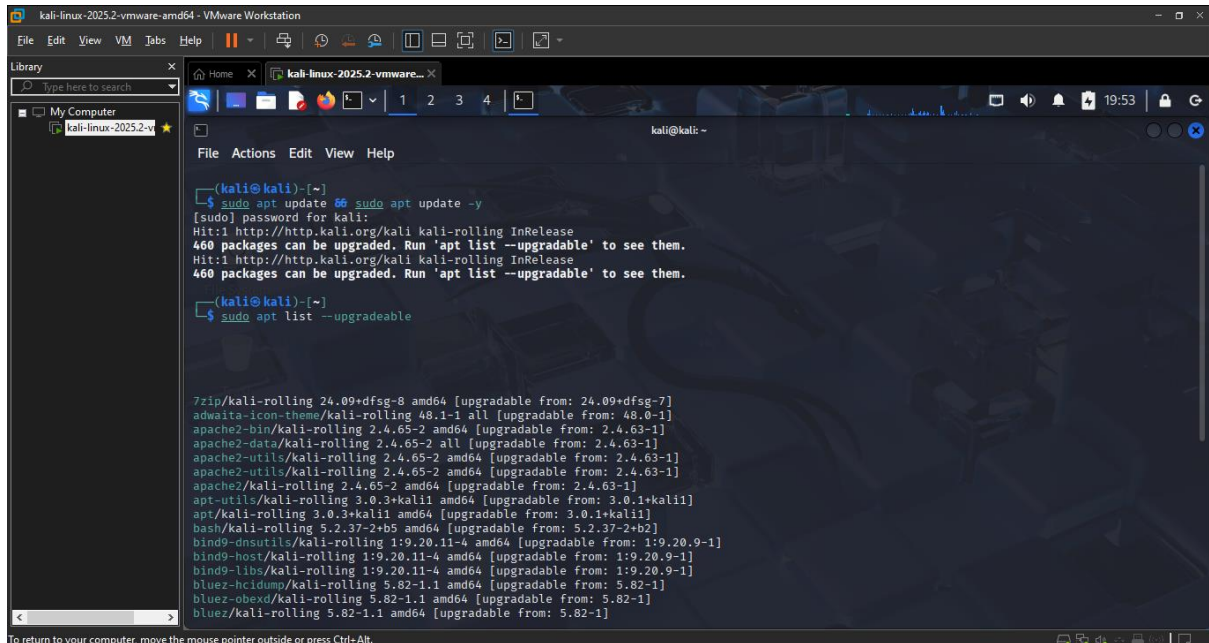


IJEI CHUKWUMA FUMNANYA DIVINE'S CYBERSECURITY ASSIGNMENT 4

OBSERVATION OF BASIC LINUX SECURITY PRACTICES ON A TEST SYSTEM

- **sudo apt update && sudo apt upgrade -y**



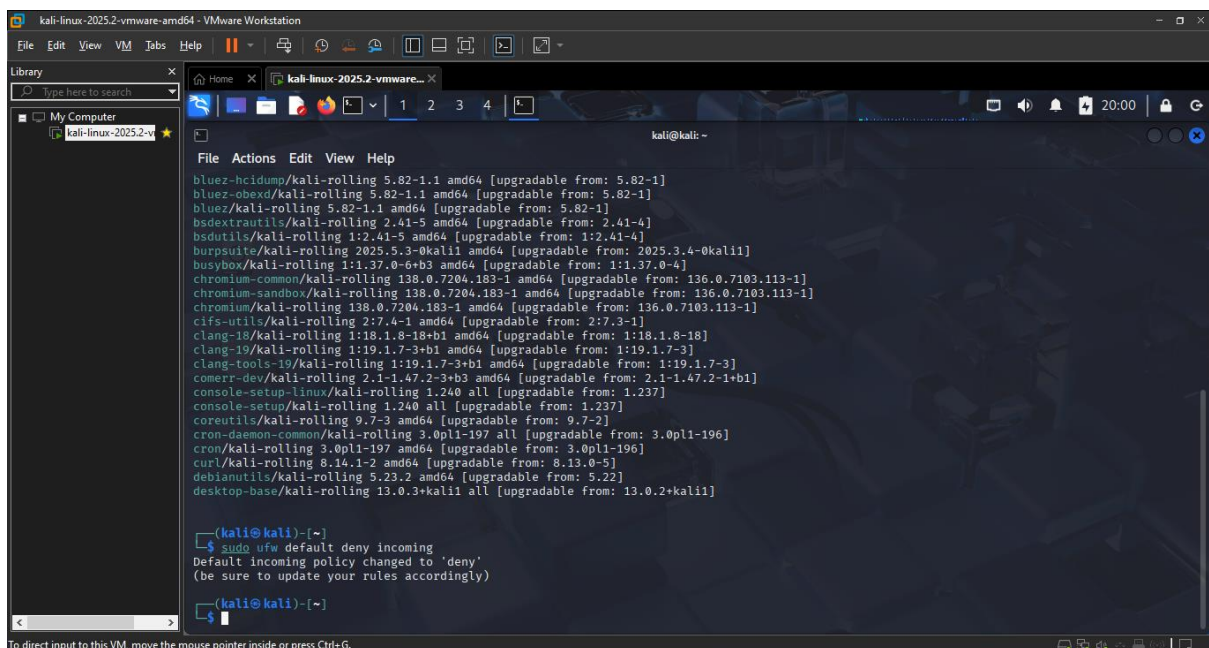
```
kali@kali:~$ sudo apt update && sudo apt upgrade -y
[sudo] password for kali:
Hit:1 http://http.kali.org/kali kali-rolling InRelease
460 packages can be upgraded. Run 'apt list --upgradable' to see them.
Hit:1 http://http.kali.org/kali kali-rolling InRelease
460 packages can be upgraded. Run 'apt list --upgradable' to see them.

(kali@kali)~$ sudo apt list --upgradable

7zip/kali-rolling 24.09+dfsg-8 amd64 [upgradable from: 24.09+dfsg-7]
adwaita-icon-theme/kali-rolling 48.1-1 all [upgradable from: 48.0-1]
apache2-bin/kali-rolling 2.4.65-2 amd64 [upgradable from: 2.4.63-1]
apache2-data/kali-rolling 2.4.65-2 all [upgradable from: 2.4.63-1]
apache2-utils/kali-rolling 2.4.65-2 amd64 [upgradable from: 2.4.63-1]
apache2-utils/kali-rolling 2.4.65-2 amd64 [upgradable from: 2.4.63-1]
apache2/kali-rolling 2.4.65-2 amd64 [upgradable from: 2.4.63-1]
apt-utils/kali-rolling 3.0.3+kali1 amd64 [upgradable from: 3.0.1+kali1]
apt/kali-rolling 3.0.3+kali1 amd64 [upgradable from: 3.0.1+kali1]
bash/kali-rolling 5.2.37-2+b5 amd64 [upgradable from: 5.2.37-2+b2]
bind9-dnsutils/kali-rolling 1:9.20.11-4 amd64 [upgradable from: 1:9.20.9-1]
