Linux

Assignment-2

1. In Linux FHS (Filesystem Hierarchy Standard) what is the /?

In the Linux FHS (Filesystem Hierarchy Standard), the / directory is the root directory, which is the starting point of the directory tree.

2. What is stored in each of the following paths?
/bin, /sbin, /usr/bin and /usr/sbin
/etc
/home
/var
/tmp

/bin: This directory contains executable programs that are essential for booting the system and running basic system commands. Examples include ls, cp, and mv.
/sbin: This directory contains executable programs that are essential for system administration, such as disk partitioning and system recovery. Examples include fdisk and iptables.
/usr/bin: This directory contains non-essential executable programs that are not required for system administration or booting the system. Examples include games and productivity software.
/usr/sbin: This directory contains non-essential executable programs that are not required for system administration, but may be necessary for certain system services. Examples include httpd and sendmail.
/etc: This directory contains system configuration files. Examples include passwd and fstab.
/home: This directory contains the home directories of regular users.
/var: This directory contains variable data files, such as logs and spool files.
/tmp: This directory is used to store temporary files. It is typically emptied on reboot, and may be stored in memory for faster access.

3. What is special about the /tmp directory when compared to other directories?

The important difference is that the /tmp directory is often stored in memory (i.e., RAM) rather than on disk, which allows for faster access to the temporary files. This can be especially important for applications that need to access temporary files frequently, such as compilers or editors.

4. What kind of information one can find in /proc?

The /proc directory is a special filesystem that contains information about running processes and system configuration. One can find various system and process related information such as memory usage, CPU usage, system uptime, kernel version, loaded modules, etc.

5. What makes /proc different from other filesystems?

The /proc filesystem is different from other filesystems because it is not a physical filesystem, but rather a virtual filesystem that is generated on the fly by the kernel. It provides an interface to kernel data structures that can be used to extract information about the system and its processes.

6. True or False? only root can create files in /proc

It is true that only root can create files in the /proc directory.

7. What can be found in /proc/cmdline?

The /proc/cmdline file contains the command line parameters that were passed to the kernel when it was booted. This includes information such as the kernel version, the root device, and any kernel parameters that were specified at boot time.

8. In which path can you find the system devices (e.g. block storage)?

System devices, such as block storage devices, can typically be found in the /dev directory.

Permissions

9. How to change the permissions of a file?

To change the permissions of a file, you can use the chmod command in a terminal. The syntax is:

```
chmod [permissions] [filename]
```

10. What does the following permissions mean?:
777
644
750

chmod 777 myfile.txt: This gives read, write, and execute permissions to the owner, group, and everyone else who has access to the file.

chmod 644 myfile.txt: This gives read and write permissions to the owner, and read-only permissions to the group and everyone else.

chmod 750 myfile.txt: This gives read, write, and execute permissions to the owner, and read and execute permissions to the group. Everyone else has no access to the file.

11. What this command does? chmod +x some_file

The command chmod +x some_file gives execute permission to the owner of the file some_file. Specifically, the +x option adds the execute permission to the file for the owner.

12. Explain what is setgid and setuid

Setgid and setuid are special permissions in Linux/Unix systems. Setgid allows a user who is a member of a group that owns a file or directory to inherit the group ownership of that file or directory when creating new files or directories within it. Setuid allows a user to execute a program with the permissions of the file's owner, rather than the user's own permissions.

13. What is the purpose of sticky bit?

The purpose of the sticky bit is to restrict the deletion of files in a directory. When the sticky bit is set on a directory, only the owner of a file or the root user can delete the file, even if other users have write permission to the directory. The sticky bit is represented by the "t" bit in the permissions of a directory.

14. What the following commands do?
chmod
chown
Chgrp

chmod is a command in Linux/Unix systems used to change the permissions of files and directories. It can be used to add or remove read, write, and execute permissions for the owner, group, and other users.

chown is a command used to change the ownership of files or directories. It can be used to change the user owner and/or group owner of a file or directory.

chgrp is a command used to change the group ownership of a file or directory. It can be used to change the group owner of a file or directory to a specific group.

15. What is sudo? How do you set it up?

sudo is a command in Linux/Unix systems that allows a user to execute commands as another user, typically the root user, which has administrative privileges. By default, only the root user has the permissions to perform certain administrative tasks, but sudo allows other users to perform those tasks as well.

16. True or False? In order to install packages on the system one must be the root user or use the sudo command

True. In order to install packages on the system, one must have administrative privileges, which can be achieved by either logging in as the root user or using the sudo command to execute the package installation command with administrative privileges.

