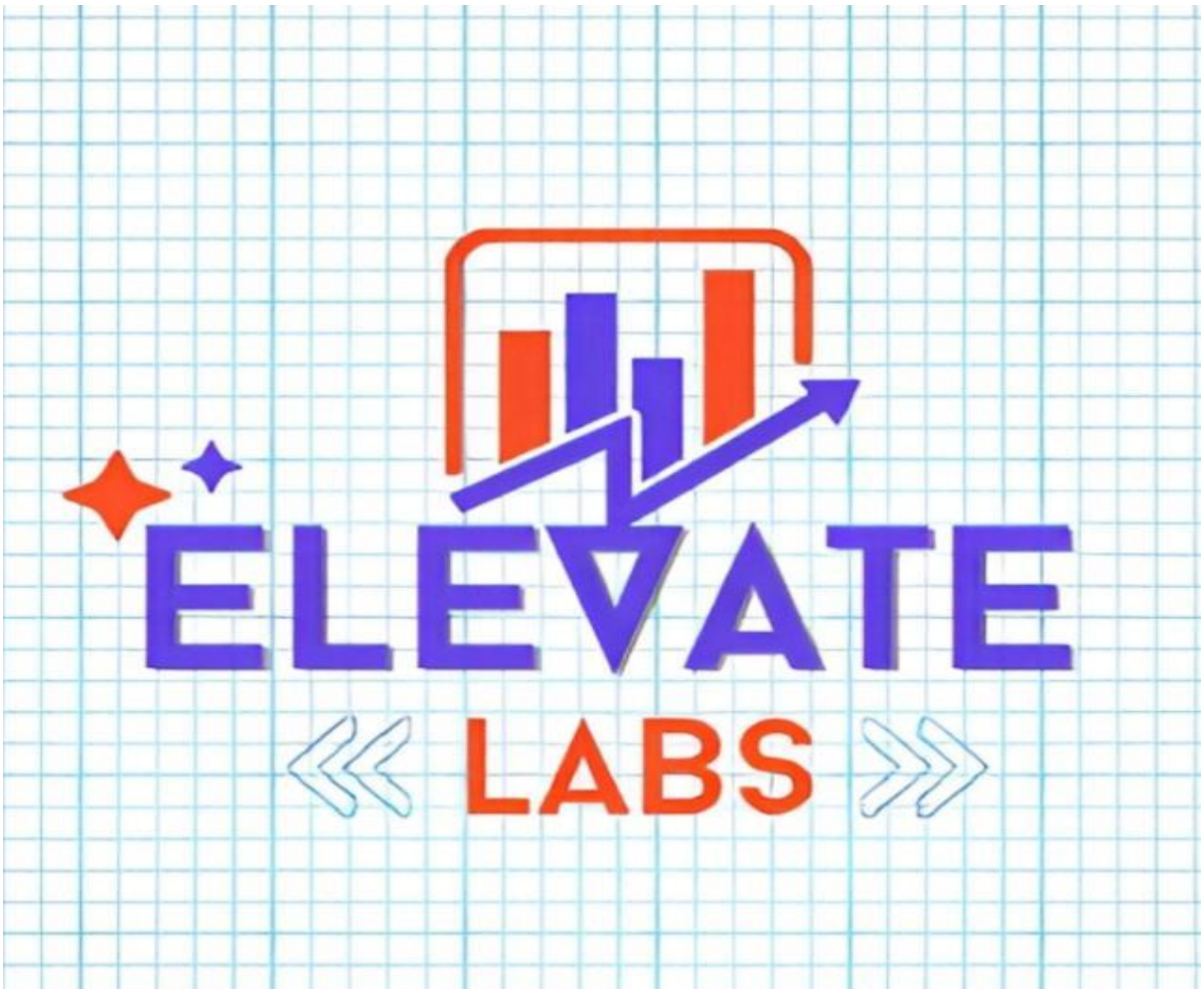
TASK 14 Linux Server Hardening & Secure Configuration



Name: - Ch Rohit Kumar

University: - Aditya University

Gmail: - rohitkumarrrr20@gmail.com

1. Review default Linux system settings to understand users, services, and open ports.

Here is the theory explanation suitable for exams (written format, no commands focus, more conceptual):

Review of Default Linux System Settings (Theory)
Reviewing default Linux system settings is an important task in system administration and cybersecurity. It helps administrators understand how the operating system is configured after installation and ensures that the system is secure and properly managed.
The main areas to review include:

- ***Users and user accounts***
- ***System services***
- ***Open ports and network configuration***
- ***System security settings***

---

1. Users in Linux
Linux is a multi-user operating system, meaning multiple users can access and use the system simultaneously.
Types of Users:

1. ***Root User***
    - ***The superuser of the system.***
    - ***Has complete control over the system.***
    - ***Can install software, modify system files, and manage other users.***
2. ***System Users***
    - ***Created automatically during installation.***
    - ***Used by services and background processes.***
    - ***Usually do not have login access.***
3. ***Regular Users***
    - ***Created by the administrator.***
    - ***Used for daily tasks.***
    - ***Have limited permissions for security reasons.***

