Linux Assignment-2

1. **In Linux FHS (Filesystem Hierarchy Standard) what is the /?**
   In Linux FHS (Filesystem Hierarchy Standard), the / (slash) refers to the root directory. It is the top-level directory in the Linux file system hierarchy, and all other directories and files are located under it.

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 essential binary executable files that are needed for system booting and running. Commands available to all users, such as cat, ls, cp, mkdir, etc. are stored here.

   /boot - The boot directory contains the files required for booting the system, including the kernel, bootloader, and initial RAM disk.

   /dev - The dev directory is a special directory in Linux that contains device files which represent the physical and logical devices on the system, including hard drives, USB devices, keyboards, etc.

   /etc - This directory contains system-wide configuration files that are used by various programs and services running on the system. These include network configuration files, user account settings, startup and shutdown scripts, etc.

   /home - The home directory is where user home directories are located by default. Each user has a subdirectory here that contains their own files and settings.

   /lib - The lib directory contains shared libraries that are used by various programs and services running on the system.

   /mnt - This directory is used for mounting external file systems, such as USB drives or network shares.

   /opt - This directory is used for installing third-party software packages that are not provided by the Linux distribution's package manager.

   /tmp - The tmp directory is used for storing temporary files that are created by various programs and services running on the system.

/usr - This directory contains the bulk of the system's non-essential files, including system libraries, documentation, games, etc.

/var - The var directory is used for storing variable data that changes frequently, such as log files, system mail, spool directories, and temporary files used by various programs and services.

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

The /tmp directory is special in several ways when compared to other directories in a Linux file system:

Temporary files: The /tmp directory is specifically designed for storing temporary files. These files are typically created by various programs and services running on the system and are intended to be deleted after they are no longer needed. Therefore the /tmp directory is often cleaned automatically at regular intervals by the system's cleaning mechanisms.

Public write access: By default, the /tmp directory is accessible to all users on the system and allows public write access. This means that any user or program can create, modify, or delete files in the /tmp directory. However, this also makes it a potential security risk, as any user or program can potentially interfere with the files of other users or programs in this directory.

Volatility: The contents of the /tmp directory are typically cleared upon reboot or during system maintenance, unlike other directories such as /home or /var, which store files persistently. This means that any files stored in the /tmp directory are not meant to be used or relied upon for long-term storage, and should be considered volatile.

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

Information about running processes, system hardware, kernel configuration, network and file system information can be found in the /proc directory in Linux

5. What makes /proc different from other filesystems?

The main difference between the /proc directory and other filesystems is that /proc is a virtual filesystem that is dynamically generated by the kernel, whereas other filesystems are backed by physical storage media.

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

False

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

Some common information that can be found in /proc/cmdline includes:

- *The name and location of the kernel image file*
- *The root filesystem location and type*
- *Any additional kernel boot parameters or options*
- *Boot loader options such as timeout, default boot image, etc.*

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

The system devices, including block storage devices such as hard disks and USB drives, can be found in the /dev directory in Linux.

Permissions

9. How to change the permissions of a file?

*In Linux, you can change the permissions of a file using the chmod command. The chmod command stands for "change mode", and it allows you to modify the read, write, and execute permissions for the file owner, group, and other users***.**

10. What does the following permissions mean?

777

644

750

The permissions in Linux are represented by a three-digit number, where each digit corresponds to the permissions for the file owner, group, and other users, respectively. Each digit is a sum of the following values:

4: read permission
2: write permission
1: execute permission
So, the following permissions mean:

777: The file owner, group, and other users have read, write, and execute permissions on the file.
644: The file owner has read and write permissions, and the group and other users have only read permissions on the file.
750: The file owner has read, write, and execute permissions, the group has read and execute permissions, and other users have no permissions on the file.

11.What this command does? chmod +x some_file

The command "chmod +x some_file" is used in Linux/Unix systems to change the permissions of a file named "some_file" and grant it executable permission for the owner of the file. Specifically, it sets the "execute" permission for the owner of the file, allowing them to run the file as a program or script.

12. Explain what is setgid and setuid

setgid, or "set group ID", is a permission that can be set on executable files or directories. When a file or directory has the setgid permission set, any user who runs the file or creates a new file in the directory will have their group ownership set to the same group as the file or directory. This is useful for shared directories where multiple users need to access and modify files.

setuid, or "set user ID", is a permission that can also be set on executable files. When a file has the setuid permission set and is executed by a user, the program runs with the privileges of the owner of the file rather than the user who executed it. This is useful for programs that need to perform tasks that require higher privileges than the user running the program has.

