Strengthening System Security on Linux Servers

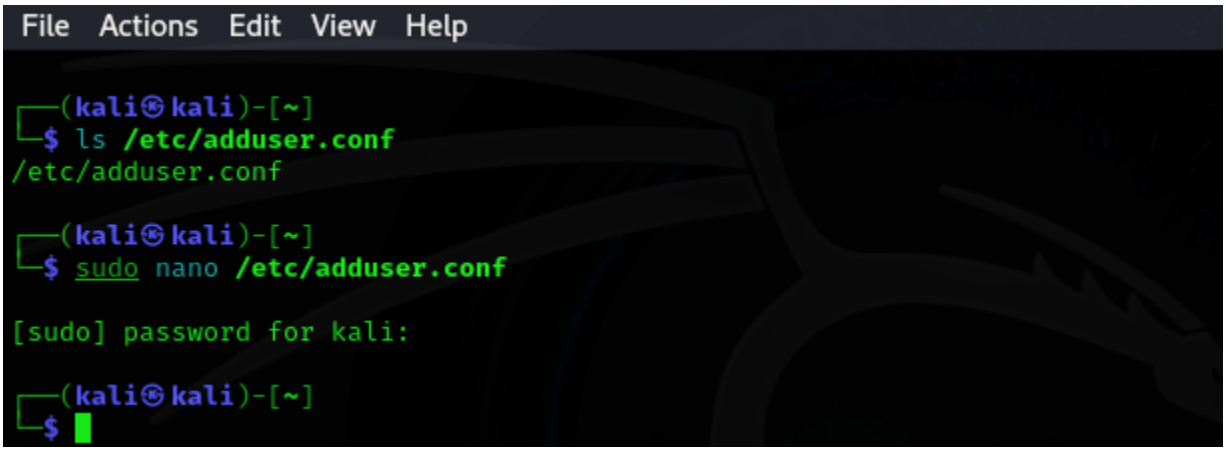
Objective:

1. To understand and apply fundamental Linux security measures for file systems, network services, remote access, and system monitoring.

2. To gain hands-on experience with tools and practices for enhancing system security.

Exercise 1: Locate and open the adduser.conf File

The location of adduser.conf is verified and using sudo nano /etc/adduser.conf to edit configuration.