Each user has:

- ***A unique User ID (UID)***
- ***A home directory***
- ***A default shell***
- ***Group membership***

Reviewing users helps detect:

- ***Unauthorized accounts***
- ***Unnecessary privileged users***
- ***Misconfigured permissions***

---

2. Services in Linux

Services are background programs that start during boot and run continuously to perform specific tasks.
Examples:

- ***SSH service (remote login)***
- ***Web server (Apache/Nginx)***
- ***Database server (MySQL)***
- ***Cron (scheduled tasks)***

Modern Linux systems use systemd to manage services.
Services can be:

- ***Running***
- ***Stopped***
- ***Enabled (start at boot)***
- ***Disabled***

Reviewing services helps to:

- ***Identify unnecessary services***
- ***Reduce system resource usage***
- ***Minimize attack surface***
- ***Improve security***

If too many services are running, the system becomes more vulnerable to attacks.

---

3. Open Ports and Network Settings
Ports are communication endpoints used by services to send and receive data over a network.
For example:

- ***Port 22 → SSH***
- ***Port 80 → HTTP***
- ***Port 443 → HTTPS***

When a service runs, it may open a port to accept network connections.
Reviewing open ports helps:

- ***Identify which services are accessible from outside***
- ***Detect suspicious or unauthorized services***
- ***Prevent remote exploitation***

Unnecessary open ports increase security risks.

---

4. Default Security Settings
After installation, Linux has default security configurations, such as:

- ***File permissions***
- ***Firewall settings***
- ***Password policies***
- ***SELinux or AppArmor (security modules)***

Reviewing these settings ensures:

- ***Proper access control***
- ***Strong password enforcement***
- ***Firewall protection***

- ***Restricted root access***

---

5. Importance of Reviewing Default Settings

Reviewing default system settings is important because:

- ***It helps in system hardening.***
- ***It reduces security vulnerabilities.***
- ***It ensures compliance with security policies.***
- ***It prevents unauthorized access.***
- ***It improves system performance and stability.***

In cybersecurity, reviewing system settings is often the first step in:

- ***Security audits***
- ***Penetration testing***
- ***Incident response***
- ***Server deployment***

Check Default Users

List all users:

cat /etc/passwd

Show only normal users (UID ≥ 1000):

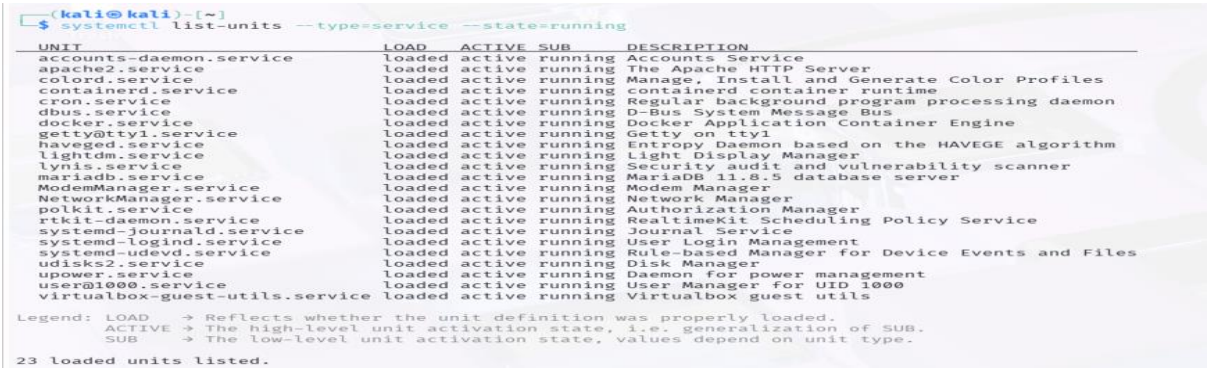
awk -F: '$3 >= 1000 {print $1}' /etc/passwd

Currently logged-in users:

Who

2. Review Groups
   cat /etc/group

3. Check Running Services Using systemctl (modern Linux):
   systemctl list-units --type=service --state=running



View all enabled services:

systemctl list-unit-files --type=service

```
┌──(kali㉿kali)-[~]
└─$ systemctl list-unit-files --type=service
UNIT FILE                                  STATE           PRESET
accounts-daemon.service                    enabled         enabled
apache-htcacheclean.service                disabled        disabled
apache-htcacheclean@.service               disabled        disabled
apache2.service                            enabled         disabled
apache2@.service                           disabled        disabled
apparmor.service                           disabled        disabled
apt-daily-upgrade.service                  static          -
apt-daily.service                          static          -
auth-rpcgss-module.service                 static          -
autovt@.service                            alias           -
avahi-daemon.service                       disabled        disabled
binfmt-support.service                     enabled         enabled
blk-availability.service                   disabled        disabled
blueman-mechanism.service                  disabled        disabled
bluetooth.service                          disabled        disabled
breakpoint-pre-basic.service               static          -
breakpoint-pre-mount.service               static          -
breakpoint-pre-switch-root.service         static          -
breakpoint-pre-udev.service                static          -
capsule@.service                           static          -
chkrootkit.service                         static          -
clamav-freshclam-once.service              disabled        disabled
clamav-freshclam.service                   disabled        disabled
colord.service                             static          -
console-getty.service                      disabled        disabled
console-setup.service                      enabled         enabled
container-getty@.service                   static          -
containerd.service                         disabled        disabled
cron.service                               enabled         enabled
cryptdisks-early.service                   masked          disabled
cryptdisks.service                         masked          disabled
dbus-org.freedesktop.hostname1.service     alias           -
dbus-org.freedesktop.locale1.service       alias           -
dbus-org.freedesktop.login1.service        alias           -
dbus-org.freedesktop.ModemManager1.service alias           -
dbus-org.freedesktop.nm-dispatcher.service alias           -
dbus-org.freedesktop.timedate1.service     alias           -
dbus-org.freedesktop.timesync1.service     alias           -
```

4. Check Open Ports Using ss (recommended):
   ss -tuln

```
┌──(kali㉿kali)-[~]
└─$ ss -tuln

Netid  State   Recv-Q  Send-Q   Local Address:Port    Peer Address:Port
tcp    LISTEN  0       4096     127.0.0.1:40231       0.0.0.0:*
tcp    LISTEN  0       80       127.0.0.1:3306        0.0.0.0:*
tcp    LISTEN  0       511            *:80                  *:*
tcp    LISTEN  0       511            *:9876                *:*
```

5. See Which Service Uses Which Port
   sudo ss -tulnp

```
┌──(kali㉿kali)-[~]
└─$ sudo ss -tulnp

[sudo] password for kali:
Netid  State   Recv-Q  Send-Q   Local Address:Port    Peer Address:Port  Process
tcp    LISTEN  0       4096     127.0.0.1:40231       0.0.0.0:*      users:(("containerd",pid=604,fd=9))
tcp    LISTEN  0       80       127.0.0.1:3306        0.0.0.0:*      users:(("mariadbd",pid=689,fd=28))
tcp    LISTEN  0       511            *:80                  *:*      users:(("apache2",pid=719,fd=4),("apache2",pid=718,fd=4),("apache2",pid=717,fd=4),("apache2",pid=716,fd=4),("apache2",pid=715,fd=4),("apache2",pid=590,fd=4))
tcp    LISTEN  0       511            *:9876                *:*      users:(("apache2",pid=719,fd=6),("apache2",pid=718,fd=6),("apache2",pid=717,fd=6),("apache2",pid=716,fd=6),("apache2",pid=715,fd=6),("apache2",pid=590,fd=6))
```

Identify Startup Services

systemctl list-unit-files | grep enabled

```
┌──(kali㉿kali)-[~]
└─$ systemctl list-unit-files | grep enabled
accounts-daemon.service                    enabled         enabled
apache2.service                            enabled         disabled
binfmt-support.service                     enabled         enabled
console-setup.service                      enabled         enabled
cron.service                               enabled         enabled
docker.service                             enabled         enabled
getty@.service                             enabled         enabled
haveged.service                            enabled         enabled
keyboard-setup.service                     enabled         enabled
lightdm.service                            enabled         disabled
mariadb.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
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
systemd-timesyncd.service                  enabled         enabled
systemd-tpm2-clear.service                 disabled        enabled
virtualbox-guest-utils.service             enabled         enabled
dm-event.socket                            enabled         enabled
```

2. Remove unused user accounts and restrict sudo access based on least privilege

Removing unused user accounts and restricting sudo access are important security practices in Linux system administration.

Unused accounts can become security risks because attackers may exploit old or inactive accounts to gain unauthorized access. Regularly reviewing and deleting unnecessary user accounts helps reduce the attack surface and improves system security.

Sudo allows users to perform administrative tasks with root privileges. However, giving sudo access to all users is dangerous. According to the **Principle of Least Privilege**, users should only have the minimum permissions required to perform their tasks.

Restricting sudo access:

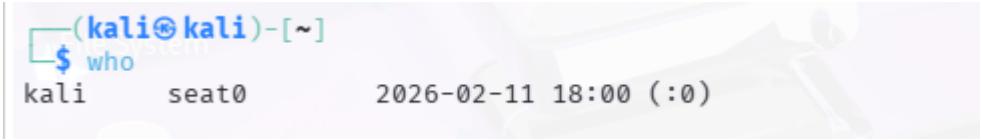
- Prevents misuse of administrative privileges
- Reduces risk of system damage
- Protects sensitive data
- Minimizes privilege escalation attacks

1. Identify Existing Users
   cat /etc/passwd

Show normal users only:

awk -F: '$3 >= 1000 {print $1}' /etc/passwd

2. Check Currently Logged-in Users
   Who



3. Remove Unused User Account Delete user (keep home directory):
   sudo userdel username(give name)
4. Lock an Account (instead of deleting)
   sudo passwd -l username(create the user again sample)
   Unlock:
   sudo passwd -u username

5. View Users with Sudo Access
   getent group sudo

6. Remove User from Sudo Group
   sudo deluser username sudo

7. Grant Sudo Access (Only When Required)
   sudo usermod -aG sudo username

8. Edit Sudo Permissions (Advanced – Least Privilege)
   Open sudoers file safely:
   sudo visudo

9. Verify Sudo Rights
   Login as user: su
   username Test: sudo -l

4. Disable root login and configure SSH using key-based authentication

Disabling root login and configuring SSH with key-based authentication are important security measures in Linux systems.