13. What is the purpose of sticky bit?

The sticky bit is a special permission bit that can be set on directories in Unix and Unix-like operating systems like Linux. When the sticky bit is set on a directory, it restricts the ability to delete or rename files within that directory to the owner of the file, the owner of the directory, and the root user.

14. What the following commands do?
chmod
chown
Chgrp

chmod: Changes the permissions of a file or directory.
chown: Changes the owner of a file or directory.
*chgrp: Changes the group owner of a file or directory***.**

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

sudo is a command in Unix and Unix-like operating systems that allows a user to run commands with elevated privileges. It stands for "superuser do".

To set up sudo:

a. First, you need to have administrative privileges on the system.
b. Install sudo package if it is not already installed on the system.
c. Open the /etc/sudoers file using the visudo command.
d. Add a new line to the file that specifies the user or group that you want to grant sudo privileges to, and the commands or types of commands they can run with sudo.
e. Save the file and exit.

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

True.

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

ACL stands for Access Control List. It is a permission mechanism used on Unix and Unix-like operating systems to provide more fine-grained control over file and directory access permissions than the traditional Unix permission model.

While traditional Unix permissions only provide three levels of permission control (read, write, and execute) for three types of users (owner, group, and other), ACLs allow you to set permissions for specific users or groups on a per-file or per-directory basis.

ACLs can be useful in a variety of scenarios, such as:

Shared directories where multiple users need different levels of access to different files or directories.
Complex file permission scenarios where traditional Unix permissions are not sufficient. Providing temporary access to a file or directory to a user who does not belong to the group that owns the file or directory. Giving specific users or groups read or write access to a single file within a directory where the rest of the files are read-only. Overall, ACLs can be a useful tool for providing more granular control over file and directory permissions, but they can also introduce complexity and potential security risks if not used carefully. It is important to understand the nuances of ACLs and how they interact with traditional Unix permissions before using them in production environments.

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

There are many reasons why creating a file might fail. Here are three possible reasons:

a. *Permission issue: You don't have the right permission to create a file in the folder you're trying to create it in. This could be because the folder is protected, or you don't have the right access level.*

b. *Disk space full: There is not enough space left on the device to create the file. This could be because you've reached the maximum capacity of the device, or because you don't have enough free space left.*

c. Invalid filename: The file name you're trying to use contains characters that are not allowed or is too long. Different file systems have different rules for how files can be named.

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

If a user accidentally executed the command "chmod -x $(which chmod)", it means that they have removed the execute permission from the "chmod" command itself, making it impossible to modify file permissions using the "chmod" command.

To fix this issue, you will need to restore the execute permission for the "chmod" command. There are several ways to do this, but one way is:

a. Log in as the root user, or a user with sudo privileges.
b. Find the location of the "chmod" command using the "which chmod" command. This will give you the path to the "chmod" command on your system.
c. Use the "chmod" command with the numerical mode to add the execute permission back to the "chmod" command.
The command will be: chmod +x /path/to/chmod
Replace "/path/to/chmod" with the path you obtained from the "which chmod" command.
After running this command, the "chmod" command should be restored and you should be able to use it to modify file permissions again.

Scenarios

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

a. To copy a file to a remote Linux host, you can use the "scp" command, which stands for "secure copy".

b. Open a terminal on your local machine and navigate to the directory where the file you want to copy is located.

c. the "scp" command to copy the file to the remote host.
The basic syntax of the command is: scp /path/to/local/file username@remote_host:/path/to/remote/directory
Replace "/path/to/local/file" with the path to the file you want to copy on your local machine, "username" with your username on the remote host, "remote_host" with the IP address or hostname of the remote host, and "/path/to/remote/directory" with the directory on the remote host where you want to copy the file.

d. If this is the first time you're connecting to the remote host, you may be prompted to verify the host's key fingerprint. Type "yes" to confirm.

e. If the remote host requires a password for the specified username, you will be prompted to enter it.

f. Once you've entered the password (if required), the file will be copied to the remote host.

21. How to generate a random string?

To generate a random string in Linux, you can use the "openssl" command, which includes a built-in pseudo-random number generator. Here's an example of how to use it to generate a 10-character random string:

openssl rand -base64 15 | tr -dc 'a-zA-Z0-9' | head -c10; echo .

This command generates a random string of 15 characters, then uses the "tr" command to delete any characters that are not letters or digits, and finally uses the "head" command to select the first 10 characters

.

