CL-2006 Operating Systems

LAB - 14

Squid proxy server
Linux hardening, security and firewall

Objective:

In today's OS lab, we will cover essential topics in securing a Linux system, including hardening techniques, firewall configuration, and security best practices, as well as setting up and configuring a Squid proxy server. By the end of this lab, we will have developed the knowledge and skills necessary to build and maintain secure Linux systems with Squid proxy server, critical for any modern computing environment.

Introduction to Squid

Squid is a popular open-source caching and proxy server that runs on Linux and Unix-like operating systems. It provides a range of features, including caching web pages, filtering requests, and logging user activity. Squid is commonly used by organizations to improve network performance by reducing bandwidth usage and speeding up web page loading times for users. Squid can also be used for content filtering, blocking access to certain websites, and enforcing access control policies

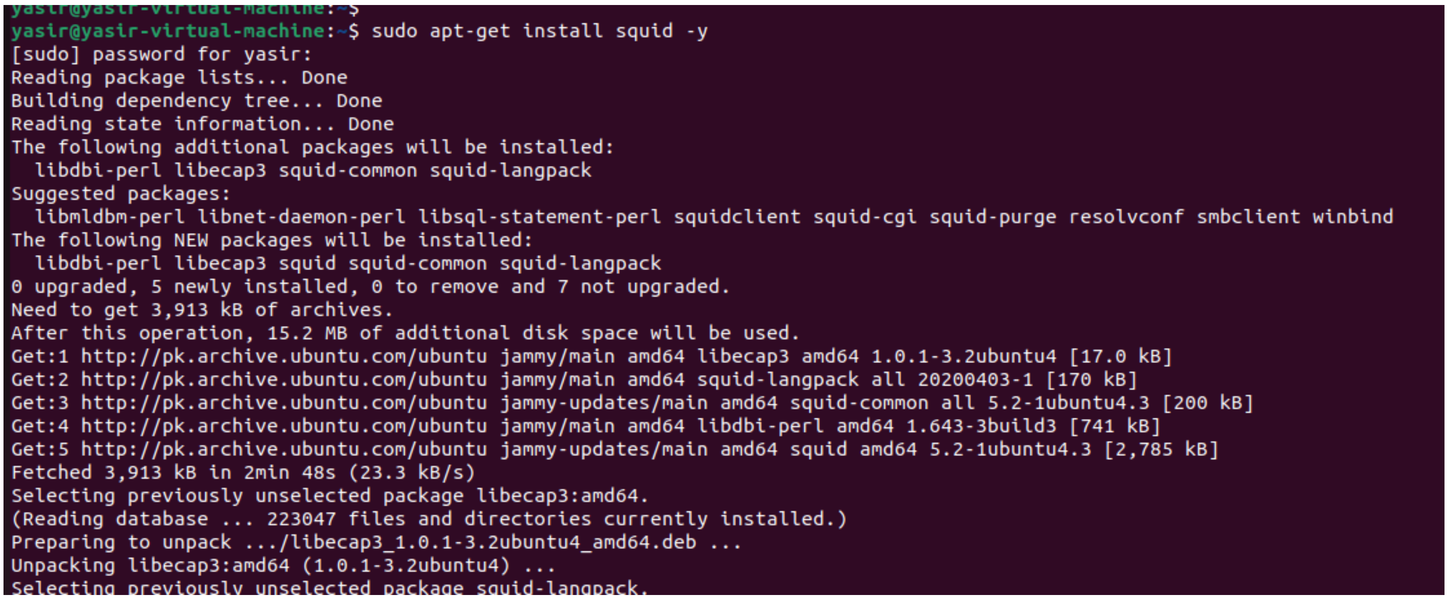
Step by step squid installation and configuration is given below

1. To install the squid package, run the following command:

```
sudo apt-get update
```

```
sudo apt-get install squid
```

Note: The `-y` flag allows skipping the installation confirmation prompt.



```
yasir@yasir-virtual-machine:~$
yasir@yasir-virtual-machine:~$ sudo apt-get install squid -y
[sudo] password for yasir:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libdbi-perl libecap3 squid-common squid-langpack
Suggested packages:
  libmldbm-perl libnet-daemon-perl libsql-statement-perl squidclient squid-cgi squid-purge resolvconf smbclient winbind
The following NEW packages will be installed:
  libdbi-perl libecap3 squid squid-common squid-langpack
0 upgraded, 5 newly installed, 0 to remove and 7 not upgraded.
Need to get 3,913 kB of archives.
After this operation, 15.2 MB of additional disk space will be used.
Get:1 http://pk.archive.ubuntu.com/ubuntu jammy/main amd64 libecap3 amd64 1.0.1-3.2ubuntu4 [17.0 kB]
Get:2 http://pk.archive.ubuntu.com/ubuntu jammy/main amd64 squid-langpack all 20200403-1 [170 kB]
Get:3 http://pk.archive.ubuntu.com/ubuntu jammy-updates/main amd64 squid-common all 5.2-1ubuntu4.3 [200 kB]
Get:4 http://pk.archive.ubuntu.com/ubuntu jammy/main amd64 libdbi-perl amd64 1.643-3build3 [741 kB]
Get:5 http://pk.archive.ubuntu.com/ubuntu jammy-updates/main amd64 squid amd64 5.2-1ubuntu4.3 [2,785 kB]
Fetched 3,913 kB in 2min 48s (23.3 kB/s)
Selecting previously unselected package libecap3:amd64.
(Reading database ... 223047 files and directories currently installed.)
Preparing to unpack .../libecap3_1.0.1-3.2ubuntu4_amd64.deb ...
Unpacking libecap3:amd64 (1.0.1-3.2ubuntu4) ...
Selecting previously unselected package squid-langpack.
```

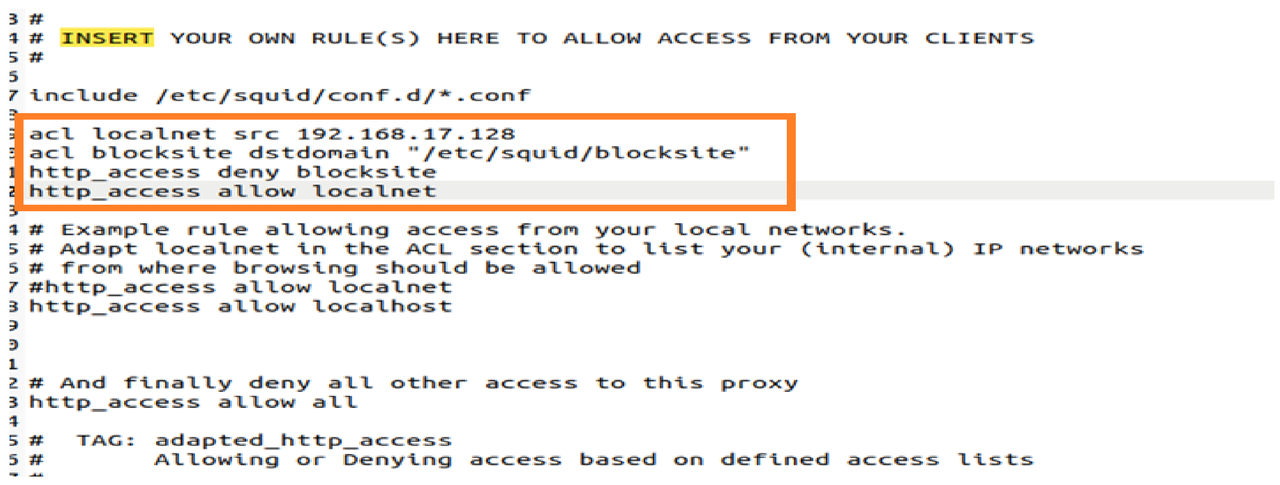
2. In step 2 we will do squid configuration

The main configuration file for Squid is **/etc/squid/squid.conf**. Here are some common configuration changes you might make:

To make configuration use the comand

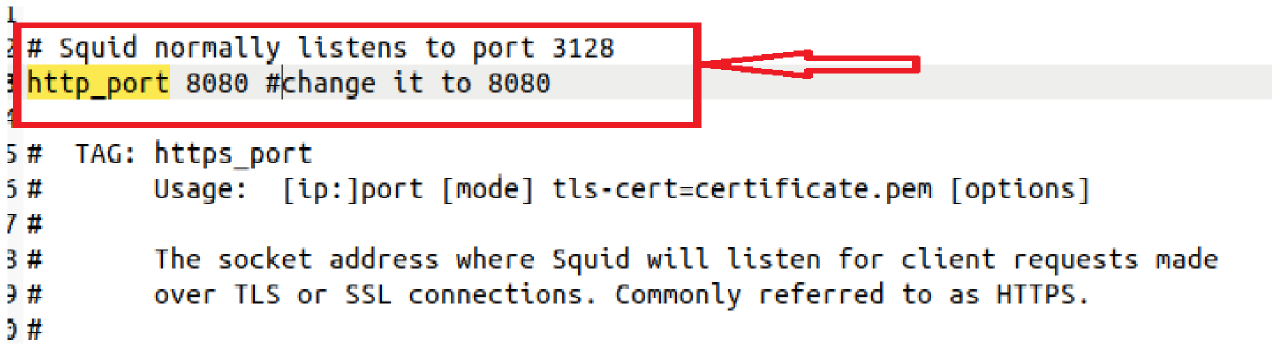
sudo Gedit /etc/squi.conf

3. Do the following configuratio



```
#
# INSERT YOUR OWN RULE(S) HERE TO ALLOW ACCESS FROM YOUR CLIENTS
#

include /etc/squid/conf.d/*.conf

acl localnet src 192.168.17.128
acl blocksite dstdomain "/etc/squid/blocksite"
http_access deny blocksite
http_access allow localnet

# Example rule allowing access from your local networks.
# Adapt localnet in the ACL section to list your (internal) IP networks
# from where browsing should be allowed
#http_access allow localnet
http_access allow localhost



# And finally deny all other access to this proxy
http_access allow all

#   TAG: adapted_http_access
#        Allowing or Denying access based on defined access lists
```

Change the default port

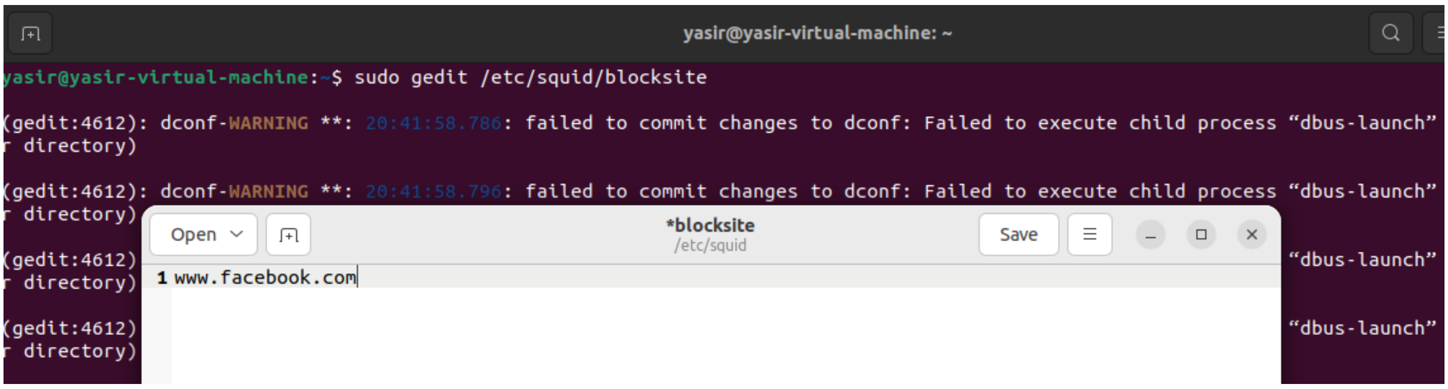


```
# Squid normally listens to port 3128
http_port 8080 #change it to 8080

#  TAG: https_port
#       Usage:  [ip:]port [mode] tls-cert=certificate.pem [options]
#
#       The socket address where Squid will listen for client requests made
#       over TLS or SSL connections. Commonly referred to as HTTPS.
#
```