By default, the **root user** has full administrative privileges. Allowing direct root login increases the risk of brute-force attacks and unauthorized access. Disabling root login ensures that attackers cannot directly access the system as the superuser. Instead, administrators must log in as a regular user and use sudo for administrative tasks, which improves accountability and security.

SSH (Secure Shell) is used for remote access to Linux systems. Password-based authentication can be vulnerable to brute-force and dictionary attacks. Key-based authentication is more secure because it uses a pair of cryptographic keys:

- **Public key** (stored on the server)
- **Private key** (kept securely by the user)

Only users with the correct private key can access the system, making it much harder for attackers to break in.

Benefits:

- Prevents direct root attacks
- Protects against password-based hacking

- Enhances remote access security
- Improves overall system hardening

STEP 1: Check SSH Status
sudo systemctl status ssh

STEP 2: Create SSH Key (Client Side)
Run on your local machine:
ssh-keygen

STEP 3: Copy Public Key to Server

STEP 4: Edit SSH Configuration Open config file:

sudo nano /etc/ssh/sshd_config

Find and modify these lines:

PermitRootLogin no

PasswordAuthentication no

PubkeyAuthentication yes

STEP 5: Restart SSH

sudo systemctl restart ssh

STEP 6: Test Login From terminal:

ssh testuser@localhost

TEP 7: Confirm Root Login Disabled

Try: ssh root@localhost

it should fail

5. Update system packages and enable automatic security updates.
   Updating system packages and enabling automatic security updates are essential practices for maintaining a secure and stable Linux system.
   Software packages may contain bugs or security vulnerabilities. Developers regularly release updates and patches to fix these issues. If the system is not updated, attackers can exploit known vulnerabilities to gain unauthorized access.
   Regularly updating the system ensures:

- Security vulnerabilities are patched
- System performance and stability improve
- New features and improvements are applied
- Compatibility issues are reduced

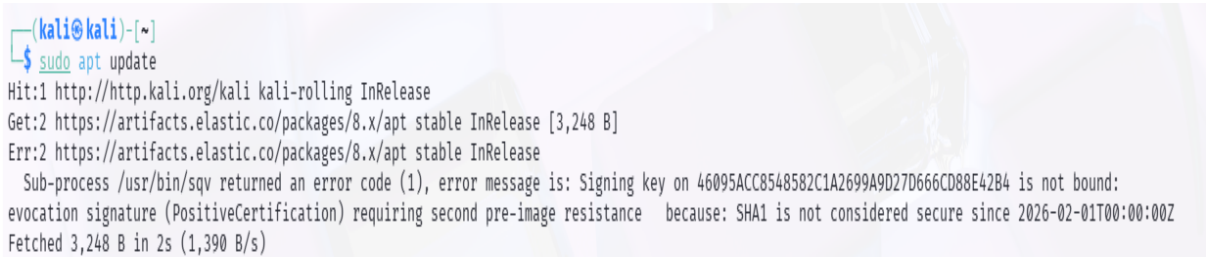
Enabling **automatic security updates** ensures that critical patches are installed automatically without manual intervention. This reduces the risk of forgetting updates and keeps the system protected against newly discovered threats.

Benefits:

- Protects against known exploits
- Reduces risk of cyberattacks
- Maintains system reliability
- Saves administrative time