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

To generate a random string of 7 characters in Linux, you can use the "openssl" command and the "head" command to select the first 7 characters.

Here's an example command: openssl rand -hex 5 | head -c 7; echo

This command generates a random string of 5 hexadecimal characters, then uses the "head" command to select the first 7 characters. The "echo" command at the end is used to add a newline character to the output.

Systemd

23. What is systemd?

systemd is a system and service manager for Linux operating systems that handles starting and stopping system services, managing system processes, handling system logging, and providing other core functionalities of the operating system. It is designed to improve system performance and reduce boot time by introducing a parallel start-up process for system services.

24. How to start or stop a service?

- *To start or stop a service in Linux, you can use the "systemctl" command.*
- *To start a service, use the command "sudo systemctl start service_name".*
- *To stop a service, use the command "sudo systemctl stop service_name".*
- *To restart a service, use the command "sudo systemctl restart service_name".*

25. How to check the status of a service?

To check the status of a service in Linux, you can use the "systemctl" command.

To check the status of a service, use the command "sudo systemctl status service_name".

This will display the current status of the service, including whether it is running or not, any error messages or warnings, and other relevant information.

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

- *To display logs on a system which uses systemd, you can use the "journalctl" command, which is a utility for querying and displaying logs collected by systemd's journal service.*
- *To display all logs, use the following command: sudo journalctl*
- *To display the logs for a specific service, replace "service_name" with the name of the service: sudo journalctl -u service_name*
- *You can also use additional options to filter the logs based on various criteria, such as time range, log level, or keyword search. For example, to display the logs for the last hour, use: sudo journalctl --since "1 hour ago"*

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

To make a certain process or application a service on Linux, you can create a systemd unit file that describes the service and its properties. Here are the general steps:

a. *Create a new unit file with a name that ends in ". service" in the /etc/systemd/system/ directory. For example, to create a service for "myapp", use: sudo nano /etc/systemd/system/myapp.service*

b. *In the unit file, define the service by specifying its name, description, and other properties.*

For example:

```
[Unit]
Description=My Application Service
After=network.target

[Service]
Type=simple
ExecStart=/path/to/myapp

[Install]
WantedBy=multi-user. target
```

c. *Save and close the unit file.*

d. *Reload the systemd configuration to pick up the new service: sudo systemctl daemon-reload*

e. *Start the service using systemctl: sudo systemctl start myapp*

f. *Check the status of the service to ensure it is running: sudo systemctl status myapp*

g. *Enable the service to start automatically on boot: sudo systemctl enable myapp*

28. Troubleshooting and Debugging

Troubleshooting and debugging are processes used to identify and solve problems in a system or application. The goal of troubleshooting is to identify the cause of the problem, while debugging involves finding and fixing errors in code.

29. Where system logs are located?

System logs are located in different directories depending on the operating system and the logging configuration. In Linux systems that use the systemd logging system, system logs are typically located in the /var/log directory. Some common system log files in this directory include:

/var/log/messages: General system messages and events
/var/log/auth.log: Authentication-related events
/var/log/kern.log: Kernel-related events
/var/log/syslog: General system messages (including those from applications)
*In addition to these system log files; specific applications may also have their own log files located in various directories***.**

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. This will allow you to continuously monitor the file and display new content as it is added.

The command syntax is as follows: tail -f /path/to/file
Replace /path/to/file with the actual path and filename of the file you want to monitor.
You can stop the monitoring process by pressing CTRL+C

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

Ping: A tool used to test network connectivity between two devices by sending packets and measuring the response time.

Traceroute: A tool used to identify the path that network traffic takes from one device to another. It sends packets with increasingly large time-to-

live (TTL) values and records the IP addresses of the routers that forward the packets.

Netstat: A tool used to display network statistics and active network connections on a device.

Tcpdump: A tool used to capture and analyze network traffic in real-time.

Wireshark: A network protocol analyzer that allows you to capture and examine packets in detail.

Nmap: A network exploration and security auditing tool that can scan networks to identify open ports, operating systems, and services.

These tools can help identify and troubleshoot issues related to network connectivity, performance, and security.

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

1.df: A command-line tool used to display disk usage information for file systems mounted on a Linux system.

2.du: A command-line tool used to estimate file space usage for a directory or file.

3.fsck: A command-line tool used to check and repair inconsistencies in file systems.

4.lsof: A command-line tool used to list open files and the processes that opened them.

5.strace: A command-line tool used to trace system calls and signals made by a process.