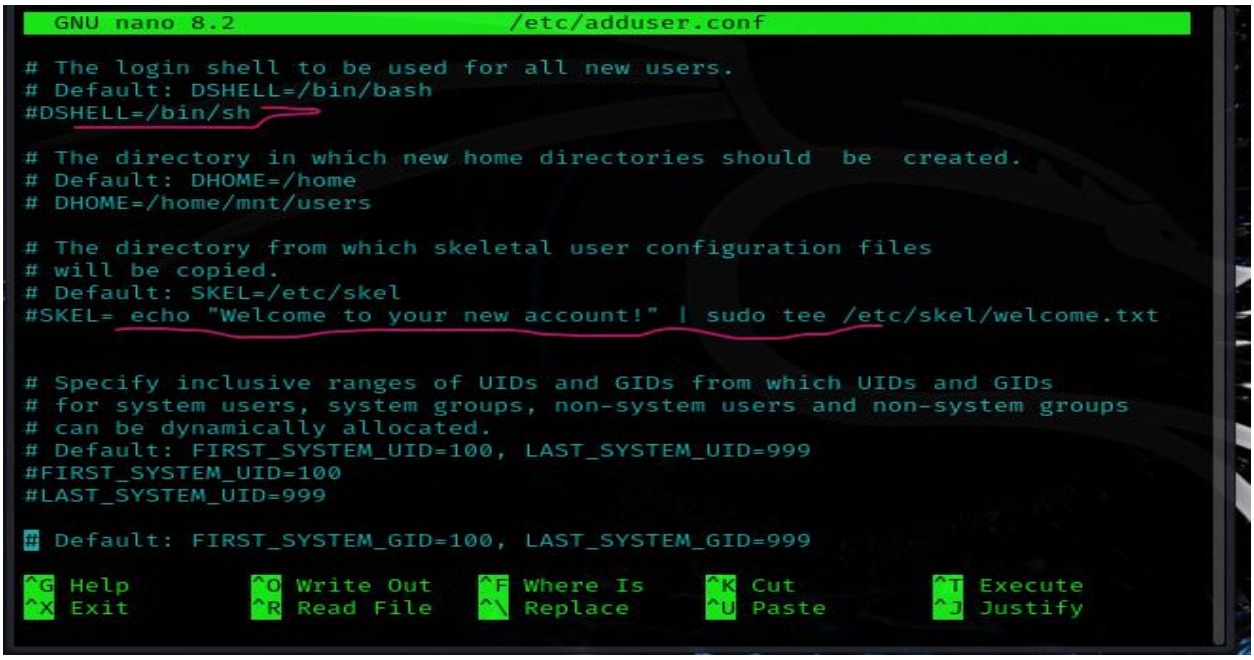


Below picture shows the editor environment

```
  GNU nano 8.2                    /etc/adduser.conf
# The login shell to be used for all new users.
# Default: DSHELL=/bin/bash
#DSHELL=/bin/bash

# The directory in which new home directories should  be  created.
# Default: DHOME=/home
# DHOME=/home

# The directory from which skeletal user configuration files
# will be copied.
# Default: SKEL=/etc/skel
#SKEL=/etc/skel

# Specify inclusive ranges of UIDs and GIDs from which UIDs and GIDs
# for system users, system groups, non-system users and non-system groups
# can be dynamically allocated.
# Default: FIRST_SYSTEM_UID=100, LAST_SYSTEM_UID=999
#FIRST_SYSTEM_UID=100
#LAST_SYSTEM_UID=999

# Default: FIRST_SYSTEM_GID=100, LAST_SYSTEM_GID=999
#FIRST_SYSTEM_GID=100
#LAST_SYSTEM_GID=999

^G Help        ^O Write Out   ^F Where Is    ^K Cut         ^T Execute
^X Exit        ^R Read File   ^\ Replace     ^U Paste       ^J Justify
```

Exercise 2: Analyze Key Configuration Parameters

Parameters like group and user were observed, Identify the parameter DHOME=/home, specifying where new users' home directories will be created, Skel Directory show the Locate SKEL=/etc/skel, which contains default files copied to a new user's home directory. The Default Shell is used to check the parameter DSHELL=/bin/bash, which defines the default login shell for new users, FIRST_UID: The first UID (User ID) to be assigned to new users, LAST_UID: The last UID number that can be assigned to regular users, FIRST_GID: The first GID (Group ID) to be assigned to a user primary group, LAST_GID: The last GID number available for regular user groups, USERGROUPS: Controls whether to create a user with a group of the same name.



Exercise 3: Modify the Configuration

The file was modified and configured to sweet the setting below as shown in the picture.

1. Default Home Directory Location was modify the DHOME value to /mnt/users in the adduser.conf file.
2. The Customization of UID and GID Ranges by Changing the FIRST_UID to 2000 and LAST_UID to 2999, and Change FIRST_GID to 3000 and LAST_GID to 3999.
3. Disable User-Specific Groups by Set USERGROUPS=no in the configuration file.
4. Add Custom Files to the Skel Directory in path /etc/skel to customize file using echo "Welcome to your new account!" | sudo tee /etc/skel/welcome.txt
5. Set a Different Default Shell by Change DSHELL to /bin/sh. Save the file.

```
  GNU nano 8.2                    /etc/adduser.conf

# The login shell to be used for all new users.
# Default: DSHELL=/bin/bash
#DSHELL=/bin/sh

# The directory in which new home directories should  be  created.
# Default: DHOME=/home
# DHOME=/home/mnt/users

# The directory from which skeletal user configuration files
# will be copied.
# Default: SKEL=/etc/skel
#SKEL= echo "Welcome to your new account!" | sudo tee /etc/skel/welcome.txt


# Specify inclusive ranges of UIDs and GIDs from which UIDs and GIDs
# for system users, system groups, non-system users and non-system groups
# can be dynamically allocated.
# Default: FIRST_SYSTEM_UID=100, LAST_SYSTEM_UID=999
#FIRST_SYSTEM_UID=100
#LAST_SYSTEM_UID=999

# Default: FIRST_SYSTEM_GID=100, LAST_SYSTEM_GID=999

^G Help        ^O Write Out   ^F Where Is    ^K Cut         ^T Execute
^X Exit        ^R Read File   ^\ Replace     ^U Paste       ^J Justify
```

Exercise 4: Test the Changes

A new user was added **sudo adduser testuser**, the changes were verified by Check the home directory of the new user using **ls /mnt/users/testuser** and confirm that the welcome.txt file exists in the home directory. the UID, GID, and default shell for the new user was noted by using command **grep testuser /etc/passwd.**

```
┌──(kali㉿kali)-[~]
└─$ sudo adduser testuser

info: Adding user `testuser' ...
info: Selecting UID/GID from range 2000 to 29999 ...
info: Adding new group `testuser' (2000) ...
info: Adding new user `testuser' (2000) with group `testuser (2000)' ...
info: Creating home directory `/home/mnt/users/testuser' ...
info: Copying files from `/etc/skel' ...
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for testuser
Enter the new value, or press ENTER for the default
        Full Name []:
        Room Number []:
        Work Phone []:
        Home Phone []:
        Other []:
Is the information correct? [Y/n] Y
info: Adding new user `testuser' to supplemental / extra groups `users' ...
info: Adding user `testuser' to group `users' ...

┌──(kali㉿kali)-[~]
└─$
```

```
┌──(kali㉿kali)-[~]
└─$ ls /home/mnt/users/testuser

ls: cannot open directory '/home/mnt/users/testuser': Permission denied

┌──(kali㉿kali)-[~]
└─$ sudo ls /home/mnt/users/testuser

welcome.txt

┌──(kali㉿kali)-[~]
└─$
```

```
File  Actions  Edit  View  Help

┌──(kali㉿kali)-[~]
└─$ grep testuser /etc/passwd
testuser:x:2000:2000:,,,:/home/mnt/users/testuser:/bin/sh

┌──(kali㉿kali)-[~]
└─$
```

Access Control Lists (ACLs)