4. Now go to black site file using comand

   sudo gedit /etc/squid/blocksite

   add all the sites you want to block

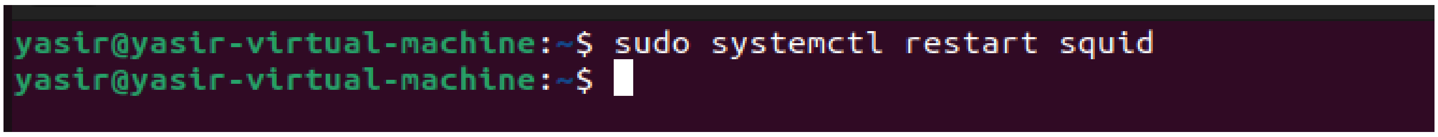


```
yasir@yasir-virtual-machine: ~
asir@yasir-virtual-machine:~$ sudo gedit /etc/squid/blocksite

(gedit:4612): dconf-WARNING **: 20:41:58.786: failed to commit changes to dconf: Failed to execute child process “dbus-launch”
r directory)

(gedit:4612): dconf-WARNING **: 20:41:58.796: failed to commit changes to dconf: Failed to execute child process “dbus-launch”
r directory)

(gedit:4612)                                                                                                 “dbus-launch”
r directory)

(gedit:4612)                                                                                                 “dbus-launch”
r directory)
```

```
*blocksite
/etc/squid
1 www.facebook.com
```

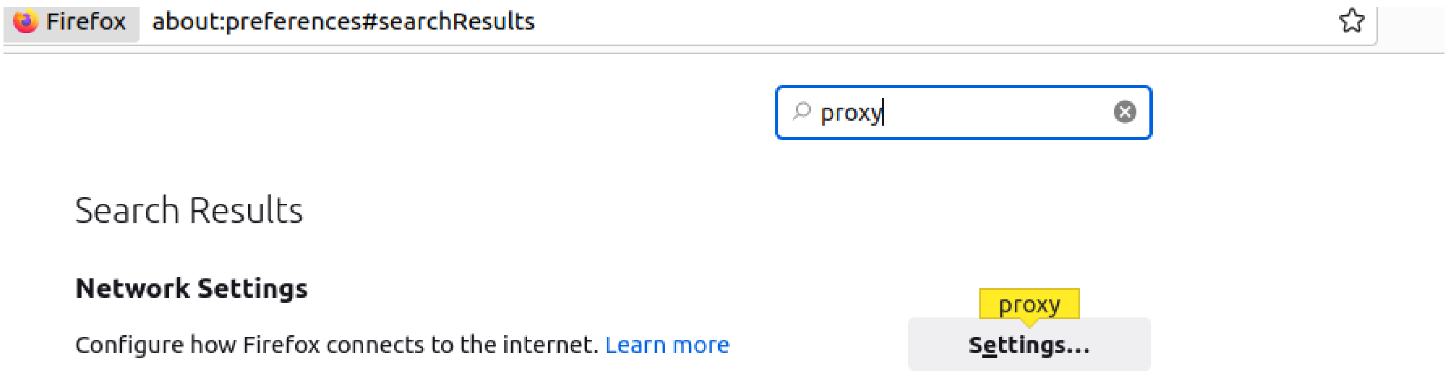
5. Now restart squid using command

   sudo systemctl restart squid



```
yasir@yasir-virtual-machine:~$ sudo systemctl restart squid
yasir@yasir-virtual-machine:~$
```

6. Now go to browser Go to setting and search for proxy



Firefox about:preferences#searchResults

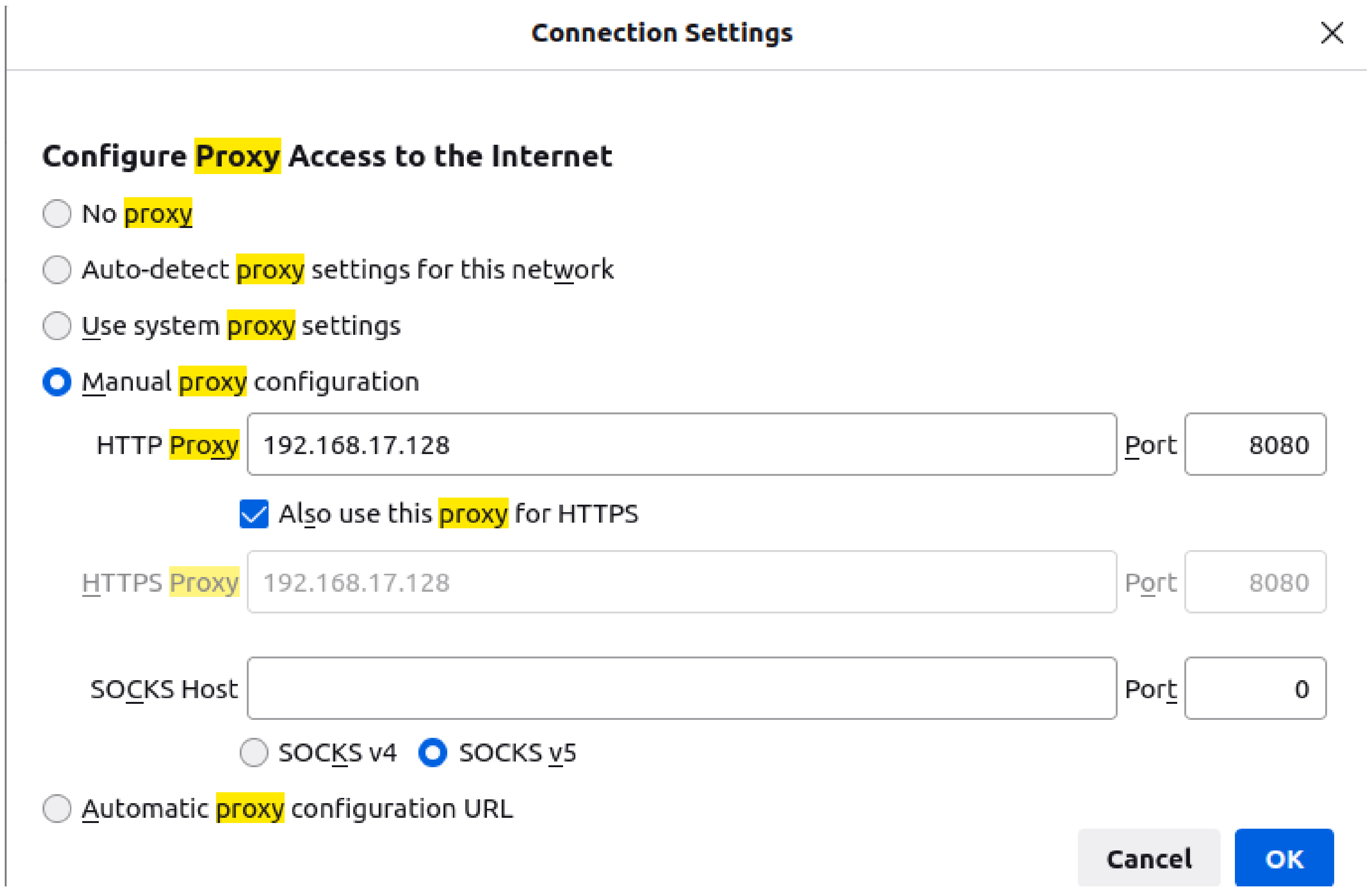
proxy

Search Results

Network Settings

Configure how Firefox connects to the internet. Learn more

7. Click on settings Select manual proxy configuration and your ip and default por


Connection Settings

Configure Proxy Access to the Internet

- No proxy
- Auto-detect proxy settings for this network
- Use system proxy settings
- [✓ Manual proxy configuration]

HTTP Proxy 192.168.17.128 Port 8080

[✓] Also use this proxy for HTTPS

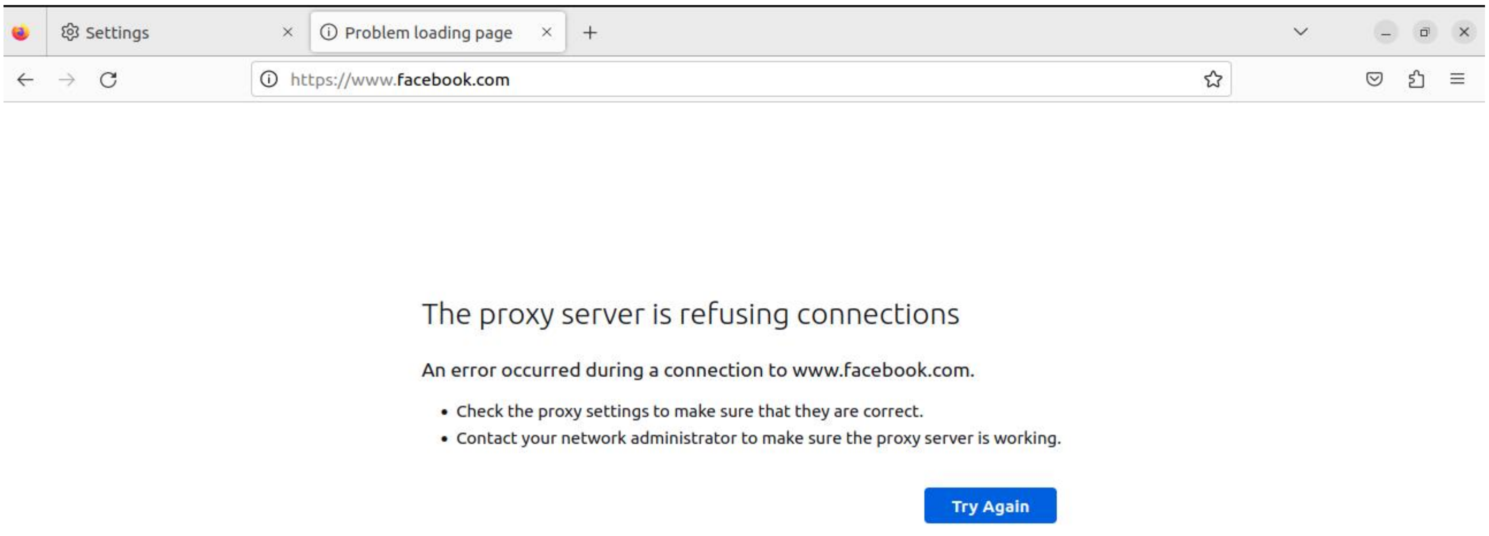
HTTPS Proxy 192.168.17.128 Port 8080

SOCKS Host Port 0

SOCKS v4 [✓ SOCKS v5]

- Automatic proxy configuration URL

8. Now Browse for site you block



https://www.facebook.com

The proxy server is refusing connections

An error occurred during a connection to www.facebook.com.

- Check the proxy settings to make sure that they are correct.
- Contact your network administrator to make sure the proxy server is working.

Introduction to Linux hardening security and firewall

Linux hardening refers to the process of securing a Linux system by taking a proactive approach to mitigate security threats and vulnerabilities. The goal of hardening is to make it more difficult for attackers to compromise the system, by reducing the attack surface and implementing security best practices

Some basic Linux hardening techniques that can be used include:

1. Disabling unnecessary services and applications
2. Keeping the system up-to-date with security patches and updates
3. Configuring and using a firewall to control network traffic
4. Implementing strong password policies
5. Configuring file system permissions to restrict access to sensitive files and directories
6. Implementing user management best practices, such as limiting user privileges and monitoring user activity
7. Linux hardening tools

By applying these techniques and other security best practices, you can improve the security posture of your Linux system and reduce the risk of compromise by malicious actors

1. Disabling unnecessary services and applications

To reduce the risk of compromise, it is recommended to disable any services or applications that are not needed. To disable a service on a Linux system,

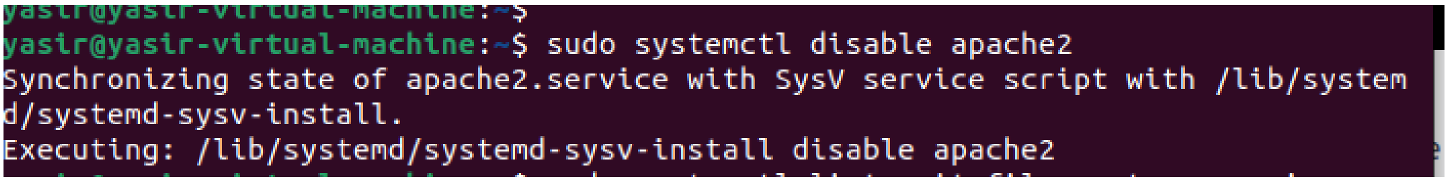
you can use the **systemctl** command, which is used to control the systemd system and service manager.