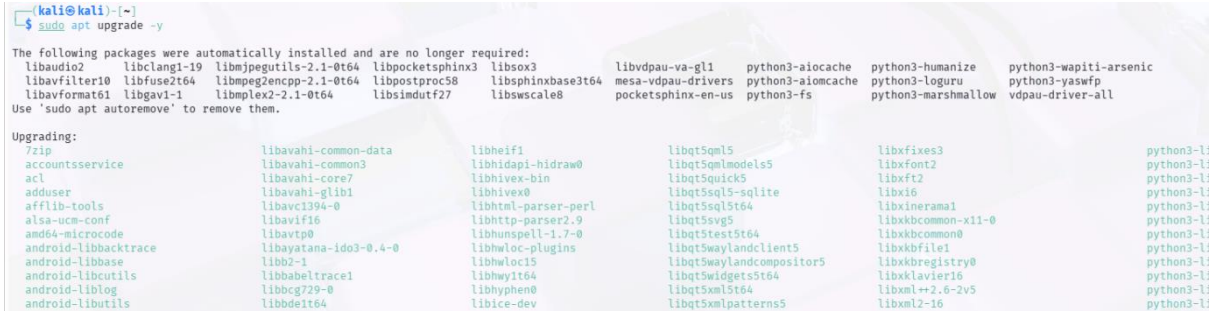
STEP 1: Update Package List

sudo apt update



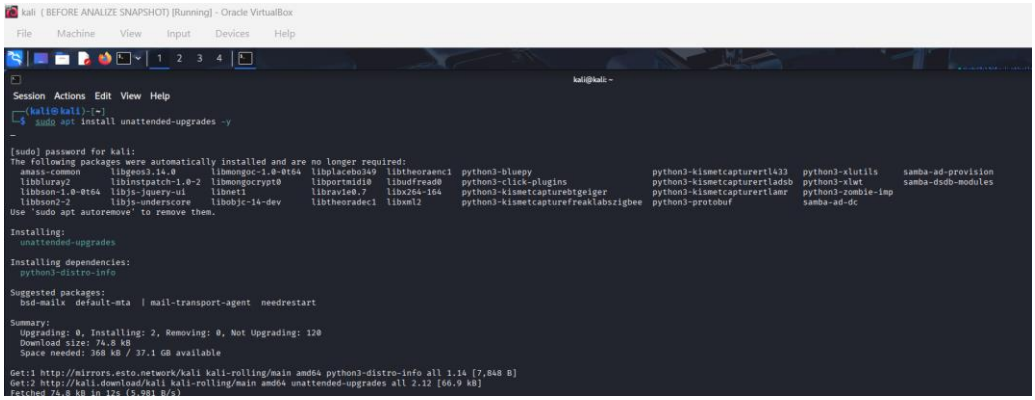
STEP 2: Upgrade Installed Packages

sudo apt upgrade -y



STEP 4: Install unattended-upgrades

sudo apt install unattended-upgrades -y

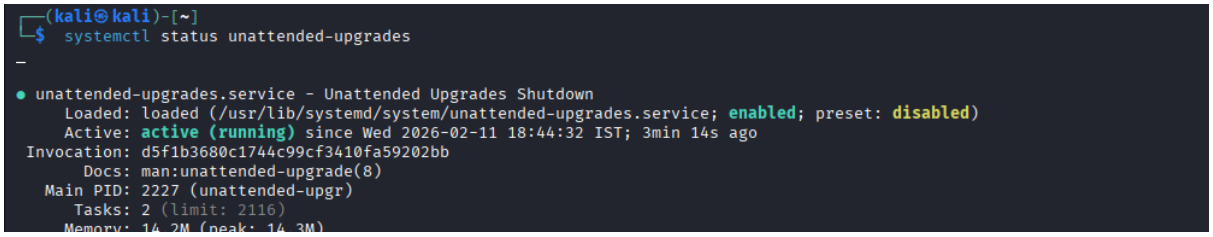


STEP 5: Enable Automatic Updates

sudo dpkg-reconfigure --priority=low unattended-upgrades

STEP 6: Verify Status

systemctl status unattended-upgrades



6. Configure a firewall to allow only required network traffic
   Configure a firewall to allow only required network traffic by following the **principle of least privilege** — deny everything by default, then explicitly allow only what is necessary.

   Define Your Required Traffic

   Before configuring anything, identify:

- Which **services** must be accessible (e.g., web, SSH, database)
- Which **ports** they use
- Which **IP addresses or networks** should have access
- Whether access is **inbound, outbound, or both**

Example:

- Web server → Allow TCP 80 (HTTP) and 443 (HTTPS)
- Admin access → Allow TCP 22 (SSH) from admin IP only
- Database → Allow TCP 3306 only from application server

Linux Firewall (UFW – Ubuntu/Debian)

1️⃣ Enable firewall

sudo ufw enable

2️⃣ Set default deny policy

sudo ufw default deny incoming

sudo ufw default allow outgoing

3️⃣ Allow required services

sudo ufw allow 80/tcp

sudo ufw allow 443/tcp

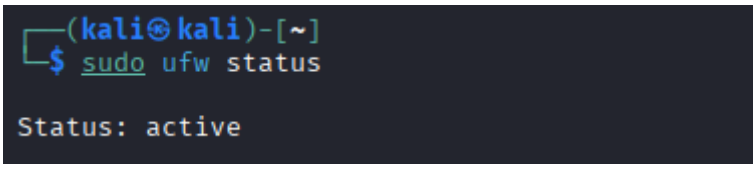
sudo ufw allow from 192.168.1.10 to any port 22