Exercise 1: Create the Project Folder and Test Files

Both project folders and files was created using sudo mkdir /projects/team_project, and sudo touch /projects/team_project/{file1.txt,file2.txt}. the ownership folder was change to root and group to developers using sudo chown -R root:developers /projects/team_project and default permissions was set to allow group access with command sudo chmod 770 /projects/team_project.

```
┌──(kali㉿kali)-[~]
└─$ sudo addgroup developers
info: Selecting GID from range 3000 to 39999 ...
info: Adding group `developers' (GID 3000) ...

┌──(kali㉿kali)-[~]
└─$ sudo chown -R root:developers projects/team_project

┌──(kali㉿kali)-[~]
└─$ sudo chmod 770 projects/team_project


┌──(kali㉿kali)-[~]
```

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

total 4
drwxrwx--T+ 2 root developers 4096 Dec 29 10:53 team_project

┌──(kali㉿kali)-[~/projects]
└─$
```

Exercise 2: Configure ACLs for Each User

Access control lists was configure for each user, Alice was granted Full Permissions (Read, Write, Execute) with command sudo setfacl -m u:alice:rwx /projects/team_project, Bob was given Read-Only Permissions via sudo setfacl -m u:bob:rx /projects/team_project command and Charlie was granted Read and Write Permissions Without Deletion. This deletion was prevent through the set of the sticky bit and assign specific write permissions with command sudo chmod +t /projects/team_project, sudo setfacl -m u:charlie:rw /projects/team_project. To check the ACLs setting for the directory, used getfacl /projects/team_project.

```
┌──(kali㉿kali)-[~]
└─$ getfacl projects/team_project

# file: projects/team_project
# owner: root
# group: developers
# flags: --t
user::rwx
user:alice:rwx
user:bob:r-x
user:charlie:rw-
group::rwx
mask::rwx
other::---


┌──(kali㉿kali)-[~]
└─$
```

Exercise 3: Test the Permissions

Testing for each user permissions, Alice using command sudo su - alice cd/projects/team_project echo "Alice can write" > file1.txt, and this was allowed because it has all the permission, for Bob this was failed because he was granted only read and execute permission, sudo su - bob cd /projects/team_project cat file1.txt echo "Bob can write" >> file1.txt. Finally for Charlie using

sudo su - charlie cd /projects/team_project echo "Charlie can write" > file2.txt rm file1.txt, This failed due to the sticky bit.

```
┌──(kali㉿kali)-[~]
└─$ ls
Desktop    ismal2.txt     manage.txt  Pictures  Public
Documents  ismal2.txt.nc  Music       project   Templates
Downloads  ismal.txt      myfolder    projects  Videos

┌──(kali㉿kali)-[~]
└─$ sudo touch projects/team_project/{file1.txt,file2.txt}

┌──(kali㉿kali)-[~]
└─$ sudo ls projects/team_project/{file1.txt,file2.txt}
projects/team_project/file1.txt  projects/team_project/file2.txt

┌──(kali㉿kali)-[~]
└─$
```

For Alice

```
┌──(kali㉿kali)-[~]
└─$ sudo su - alice
cd projects/team_project
$ ls
file1.txt  welcome.txt
$ cat file1.txt
Alice can write
$
```

For Bob

```
┌──(kali㉿kali)-[~]
└─$ sudo su - bob
$ cd projects/team_project
-sh: 1: cd: can't cd to projects/team_project
$ cd projects
-sh: 2: cd: can't cd to projects
$
```

For charlie

```
┌──(kali㉿kali)-[~]
└─$ sudo su - charlie
$ cd projects
-sh: 1: cd: can't cd to projects
$ cd projects/team_project
-sh: 2: cd: can't cd to projects/team_project
$
```

Exercise 4: Manage Default ACLs

To manage the Access control list, set default ACLs, so new files inherit the permissions

```
┌──(kali㉿kali)-[~]
└─$ sudo getfacl projects/team_project
# file: projects/team_project
# owner: root
# group: developers
# flags: --t
user::rwx
user:alice:rwx
user:bob:r-x
user:charlie:rw-
group::rwx
mask::rwx
other::r-x
default:user::rwx
default:user:alice:rwx
default:user:bob:r-x
default:user:charlie:rw-
default:group::rwx
default:mask::rwx
default:other::r-x
```

Sudo and Privilege Management

In this scenario, john user account was created with full administrative privileges, mary account was given read and execute permission and paul

Exercise 1: Create User Accounts

User account was created using this command sudo useradd -m john, sudo useradd -m mary and sudo useradd -m paul, and password settings with sudo passwd john sudo passwd mary sudo passwd paul.