bind9-host/kali-rolling 1:9.20.11-4 amd64 [upgradable from: 1:9.20.9-1]
bind9-libs/kali-rolling 1:9.20.11-4 amd64 [upgradable from: 1:9.20.9-1]
bluez-hcidump/kali-rolling 5.82-1.1 amd64 [upgradable from: 5.82-1]
bluez-obexd/kali-rolling 5.82-1.1 amd64 [upgradable from: 5.82-1]
bluez/kali-rolling 5.82-1.1 amd64 [upgradable from: 5.82-1]
```

This ensures that all Kali packages are updated and have the latest security patches.

- **sudo ufw default deny incoming**



```
kali@kali:~$ sudo ufw default deny incoming
Default incoming policy changed to 'deny'
(be sure to update your rules accordingly)

(kali@kali)~$
```

This blocks all incoming net connections by default

- **sudo ufw default allow outgoing**

```
(kali@kali)-[~]
$ sudo ufw default allow outgoing
Default outgoing policy changed to 'allow'
(be sure to update your rules accordingly)
(kali@kali)-[~]
$
```

mouse pointer inside or press Ctrl+G.

This allows all outgoing net connections

- **sudo ufw allow ssh**

```
kali-linux-2025.2-vmware-amd64 - VMware Workstation
File Edit View VM Tabs Help
Library
Type here to search
My Computer
kali-linux-2025.2-vm
kali@kali: ~
File Actions Edit View Help
(kali@kali)-[~]
$ sudo ufw allow ssh
Rules updated
Rules updated (v6)
(kali@kali)-[~]
$
```

To direct input to this VM, move the mouse pointer inside or press Ctrl+G.