4️⃣ Check status

sudo ufw status verbose

STEP 2: Check Firewall Status

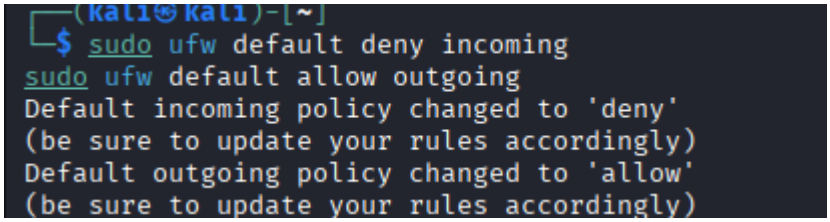
sudo ufw status



STEP 3: Set Default Policies (Deny Everything)

sudo ufw default deny incoming

sudo ufw default allow outgoing



STEP 4: Allow Required Services

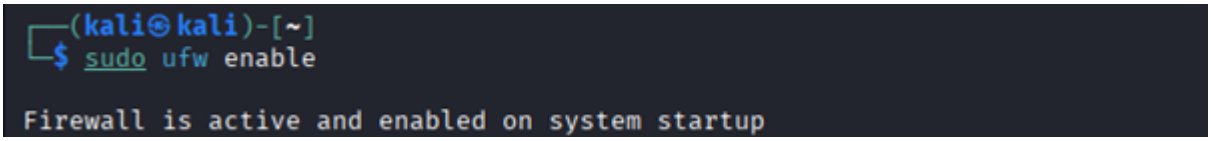
Only Allow SSH:

sudo ufw allow ssh

STEP 5: Enable Firewall

sudo ufw enable

STEP 6: Verify Rules sudo ufw status verbose



STEP 6: Verify Rules

sudo ufw status verbose

7 Stop and disable unnecessary services running on the server.

Attack Surface Reduction

Each service:

- Opens network ports
- Accepts input
- Has executable code

More services = more potential entry points.

Fewer services = fewer opportunities for exploitation.

Principle of Least Functionality

A core security best practice:

A system should provide only essential capabilities.

This minimizes:

- Exploitable software
- Misconfiguration risks
- Privilege escalation paths

Vulnerability Containment

Even trusted services can have:

- Zero-day vulnerabilities

- Misconfigurations
- Weak authentication

If the service isn't running, it cannot be exploited.

Performance & Stability

Unnecessary services:

- Consume CPU and memory
- Increase boot time
- Create log noise
- Increase maintenance overhead

Minimal systems are:

- Easier to monitor
- Easier to audit
- Easier to secure

STEP 1: List Running Services
systemctl list-units --type=service --state=running

```
┌──(kali㉿kali)-[~]
└─$ systemctl list-units --type=service --state=running

  UNIT                            LOAD   ACTIVE SUB     DESCRIPTION
  accounts-daemon.service         loaded active running Accounts Service
  colord.service                  loaded active running Manage, Install and Generate Color Profiles
  cron.service                    loaded active running Regular background program processing daemon
  dbus.service                    loaded active running D-Bus System Message Bus
  getty@tty1.service              loaded active running Getty on tty1
  haveged.service                 loaded active running Entropy Daemon based on the HAVEGE algorithm
  lightdm.service                 loaded active running Light Display Manager
  ModemManager.service            loaded active running Modem Manager
  NetworkManager.service          loaded active running Network Manager
  polkit.service                  loaded active running Authorization Manager
  rtkit-daemon.service            loaded active running RealtimeKit Scheduling Policy Service
  systemd-journald.service        loaded active running Journal Service
  systemd-logind.service          loaded active running User Login Management
  systemd-udevd.service           loaded active running Rule-based Manager for Device Events and Files
  tor@default.service             loaded active running Anonymizing overlay network for TCP
  udisks2.service                 loaded active running Disk Manager
  unattended-upgrades.service     loaded active running Unattended Upgrades Shutdown
  upower.service                  loaded active running Daemon for power management
  user@1000.service               loaded active running User Manager for UID 1000
  virtualbox-guest-utils.service loaded active running Virtualbox guest utils
```

STEP 2: List Enabled Services (Start at Boot)
systemctl list-unit-files --type=service | grep enabled

STEP 3: Identify Unnecessary Services (Examples) Common services you may disable (only if not needed):

```
┌──(kali㉿kali)-[~]
└─$ systemctl list-unit-files --type=service | grep enabled
accounts-daemon.service                    enabled         enabled
console-setup.service                      enabled         enabled
cron.service                               enabled         enabled
getty@.service                             enabled         enabled
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
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
systemd-timesyncd.service                  enabled         enabled
systemd-tpm2-clear.service                 disabled        enabled
tor.service                                enabled         disabled
tor@default.service                        enabled-runtime disabled
ufw.service                                enabled         enabled
```