```
┌──(kali㉿kali)-[~]
└─$ sudo useradd -m john
[sudo] password for kali:

┌──(kali㉿kali)-[~]
└─$ sudo useradd -m mary

┌──(kali㉿kali)-[~]
└─$ sudo useradd -m paul

┌──(kali㉿kali)-[~]
└─$ sudo passwd john
New password:
Retype new password:
passwd: password updated successfully

┌──(kali㉿kali)-[~]
└─$ sudo passwd mary
New password:
Retype new password:
passwd: password updated successfully

┌──(kali㉿kali)-[~]
└─$
```

Exercise 2: Configure Sudo Privileges

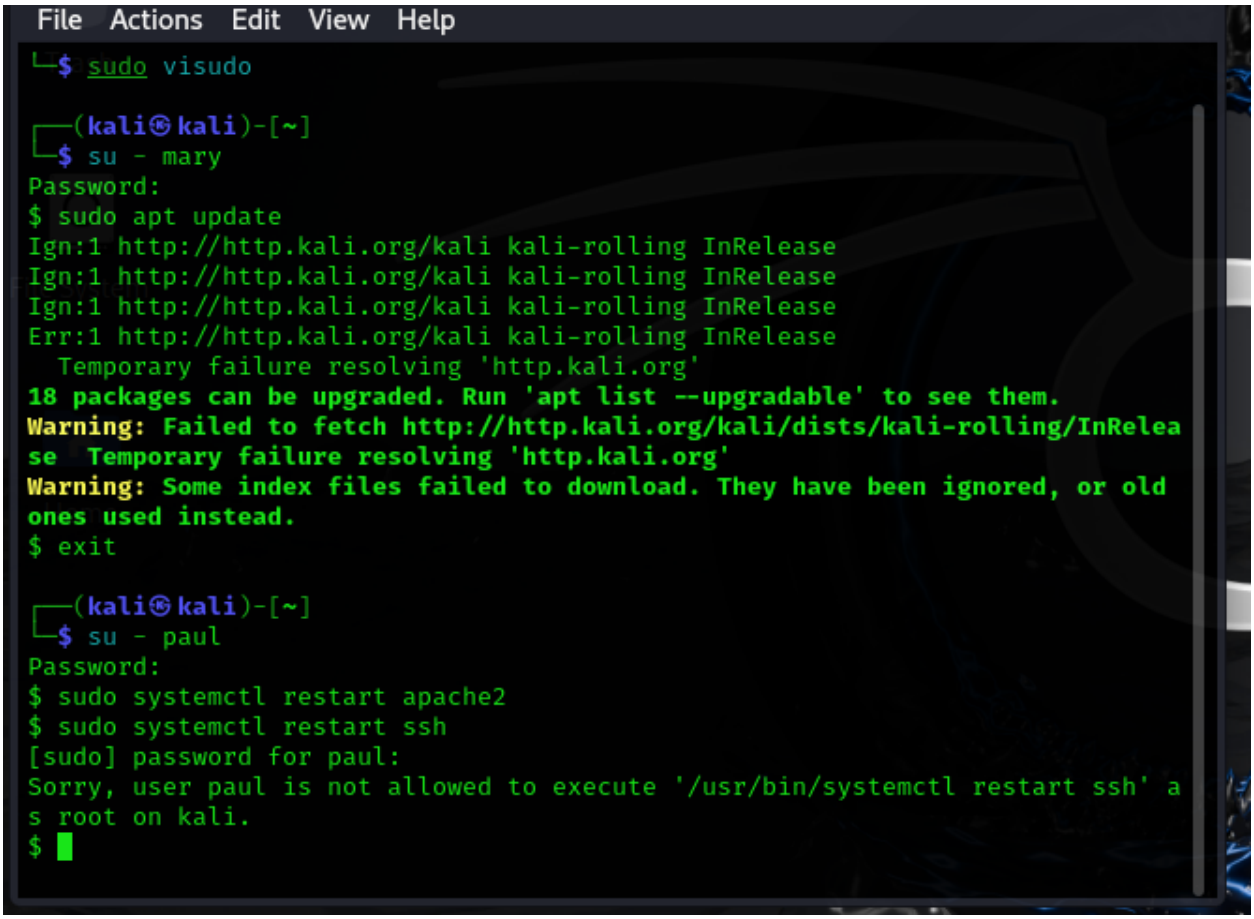
John is configured with superuser privilege and added to the sudo group using sudo usermod -aG sudo john and it was verified su - john sudo whoami, command which returned "root".

```
kali@kali: ~
File  Actions  Edit  View  Help
┌──(kali㉿kali)-[~]
└─$ sudo usermod -aG sudo john
[sudo] password for kali:

┌──(kali㉿kali)-[~]
└─$ su - john
Password:
su: Authentication failure

┌──(kali㉿kali)-[~]
└─$ su - john sudo whoami
Password:
-sh: 0: cannot open sudo: No such file

┌──(kali㉿kali)-[~]
└─$ su - john
Password:
$ sudo whoami
[sudo] password for john:
root
$
```

To grant mary privilege of managing system updates, sudoers privilege was edited and the following rule was added to sudo using sudo visudo . mary ALL=(ALL) NOPASSWD: /usr/bin/apt update, /usr/bin/apt upgrade, the path for update and upgrade was added because that

is the specific task mary was allow to do with sudo privileges, assuming we put ALL in in front of the nopasswd without specify path, mary will have all privileges.

Paul was also grant privileges to manage a specific service, by using paul ALL=(ALL) NOPASSWD: /bin/systemctl restart apache2, /bin/systemctl restart mysql. Paul will be able to restart both apache2 and mysql service and his work were limited to these two, he can not restart any other services except the two. Below screenshot show it testing.



Exercise 3: Restrict Access to the Sudo Command

To ensure restriction to sudo access by users and know which user have access to sudo group using getent group sudo, this will give us the username and that can be remove by using command sudo deluser <username> sudo. As shown below.