6.iostat: A command-line tool used to monitor input/output (I/O) statistics for storage devices and file systems.

These tools can help identify and troubleshoot issues related to disk usage, file system corruption, open files, and storage performance.

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

Some commonly used tools for troubleshooting and debugging process issues include:

1.ps: A command-line tool used to display information about running processes on a system.
2.top: A command-line tool used to monitor system resources and processes in real-time.
3.htop: A command-line tool similar to top but with a more user-friendly interface and additional features.
4.strace: A command-line tool used to trace system calls and signals made by a process.
5.lsof: A command-line tool used to list open files and the processes that opened them.
6.kill: A command-line tool used to send signals to processes, allowing them to be terminated or reloaded.

*These tools can help identify and troubleshoot issues related to process status, resource usage, system calls, open files, and process termination***.**

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

Some commonly used tools for debugging CPU related issues include:

1.top: A command-line tool used to monitor system resources and processes in real-time, including CPU usage.
2.htop: A command-line tool similar to top but with a more user-friendly interface and additional features, including CPU usage.
3.vmstat: A command-line tool used to monitor system performance, including CPU usage, memory usage, and I/O statistics.
4.strace: A command-line tool used to trace system calls and signals made by a process, including CPU usage.
5.perf: A command-line tool used to analyze and trace system performance, including CPU usage and system calls.

These tools can help identify and troubleshoot issues related to high CPU usage, CPU-bound processes, and system performance related to CPU usage.

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

If I receive a call from someone claiming that their system is slow, I would follow these steps:

1.Ask the user to provide more details about the system and the issue they are experiencing, such as the type of system (desktop, laptop, server), the operating system, and the specific symptoms they are noticing.
2.Check the system's resource usage, such as CPU, memory, and disk usage, using command-line tools such as top, htop, or vmstat.

3.Check the system logs for any error messages or warnings related to system performance or resource usage.
4.Check for any running processes that may be consuming excessive resources or causing performance issues using command-line tools such as ps or lsof.
5.Check for any updates or patches that may be available for the system or its applications.
6.Advise the user on any recommended actions to improve system performance, such as optimizing resource usage, cleaning up disk space, or upgrading hardware.

Overall, the goal is to gather more information about the issue, identify potential causes, and provide recommendations or solutions to resolve the performance problem.

36. Explain iostat output

iostat is a command-line tool used to monitor input/output (I/O) statistics for devices and partitions on a Linux system. The output of iostat includes several fields that provide information about the I/O activity on the system, including:

1.Device name: the name of the device or partition being monitored.
2.tps (transactions per second): the number of read/write operations completed per second.
3.kB_read/s: the amount of data read from the device or partition per second, measured in kilobytes.
4.kB_wrtn/s: the amount of data written to the device or partition per second, measured in kilobytes.
5.kB_read: the total amount of data read from the device or partition since the system was started, measured in kilobytes.
6.kB_wrtn: the total amount of data written to the device or partition since the system was started, measured in kilobytes.
7.%util: the percentage of time the device or partition was busy servicing I/O requests.

37. How to debug binaries?

Debugging binaries typically involves using a debugger tool to inspect and manipulate the running state of the binary. The most common tool for debugging binaries on Linux systems is the GNU Debugger (GDB). Here are the basic steps to debug a binary using GDB:

1.Compile the binary with debugging symbols: When compiling the binary, include the -g flag to add debugging symbols to the resulting binary. For example: gcc -g -o myprogram myprogram.c
2.Start GDB: Start GDB with the binary as an argument. For example: gdb myprogram

3.Set breakpoints: Set breakpoints at points in the code where you want to pause execution and inspect the state of the program. For example: break main
4.Run the program: Start the program with the "run" command. For example: run
5.Inspect the program state: Use various GDB commands to inspect the state of the program, such as "print" to print the value of a variable, or "step" to step through the code line by line.
6.Continue running the program: Use the "continue" command to continue running the program after a breakpoint is hit.

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

CPU load refers to the number of processes or threads that are waiting to be executed by the CPU at any given time. When the CPU is fully loaded, there are no idle processes waiting to be executed, and the load reaches 100%. High CPU load can indicate that the CPU is struggling to keep up with the demands of the system.

CPU utilization, on the other hand, is a measure of how much of the CPU's processing capacity is being used at a particular time. This can be expressed as a percentage of the total available processing power. High CPU utilization can indicate that the CPU is busy with processing tasks, but it doesn't necessarily mean that the system is experiencing performance issues.