This would keep remote management working if SSH (Secure Shell) is required

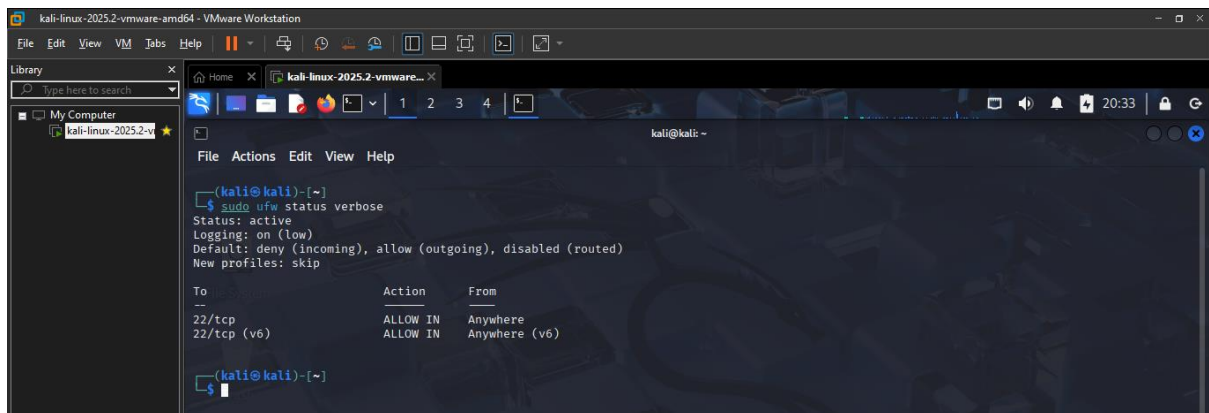
- **sudo ufw enable**

```
kali-linux-2025.2-vmware-amd64 - VMware Workstation
File Edit View VM Tabs Help
Library
Type here to search
My Computer
kali-linux-2025.2-vm
kali@kali: ~
File Actions Edit View Help
(kali@kali)-[~]
$ sudo ufw status
Status: inactive
(kali@kali)-[~]
$ sudo su
(root@kali)-[/home/kali]
# ufw enable
Firewall is active and enabled on system startup
(root@kali)-[/home/kali]
# ufw status
Status: active
To Action From
--
22/tcp ALLOW Anywhere
22/tcp (v6) ALLOW Anywhere (v6)
(root@kali)-[/home/kali]
# ufw disable
Firewall stopped and disabled on system startup
(root@kali)-[/home/kali]
# ufw enable
Firewall is active and enabled on system startup
(root@kali)-[/home/kali]
# exit
```

To direct input to this VM, move the mouse pointer inside or press Ctrl+G.

I utilized “root” here but it’s still the same thing as using “sudo ufw...” and this stage is to confirm that fire wall is active

- **sudo ufw status verbose**

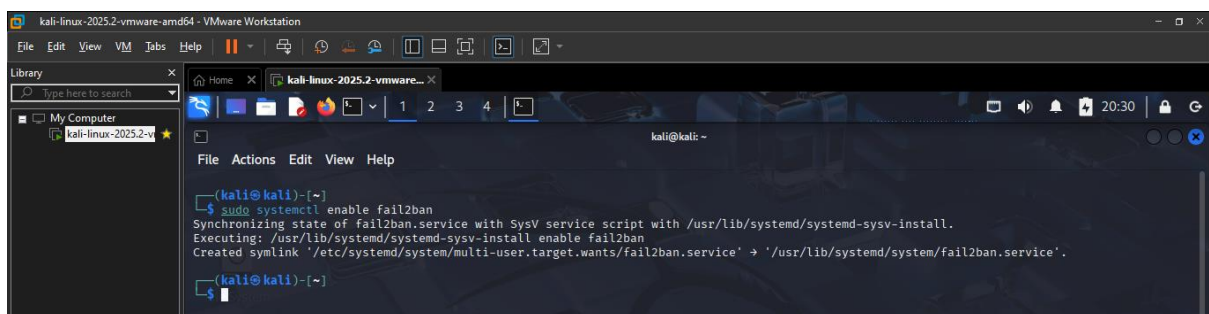


```
kali@kali:~$ sudo ufw status verbose
Status: active
Logging: on (low)
Default: deny (incoming), allow (outgoing), disabled (routed)
New profiles: skip

To Action From
--
22/tcp ALLOW IN Anywhere
22/tcp (v6) ALLOW IN Anywhere (v6)
```

