Midterm Solution

1 pfSense (25 pts)

- **Subtask 1**: VM of pfSense should have its WAN network card `bridged` to the host one. (In VirtualBox, Setting -> Network -> Adapter 1 -> Attached to -> Bridged Adpator)
- Set up 3 VLAN interfaces on LAN. (`VLAN7`, `VLAN22` and `VLAN99`)
  - Remember to use `static IPv4`.
  - Size of netmask should be less than 31.
  - Remember to enable the interface.
- **Subtask 2**: Enable DHCP servers in `VLAN99` interfaces. (Services -> DHCP server)
- Rules (**Subtask 3&5**):
  - `VLAN7`:
    1. BLOCK any to `this firewall` in 22 and 443 port.
    2. PASS any to any
  - `VLAN22`:
    1. BLOCK any to `this firewall` in 22 and 443 port.
    2. PASS any to any
  - `VLAN99`:
    1. PASS any to any
  - `WAN`:
    1. BLOCK any to `this firewall` in 22 and 443 port.
    2. PASS any to any
- **Subtask 4**:
  Firewall -> NAT -> Port Forward -> add
  - Destination IP = pfSense's WAN IP
  - Destination Port = [1024 - 65535]
  - Redirect IP = IP of your VM in VLAN 22
  - Redirect port = port of your ssh server (usually 22)

To achieve **Subtask 6**, do nothing in outbound NAT settings of pfsense.

2 Wireshark (25 pts)

1. (a) Find addresses of DNS servers in the CSIE department in DHCP ACK packet responded by the DHCP server, which are `140.112.30.21` and `140.112.30.12`.
     - To find DHCP packets, use `bootp` as displaying filter.

   (b) The filter would be:
   `dns && (ip.dst == 140.112.30.21 || ip.dst == 140.112.30.12)`

2. There are 4 stages in DHCP request, and it is confirmed as completed when a client receives `ACK` packet from the server. Simply by `bootp`, we can see those DHCP requests that have not reached the final `ACK` state. Thus, the answer will be shown by checking the MAC addresses in the `bootstrap protocol`, which are as follows.
   - 84:38:35:4e:b4:a0
   - e0:ac:cb:69:44:90
   - 48:4b:aa:00:1c:56
   - d4:f4:6f:98:09:54
   - (78:f8:82:b1:74:70)

3. `http && http.user_agent contains "Mac"`
```
54.243.198.221
206.108.53.86
163.28.5.40
163.28.5.35
```

3 Packet Tracer (25 pts)

```
*****switch0*****

enable
conf t
enable secret q_mao

username admin secret admin (If exists, use 'no username admin')
line vty 0
login local
exit

conf t
int Fa0/1
switchport access vlan 7

int Fa0/2
switchport access vlan 8

int Fa0/24
switchport access vlan 99

int vlan99
ip address 192.168.99.2 255.255.255.0

int range Gi0/1 - 2
switchport mode trunk
channel-group 1 mode active

*****switch 1*****

enable
conf t

int Fa0/1
switchport access vlan 7

int Fa0/2
switchport access vlan 8

int range Gi0/1 - 2
switchport mode trunk
channel-group 1 mode active
```

4 Strange SSH (15 pts)

```
ssh -p 9753 nasa_meow@140.112.30.52 -i private_key_file_in_the_pcap
```

5 Stupid encodings (15 pts)

```
#!/bin/sh
from="Base{32,64}_Is_Stupid_But_Sometimes_Useful"
ans="195a30a1d1561cbc0ae7c488b93d037f6b713354"
for a in base{32,64}; do
    for b in base{32,64}; do
        for c in base{32,64}; do
            for d in base{32,64}; do
                for e in base{32,64}; do
             stupid=$(eval "echo '$from' | $a | $b | $c | $d | $e | sha1sum" | cut -f1 -d' ')
                    if [ "$stupid" = "$ans" ]; then
                        echo "$a -> $b -> $c -> $d -> $e";
                        exit 0;
                    fi
                done
            done
        done
    done
done
```

6 Debian Mirror (15 pts)

```
#!/bin/sh

# Tell bash to loop by line, instead of by space
IFS=$'\n'

for l in `tail -n +2 country.csv`; do
    ip=$(echo "$l" | cut -f1 -d',' | tr '[A-Z]' '[a-z]');
    country=$(echo "$l" | cut -f2 -d',');
    res=$(dig +short ftp.$ip.debian.org 2> /dev/null);
    if [ ! -z "$res" ]; then
        echo "$country ftp.$ip.debian.org:";
        resolved=n
        while IFS= read -r line; do
            # CNAME Record
            if echo "$line" | grep '[a-z]' > /dev/null; then
                echo -n "$line => ";
            # A Record
            else
                provider=$(curl ipinfo.io/$line 2> /dev/null | \
                    grep '"org":' | cut -f4 -d'"');
                if [ ! -z "$provider" ]; then
                    echo "$line, provided by $provider"
                    resolved=y
                fi
            fi
        done <<< "$(echo -e "$res")"

        if [ "$resolved" = "n" ]; then
            echo "unresolvable";
        fi

        echo;
    fi
done
```