In summary, CPU load measures how many processes are waiting to be executed, while CPU utilization measures how much of the CPU's processing capacity is being used at a given time. Both metrics are useful for understanding how a system is performing, but they provide different types of information.

39. How you measure time execution of a program?

One common way to measure the time execution of a program is to use the "time" command in the terminal.
To use it, simply prepend the "time" command to the program or command you want to run, like this: time my_program

When the program finishes running, "time" will output some statistics about the execution, including the real time (i.e., wall clock time) it took to execute the program, the user time (i.e., the amount of CPU time used by the program in user mode), and the system time (i.e., the amount of CPU time used by the program in kernel mode).

You can also use programming languages' built-in time functions or libraries to measure execution time within the code itself. For example, in

Python, you can use the "time" module to measure the time execution of a specific function or block of code.

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 in Linux and terminate it:

Use the lsof command with the path of the file to list all the processes accessing it.
Find the process ID (PID) of the process that is writing to the file.
Use the kill command with the appropriate signal (such as SIGTERM for a graceful termination) and the PID of the process to terminate it.
If the process does not exit within a certain time frame, use the SIGKILL signal to forcefully terminate it.

Kernel

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

The kernel is the core component of an operating system. It is a low-level software that interacts directly with the underlying hardware of a computer and provides a platform for higher-level software to run on.

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

To find out which kernel version your Linux system is using, you can use the uname -r command or check the contents of the /proc/version file using the cat /proc/version command.

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 running kernel on demand. Kernel modules allow developers to add new functionality to the kernel without having to recompile the entire kernel.

To load a new kernel module, you can use the modprobe command with the name of the module. The modprobe command will automatically load any dependencies required by the module.

44. Explain user space vs. kernel space

In Linux, user space refers to the area of memory and resources where user-level applications and programs run, while kernel space refers to the area of memory and resources where the kernel and low-level system processes run.

User space is isolated from the kernel space for security and stability reasons. User space programs can only interact with the kernel through defined interfaces, such as system calls and device files.

*Kernel space, on the other hand, has unrestricted access to the system hardware and resources, such as the CPU, memory, and storage devices. The kernel provides services and abstractions to user space programs through system calls, which allow user space programs to interact with the kernel and access system resources***.**

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

You can change the configuration of the Linux kernel in two phases of its lifecycle:

During kernel compilation: When you are compiling the kernel, you can modify its configuration by enabling or disabling various features, such as device drivers, filesystems, and network protocols. This is typically done by running the make menuconfig or make config command in the kernel source directory.

*During runtime: After the kernel has been compiled and is running, you can modify its configuration by changing various parameters and options through the sysctl interface. The sysctl interface allows you to view and modify kernel parameters that control the behaviour of various system components, such as networking, memory management, and process scheduling***.**

46. Where can you find kernel's configuration?

The kernel configuration is stored in a file called .config in the root of the kernel source directory.

If you are using a pre-built kernel that came with your Linux distribution, you can find its configuration in the /boot directory. The configuration file is usually named config-<kernel-version>, where <kernel-version> is the version number of the kernel.

You can also view the current kernel configuration at runtime by examining the /proc/config.gz file. This file contains a compressed version of the kernel configuration and can be viewed using the zcat or gunzip command.

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 called cmdline. This file is located in the /proc virtual filesystem.

48. How to list kernel's runtime parameters?

You can list the kernel's runtime parameters using the sysctl command. The sysctl command is used to view, set, and modify kernel parameters at runtime.

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

Yes, running sysctl -a as a regular user vs. root user will produce different results.

Many of the kernel parameters listed by sysctl -a require root privileges to read. If you run the command as a regular user, you will only see the parameters that are accessible to regular users. These are typically system settings that are not considered sensitive or security critical.

If you run sysctl -a as the root user, you will see the full list of kernel parameters, including those that require root privileges to access. This will give you a more complete picture of the system's configuration and performance.