```
kali@kali: ~
File  Actions  Edit  View  Help
┌──(kali㉿kali)-[~]
└─$ getent group sudo
sudo:x:27:kali,john

┌──(kali㉿kali)-[~]
└─$ sudo deluser john sudo
info: Removing user `john' from group `sudo' ...

┌──(kali㉿kali)-[~]
└─$
```

Exercise 4: Logging and Monitoring Sudo Usage

```
File  Actions  Edit  View  Help
┌──(kali㉿kali)-[~]
└─$  sudo tail -f /var/log/auth.log
2025-01-03T02:25:14.775961-05:00 kali lightdm: pam_unix(lightdm-greeter:sessi
on): session opened for user lightdm(uid=112) by (uid=0)
2025-01-03T02:27:48.629558-05:00 kali lightdm: gkr-pam: unable to locate daem
on control file
2025-01-03T02:27:48.636441-05:00 kali lightdm: gkr-pam: stashed password to t
ry later in open session
2025-01-03T02:27:49.502380-05:00 kali lightdm: pam_unix(lightdm-greeter:sessi
on): session closed for user lightdm
2025-01-03T02:27:49.506410-05:00 kali lightdm: pam_unix(lightdm-greeter:sessi
on): session closed for user lightdm
2025-01-03T02:27:49.559465-05:00 kali lightdm: pam_systemd(lightdm-greeter:se
ssion): Failed to release session: Transport endpoint is not connected
2025-01-03T02:27:49.633595-05:00 kali systemd-logind[504]: Removed session c4
.
2025-01-03T02:28:00.053268-05:00 kali systemd-logind[504]: Removed session 14
.
2025-01-03T02:29:14.647042-05:00 kali sudo:     kali : TTY=pts/0 ; PWD=/home/
kali ; USER=root ; COMMAND=/usr/bin/tail -f /var/log/auth.log
2025-01-03T02:29:14.662996-05:00 kali sudo: pam_unix(sudo:session): session o
pened for user root(uid=0) by kali(uid=1000)
```

The above screenshot shows that only kali has root user and has logging sudo command execution.

Securing Network Services and Port

To secure network service and port, it essential to know which service and opens are open to determine whether it necessary to be open or not. To know this sudo ss -tuln command is run which show protocol in use and port.

To stop unnecessary service (e.g., Apache), sudo systemctl stop apache2 sudo systemctl disable apache2 command is used.

```
┌──(kali㉿kali)-[~]
└─$ sudo ss -tuln
Netid  State   Recv-Q   Send-Q     Local Address:Port     Peer Address:Port
tcp    LISTEN  0        511                   *:80                 *:*

┌──(kali㉿kali)-[~]
└─$ sudo systemctl stop apache2

┌──(kali㉿kali)-[~]
└─$ sudo systemctl disable apache2
Synchronizing state of apache2.service with SysV service script with /usr/lib
/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install disable apache2

┌──(kali㉿kali)-[~]
```

Exercise 4: SSH and Remote Access

Ssh port was edited from 22 which default port to port 2222 and the service was restarted using command sudo systemctl restart sshd or ssh.

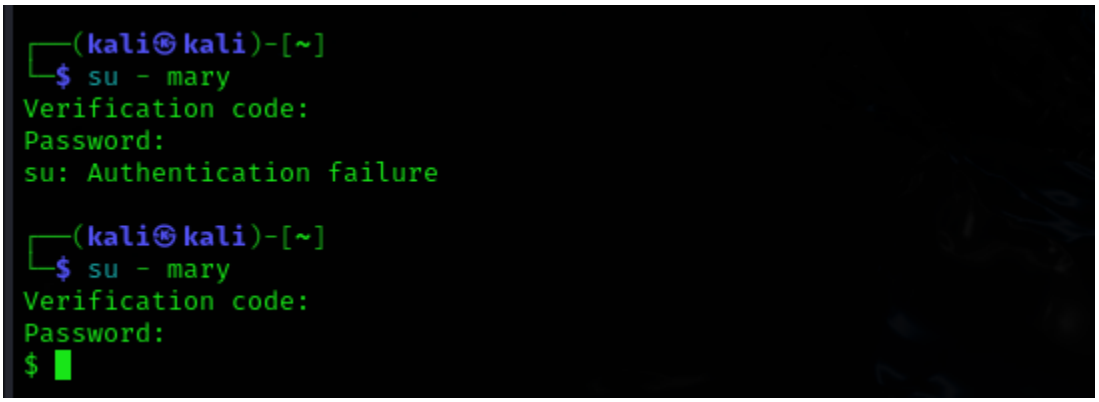
```
kali@kali: ~
File  Actions  Edit  View  Help

┌──(kali㉿kali)-[~]
└─$ sudo nano /etc/ssh/sshd_config

┌──(kali㉿kali)-[~]
└─$ sudo systemctl restart sshd
Failed to restart sshd.service: Unit sshd.service not found.