17. Explain what are ACLs. For what use cases would you recommend to use them?

ACLs (Access Control Lists) are a mechanism in Linux/Unix systems that allow more granular control over file and directory permissions than traditional Unix file permissions. ACLs allow you to set permissions for specific users and groups, rather than just the owner, group, and others as in traditional Unix file permissions. ACLs are useful in scenarios where you need to grant different levels of access to different users or groups for the same file or directory. For example, if you have a file that needs to be accessible by multiple users but you want to restrict some users from modifying it, you can use ACLs to achieve that.

18. You try to create a file but it fails. Name at least three different reason as to why it could happen

- Insufficient permissions: If you don't have the necessary permissions to create files in the target directory, the file creation will fail.
- Insufficient disk space: If the disk is full or there isn't enough space for the new file, the file creation will fail.
- File name or path errors: If the file name or path contains invalid characters or exceeds the maximum length allowed by the filesystem, the file creation will fail.

19. A user accidentally executed the following chmod -x $(which chmod). How to fix it?

If a user accidentally executes the command chmod -x $(which chmod), it will remove the execute permission from the chmod command, which is required to change file permissions. To fix it, you can use the full path to the chmod command, which is /bin/chmod, to restore the execute permission. The command to fix it would be sudo /bin/chmod +x $(which chmod).

Scenarios

20. You would like to copy a file to a remote Linux host. How would you do?

To copy a file to a remote Linux host, you can use the scp (secure copy) command. The syntax is:
scp /path/to/local/file username@remotehost:/path/to/destination
Replace /path/to/local/file with the path to the file you want to copy, username with the username of the remote host, remotehost with the hostname or IP address of the remote host, and /path/to/destination with the destination path on the remote host.

21. How to generate a random string?

To generate a random string in Linux/Unix, you can use the openssl rand command. For example, to generate a random string of 10 characters, you can use the following command:
openssl rand -hex 5

22. How to generate a random string of 7 characters?

To generate a random string of 7 characters, you can modify the command above to generate half the number of characters and then use the head command to truncate the output:
openssl rand -hex 4 | head -c 7

Systemd

23. What is systemd?

systemd is a system and service manager for Linux operating systems. It is responsible for managing the boot process, system services, and other system processes. It replaces the traditional SysV init system and provides many features such as on-demand starting of daemons, parallel startup of system services, and dependency-based service control.

24. How to start or stop a service?

To start or stop a service on a system that uses systemd, you can use the systemctl command. For example, to start the Apache web server service, you can run the command:

sudo systemctl start apache2

To stop the Apache web server service, you can run the command:

sudo systemctl stop apache2

25. How to check the status of a service?

To check the status of a service, you can use the systemctl command with the status option. For example, to check the status of the Apache web server service, you can run the command:

sudo systemctl status apache2

26. On a system which uses systemd, how would you display the logs?

On a system that uses systemd, you can display the logs using the journalctl command. For example, to display the logs for the Apache web server service, you can run the command:

sudo journalctl -u apache2

27. Describe how to make a certain process/app a service

To make a certain process or application a service, you can create a systemd unit file. The unit file contains information about the service, such as the service name, description, startup commands, and dependencies.

28. Troubleshooting and Debugging

Troubleshooting and debugging can be done using various tools and techniques, depending on the specific issue. Some common tools and techniques include:

- Checking system logs using the journalctl command or log files in /var/log/
- Checking service status using the systemctl status command
- Checking network connectivity using the ping or traceroute commands
- Using the strace or ltrace commands to trace system calls and library calls made by a process
- Using the gdb command to debug a process
- Checking system resource usage using the top or htop commands

29. Where system logs are located?

System logs are located in the /var/log/ directory on most Linux systems. Each service or application typically has its own log file, which can be viewed using the appropriate log viewer or by using the journalctl command. Common log files include /var/log/messages, /var/log/syslog, and /var/log/auth.log.

30. How to follow file's content as it being appended without opening the file every time?

To follow a file's content as it is being appended without opening the file every time, you can use the "tail" command with the "-f" option. For example, to follow the content of a file named "mylog.txt", you can run the following command:

tail -f mylog.txt

31. What are you using for troubleshooting and debugging network issues?

For troubleshooting and debugging network issues, some common tools include:

- ping: checks network connectivity between two hosts
- traceroute: traces the route taken by packets between two hosts
- netstat: displays network connections and statistics
- tcpdump: captures network traffic for analysis
- Wireshark: a GUI tool for analyzing network traffic

32. What are you using for troubleshooting and debugging disk & file system issues?

For troubleshooting and debugging disk and file system issues, some common tools include:

- df: displays disk space usage and available space
- du: displays disk usage by file or directory
- fsck: checks and repairs file system errors
- lsof: lists open files and processes