This shows how uncomplicated firewall rules are applied

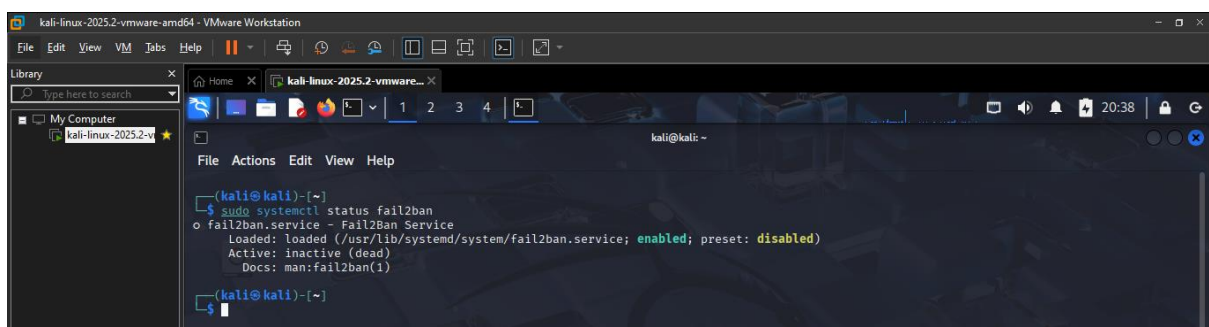
- **sudo systemctl enable fail2ban**



```
kali@kali:~$ sudo systemctl enable fail2ban
Synchronizing state of fail2ban.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable fail2ban
Created symlink '/etc/systemd/system/multi-user.target.wants/fail2ban.service' → '/usr/lib/systemd/system/fail2ban.service'.
```

This protects the system from brute-force attacks by monitoring failed attempts and temporarily banning offending IP (Internet protocol) addresses

- **sudo systemctl status fail2ban**



```
kali@kali:~$ sudo systemctl status fail2ban
○ fail2ban.service - Fail2Ban Service
   Loaded: loaded (/usr/lib/systemd/system/fail2ban.service; enabled; preset: disabled)
   Active: inactive (dead)
     Docs: man:fail2ban(1)
```

This shows the status of fail2ban in your system

- `sudo nano /etc/ssh/sshd_config`

The screenshot shows a Kali Linux virtual machine running in VMware Workstation. The terminal window is open to the `/etc/ssh/sshd_config` file using the `nano` editor. The file contains various configuration options for the SSH daemon, including port settings, host keys, ciphers, logging, and authentication methods. The `PermitRootLogin` option is currently set to `prohibit-password`.

```

#Port 22
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::

#HostKey /etc/ssh/ssh_host_rsa_key
#HostKey /etc/ssh/ssh_host_ecdsa_key
#HostKey /etc/ssh/ssh_host_ed25519_key

# Ciphers and keying
#RekeyLimit default none

# Logging
#SyslogFacility AUTH
#LogLevel INFO

# Authentication:

#LoginGraceTime 2m
#PermitRootLogin prohibit-password
#StrictModes no
#MaxAuthTries 6
#MaxSessions 10

#PubkeyAuthentication yes

Expect .ssh/authorized_keys2 to be disregarded by default in future.

```

This is where I disable root login for SSH (`Permitrootlogin: Yes => No`) which adds an extra layer of security by forcing attackers to guess both a username and password

- `sudo systemctl list-unit-files --type=service | grep enabled`

The screenshot shows the same Kali Linux VM with the terminal window displaying the output of the command `sudo systemctl list-unit-files --type=service | grep enabled`. The output lists various system services and their status (enabled, disabled, masked, or runtime).