Here's the basic syntax for disabling a service using systemctl

```
sudo systemctl disable <service-name>
```

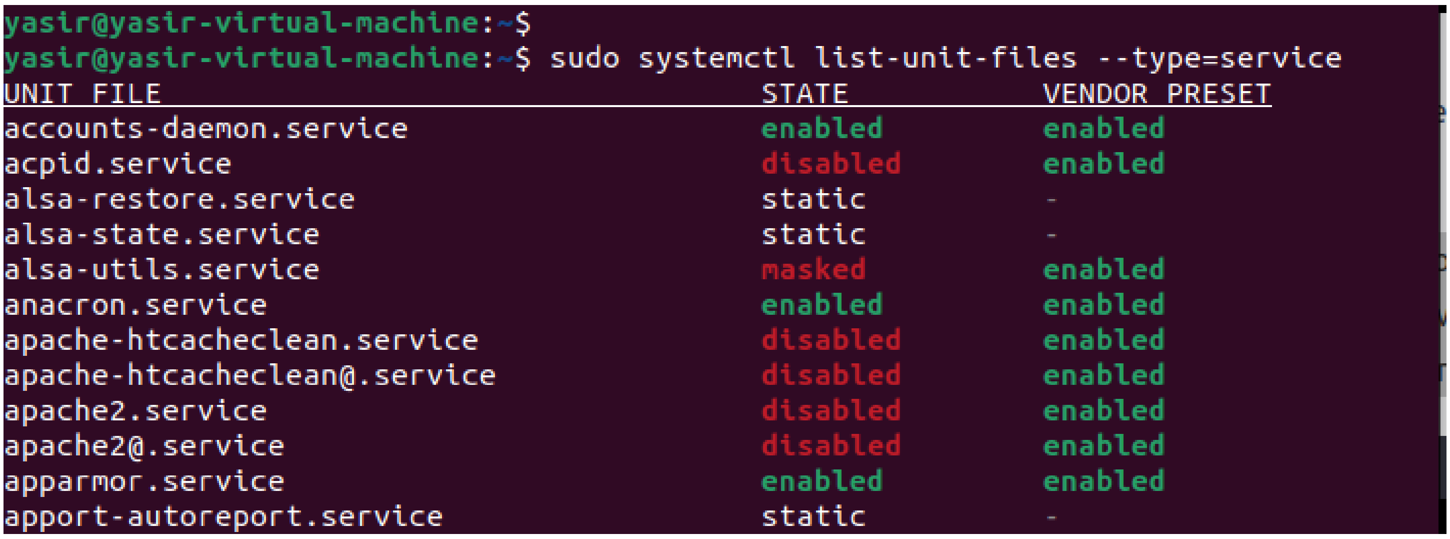
Replace <service-name> with the name of the service you want to disable.

For example, to disable the Apache web server service on a Ubuntu system, you would run the following command:



```
yasir@yasir-virtual-machine:~$
yasir@yasir-virtual-machine:~$ sudo systemctl disable apache2
Synchronizing state of apache2.service with SysV service script with /lib/system
d/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install disable apache2
```

We can use the **systemctl list-unit-files --type=service** command to list all services on the system and their current status (enabled or disabled). The output will show a list of all services, with [enabled] or [disabled] displayed next to each service name



```
yasir@yasir-virtual-machine:~$
yasir@yasir-virtual-machine:~$ sudo systemctl list-unit-files --type=service
UNIT FILE                                  STATE           VENDOR PRESET
accounts-daemon.service                    enabled         enabled
acpid.service                              disabled        enabled
alsa-restore.service                       static          -
alsa-state.service                         static          -
alsa-utils.service                         masked          enabled
anacron.service                            enabled         enabled
apache-htcacheclean.service                disabled        enabled
apache-htcacheclean@.service               disabled        enabled
apache2.service                            disabled        enabled
apache2@.service                           disabled        enabled
apparmor.service                           enabled         enabled
apport-autoreport.service                  static          -
```

2. Keeping the system up-to-date with security patches and updates

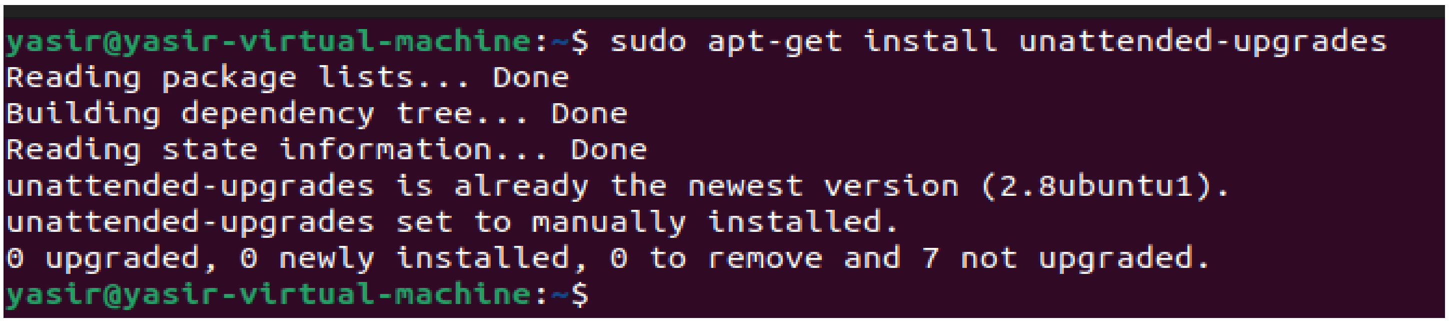
Keeping your Linux system up-to-date with the latest security patches and updates is an essential step in maintaining its security

Configure automatic updates

configure your system to automatically download and install security updates. Most Linux distributions have built-in tools to manage updates, such as apt for Debian-based distributions like Ubuntu or yum for Red Hat-based distributions like CentOS.

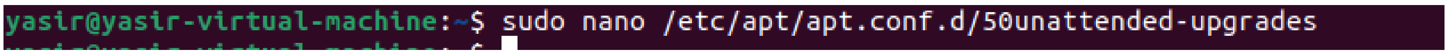
On Ubuntu, you can use the unattended-upgrades package to configure automatic updates. First, install the package using the following command:

sudo apt-get install unattended-upgrades

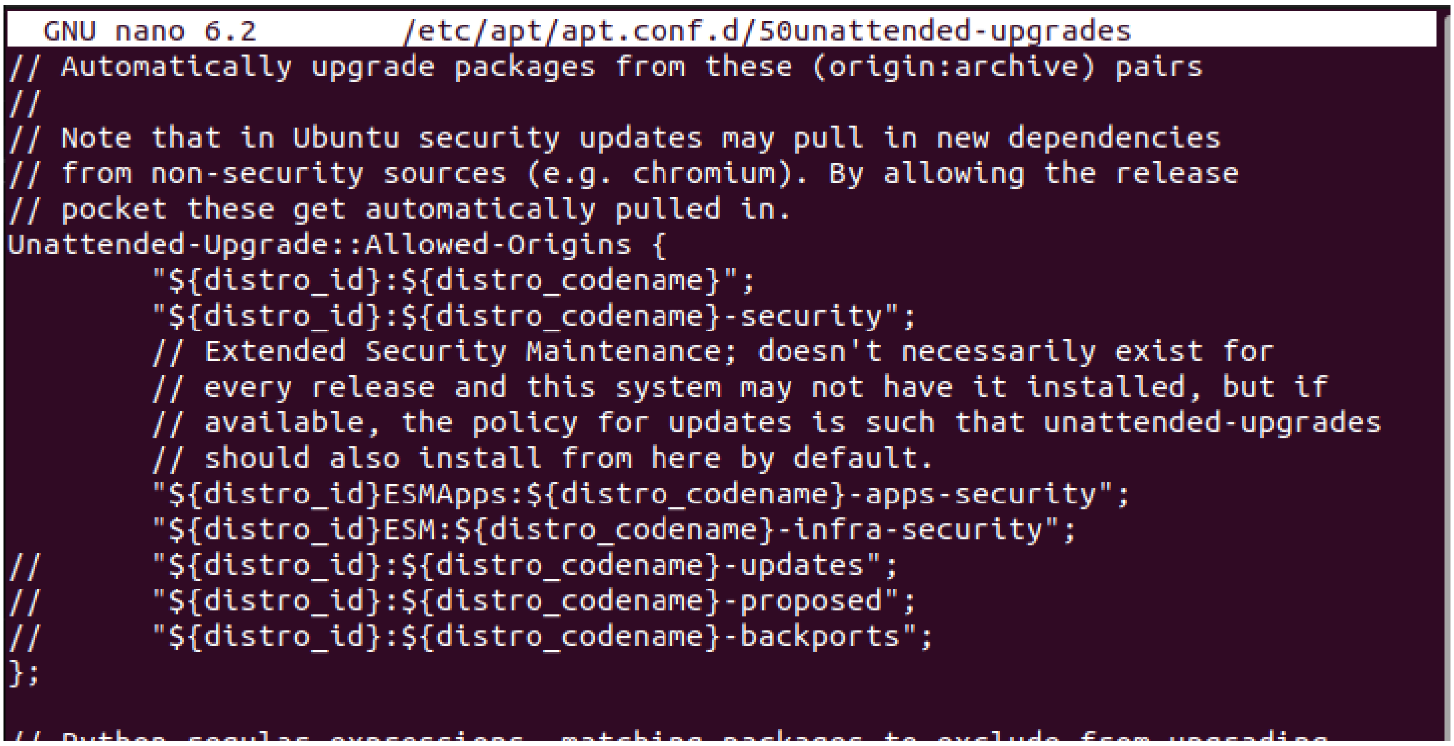


```
yasir@yasir-virtual-machine:~$ sudo apt-get install unattended-upgrades
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
unattended-upgrades is already the newest version (2.8ubuntu1).
unattended-upgrades set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 7 not upgraded.
yasir@yasir-virtual-machine:~$
```

Then, edit the **/etc/apt/apt.conf.d/50unattended-upgrades** configuration file to specify which updates you want to install automatically:



```
yasir@yasir-virtual-machine:~$ sudo nano /etc/apt/apt.conf.d/50unattended-upgrades
```



```
  GNU nano 6.2          /etc/apt/apt.conf.d/50unattended-upgrades
// Automatically upgrade packages from these (origin:archive) pairs
//
// Note that in Ubuntu security updates may pull in new dependencies
// from non-security sources (e.g. chromium). By allowing the release
// pocket these get automatically pulled in.
Unattended-Upgrade::Allowed-Origins {
        "${distro_id}:${distro_codename}";
        "${distro_id}:${distro_codename}-security";
        // Extended Security Maintenance; doesn't necessarily exist for
        // every release and this system may not have it installed, but if
        // available, the policy for updates is such that unattended-upgrades
        // should also install from here by default.
        "${distro_id}ESMApps:${distro_codename}-apps-security";
        "${distro_id}ESM:${distro_codename}-infra-security";
//      "${distro_id}:${distro_codename}-updates";
//      "${distro_id}:${distro_codename}-proposed";
//      "${distro_id}:${distro_codename}-backports";
};
```

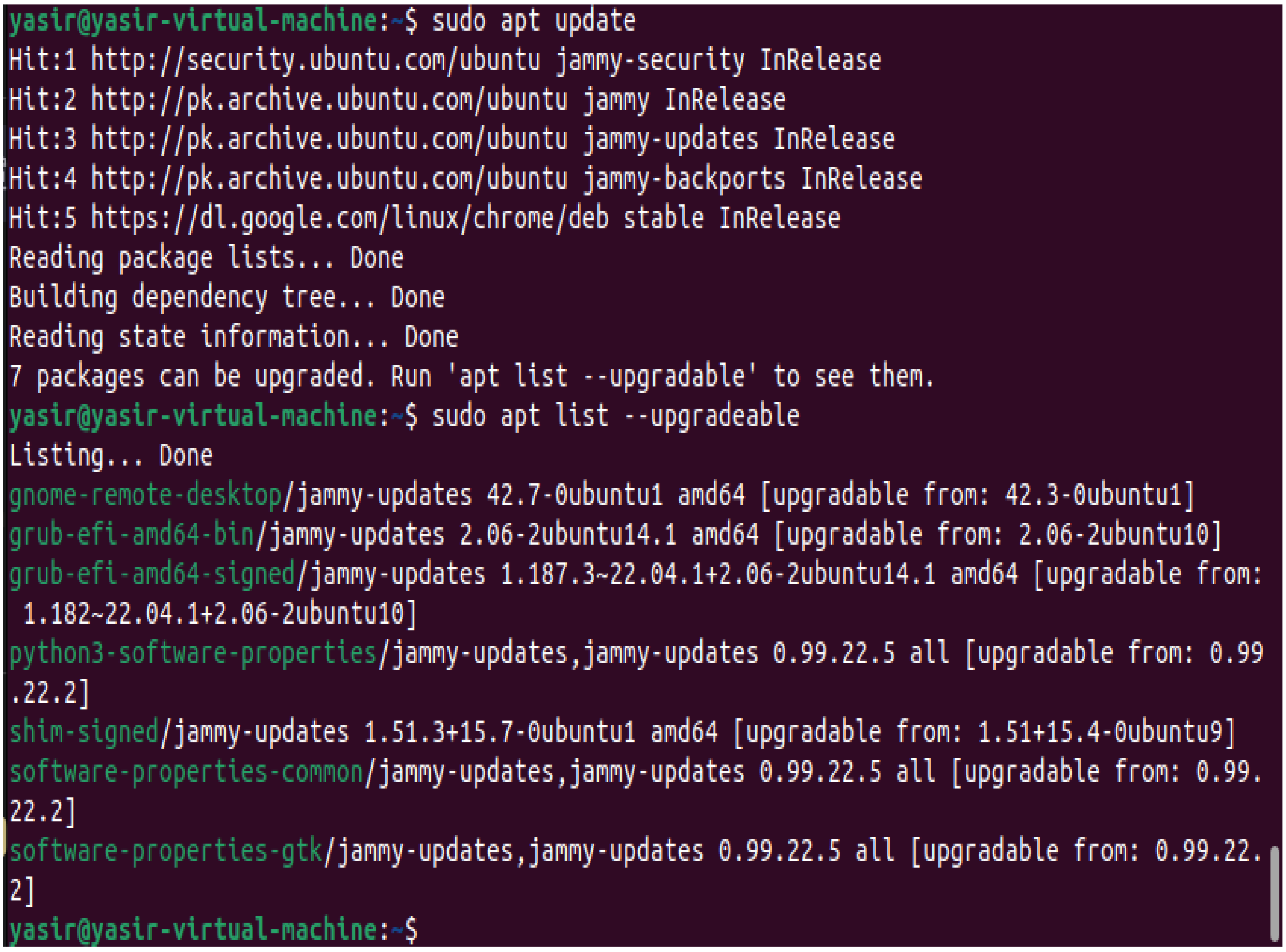
Check for updates manually:

You can also check for updates manually using the appropriate package management tool for your distribution. On Ubuntu, you can use the apt command to check for available updates:

sudo apt apdate

sudo apt list –upgradable

This will update the package repository information and show a list of available updates



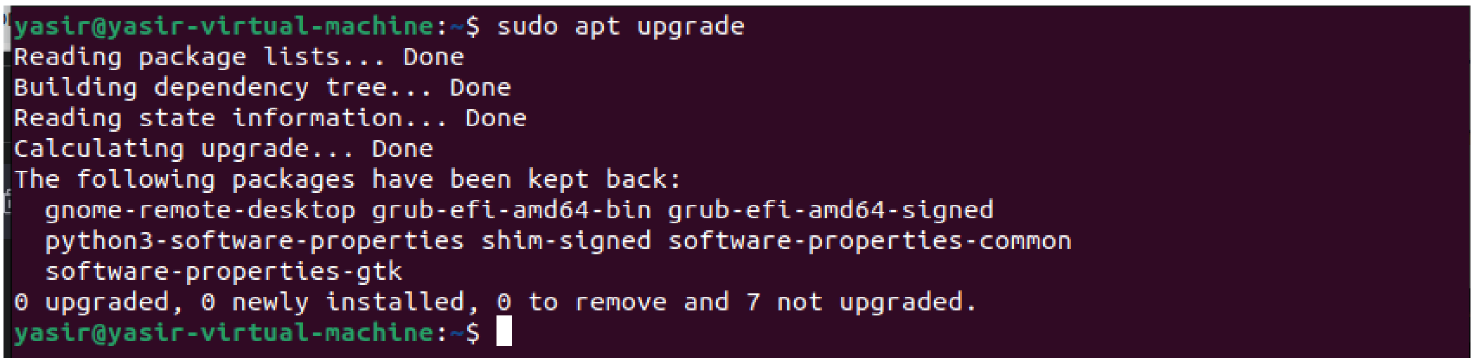
```
yasir@yasir-virtual-machine:~$ sudo apt update
Hit:1 http://security.ubuntu.com/ubuntu jammy-security InRelease
Hit:2 http://pk.archive.ubuntu.com/ubuntu jammy InRelease
Hit:3 http://pk.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:4 http://pk.archive.ubuntu.com/ubuntu jammy-backports InRelease
Hit:5 https://dl.google.com/linux/chrome/deb stable InRelease
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
7 packages can be upgraded. Run 'apt list --upgradable' to see them.
yasir@yasir-virtual-machine:~$ sudo apt list --upgradeable
Listing... Done
gnome-remote-desktop/jammy-updates 42.7-0ubuntu1 amd64 [upgradable from: 42.3-0ubuntu1]
grub-efi-amd64-bin/jammy-updates 2.06-2ubuntu14.1 amd64 [upgradable from: 2.06-2ubuntu10]
grub-efi-amd64-signed/jammy-updates 1.187.3~22.04.1+2.06-2ubuntu14.1 amd64 [upgradable from:
 1.182~22.04.1+2.06-2ubuntu10]
python3-software-properties/jammy-updates,jammy-updates 0.99.22.5 all [upgradable from: 0.99
.22.2]
shim-signed/jammy-updates 1.51.3+15.7-0ubuntu1 amd64 [upgradable from: 1.51+15.4-0ubuntu9]
software-properties-common/jammy-updates,jammy-updates 0.99.22.5 all [upgradable from: 0.99.
22.2]
software-properties-gtk/jammy-updates,jammy-updates 0.99.22.5 all [upgradable from: 0.99.22.
2]
yasir@yasir-virtual-machine:~$
```

Install security updates:

Once you've identified which updates you need, install them using the package management tool. On Ubuntu, you can use the apt command to install updates:

sudo apt upgrade

This will install all available updates, including security updates.



```
yasir@yasir-virtual-machine:~$ sudo apt upgrade
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Calculating upgrade... Done
The following packages have been kept back:
  gnome-remote-desktop grub-efi-amd64-bin grub-efi-amd64-signed
  python3-software-properties shim-signed software-properties-common
  software-properties-gtk
0 upgraded, 0 newly installed, 0 to remove and 7 not upgraded.
yasir@yasir-virtual-machine:~$
```

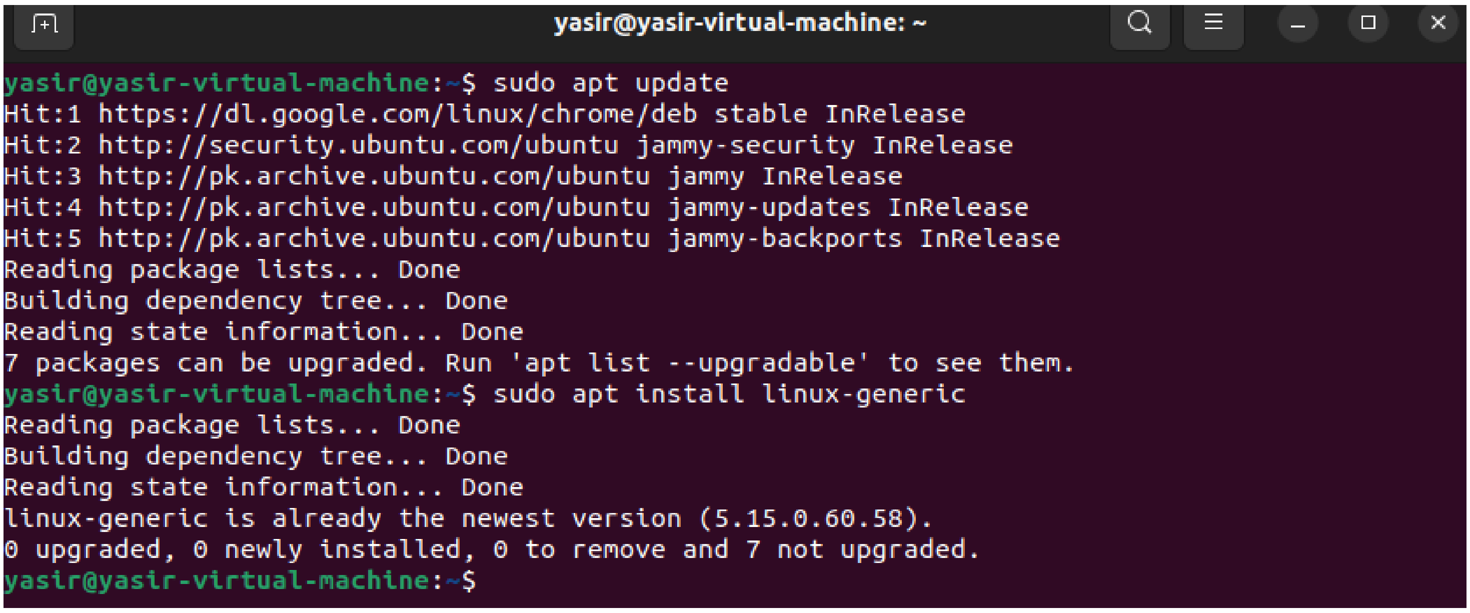
Keep your kernel up-to-date

In addition to software updates, it's also important to keep your Linux kernel up-to-date with security patches. On Ubuntu, you can use the apt command to update your kernel:

sudo apt update

sudo apt install linux-generic

This will install the latest version of the Linux kernel for your distribution.



```
yasir@yasir-virtual-machine:~$ sudo apt update
Hit:1 https://dl.google.com/linux/chrome/deb stable InRelease
Hit:2 http://security.ubuntu.com/ubuntu jammy-security InRelease
Hit:3 http://pk.archive.ubuntu.com/ubuntu jammy InRelease
Hit:4 http://pk.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:5 http://pk.archive.ubuntu.com/ubuntu jammy-backports InRelease
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
7 packages can be upgraded. Run 'apt list --upgradable' to see them.
yasir@yasir-virtual-machine:~$ sudo apt install linux-generic
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
linux-generic is already the newest version (5.15.0.60.58).
0 upgraded, 0 newly installed, 0 to remove and 7 not upgraded.
yasir@yasir-virtual-machine:~$
```

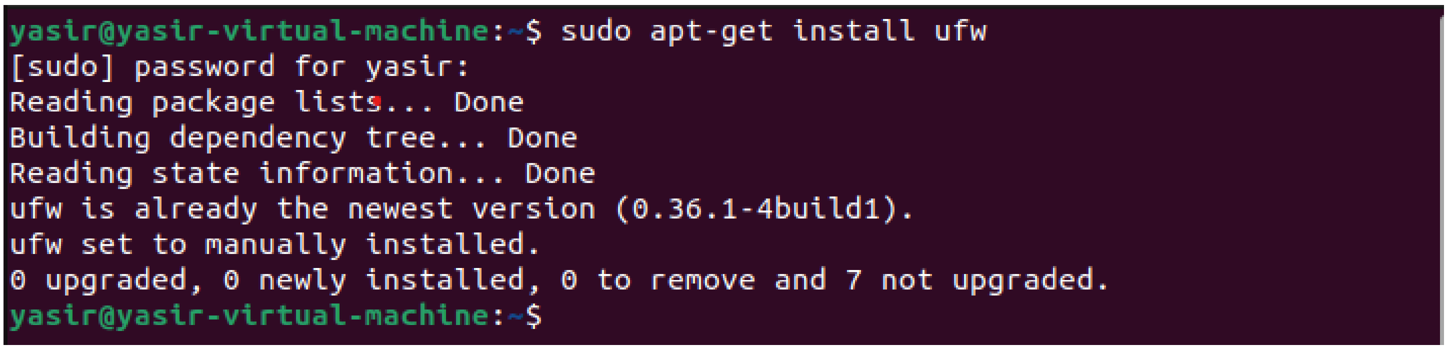
By following these steps, you can keep your Linux system up-to-date with the latest security patches and updates, which can help to prevent security vulnerabilities and keep your system secure.