33. What are you using for troubleshooting and debugging process issues?

For troubleshooting and debugging process issues, some common tools include:

- ps: displays information about processes
- top: displays real-time information about system processes and resource usage
- strace: traces system calls made by a process
- ltrace: traces library calls made by a process
- gdb: a debugger for C and C++ programs

34. What are you using for debugging CPU related issues?

For debugging CPU-related issues, some common tools include:

- top: displays real-time information about system processes and resource usage
- htop: an improved version of top with additional features
- sar: collects and reports system resource usage
- perf: a profiling tool for Linux systems

35. You get a call from someone claiming "my system is SLOW". What do you do?

When someone claims their system is slow, the first step is to gather more information about the issue. This can be done by asking questions such as:

- What specific tasks are slow?
- Has anything changed recently on the system?
- Are there any error messages or warnings?
- Is the system running out of memory or disk space?
- Are there any network issues?

36. Explain iostat output

iostat is a command-line tool for monitoring system input/output (I/O) statistics. It displays information such as disk utilization, disk I/O rates, and CPU utilization. Here is an example output:

Device	tps	kB_read/s	kB_wrtn/s	kB_dscd/s	%util
sda	9.67	114.33	78.33	0.00	11.11

- tps: the number of I/O operations per second

- kB_read/s: the amount of data read from the disk per second in kilobytes
- kB_wrtn/s: the amount of data written to the disk per second in kilobytes
- kB_dscd/s: the amount of data discarded from the disk cache per second in kilobytes
- %util: the percentage of time the disk was busy handling I/O requests

37. How to debug binaries?

To debug binaries, you can use a debugger such as gdb. This allows you to step through the code, set breakpoints, inspect variables and memory, and more. Here is an example command to run a binary with gdb:
gdb /path/to/binary

38. What is the difference between CPU load and utilization?

CPU load and utilization are related but different metrics. CPU load refers to the number of processes waiting to be executed or waiting for system resources, while CPU utilization refers to the percentage of time the CPU is actively executing processes. A high CPU load can indicate a system

39. How you measure time execution of a program?

Using the time command:
The time command is a built-in command in Linux that can be used to measure the execution time of a program. To use it, simply type the word time followed by the command you want to measure. For example, to measure the time it takes to run the ls command, type:

time ls

Scenarios

40. You have a process writing to a file. You don't know which process exactly, you just know the path of the file. You would like to kill the process as it's no longer needed. How would you achieve it?

To identify the process writing to a file and then kill it, you can use the lsof command which stands for "list open files". Here are the steps to achieve this:

1. Run the lsof command on the file path to get the process ID (PID) of the process writing to the file:

lsof <file_path>

This will display a list of processes that have the file open, along with their PIDs, user IDs, and other information.

2. Identify the process ID (PID) of the process that is writing to the file.

3. Kill the process using the kill command and the process ID (PID) you identified in step 2:

   kill <pid>

This will send a signal to the process asking it to terminate. If the process does not terminate after a certain amount of time, you can use the kill -9 command which sends a stronger signal that will forcibly terminate the process.

Kernel

41. What is a kernel, and what does it do?

The kernel is the core component of the operating system that manages system resources such as CPU, memory, and input/output devices. It acts as an interface between the hardware and the software, providing services to applications and other system components. The kernel is responsible for managing processes, handling system calls, managing memory, and providing security and access control.

42. How do you find out which Kernel version your system is using?

To find out which kernel version your system is using, you can use the uname command with the -r option. Open a terminal and run the following command:

uname -r

43. What is a Linux kernel module and how do you load a new module?

A Linux kernel module is a piece of code that can be loaded and unloaded from the kernel at runtime, without the need to reboot the system. It allows you to add new functionality to the kernel or modify existing functionality. To load a new module, you can use the modprobe command followed by the name of the module. For example:

sudo modprobe <module_name>

44. Explain user space vs. kernel space

User space and kernel space are two distinct areas of memory in the operating system. User space is the memory area where user-level applications run, while kernel space is the memory area where the kernel and device drivers run. User space programs can only access a limited set of system resources, while kernel space programs have access to all system resources.

45. In what phases of kernel lifecycle, can you change its configuration?

In the kernel lifecycle, you can change the configuration during the compile time, the boot time, or the runtime. During the compile time, you can modify the kernel configuration

options before building the kernel. During the boot time, you can pass command-line arguments to the kernel using the bootloader configuration. During the runtime, you can modify kernel configuration options using the sysctl command or by modifying the files in the /proc and /sys file systems.