- Bluetooth
- cups (printing)
- apache2
- avahi-daemon

STEP 4: Stop a Service Example: stop Bluetooth
sudo systemctl stop Bluetooth

STEP 5: Disable Service from Startup
sudo systemctl disable Bluetooth

STEP 6: Verify Service is Disabled
systemctl status Bluetooth

STEP 8: Confirm Reduced Services
systemctl list-units --type=service --state=running

```
┌──(kali㉿kali)-[~]
└─$ systemctl list-units --type=service --state=running
  UNIT                          LOAD   ACTIVE SUB     DESCRIPTION
  accounts-daemon.service       loaded active running Accounts Service
  colord.service                loaded active running Manage, Install and Generate Color Profiles
  cron.service                  loaded active running Regular background program processing daemon
  dbus.service                  loaded active running D-Bus System Message Bus
  getty@tty1.service            loaded active running Getty on tty1
  haveged.service               loaded active running Entropy Daemon based on the HAVEGE algorithm
  lightdm.service               loaded active running Light Display Manager
  ModemManager.service          loaded active running Modem Manager
  NetworkManager.service        loaded active running Network Manager
  polkit.service                loaded active running Authorization Manager
  rtkit-daemon.service          loaded active running RealtimeKit Scheduling Policy Service
  systemd-journald.service      loaded active running Journal Service
  systemd-logind.service        loaded active running User Login Management
  systemd-udevd.service         loaded active running Rule-based Manager for Device Events and Files
  tor@default.service           loaded active running Anonymizing overlay network for TCP
  udisks2.service               loaded active running Disk Manager
  unattended-upgrades.service   loaded active running Unattended Upgrades Shutdown
  upower.service                loaded active running Daemon for power management
  user@1000.service             loaded active running User Manager for UID 1000
  virtualbox-guest-utils.service loaded active running Virtualbox guest utils
```

7. Secure file permissions for sensitive system and configuration files
   The Core Principle

Every file has:

- **Owner**
- **Group**
- **Permissions (read, write, execute)**

Security goal:

Only the minimum necessary users/processes can access sensitive files.

Why This Matters

Sensitive files often contain:

- Password hashes
- API keys
- Private keys
- Database credentials
- System configurations
- Security policies

If improperly permissioned:

- Attackers can escalate privileges
- Malware can modify system behavior
- Credentials can be stolen

Apply the Principle of Least Privilege

- Users access only what they need
- Services run under dedicated low-privilege accounts
- No world-readable sensitive files

Protect Critical System Files

Examples (Linux):

- /etc/passwd
- /etc/shadow
- /etc/sudoers
- SSH private keys
- Web server configs
- Application .env files

STEP 1: Check Current Permissions ls

-l /etc/passwd /etc/shadow /etc/sudoers

```
┌──(kali㉿kali)-[~]
└─$ ls -l /etc/passwd /etc/shadow /etc/sudoers

-rw-r--r-- 1 root root   3328 Nov 18 19:08 /etc/passwd
-rw-r----- 1 root shadow 1470 Nov 18 19:08 /etc/shadow
-r--r----- 1 root root   1714 Aug 17 16:11 /etc/sudoers
```

STEP 2: Secure /etc/shadow (Most Sensitive)

sudo chown root:shadow /etc/shadow

sudo chmod 640 /etc/shadow

```
┌──(kali㉿kali)-[~]
└─$ sudo chown root:shadow /etc/shadow
sudo chmod 640 /etc/shadow
```

Verify: ls -l /etc/shadow

```
┌──(kali㉿kali)-[~]
└─$ ls -l /etc/shadow
-rw-r----- 1 root shadow 1470 Nov 18 19:08 /etc/shadow
```

STEP 3: Secure /etc/passwd

sudo chmod 644 /etc/passwd

```
┌──(kali㉿kali)-[~]
└─$ sudo chmod 644 /etc/passwd
```

STEP 4: Secure sudoers File

sudo chmod 440 /etc/sudoers

```
┌──(kali㉿kali)-[~]
└─$ sudo chmod 440 /etc/sudoers
```

Verify: ls -l /etc/sudoers

```
┌──(kali㉿kali)-[~]
└─$ ls -l /etc/sudoers

-r--r----- 1 root root 1714 Aug 17 16:11 /etc/sudoers
```

STEP 5: Secure SSH Configuration

sudo chmod 600 /etc/ssh/sshd_config

sudo chown root:root /etc/ssh/sshd_config

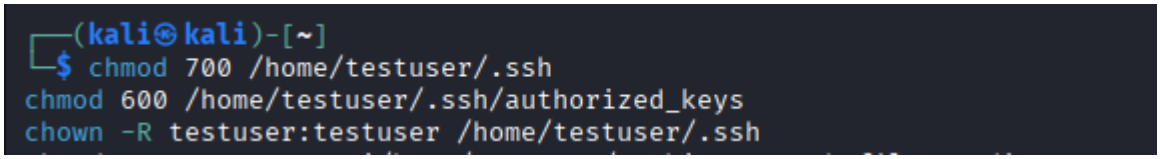
```
┌──(kali㉿kali)-[~]
└─$ sudo chmod 600 /etc/ssh/sshd_config
sudo chown root:root /etc/ssh/sshd_config
```