7 Simple Password Generator (15 pts)

```
#!/usr/bin/env bash

num_of_pass=6
min_pass_len=8
max_pass_len=10

password_dict="abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ1234567890"
dict_len=${#password_dict}

checknum ()
{
    if [ "$2" -eq "$2" ] 2>/dev/null; then
        return
    else
        >&2 printf "Expected an integer after %s\n" "$1"
        exit 2
    fi
}

while [ "$#" -gt 0 ]; do
    case $1 in
        -n)
        checknum "$1" "$2"
        num_of_pass="$2"
        shift
        ;;
        -m)
        checknum "$1" "$2"
        min_pass_len="$2"
        shift
        ;;
        -x)
        checknum "$1" "$2"
        max_pass_len="$2"
        shift
        ;;
        *)
        >&2 printf "Unknown argument: %s\n" "$1"
        exit 1
        ;;
    esac
    shift
done

if [ "$min_pass_len" -gt "$max_pass_len" ]; then
    max_pass_len="$min_pass_len"
fi
```

```
count=0

while [ "$count" -lt "$num_of_pass" ]; do
    range=$(( max_pass_len - min_pass_len + 1 ))
    len=$(( ( RANDOM % range ) + min_pass_len ))

    password=""
    while [ "$len" -gt 0 ]; do
        index=$(( ( RANDOM % dict_len ) ))
        char=${password_dict:$index:1}
        password="$password$char"
        len=$(( len - 1 ))
    done
    printf "%s\n" "$password"

    count=$(( count + 1 ))
done
```

8 Where are those attackers from? (15 pts)

```
#!/bin/sh

for ip in $(cat path-to-log-file \
        | grep NOTICE \
        | grep Ban \
        | grep -oE '([0-9]+\.){3}([0-9]+)' \
        | sort -u ); do
    geoiplookup "$ip" 2>/dev/null \
    | cut -f2 -d':' \
    | grep -v "not found"
done | sort | uniq -c | sort -nr
```

9 Yet Another Arch (20 pts)

Skills required for solving this problem

- Really know how to install OS.
- Really know how to manipulate LVM.
- Really understand fstab.
- Know commands that show information that help debug.

1. Create a VM with virtualbox and add the second hard drive before booting.
2. In the setting page
   - Click "installation destination".
   - Choose two disks.
   - Select "I will configure partition".
   - Click "done".
   - Click "Click here to create them automatically".
   - Set the size of boot partition and swap partition to smaller size.
   - Click + to add partition of /home. Set desired capacity to 87 MiB.
   - Click "Done" twice and click "Accept Changes".
3. Click begin install.
4. Then CentOS should be installed.
5. Shutdown the machine and boot from Arch installation CD.
6. rm partion /dev/sda2 and mkpart it back with parted to enlarge /dev/sda2.
7. Use pvresize /dev/sda2 to enlarge PV.
8. lvcreate cl -L 5GiB -n arch.
9. mkfs.ext4 /dev/cl/arch
10. Create directories and mount /dev/cl/arch /mnt, mount /dev/cl/boot /dev/cl/boot, mount /dev/cl/home /mnt/home
11. pacstrap /mnt base
12. arch-chroot /mnt
13. pacman -S grub
14. grub-mkconfig > grub
15. Copy menu entry in grub file paste it to /boor/grub2/grub.cfg.

Note that it is a nasty hack!!

10 Enlarge (15 pts)

Though there seem to be some fancy ways to shrink a filesystem online, the easiest to do so is to boot with installation CD. Thus, one have to know they can utilize installation CD to complete this problem.

```
lvchange -a y /dev/cl/root
lvchange -a y /dev/cl/home
e2fsck -f /dev/cl/root
resize2fs /dev/cl/root 1.5G
lvresize /dev/cl/root -L 1.5G
lvresize /dev/cl/home -L 2G
resize2fs /dev/cl/home
```

11 Virtualize (15 pts)

Install packages

```
yum install virt-install qemu-kvm libvirt
```

Uncomment the following line in `/etc/libvirt/libvirtd.conf`. The first line is to set the group of socket file of libvirt. The second line is to set the permission of the socket file . The third line is to allow user that has access to the socket file of libvirtd to use libvirt.

```
unix_sock_group = "libvirt"
unix_sock_rw_perms = "0770"
auth_unix_rw = "none"
```

Then restart `libvrtd` to apply the configuration.

```
systemctl restart libvirtd
```

To enable user `user` to create VM, do

```
usermod -aG libvirt user
```

Then for user `user`, to create VM as usual, simply add `--connect qemu:///system` after original commands.
To generate password protected ssh key, simply set non-empty password when doing `ssh-keygen`.
To let the system "caches" the password, since GNOME is so powerful, it will cache it automatically once correct password is entered.