46. Where can you find kernel's configuration?

The kernel configuration can be found in the **/usr/src/linux** directory, where the kernel source code is stored. The configuration file is named config or .config.

47. Where can you find the file that contains the command passed to the boot loader to run the kernel?

The file that contains the command passed to the boot loader to run the kernel is the bootloader configuration file. The location and name of the file may vary depending on the bootloader used by the system. For example, on a system using GRUB, the configuration file is typically located at **/boot/grub/grub.cfg.**

48. How to list kernel's runtime parameters?

To list the kernel's runtime parameters, you can use the sysctl command with the -a option. Open a terminal and run the following command:

sudo sysctl -a

49. Will running sysctl -a as a regular user vs. root, produce different result?

Running sysctl -a as a regular user vs. root will produce different results. Regular users can only view the kernel parameters that are marked as readable by non-privileged users. Some parameters may require root privileges to be accessed.

50. You would like to enable IPv4 forwarding in the kernel, how would you do it?

To enable IPv4 forwarding in the kernel, you can use the sysctl command to modify the value of the net.ipv4.ip_forward parameter. Open a terminal and run the following command:

sudo sysctl -w net.ipv4.ip_forward=1

This will enable IPv4 forwarding in the kernel. If you want to make this change permanent, you can add the following line to the /etc/sysctl.conf configuration file:

net.ipv4.ip_forward = 1

51. How sysctl applies the changes to kernel's runtime parameters the moment you run sysctl command?

When you run the sysctl command to modify a kernel runtime parameter, the changes are applied immediately to the running kernel of the system. The sysctl command interacts with the proc filesystem to modify the runtime parameters of the kernel.

52. How changes to kernel runtime parameters persist? (applied even after reboot to the system for example)

By default, the changes you make to kernel runtime parameters using sysctl are not persisted across system reboots. To persist the changes, you can add the corresponding sysctl command to a system startup script, such as /etc/rc.local on some systems, or use a configuration management tool to set the values.

53. Are the changes you make to kernel parameters in a container, affects also the kernel parameters of the host on which the container runs?

No, changes you make to kernel parameters inside a container only affect the kernel of that container and not the host system. Containers are isolated from the host system, and each container runs its own instance of the kernel. However, some kernel parameters may affect the behavior of the container or its ability to run, so it's important to understand the interactions between the container and the host kernel.

SSH

54. What is SSH? How to check if a Linux server is running SSH?

SSH (Secure Shell) is a secure network protocol used for remote access and control of computers over a network. To check if a Linux server is running SSH, you can try connecting to it using an SSH client, such as OpenSSH or PuTTY. If the server is running SSH, it should respond and prompt you for authentication.

55. Why SSH is considered better than telnet?

SSH is considered better than Telnet because SSH provides secure encrypted communication between the client and server, while Telnet sends unencrypted data over the network, making it vulnerable to interception and eavesdropping. SSH also supports key-based authentication, providing a more secure and convenient way to authenticate than Telnet's password-based authentication.

56. What is stored in ~/.ssh/known_hosts?

The **~/.ssh/known_hosts** file is used by the SSH client to store public keys of remote hosts that the client has previously connected to. When a client connects to a remote host, the host's public key is checked against the known_hosts file to verify its authenticity.

57. You try to ssh to a server and you get "Host key verification failed". What does it mean?

"Host key verification failed" error message usually means that the public key of the remote host that you are connecting to has changed since the last time you connected, and the SSH client cannot verify the authenticity of the host. This could be due to a legitimate change, such as the host being reinstalled, or it could be a sign of a security breach, such as a man-in-the-middle attack. You can solve this issue by removing the old host key from your ~/.ssh/known_hosts file and adding the new one.

58. What is the difference between SSH and SSL?

SSH and SSL (Secure Sockets Layer) are both encryption protocols used to secure communication over a network. However, SSH is designed for secure remote access and control of computers, while SSL is designed for secure communication between web servers and clients, such as web browsers.

59. What ssh-keygen is used for?

ssh-keygen is a tool used to generate, manage, and convert SSH public and private key pairs for authentication. It is used to create a new key pair or to convert an existing key to a different format. The generated public key is usually placed on the server that you want to connect to, while the private key is kept on your local machine for authentication.

60. What is SSH port forwarding?

SSH port forwarding (also known as SSH tunneling) is a feature of SSH that allows you to create a secure encrypted connection between a local machine and a remote server and forward network traffic through that connection. This is useful for securely accessing remote resources that may not be directly accessible from your local network, such as a database server or a web application. With SSH port forwarding, you can create a tunnel between your local machine and the remote server and forward network traffic through that tunnel, as if the remote resource was directly accessible from your local machine.