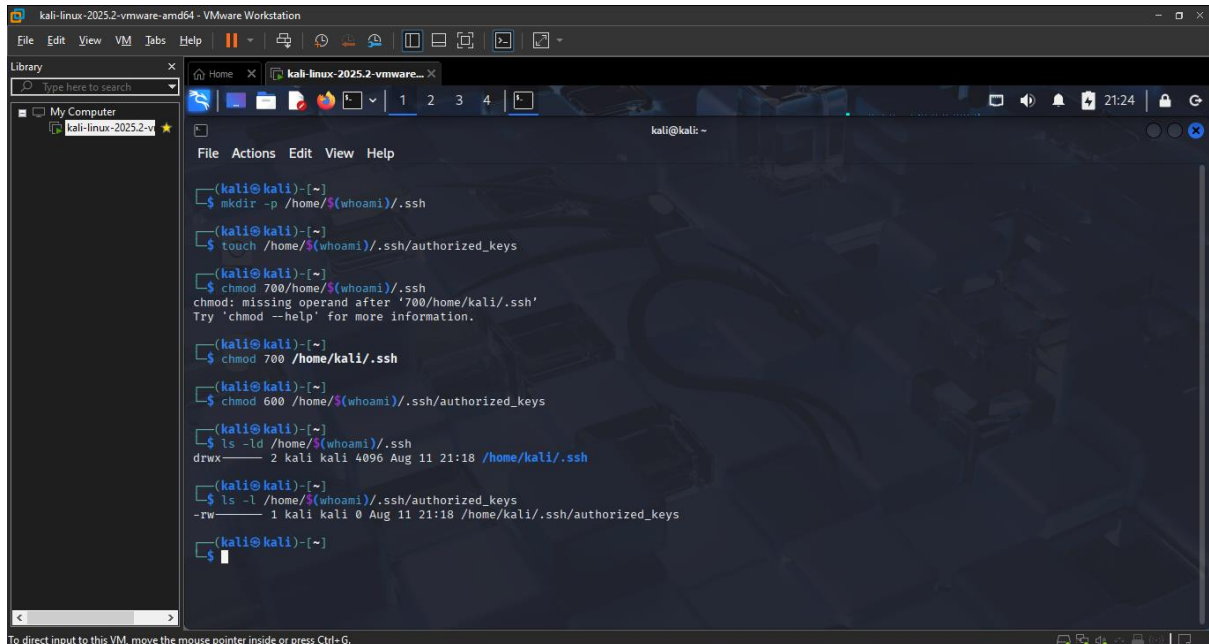
```

(kali@kali)-[~]
└─$ sudo systemctl list-unit-files --type=service | grep enabled
accounts-daemon.service          enabled          enabled
console-setup.service           enabled          enabled
cron.service                    enabled          enabled
fail2ban.service                enabled          disabled
getty@.service                  enabled          enabled
grub-install-devices.service    enabled          disabled
haveged.service                 enabled          enabled
keyboard-setup.service          enabled          enabled
lightdm.service                 enabled          disabled
ModemManager.service            enabled          enabled
networking.service              enabled          enabled
NetworkManager-dispatcher.service enabled          disabled
NetworkManager-wait-online.service enabled          disabled
NetworkManager.service          enabled          enabled
nfs-common.service              masked           enabled
open-vm-tools.service           enabled          enabled
regenerate-ssh-host-keys.service enabled          enabled
rsync.service                   disabled         enabled
rtkit-daemon.service            disabled         enabled
smartmontools.service           enabled          enabled
sudo.service                    masked           enabled
systemd-confext.service         disabled         enabled
systemd-fsck-root.service       enabled-runtime disabled
systemd-network-generator.service disabled         enabled
systemd-networkd-wait-online.service disabled         enabled
systemd-networkd.service        disabled         enabled
systemd-pstore.service          enabled          enabled
systemd-remount-fs.service       enabled-runtime disabled
systemd-sysext.service          disabled         enabled

```

From a detour research, I found out this is used to disable unused services which reduces the number of potential vulnerabilities attackers can exploit

- `mkdir -p /home/${whoami}/.ssh`
- `touch /home/${whoami}/.ssh/authorized_keys`
- `chmod 700 /home/${whoami}/.ssh`
- `chmod 600 /home/${whoami}/.ssh/authorized_keys`
- `ls -ld /home/${whoami}/.ssh`
- `ls -l /home/${whoami}/.ssh/authorized_keys`



```

kali@kali:~$ mkdir -p /home/${whoami}/.ssh
kali@kali:~$ touch /home/${whoami}/.ssh/authorized_keys
kali@kali:~$ chmod 700 /home/${whoami}/.ssh
chmod: missing operand after '700/home/kali/.ssh'
Try 'chmod --help' for more information.
kali@kali:~$ chmod 700 /home/kali/.ssh
kali@kali:~$ chmod 600 /home/${whoami}/.ssh/authorized_keys
kali@kali:~$ ls -ld /home/${whoami}/.ssh
drwx----- 2 kali kali 4096 Aug 11 21:18 /home/kali/.ssh
kali@kali:~$ ls -l /home/${whoami}/.ssh/authorized_keys
-rw----- 1 kali kali 0 Aug 11 21:18 /home/kali/.ssh/authorized_keys
kali@kali:~$

```

Initially I had no “.ssh” folder or even an “.ssh/authorized-keys” file so I had to set up the folder and the file giving the file a directory in the system and thereafter I applied the permissions “chmod 700” for “Read + write + execute” and “chmod 600” for “Read + write only”.