Linux Assignment - 2 (*ALEEN ULLA)*

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

The root directory "/" in the Linux FHS is the top-level directory that contains all the files and directories necessary for the operating system to function properly.

2.What is stored in each of the following paths?

/bin, /sbin, /usr/bin and /usr/sbin
/etc
/home
/var
/tmp

- */bin: Essential command binaries (executable programs) that are required to boot the system and perform basic system maintenance.*
- */sbin: Essential system binaries (executable programs) that are required for system administration and maintenance.*
- */usr/bin: Non-essential command binaries (executable programs) that are used by general users, but not required for system boot or maintenance.*
- */usr/sbin: Non-essential system binaries (executable programs) that are used by system administrators, but not required for system boot or maintenance.*
- */etc: Configuration files that are used by the system and its applications.*
- */home: Home directories for individual users, where they can store their personal files and data.*
- */var: Variable files that contain system-generated data and logs, such as log files, database files, and temporary files.*
- */tmp: Temporary files that are created by the system and its applications, and are intended to be deleted regularly to free up disk space.*

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

The /tmp directory is typically cleaned out automatically on a regular basis, whereas other directories require manual deletion of files.

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

The /proc directory contains a virtual filesystem that provides system information in real-time, such as information about hardware, running processes, and kernel configuration.

5. What makes /proc different from other filesystems?

The /proc filesystem is a virtual filesystem that does not contain real files, but rather provides information about running processes and system resources in a hierarchical structure that can be accessed like a regular filesystem.

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

False.

Regular users can create some files and directories in the /proc filesystem, but they are typically limited to files and directories that correspond to their own processes and do not have the ability to create system-level files or modify system-wide settings. The ability to create and modify files in the /proc filesystem is generally restricted to users with administrative privileges or root access.

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

The /proc/cmdline file contains the command-line arguments that were passed to the Linux kernel when it was booted, including boot options, kernel parameters, and other system configuration settings.

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

In Linux, system devices such as block storage devices can be found in the "/dev" directory. This directory contains special files that represent physical and logical devices attached to the system, including hard drives, USB devices, CD-ROMs, and more. The device files in the "/dev" directory are used by the system to communicate with the devices and perform operations such as reading, writing, and formatting data.

Permissions

9. How to change the permissions of a file?

You can change the permissions of a file in Linux using the "chmod" command followed by the desired permission settings and the file name. Here is the basic syntax for the chmod command

chmod [options] permissions filename

The permissions are specified using a three-digit number or a combination of letters and symbols that represent the read, write, and execute permissions for the owner, group, and others. For example:

chmod 755 myfile.txt

10. What does the following permissions mean?:

777
644
750

- *777: This set of permissions gives the owner, group, and others full read, write, and execute permissions on the file or directory.*
- *644: This set of permissions gives the owner read and write permissions, and the group and others read-only permissions on the file.*
- *750: This set of permissions gives the owner read, write, and execute permissions, the group read and execute permissions, and others no permissions on the file or directory.*

11. What this command does? chmod +x some_file

The command "chmod +x some_file" sets the executable permission (+x) on the file named "some_file". This allows the file to be executed as a program or script.

The "+" sign indicates that the permission is being added, while "x" represents the executable permission. The "chmod" command changes the file permissions, and the argument "+x" specifies the permission to be added.

12. Explain what is setgid and setuid

- *setgid (set group ID) is a file permission bit that allows a user to inherit the group ownership of a file when creating new files in a directory, regardless of their default group ownership. This can be useful when multiple users need to work on the same files and ensure that the group ownership remains consistent.*

- *setuid (set user ID) is a file permission bit that allows a user to run an executable file with the permissions of the file's owner, rather than the permissions of the user running the file. This can be used in certain cases where a program requires elevated privileges to run certain operations, such as changing system settings or accessing sensitive data.*

13. What is the purpose of sticky bit?

The purpose of the sticky bit in Linux is to allow only the owner of a file or directory to delete or rename it, even if other users have write permission on the same directory. This is useful in scenarios where multiple users have write access to a shared directory, but only the owner should have the ability to delete or rename their own files. When the sticky bit is set on a directory, it is represented by the letter "t" in the file permission string, and can be set using the "chmod" command with the "+t" option.