┌──(kali㉿kali)-[~]
└─$ sudo systemctl restart ssh

┌──(kali㉿kali)-[~]
└─$
```

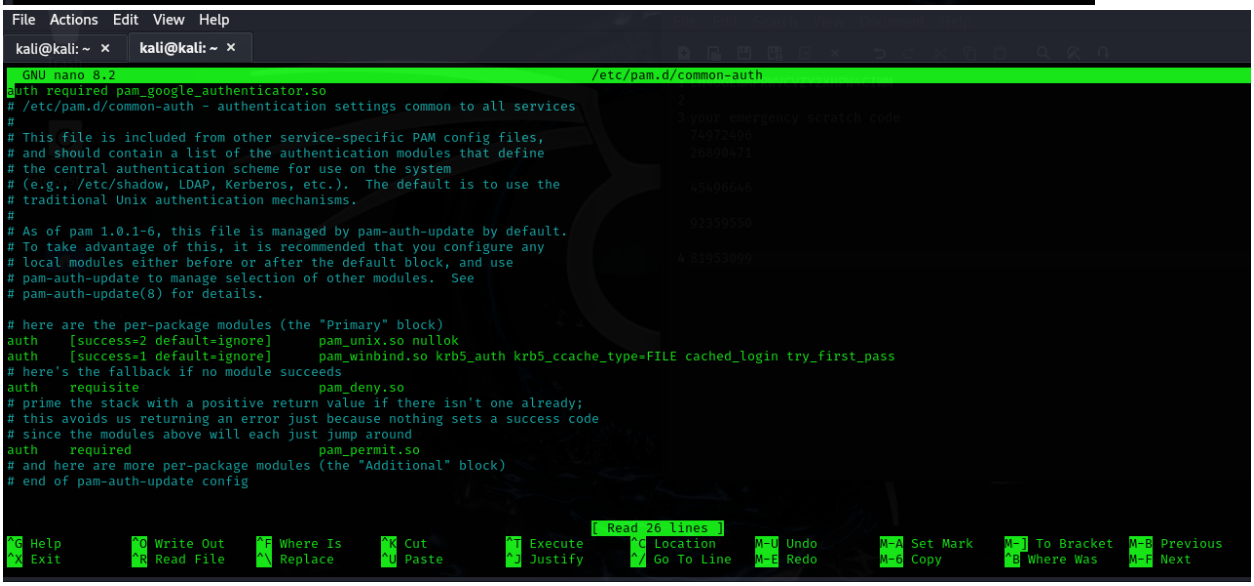
Exercise 5: System Update and Patching

An updates was check using sudo apt update and sudo apt upgrade. To automate Updates and Install unattended-upgrades, with command sudo apt install unattended-upgrades sudo dpkg-reconfigure unattended-upgrades

Implementing Two-Factor Authentication (2FA) with Google Authenticator

Two factor authentication (2fA) with using of google authentication is used to manage access control on mary log in. the implementation was added and install in both kali and phone. When the switch to user mary, a verification code was prompted and the 2FA token generated by the Google Authenticator app was provided along side with mary password and log in was successful.





Configure SSH Key-Based Authentication

On the local machine an ssh was generated with an SSH key pair (public and private keys) using command ssh-keygen -t rsa -b 4096 and save at default directory. Than public kay was copy to the server using ssh-copy-id kali@192.168.92.30. And ssh file was configured on the server by edit and ensure both are set PubkeyAuthentication: yes PasswordAuthentication: no.

Ssh was tested and no password was asked while login to the server.

root@kali: /home/kali

File Actions Edit View Help

```
[sudo] password for kali:
┌──(root㉿kali)-[/home/kali]
└─# ssh-keygen -t rsa -b 4096

Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
Enter passphrase for "/root/.ssh/id_rsa" (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa
Your public key has been saved in /root/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:0kb7xjWmRG/0kczMHRdXHWmbck31LoRnl5n7PRcf+Cw root@kali
The key's randomart image is:
+---[RSA 4096]----+
|               +%|
|              * =@|
|         . . o %=B|
|        o o o *.Bo|
|       . S . *.=+.|
|        o + = .o.*|
|           =  E +=|
|          .    . o|
|                 |
+----[SHA256]-----+

┌──(root㉿kali)-[/home/kali]
└─#
```

```
┌──(root㉿kali)-[/home/kali]
└─# ssh-copy-id mary@192.168.92.30
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/root/.ssh/id_
rsa.pub"
The authenticity of host '192.168.92.30 (192.168.92.30)' can't be established
.
ED25519 key fingerprint is SHA256:xCcEdmIFmDMN+ROefnG1skG2pNpd7wkHkywW8fnGJDU
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? y
Please type 'yes', 'no' or the fingerprint: yes
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filt
er out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are pro
mpted now it is to install the new keys
mary@192.168.92.30: Permission denied (publickey).