STEP 6: Secure User SSH Keys For testuser:

chmod 700 /home/testuser/.ssh 22

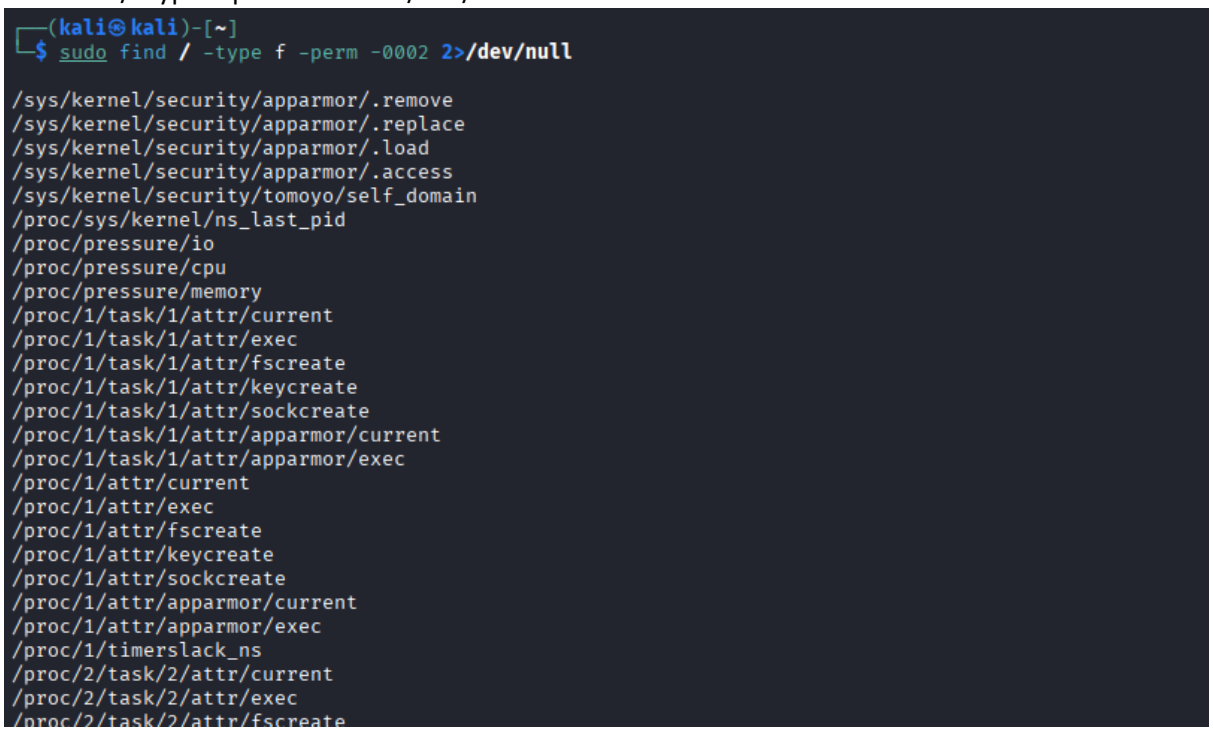
chmod 600 /home/testuser/.ssh/authorized_keys

chown -R testuser:testuser /home/testuser/.ssh



STEP 7: Find World-Writable Files (Audit)

sudo find / -type f -perm -0002 2>/dev/null



8. Review system logs to monitor authentication and system activity
   Log review supports three security goals:

1. **Detection** – Identify unauthorized access or abnormal behavior
2. **Accountability** – Trace actions to users or processes
3. **Forensics** – Investigate incidents after they occur
   Without log monitoring, breaches can remain undetected for months.
   What Logs Contain
   System logs record:

- Successful and failed logins
- Privilege escalations (e.g., sudo use)
- Service start/stop events
- Configuration changes
- File access events
- Network connections

- System errors and crashes

Logs are the system's **security diary**.

Authentication Monitoring (High Priority)

Key events to monitor:

Failed Login Attempts

- Brute-force attacks
- Credential stuffing
- Enumeration attempts

Successful Logins at Unusual Times

- Late-night access
- Access outside business hours

Logins from Unusual Locations or IPs

- Foreign IP addresses
- New geographic regions

Privilege Escalation

- Use of administrative accounts
- sudo activity
- Role or group changes

System Activity Monitoring

Look for:

Unexpected Service Changes

- Services stopping unexpectedly
- New services starting
- Disabled security tools

File Changes

- Modifications to system configuration files
- Changes to authentication files
- Altered binaries

New User Accounts

- Unexpected account creation
- Admin privilege assignment

STEP 1 View all authentication logs
sudo journalctl -u ssh

STEP 1 View all authentication logs
sudo journalctl -u ssh

View
sudo activity sudo journalctl | grep
sudo View login history
last

View system logs

sudo journalctl

Recent boot only:

sudo journalctl -b

9. Conclusion:-

Comprehensive Linux system hardening was carried out to enhance security and minimize potential attack surfaces. Default users, running services, and open ports were carefully examined to assess overall system exposure. Unnecessary user accounts were removed, and sudo privileges were limited in accordance with the principle of least privilege. SSH was secured by disabling root login and enforcing key-based authentication.

System packages were kept up to date, and automatic security updates were enabled to address known vulnerabilities promptly. A firewall was configured to permit only essential network traffic, while unnecessary services were stopped and disabled to further reduce risk. Sensitive system and configuration files were protected with appropriate file permissions, and system logs were regularly reviewed to monitor authentication events and overall system activity.

Collectively, these actions strengthened the server's security posture, enforced controlled access, improved monitoring and visibility, and established a secure baseline configuration for Linux systems.