3. Configuring and using a firewall to control network traffic

Choose a firewall: There are several firewalls available for Linux systems, including iptables, ufw, and firewalld. Each firewall has its own syntax and configuration options, so choose the one that best fits your needs and skill level.

ufw: This is a simpler firewall that is based on iptables but uses a more user-friendly syntax. It is included by default in Ubuntu and can be managed using the ufw command. You can install ufw on other Linux distributions using the following command:

sudo apt-get install ufw



```
yasir@yasir-virtual-machine:~$ sudo apt-get install ufw
[sudo] password for yasir:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
ufw is already the newest version (0.36.1-4build1).
ufw set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 7 not upgraded.
yasir@yasir-virtual-machine:~$
```

firewalld: This is a dynamic firewall that is designed to work well with modern network configurations, such as those used in cloud environments. It uses a command-line syntax but also provides graphical interfaces **for** management. You can install firewalld on Ubuntu using the following command:

sudo apt-get install firewalld

Downloading and configuring nftables: Verify that the nftables package is installed on your system by running the following command:

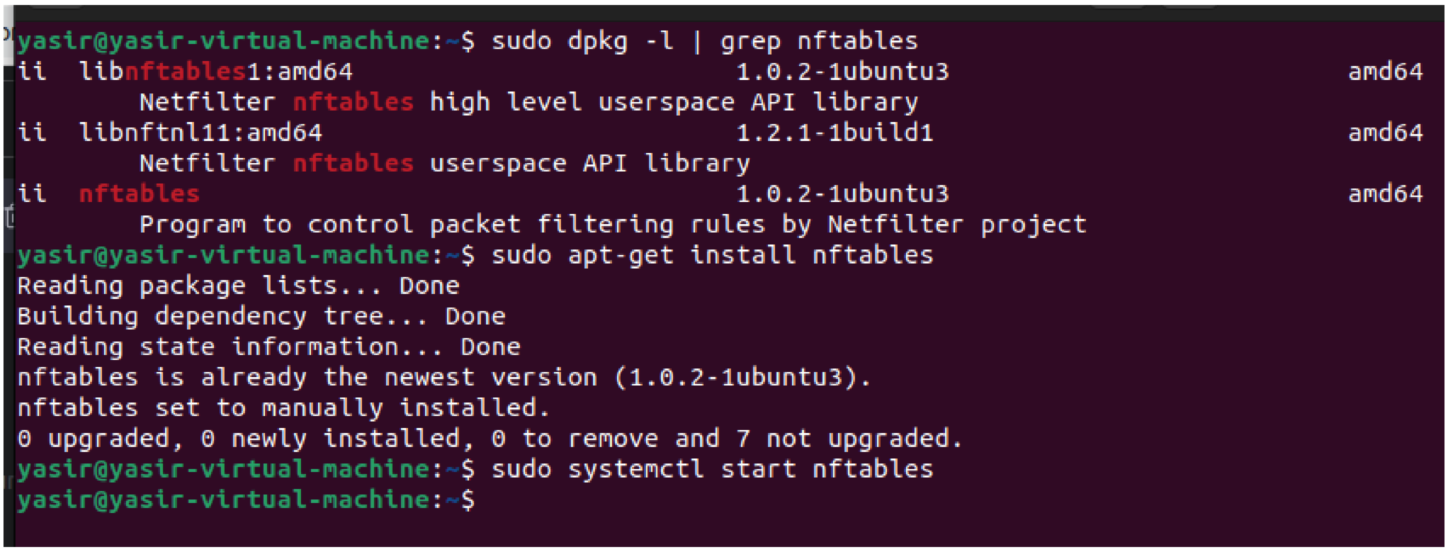
sudo dpkg -l | grep nftables

To install you can use the command

sudo apt-get install nftables

To start iptables you can use the command

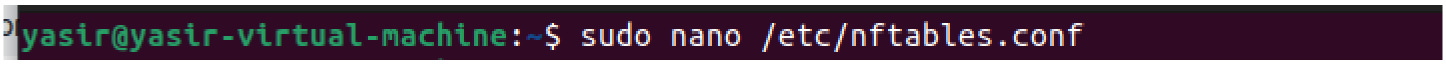
sudo systemctl start nftables



```
yasir@yasir-virtual-machine:~$ sudo dpkg -l | grep nftables
ii  libnftables1:amd64                          1.0.2-1ubuntu3                          amd64
        Netfilter nftables high level userspace API library
ii  libnftnl11:amd64                            1.2.1-1build1                           amd64
        Netfilter nftables userspace API library
ii  nftables                                    1.0.2-1ubuntu3                          amd64
        Program to control packet filtering rules by Netfilter project
yasir@yasir-virtual-machine:~$ sudo apt-get install nftables
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
nftables is already the newest version (1.0.2-1ubuntu3).
nftables set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 7 not upgraded.
yasir@yasir-virtual-machine:~$ sudo systemctl start nftables
yasir@yasir-virtual-machine:~$
```

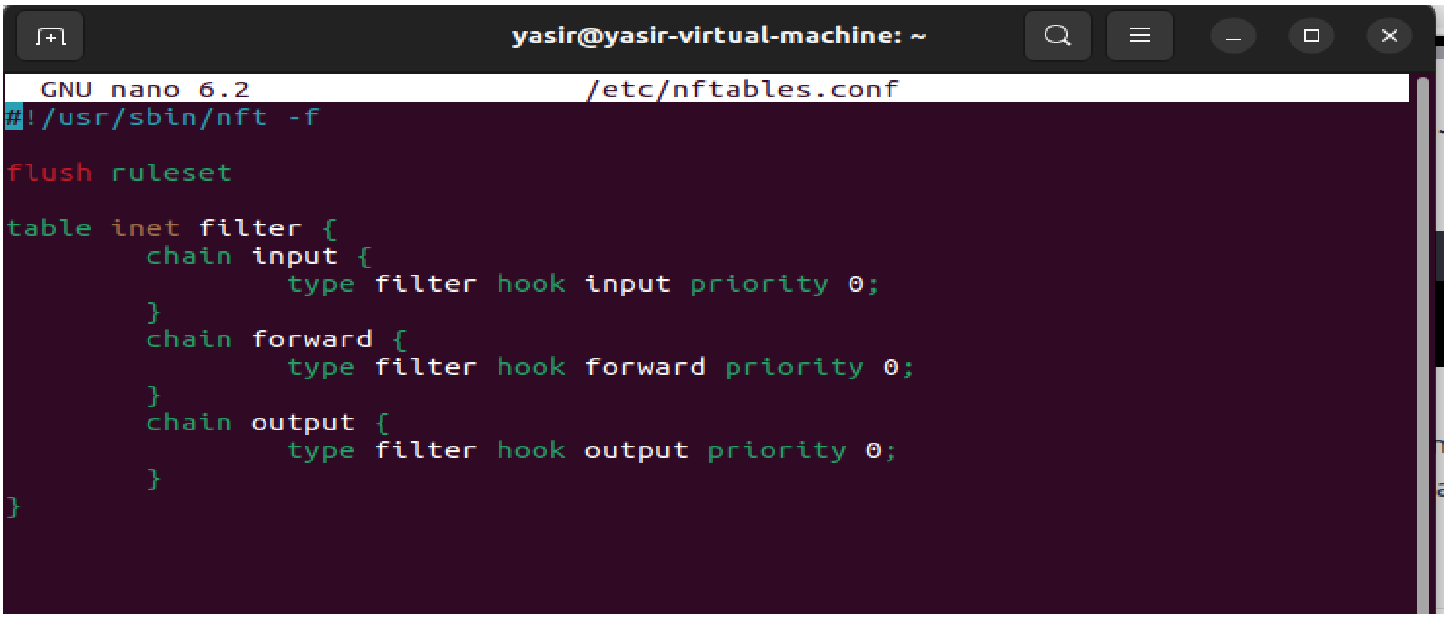
Create a new **nftables** configuration file in **/etc/nftables.conf** using your preferred text editor:

sudo nano /etc/nftables.conf



```
yasir@yasir-virtual-machine:~$ sudo nano /etc/nftables.conf
```

Define your firewall rules in the nftables configuration file. For example, the following rules allow incoming SSH connections and block all other incoming traffic:



```
yasir@yasir-virtual-machine: ~
  GNU nano 6.2                     /etc/nftables.conf
#!/usr/sbin/nft -f

flush ruleset

table inet filter {
        chain input {
                type filter hook input priority 0;
        }
        chain forward {
                type filter hook forward priority 0;
        }
        chain output {
                type filter hook output priority 0;
        }
}
```

This is a basic nftables firewall ruleset that clears any existing rules and sets up default chains for input, forward, and output traffic.

The first line specifies the path to the nft executable and tells it to load the ruleset from the file (-f option). The "flush ruleset" command clears any existing rules in the filter table.

The "table inet filter" line creates a new filter table in the IPv4 address family.

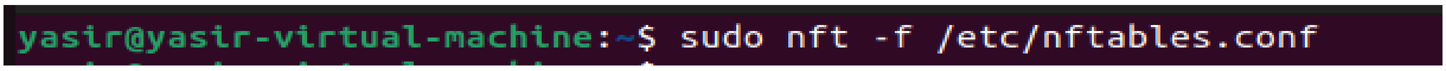
The next three lines create chains for input, forward, and output traffic, respectively. Each chain has a type of "filter" and a hook of "input", "forward", or "output", indicating where in the networking stack the chain will be processed.

The "priority 0" option sets the priority of the chain to zero, which means it will be processed before any other chains with a higher priority.

At this point, the ruleset does not have any rules for allowing or blocking traffic. These rules would need to be added to the appropriate chains in order to control network traffic.

We can Save the nftables configuration file and apply the new rules:

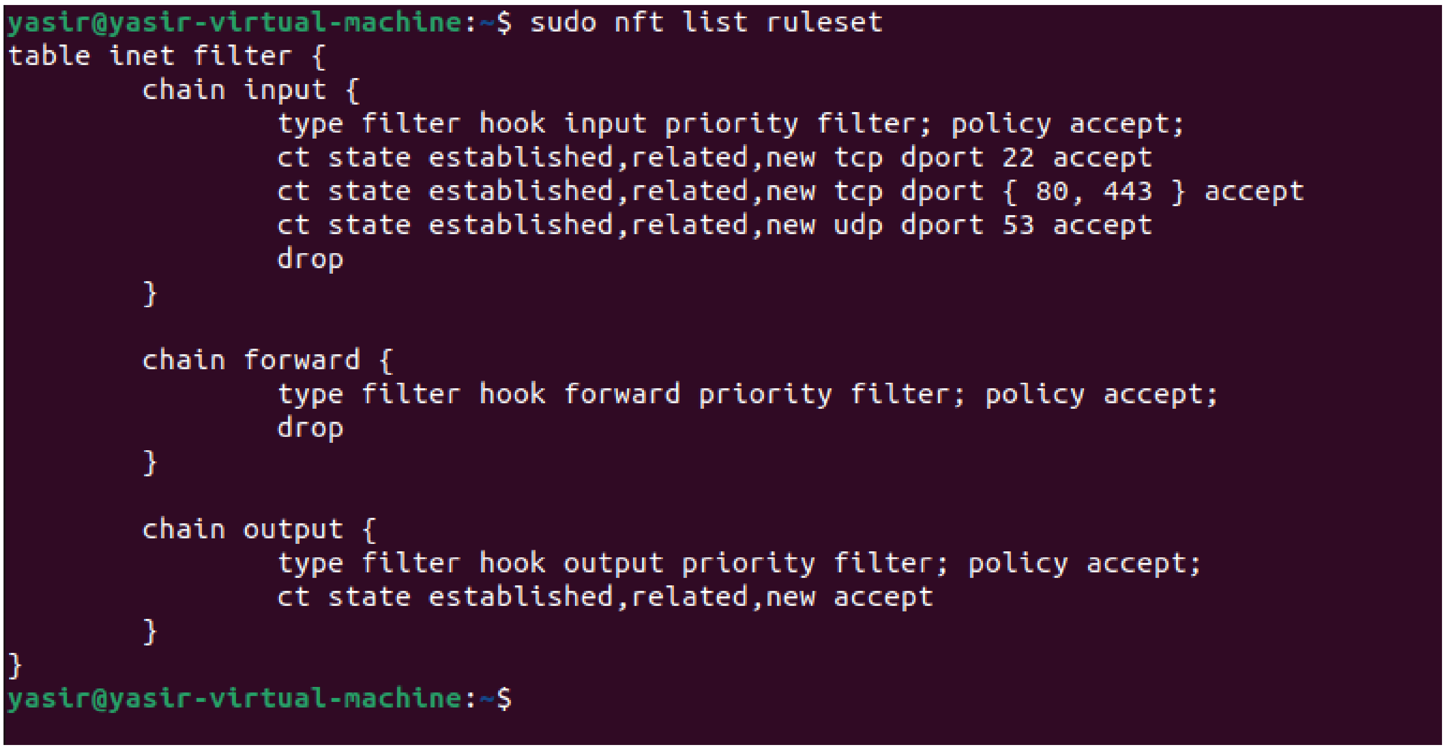
sudo nft -f /etc/nftables.conf



```
yasir@yasir-virtual-machine:~$ sudo nft -f /etc/nftables.conf
```

For Check that the rules have been applied correctly:

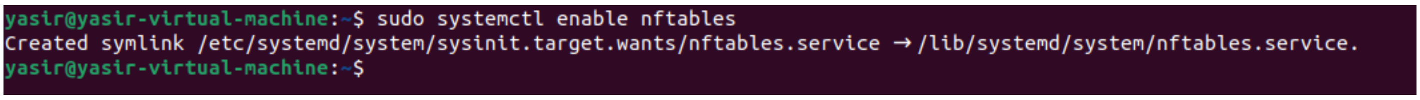
sudo nft list ruleset



```
yasir@yasir-virtual-machine:~$ sudo nft list ruleset
table inet filter {
        chain input {
                type filter hook input priority filter; policy accept;
                ct state established,related,new tcp dport 22 accept
                ct state established,related,new tcp dport { 80, 443 } accept
                ct state established,related,new udp dport 53 accept
                drop
        }

        chain forward {
                type filter hook forward priority filter; policy accept;
                drop
        }

        chain output {
                type filter hook output priority filter; policy accept;
                ct state established,related,new accept
        }
}
yasir@yasir-virtual-machine:~$
```

set nftables to start automatically at boot:

sudo systemctl enable nftables



```
yasir@yasir-virtual-machine:~$ sudo systemctl enable nftables
Created symlink /etc/systemd/system/sysinit.target.wants/nftables.service → /lib/systemd/system/nftables.service.
yasir@yasir-virtual-machine:~$
```

That's it! You now have a basic firewall configured using nftables. You can add additional rules to the nftables configuration file as needed to further customize your firewall. Note that nftables uses a different syntax than iptables, so you may need to refer to the nftables documentation to learn how to create more complex rules

4. Implementing strong password

Implementing strong password policies is an important aspect of Linux hardening. This involves setting guidelines and requirements for passwords to ensure that they are difficult to guess or crack. Strong password policies typically include the following elements:

1. Length: Passwords should be a minimum of 8 characters in length. Longer passwords are generally more secure.

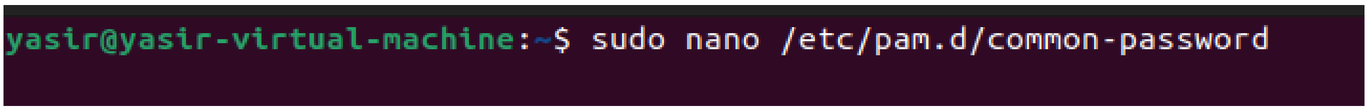
2. Complexity: Passwords should include a mix of uppercase and lowercase letters, numbers, and symbols. This makes it more difficult for attackers to guess or crack passwords using automated tools.
3. Age: Passwords should be changed periodically (for example, every 90 days) to reduce the risk of compromise due to long-term exposure.
4. History: Passwords should be unique and not repeated. Users should not be allowed to use any of their last few passwords as their new password.

   To implement strong password policies on a Linux system, follow these steps:

Set minimum password length and complexity requirements in the system's password policy file. This file is usually located at **/etc/pam.d/common-password**.

To open and configure

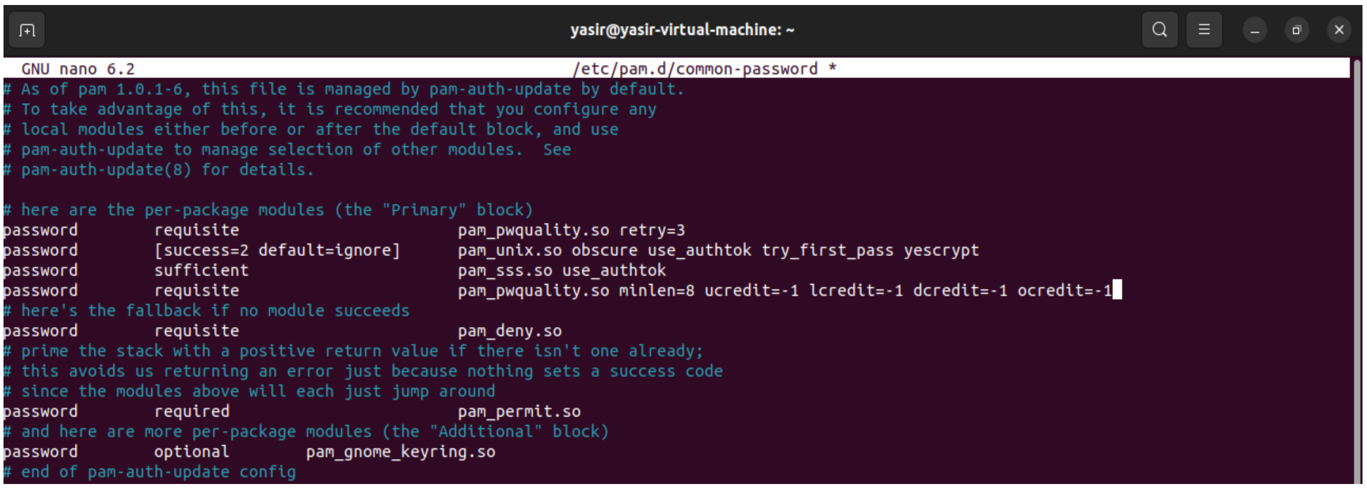
```
sudo nano etc/pam.d/common-password
```



```
yasir@yasir-virtual-machine:~$ sudo nano /etc/pam.d/common-password
```

For example, to require a minimum password length of 8 characters and at least one uppercase letter, lowercase letter, number, and symbol, add the following line to the file:

```
password        requisite                       pam_pwquality.so minlen=8 ucredit=-1 lcredit=-1 dcredit=-1 ocredit=-1
```

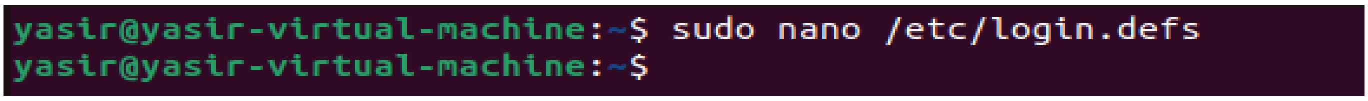


```
  GNU nano 6.2                                      /etc/pam.d/common-password *
# As of pam 1.0.1-6, this file is managed by pam-auth-update by default.
# To take advantage of this, it is recommended that you configure any
# local modules either before or after the default block, and use
# pam-auth-update to manage selection of other modules.  See
# pam-auth-update(8) for details.

# here are the per-package modules (the "Primary" block)
password        requisite                       pam_pwquality.so retry=3
password        [success=2 default=ignore]      pam_unix.so obscure use_authtok try_first_pass yescrypt
password        sufficient                      pam_sss.so use_authtok
password        requisite                       pam_pwquality.so minlen=8 ucredit=-1 lcredit=-1 dcredit=-1 ocredit=-1
# here's the fallback if no module succeeds
password        requisite                       pam_deny.so
# prime the stack with a positive return value if there isn't one already;
# this avoids us returning an error just because nothing sets a success code
# since the modules above will each just jump around
password        required                        pam_permit.so
# and here are more per-package modules (the "Additional" block)
password        optional        pam_gnome_keyring.so
# end of pam-auth-update config
```

Set password age and history requirements in the system's password policy file. This file is usually located at **/etc/login.defs.**

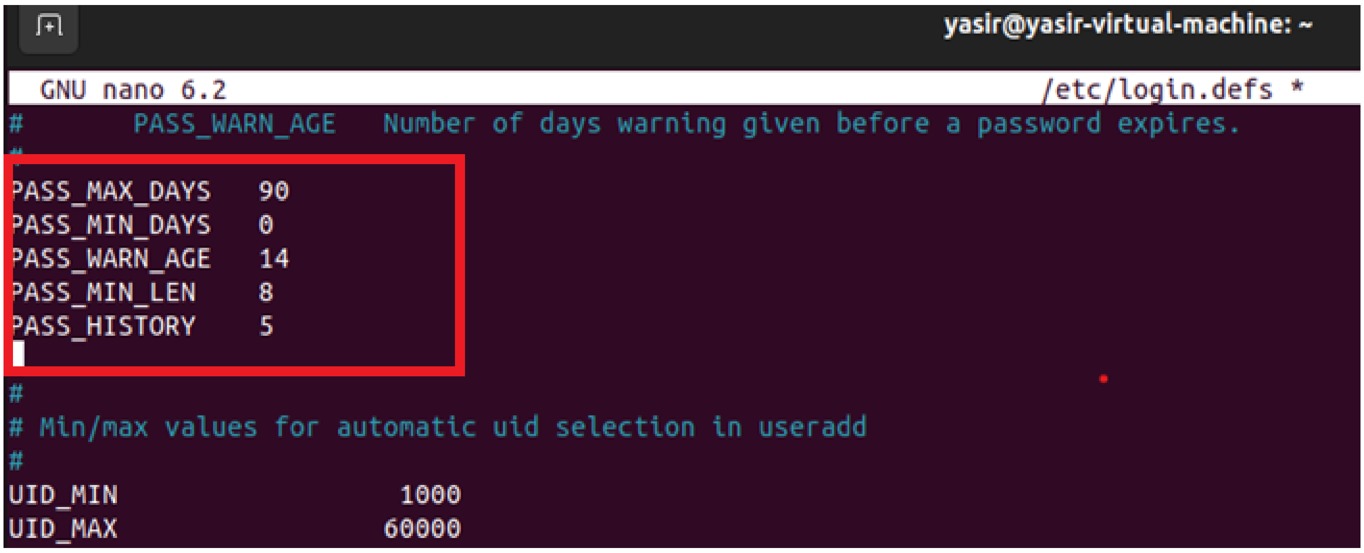
To open and configure

sudo nano /etc/login.defs.



```
yasir@yasir-virtual-machine:~$ sudo nano /etc/login.defs
yasir@yasir-virtual-machine:~$
```

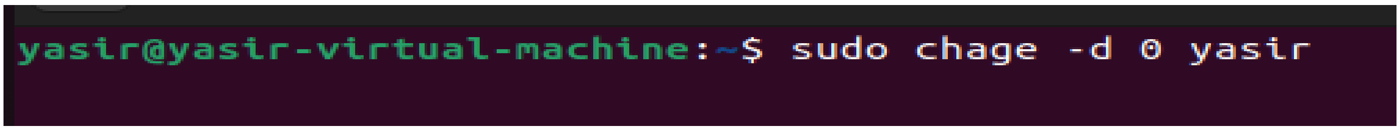
For example, to require that passwords be changed every 90 days and that users cannot use any of their last 5 passwords as their new password, add the following lines to the file:



```
yasir@yasir-virtual-machine: ~
  GNU nano 6.2                                             /etc/login.defs *
#       PASS_WARN_AGE   Number of days warning given before a password expires.
#
PASS_MAX_DAYS   90
PASS_MIN_DAYS   0
PASS_WARN_AGE   14
PASS_MIN_LEN    8
PASS_HISTORY    5

#
# Min/max values for automatic uid selection in useradd
#
UID_MIN                  1000
UID_MAX                 60000
```

Require users to change their password at the next login. This can be done using the chage command:

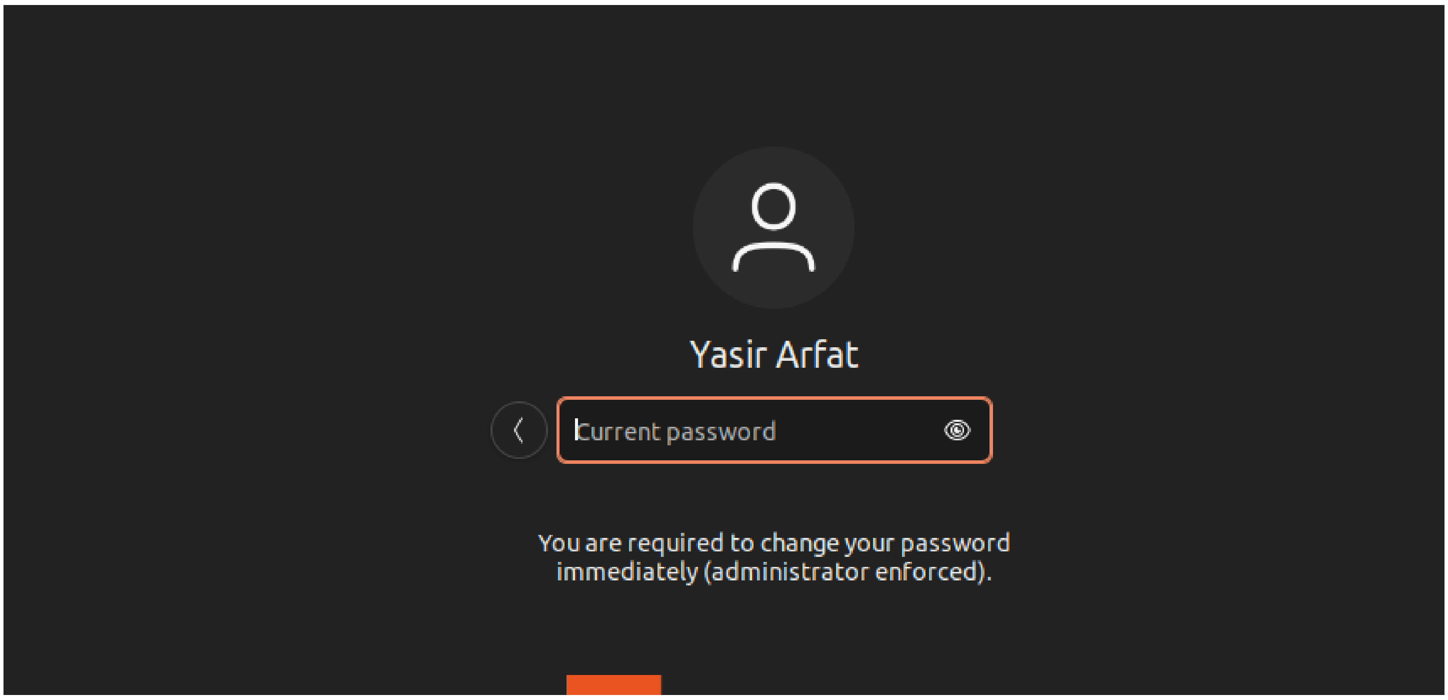
sudo chage -d 0 username



```
yasir@yasir-virtual-machine:~$ sudo chage -d 0 yasir
```

This command sets the date of the last password change to 0, which forces the user to change their password at the next login.