┌──(root㉿kali)-[/home/kali]
└─#
```

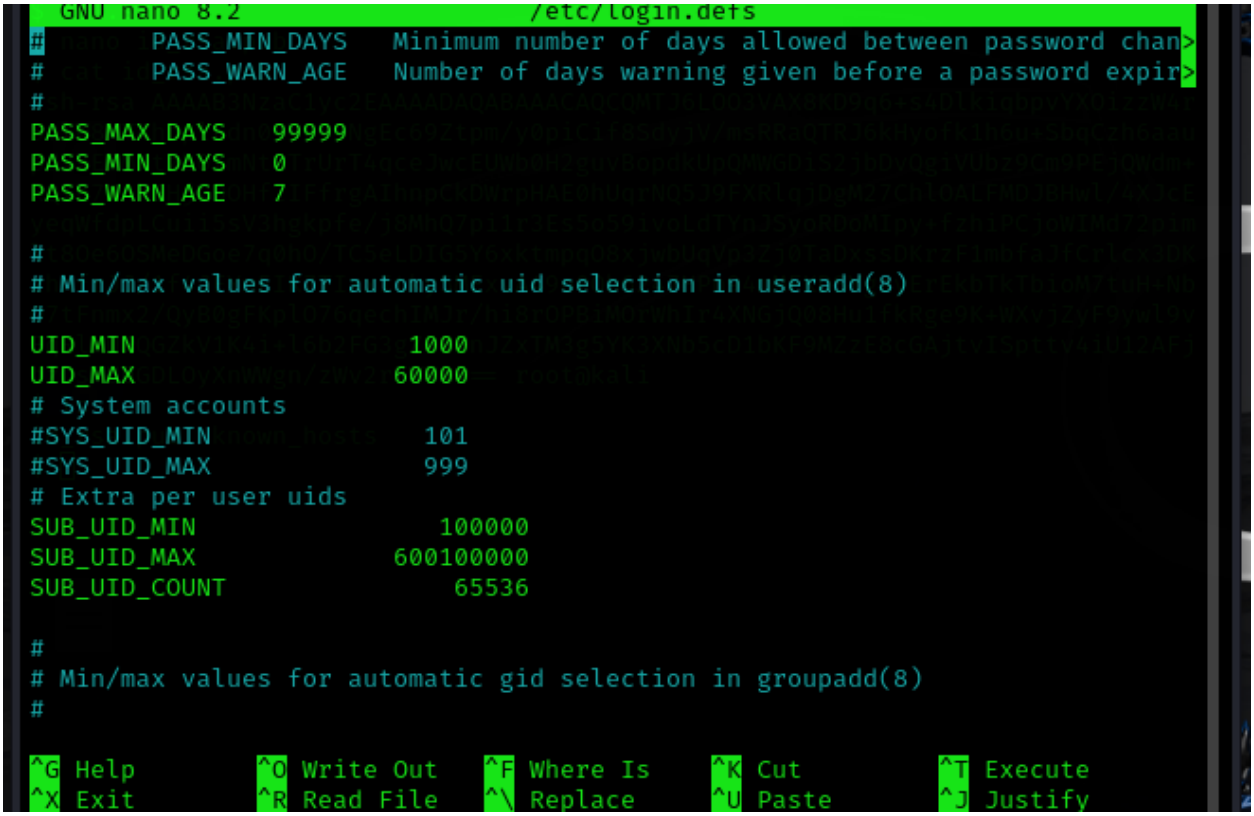
```
┌──(root㉿kali)-[~/.ssh]
└─# ssh kali
Enter passphrase for key '/root/.ssh/id_rsa':
Linux kali 6.11.2-amd64 #1 SMP PREEMPT_DYNAMIC Kali 6.11.2-1kali1 (2024-10-15
) x86_64

The programs included with the Kali GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Kali GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
┌──(root㉿kali)-[~]
└─#
```

Root login was also configured in the ssh configuration file by disable PermitRootLogin: no in the /etc/ssh/sshd_config directory, the root verification login is permissible without password

```
┌──(root㉿kali)-[~]
└─# ssh root@192.168.92.30
Enter passphrase for key '/root/.ssh/id_rsa':
Linux kali 6.11.2-amd64 #1 SMP PREEMPT_DYNAMIC Kali 6.11.2-1kali1 (2024-10-15
) x86_64

The programs included with the Kali GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Kali GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Sun Jan  5 08:36:01 2025 from 127.0.0.1
┌──(root㉿kali)-[~]
└─#
```

Configuring User Account Security with login.defs

To ensure that all user accounts adhere to these policies for better security management, the login.defs was edited and configured using sudo nano /etc/login.defs and PASS_MAX_DAYS: Maximum number of days a password is valid, PASS_MIN_DAYS: Minimum number of days between password changes, and PASS_WARN_AGE: Number of days before expiration that a warning is issued were set to 90, 7 and 14 respectively.



Inorder to test for this configuration a new user was added and sudo chage -l testuser is used to view the expiration date of the new configuration. The news was tested by login to the user and try to change it password. This was denied because the days setting is not set yet. This show in the two screenshot below

```
┌──(kali㉿kali)-[~]
└─$ sudo useradd testuser1

┌──(kali㉿kali)-[~]
└─$ sudo chage -l testuser1
Last password change                                    : Jan 05, 2025
Password expires                                        : Apr 05, 2025
Password inactive                                       : never
Account expires                                         : never
Minimum number of days between password change          : 7
Maximum number of days between password change          : 90
Number of days of warning before password expires       : 14

┌──(kali㉿kali)-[~]
└─$ sudo passwd testuser1
New password:
Retype new password:
passwd: password updated successfully
```