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

To enable IPv4 forwarding in the kernel, you need to set the value of the net.ipv4.ip_forward parameter to 1. This parameter controls whether the kernel forwards IPv4 packets between network interfaces.

```
sudo sysctl -w 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, it modifies the values of kernel runtime parameters stored in the /proc/sys/ directory.*
- *The sysctl() system call is then called to update the corresponding kernel variables with the new values.*
- *The updated kernel variables trigger the appropriate update handlers, which apply the new settings to the corresponding subsystem or component.*
- *In summary, the sysctl command applies changes to kernel's runtime parameters by updating the values of the corresponding kernel variables and triggering the appropriate update handlers to apply the new settings.*

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

When you modify kernel runtime parameters using the sysctl command, the changes are only applied temporarily and will be lost after a system

reboot. To make the changes persist across reboots, you need to update the corresponding configuration files.

The configuration files that contain the kernel runtime parameters are typically located in the /etc/sysctl. d/ directory and have a .conf extension. Each file contains a set of parameter assignments, where each assignment is in the form of parameter_name=value.

To make your changes persist, you can create a new configuration file or modify an existing one and add the parameter assignments that correspond to the changes you made using the sysctl command. When the system boots, the configuration files are read and the kernel runtime parameters are set to the values specified in the files.

*Alternatively, you can also modify the /etc/sysctl.conf file directly, which is the main configuration file for kernel runtime parameters. This file contains global settings that apply to all system components, and also allows you to include other configuration files using the include directive***.**

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?

When you modify kernel parameters inside a container, the changes only affect the kernel parameters of the container itself, and not the kernel parameters of the host on which the container runs.

This is because containers provide a virtualized environment with its own set of kernel runtime parameters, which are isolated from the host system's kernel parameters. The container's kernel runtime parameters are managed by the container's runtime environment, such as Docker or Kubernetes, and are not visible or accessible from outside the container.

However, it's worth noting that some kernel parameters are shared between the container and the host system, such as network-related parameters or file system parameters. In these cases, changing the parameter value inside the container may affect the behavior of the corresponding subsystems on the host system as well.

*In general, it's recommended to avoid modifying kernel parameters inside a container unless it's necessary, and instead use container-specific configuration options or higher-level tools to manage the container's behavior***.**

SSH

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

SSH (Secure Shell) is a network protocol used for secure remote access to a Linux (or other Unix-like) operating system. It allows a user to establish a secure and encrypted connection to a remote server, and then access and manage the server's files, applications, and services from a remote location.

systemctl status ssh

55. Why SSH is considered better than telnet?

SSH is considered better than telnet for several reasons:

- *Security: SSH provides secure encrypted communication over an insecure network, while telnet sends all data in clear text, which can be intercepted and read by anyone with access to the network. This makes SSH much more secure than telnet, especially for remote access to critical systems.*

- *Authentication: SSH provides stronger authentication mechanisms than telnet, which relies on simple username and password authentication. SSH uses public-key cryptography and supports two-factor authentication, which makes it much harder for attackers to gain unauthorized access to a system.*

- *Flexibility: SSH supports tunnelling, forwarding, and other advanced features that telnet does not provide. For example, with SSH, you can securely forward X11 graphical applications over the network, tunnel traffic through an encrypted connection, or use SSH keys for automated login without entering a password.*

- *Portability: SSH is available on most Unix-like systems, including Linux, macOS, and FreeBSD, as well as on Windows through third-party clients. Telnet, on the other hand, is less commonly used and has been largely replaced by SSH.*

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

*The ~/.ssh/known_hosts file is used by SSH clients to store information about remote hosts that the client has connected to in the past. Specifically, it stores the public key of each remote host, along with its hostname or IP address and the SSH server's cryptographic fingerprint***.**

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

If you get the error message "Host key verification failed" when trying to connect to an SSH server, it means that the client is unable to verify the authenticity of the server's host key.

58. What is the difference between SSH and SSL?

SSH (Secure Shell) and SSL (Secure Sockets Layer) are both cryptographic protocols used to provide secure communication over the internet, but they serve different purposes.

SSH is primarily used for remote access to servers, allowing users to securely log in to a remote machine and execute commands or transfer files. It is designed for secure command-line access to a remote computer system and uses a combination of public-key cryptography, symmetric-key encryption, and message authentication to protect the confidentiality and integrity of data transmitted between the client and the server.

On the other hand, SSL is primarily used to provide secure communication between web browsers and web servers, allowing users to securely transmit sensitive information such as passwords, credit card numbers, and other personal data. SSL uses a combination of public-key and symmetric-key encryption to establish a secure, encrypted connection between the client and the server.

59. What ssh-keygen is used for?

ssh-keygen is a command-line utility in Linux and other Unix-like operating systems used for generating, managing, and verifying SSH authentication keys

60. What is SSH port forwarding?

SSH port forwarding, also known as SSH tunneling, is a technique used to securely transfer data between two computers over an unsecured network, such as the internet. SSH port forwarding enables a client to securely access a service on a remote server, by forwarding traffic from a local port to a remote port through an SSH tunnel.