By implementing strong password policies, you can significantly improve the security of your Linux system and reduce the risk of unauthorized access or data breaches.



You are required to change your password immediately (administrator enforced).

5. Configuring file system permissions to restrict access to sensitive files and directories

Configuring file system permissions is an important part of securing a Linux system. It involves setting permissions on files and directories to control who can access them and what actions they can perform. Here are some steps you can follow to configure file system permissions to restrict access to sensitive files and directories

Identify sensitive files and directories: First, identify the files and directories that contain sensitive information or perform critical system functions. These might include configuration files, log files, system binaries, and user data.

Set ownership and group permissions: Next, set ownership and group permissions on these files and directories using the chown and chgrp commands. This ensures that only authorized users can access them. For example, to change the ownership of a file named sensitive.txt to user bob, you would run the command:

```
chown username sensitive.txt
```

To change the group ownership to a group called sensitive-group, you would run:

```
chgrp sensitive-group sensitive.txt
```

Set file permissions: Once you have set ownership and group permissions, you can set file permissions using the chmod command. The chmod command allows you to set read, write, and execute permissions for the file owner, group members, and other users. For example, to give the owner of a file read and write permissions, you would run the command

```
chmod u+rw file.txt
```

To give group members read permissions and others no permissions, you would run:

```
chmod g+r file.txt
```

```
chmod o-rwx file.txt
```

Use access control lists (ACLs): If you need to set more complex permissions, you can use access control lists (ACLs). ACLs allow you to set permissions for specific users or groups on a file or directory. You can set ACLs using the setfacl command. For example, to give user alice read and write permissions on a file named sensitive.txt, you would run:

```
setfacl -m u:alice:rw sensitive.txt
```

Regularly review and update permissions: Finally, it is important to regularly review and update file permissions to ensure that they are still appropriate. As your system changes and new users are added, you may need to adjust permissions to maintain security.

6. Implementing user management best practices, such as limiting user privileges and monitoring user activity

Implementing user management best practices is an important aspect of securing a Linux system. Here are some commands that can be used to implement these practices

Limit user privileges: Grant users the minimum level of privileges necessary to perform their job functions. Avoid granting administrative privileges unless necessary, and consider implementing role-based access control to limit user access to only the systems and resources they need to perform their job.

Implement strong password policies: Enforce strong password policies, such as requiring complex passwords, regular password changes, and password length requirements.

Monitor user activity: Monitor user activity to detect and respond to potential security incidents. Implement tools such as intrusion detection systems (IDS) and security information and event management (SIEM) systems to monitor user activity and detect anomalies or suspicious behavior.

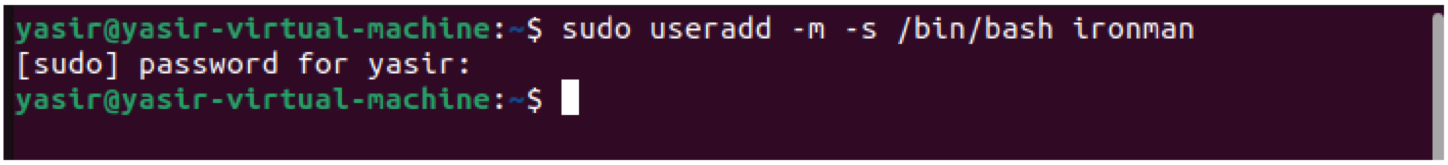
Conduct regular security training: Educate users on security best practices and provide training on how to identify and respond to security incidents.

Review and audit user accounts: Regularly review user accounts to ensure that inactive accounts are disabled and that access privileges are appropriate. Conduct periodic security audits to identify potential vulnerabilities and implement corrective actions.

Limiting user privileges:

Create a new user with limited privileges using the **useradd** command:

sudo useradd -m -s /bin/bash ironman



```
yasir@yasir-virtual-machine:~$ sudo useradd -m -s /bin/bash ironman
[sudo] password for yasir:
yasir@yasir-virtual-machine:~$ 
```

Add the user to a group with the required permissions:

sudo usermod -aG <group_name> john

Remove unnecessary permissions from a user:

sudo deluser <username> <group_name>

Limit user access to specific commands using the sudoers file:

sudo visudo

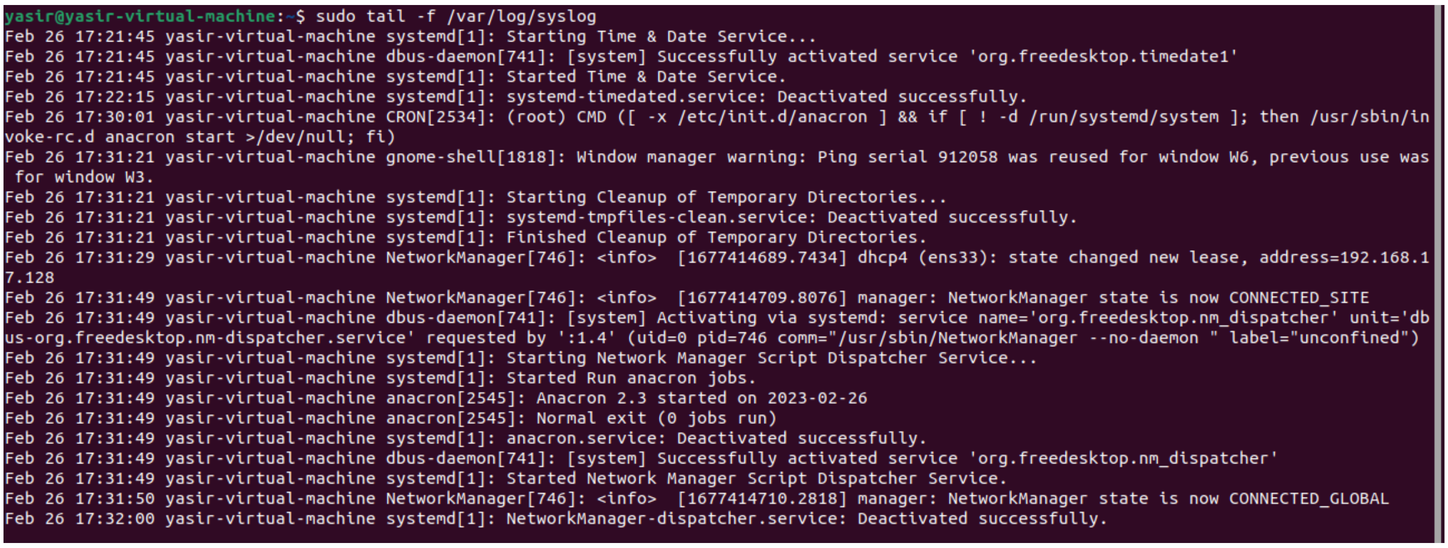
Add the following line to the file to grant a user access to a specific command:

ironman ALL=(ALL) /bin/ls

This will allow the user ironman to run the ls command using sudo

Monitoring user activity:

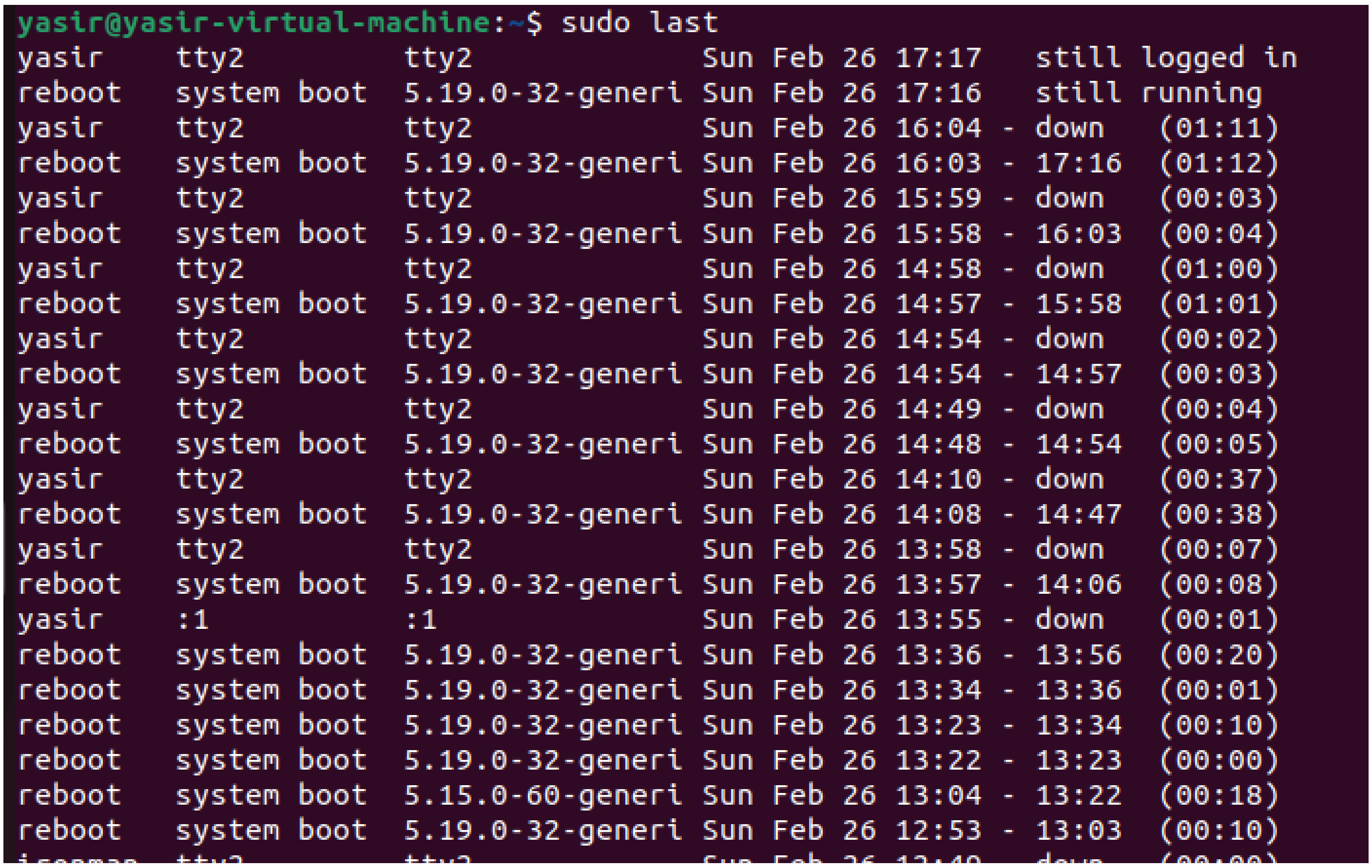
View the system log files to check for user activity sudo tail -f /var/log/syslog



```
yasir@yasir-virtual-machine:~$ sudo tail -f /var/log/syslog
Feb 26 17:21:45 yasir-virtual-machine systemd[1]: Starting Time & Date Service...
Feb 26 17:21:45 yasir-virtual-machine dbus-daemon[741]: [system] Successfully activated service 'org.freedesktop.timedate1'
Feb 26 17:21:45 yasir-virtual-machine systemd[1]: Started Time & Date Service.
Feb 26 17:22:15 yasir-virtual-machine systemd[1]: systemd-timedated.service: Deactivated successfully.
Feb 26 17:30:01 yasir-virtual-machine CRON[2534]: (root) CMD ([ -x /etc/init.d/anacron ] && if [ ! -d /run/systemd/system ]; then /usr/sbin/in
voke-rc.d anacron start >/dev/null; fi)
Feb 26 17:31:21 yasir-virtual-machine gnome-shell[1818]: Window manager warning: Ping serial 912058 was reused for window W6, previous use was
 for window W3.
Feb 26 17:31:21 yasir-virtual-machine systemd[1]: Starting Cleanup of Temporary Directories...
Feb 26 17:31:21 yasir-virtual-machine systemd[1]: systemd-tmpfiles-clean.service: Deactivated successfully.
Feb 26 17:31:21 yasir-virtual-machine systemd[1]: Finished Cleanup of Temporary Directories.
Feb 26 17:31:29 yasir-virtual-machine NetworkManager[746]: <info>  [1677414689.7434] dhcp4 (ens33): state changed new lease, address=192.168.1
7.128
Feb 26 17:31:49 yasir-virtual-machine NetworkManager[746]: <info>  [1677414709.8076] manager: NetworkManager state is now CONNECTED_SITE
Feb 26 17:31:49 yasir-virtual-machine dbus-daemon[741]: [system] Activating via systemd: service name='org.freedesktop.nm_dispatcher' unit='db
us-org.freedesktop.nm-dispatcher.service' requested by ':1.4' (uid=0 pid=746 comm="/usr/sbin/NetworkManager --no-daemon " label="unconfined")
Feb 26 17:31:49 yasir-virtual-machine systemd[1]: Starting Network Manager Script Dispatcher Service...
Feb 26 17:31:49 yasir-virtual-machine systemd[1]: Started Run anacron jobs.
Feb 26 17:31:49 yasir-virtual-machine anacron[2545]: Anacron 2.3 started on 2023-02-26
Feb 26 17:31:49 yasir-virtual-machine anacron[2545]: Normal exit (0 jobs run)
Feb 26 17:31:49 yasir-virtual-machine systemd[1]: anacron.service: Deactivated successfully.
Feb 26 17:31:49 yasir-virtual-machine dbus-daemon[741]: [system] Successfully activated service 'org.freedesktop.nm_dispatcher'
Feb 26 17:31:49 yasir-virtual-machine systemd[1]: Started Network Manager Script Dispatcher Service.
Feb 26 17:31:50 yasir-virtual-machine NetworkManager[746]: <info>  [1677414710.2818] manager: NetworkManager state is now CONNECTED_GLOBAL
Feb 26 17:32:00 yasir-virtual-machine systemd[1]: NetworkManager-dispatcher.service: Deactivated successfully.
```