```
File  Actions  Edit  View  Help
$ passwd testuser1
Changing password for testuser1.
Current password:
You must wait longer to change your password.
passwd: Authentication token manipulation error
passwd: password unchanged
$
```

Configuring System Auditing with auditd

To configure audits to track critical events and to monitor key activities such as user logins, file access, and changes to system files. Install audit using sudo apt update && sudo apt install auditd, then start the service by sudo systemctl start auditd sudo systemctl enable auditd.

View the current aduit rule, to do this, use sudo auditctl -l Command to list the current audit rules to see the default configuration, Add an audit rule to monitor successful and failed login attempts by editing the rules file sudo nano /etc/audit/rules.d/audit.rules, Add the following lines to monitor user logins -w /var/log/auth.log -p wa -k user-logins, Add Rule to Monitor Access to Sensitive Files , Add audit rules to monitor access and modification of critical system files, such as /etc/passwd -w /etc/passwd -p wa -k passwd-modifications, Save and Apply Rules, then After making the changes, save the file and reload the audit rules using sudo service auditd restart

```
File  Actions  Edit  View  Help
┌──(kali㉿kali)-[~]
└─$  sudo ausearch -k passwd-modifications
time→Sun Jan  5 10:02:44 2025
type=PROCTITLE msg=audit(1736089364.993:104): proctitle=757365726164640074657
3747573657232
type=PATH msg=audit(1736089364.993:104): item=0 name="/etc/passwd" inode=2496
292 dev=08:01 mode=0100644 ouid=0 ogid=0 rdev=00:00 nametype=NORMAL cap_fp=0
cap_fi=0 cap_fe=0 cap_fver=0 cap_frootid=0
type=CWD msg=audit(1736089364.993:104): cwd="/home/kali"
type=SYSCALL msg=audit(1736089364.993:104): arch=c000003e syscall=257 success
=yes exit=5 a0=ffffff9c a1=564be52c5ee0 a2=a0902 a3=0 items=1 ppid=168294 pid
=168295 auid=1000 uid=0 gid=0 euid=0 suid=0 fsuid=0 egid=0 sgid=0 fsgid=0 tty
=pts2 ses=3 comm="useradd" exe="/usr/sbin/useradd" subj=unconfined key="passw
d-modifications"
time→Sun Jan  5 10:02:45 2025
type=PROCTITLE msg=audit(1736089365.061:107): proctitle=757365726164640074657
3747573657232
type=PATH msg=audit(1736089365.061:107): item=4 name="/etc/passwd" inode=2496
314 dev=08:01 mode=0100644 ouid=0 ogid=0 rdev=00:00 nametype=CREATE cap_fp=0
cap_fi=0 cap_fe=0 cap_fver=0 cap_frootid=0
type=PATH msg=audit(1736089365.061:107): item=3 name="/etc/passwd" inode=2496
292 dev=08:01 mode=0100644 ouid=0 ogid=0 rdev=00:00 nametype=DELETE cap_fp=0
cap_fi=0 cap_fe=0 cap_fver=0 cap_frootid=0
type=PATH msg=audit(1736089365.061:107): item=2 name="/etc/passwd+" inode=249
```

To view generated log, a user login was demonstrated, and this do by adding user and change password, and also a modification was also simulated by modify /etc/passwd file. The both audit log was view using sudo ausearch -k user-logins and sudo ausearch -k passwd-modifications to review both user login and modifications in the files.

File Actions Edit View Help

```
┌──(kali㉿kali)-[~]
└─$ sudo ausearch -i
----
type=DAEMON_START msg=audit(01/05/2025 09:54:20.006:5642) : op=start ver=4.0
2 format=enriched kernel=6.11.2-amd64 auid=unset pid=163936 uid=root ses=uns
t subj=unconfined  res=success
----
type=USER_END msg=audit(01/05/2025 09:54:20.011:3) : pid=163908 uid=kali auid
=kali ses=3 subj=unconfined msg='op=PAM:session_close grantors=pam_limits,pam
_permit,pam_umask,pam_unix,pam_winbind acct=root exe=/usr/bin/sudo hostname=?
 addr=? terminal=/dev/pts/0 res=success'
----
type=CRED_DISP msg=audit(01/05/2025 09:54:20.011:4) : pid=163908 uid=kali aui
d=kali ses=3 subj=unconfined msg='op=PAM:setcred grantors=pam_permit acct=roo
t exe=/usr/bin/sudo hostname=? addr=? terminal=/dev/pts/0 res=success'
----
type=SERVICE_START msg=audit(01/05/2025 09:54:20.019:5) : pid=1 uid=root auid
=unset ses=unset subj=unconfined msg='unit=auditd comm=systemd exe=/usr/lib/s
ystemd/systemd hostname=? addr=? terminal=? res=success'
----
type=USER_ACCT msg=audit(01/05/2025 09:55:01.749:6) : pid=164281 uid=root aui
d=unset ses=unset subj=unconfined msg='op=PAM:accounting grantors=pam_permit
acct=root exe=/usr/sbin/cron hostname=? addr=? terminal=cron res=success'
----
type=CRED_ACQ msg=audit(01/05/2025 09:55:01.749:7) : pid=164281 uid=root auid
=unset ses=unset subj=unconfined msg='op=PAM:setcred grantors=pam_permit acct
=root exe=/usr/sbin/cron hostname=? addr=? terminal=cron res=success'
```