygen -A
RUN useradd b07902000
RUN echo "pwd" | passwd b07902000 --stdin
CMD ["/sbin/sshd", "-D"]
```

build it:

```
Docker build -t ws .
```

And then run it

```
docker run -d --name ws1 --publish 1022:22 ws
docker run -d --name ws2 --publish 2022:22 ws
```

Congrats, you have got 90%!

Back to VM host

Shutdown the running VM

```
virsh shutdown vm1
```

and clone it

```
virt clone -o vm1 -n vm2 --auto-clone
```

Start the VMs

```
virsh start vm1
virsh start vm2
```

Get into VMs

```
docker run -d --name ws1 --publish 1022:22 ws
docker run -d --name ws2 --publish 2022:22 ws
docker run -d --name ws3 --publish 3022:22 ws
docker run -d --name ws4 --publish 4022:22 ws
```

on the two VMs respectively.

Congrats, you have got 100%!

9 More on Storage! (25%) ★

Before you start, download `nasa-midterm.ova` from `linux1-4` under `/tmp2`. The username is `nasa` and the password is `nasa2018`.

There should be a network interface for you to SSH into as mentioned in class. Also, there is another network interface which you are allowed to use to access the Internet. You may simply use the command `sudo ifup <interface>` to enable the interface.

Part 1 (10%)

Create the partition table.

```
sudo parted /dev/sdb mklabel gpt
sudo parted /dev/sdb mkpart primary 0% 400Gib set 1 lvm on
sudo parted /dev/sdb mkpart primary 400Gib 100%
```

You'll need to install some package to format FAT32 filesystem, you should also remember to enable your network first.

```
sudo yum install -y dosfstools
sudo mkfs.vfat /dev/sdb2
sudo mkdir /fat
sudo mount /dev/sdb2 /fat
```

Then, the LVM and filesystem part.

```
sudo pvcreate /dev/sdb1 /dev/sdc
sudo vgcreate storage /dev/sdb1 /dev/sdc
sudo lvcreate -n bebi -L 250Gib storage
sudo lvcreate -n inm -L 150Gib storage
sudo mkfs.ext4 /dev/storage/bebi
sudo mkfs.ext4 /dev/storage/inm
sudo mkdir /bebi /inm
sudo mount /dev/storage/bebi /bebi
sudo mount /dev/storage/inm /inm
```

Part 2 (10%)

You should backup everything under `/home`.

```
sudo tar -zcvf backup.tar.gz /home
```

Then you can just easily remove the logical volume and create a smaller one.

```
sudo umount /home
sudo lvremove /dev/centos/home
sudo lvcreate -n home -L 20Gib centos
sudo lvcreate -n images -l 100%FREE centos
sudo mkfs.xfs /dev/centos/home
sudo mkfs.xfs /dev/centos/images
```

Finally, restore the data back.

```
sudo mount /home
sudo tar -zxvf backup.tar.gz -C /
```

Part 3 (5%)

You should clear the partition table first.

```
sudo dd if=/dev/zero of=/dev/sdd count=2000
```

Then, the LVM and filesystem part.

```
sudo vgextend storage /dev/sdd
sudo lvcreate -n csie -l 100%FREE storage
sudo mkfs.ext4 /dev/storage/csie
sudo mkdir /csie
sudo mount /dev/storage/csie /csie
```