Use the last command to view a list of recently logged in users:

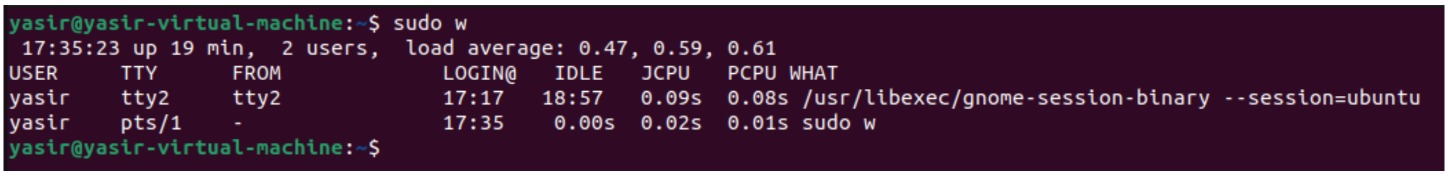
sudo last



```
yasir@yasir-virtual-machine:~$ sudo last
yasir    tty2         tty2             Sun Feb 26 17:17   still logged in
reboot   system boot  5.19.0-32-generi Sun Feb 26 17:16   still running
yasir    tty2         tty2             Sun Feb 26 16:04 - down   (01:11)
reboot   system boot  5.19.0-32-generi Sun Feb 26 16:03 - 17:16  (01:12)
yasir    tty2         tty2             Sun Feb 26 15:59 - down   (00:03)
reboot   system boot  5.19.0-32-generi Sun Feb 26 15:58 - 16:03  (00:04)
yasir    tty2         tty2             Sun Feb 26 14:58 - down   (01:00)
reboot   system boot  5.19.0-32-generi Sun Feb 26 14:57 - 15:58  (01:01)
yasir    tty2         tty2             Sun Feb 26 14:54 - down   (00:02)
reboot   system boot  5.19.0-32-generi Sun Feb 26 14:54 - 14:57  (00:03)
yasir    tty2         tty2             Sun Feb 26 14:49 - down   (00:04)
reboot   system boot  5.19.0-32-generi Sun Feb 26 14:48 - 14:54  (00:05)
yasir    tty2         tty2             Sun Feb 26 14:10 - down   (00:37)
reboot   system boot  5.19.0-32-generi Sun Feb 26 14:08 - 14:47  (00:38)
yasir    tty2         tty2             Sun Feb 26 13:58 - down   (00:07)
reboot   system boot  5.19.0-32-generi Sun Feb 26 13:57 - 14:06  (00:08)
yasir    :1           :1               Sun Feb 26 13:55 - down   (00:01)
reboot   system boot  5.19.0-32-generi Sun Feb 26 13:36 - 13:56  (00:20)
reboot   system boot  5.19.0-32-generi Sun Feb 26 13:34 - 13:36  (00:01)
reboot   system boot  5.19.0-32-generi Sun Feb 26 13:23 - 13:34  (00:10)
reboot   system boot  5.19.0-32-generi Sun Feb 26 13:22 - 13:23  (00:00)
reboot   system boot  5.15.0-60-generi Sun Feb 26 13:04 - 13:22  (00:18)
reboot   system boot  5.19.0-32-generi Sun Feb 26 12:53 - 13:03  (00:10)
```

Use the w command to view the current users logged in and their activity

sudo w

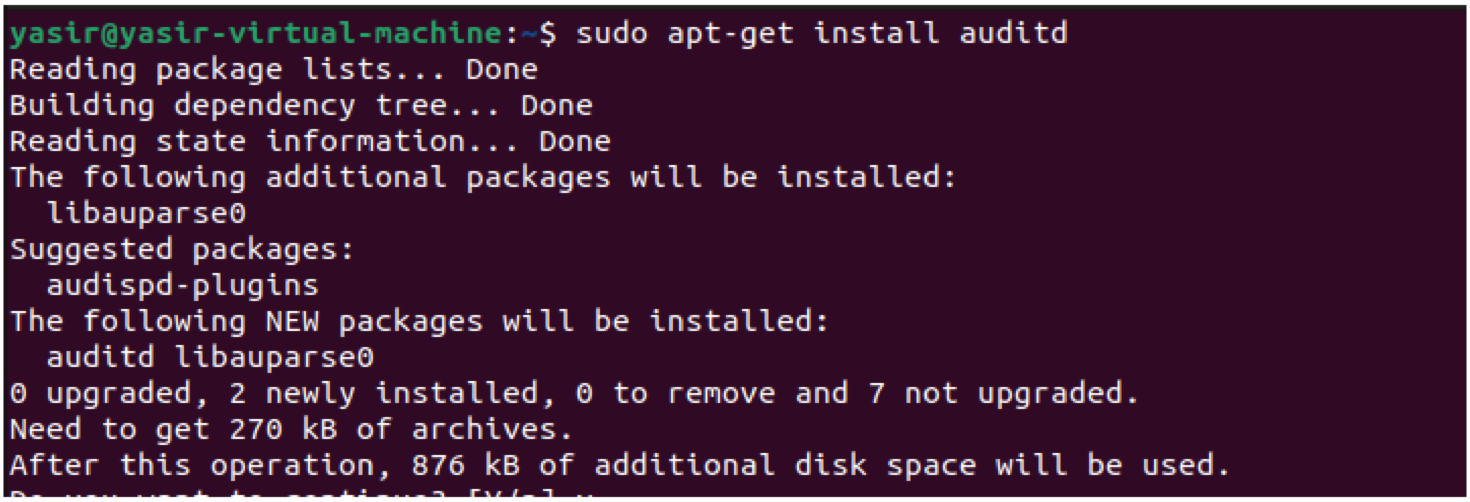


```
yasir@yasir-virtual-machine:~$ sudo w
 17:35:23 up 19 min,  2 users,  load average: 0.47, 0.59, 0.61
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
yasir    tty2     tty2             17:17   18:57   0.09s  0.08s /usr/libexec/gnome-session-binary --session=ubuntu
yasir    pts/1    -                17:35    0.00s  0.02s  0.01s sudo w
yasir@yasir-virtual-machine:~$
```

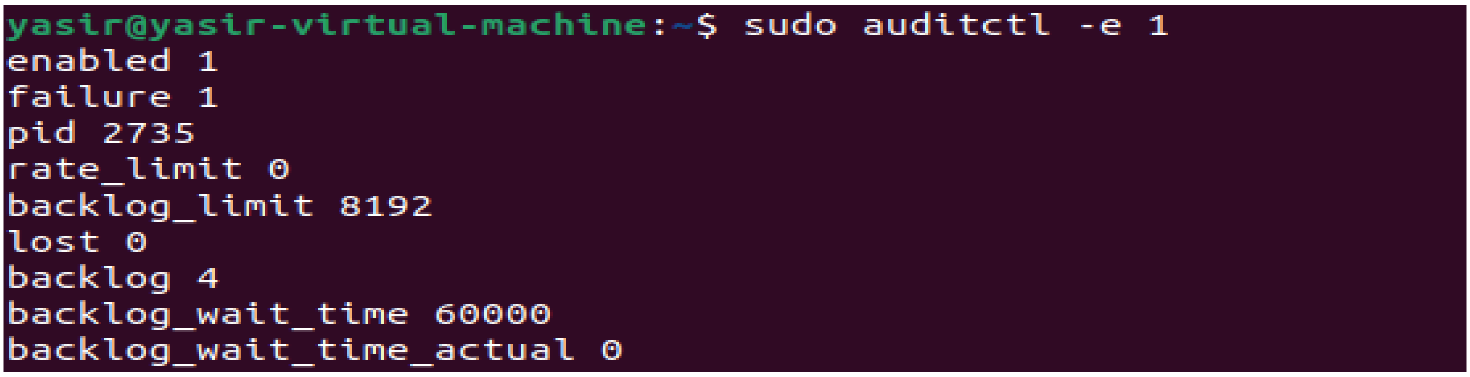
Install and configure auditd to monitor system activity:

sudo apt-get install auditd

sudo auditctl -e 1



```
yasir@yasir-virtual-machine:~$ sudo apt-get install auditd
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libauparse0
Suggested packages:
  audispd-plugins
The following NEW packages will be installed:
  auditd libauparse0
0 upgraded, 2 newly installed, 0 to remove and 7 not upgraded.
Need to get 270 kB of archives.
After this operation, 876 kB of additional disk space will be used.
```



```
yasir@yasir-virtual-machine:~$ sudo auditctl -e 1
enabled 1
failure 1
pid 2735
rate_limit 0
backlog_limit 8192
lost 0
backlog 4
backlog_wait_time 60000
backlog_wait_time_actual 0
```

7. Linux Hardening Tools

Hardening tools are software programs that automate this process, making it easier for administrators to identify and eliminate security risks. Hardening tools are essential for maintaining the security of computer systems, as they help protect against threats such as malware, ransomware, and unauthorized access. By automating security best practices and performing regular security checks, hardening tools help ensure that a system is properly configured and protected from known security threats

Here are some popular Linux hardening tools:

JShielder: A Linux hardening tool that provides an interactive menu-driven interface for configuring security options such as disabling unnecessary services, securing the network stack, and identifying system vulnerabilities

Lynis: A security scanner and compliance auditing tool for Linux systems that performs various security checks and provides a detailed report on the system's security status

OpenSCAP: A suite of tools that provides a standard for automating the assessment of security configurations on Linux systems, including system hardening, vulnerability scanning, compliance checking, and security data.

Zeus: An auditing and hardening tool for AWS environments that provides a range of security checks and automated remediation features to maintain a secure environment.

nixarmor: A Linux hardening script that provides an interactive interface for configuring security options and includes security checks such as detecting rootkits and known vulnerabilities.

These tools can be used to improve the security and harden a Linux system by checking for vulnerabilities, monitoring system activity, enforcing access control policies, and detecting and preventing malicious activity.

Conclusion

In conclusion, This lab provided valuable insights and practical guidance on enhancing the security of a Linux system. Through a variety of topics, including disabling unnecessary services, using firewalls, implementing strong passwords, and configuring file system permissions, the lab demonstrated how to reduce security risks and protect against potential threats. The Linux hardening tools covered in the lab, including JShielder, Lynis, OpenSCAP, Zeus, and nixarmor, offer valuable solutions to improve security and maintain compliance. Finally, the use of a Squid Proxy Server can enhance network security and performance by filtering out malicious traffic and caching frequently accessed web pages. By following the lab's best practices and tools, administrators can minimize security risks and maintain the integrity of their Linux systems.