14. What the following commands do?

chmod
chown
Chgrp

- *chmod: The "chmod" command is used to change the file mode (permissions) of a file or directory. It can add or remove read, write, or execute permissions for the owner, group, and others, and can be used to set special permissions like setuid, setgid, and sticky bit.*

- *chown: The "chown" command is used to change the ownership of a file or directory. It can be used to change the owner or group owner of a file or directory, and is often used to give ownership of files to a different user or group.*

- *chgrp: The "chgrp" command is used to change the group ownership of a file or directory. It can be used to change the group ownership of a file or directory to a different group, and is often used to give group ownership of files to a different group of users.*

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

Sudo is a command in Linux that allows a user with administrative privileges to execute commands or perform tasks with root privileges, without having to log in as the root user. This provides an additional layer of security by allowing users to perform administrative tasks only when necessary, rather than logging in as the root user for every task.

To set up sudo, you need to follow these steps:

Log in as a user with root privileges.
Install the "sudo" package using the package manager for your Linux distribution.

Open the sudo configuration file, typically located at /etc/sudoers, using a text editor with root privileges.
Add a line to the configuration file that grants a specific user or group the ability to run commands with root privileges using sudo.
Save the configuration file and exit the text editor.
Once sudo is set up, users who are granted access can use the "sudo" command before any other command to execute it with root privileges. For example, "sudo apt-get update" will execute the "apt-get update" command with root privileges. When a user executes a command using sudo, they will be prompted to enter their own password to verify their identity before the command is executed with root privileges.

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, a user must have administrative privileges, which can be achieved by either logging in as the root user or using the "sudo" command to run commands with root privileges. Without administrative privileges, a user will not have the necessary permissions to modify the system or install packages.

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

ACLs (Access Control Lists) are an extension to standard file permissions in Linux that allow for more fine-grained control over access to files and directories. ACLs allow administrators to specify permissions for individual users or groups beyond the traditional "owner-group-others" permission model.

ACLs can be useful in a variety of use cases, such as when multiple users or groups need different levels of access to the same file or directory, or when access control needs to be applied to a specific file or directory within a larger directory hierarchy.

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

- *Permission issues: If the user does not have the necessary permissions to create a file in the target directory or the file system, the file creation will fail.*
- *Disk space issues: If the disk or partition is full, it will not be possible to create a new file until some space is freed up.*

- *File system corruption: If the file system is corrupt or experiencing errors, it may not be possible to create a new file until the issues are resolved.*

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

The command "chmod -x $(which chmod)" removes the execute permission from the "chmod" command, making it unusable. To fix this, the user will need to restore the execute permission to the "chmod" command. This can be done using the "chown" command, provided the user has administrative privileges.

Here are the steps to fix the issue:

Log in as the root user or use the "sudo" command to gain administrative privileges.

Navigate to the directory where the "chmod" command is located by using the "which" command: "which chmod". This will display the path to the "chmod" command.

Once the path to "chmod" is known, use the "chmod" command to restore the execute permission: "chmod +x [path to chmod]".

Verify that the execute permission has been restored by running "ls -l $(which chmod)" and confirming that the "x" permission is present.

After following these steps, the "chmod" command should be usable again.

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. Here are the steps to copy a file to a remote Linux host using scp:

1.Open a terminal window on your local machine.

2.Use the "scp" command with the following syntax to copy the file to the remote host:

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, "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 path to the directory on the remote host where you want to copy the file.

3.Enter your password for the remote host when prompted.

4.Wait for the file to transfer.

After completing these steps, the file will have been copied to the specified location on the remote Linux host.

21. How to generate a random string?

Using the "uuidgen" command:
uuidgen | tr -d '-'

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

openssl rand -base64 7 | tr -dc 'a-zA-Z0-9' | head -c 7

This command generates a random string of 7 characters by first generating a random string of 7 bytes using the "-base64" option, and then removing any non-alphanumeric characters using the "tr" command. Finally, the "head" command is used to limit the output to 7 characters.

Systemd

23. What is systemd?

systemd is a system and service manager for Linux operating systems that is designed to manage the core functionality of the system, such as the initialization process, logging, network stack, and other system services. It provides a range of features such as parallel startup of system services, on-demand service activation, socket and D-Bus activation, process tracking with the cgroups functionality, and more. systemd has become the default init system for many major Linux distributions.

24. How to start or stop a service?

To start a service:
sudo systemctl start <service-name>

To stop a service:
sudo systemctl stop <service-name>

25. How to check the status of a service?

In Linux, you can check the status of a service using the "systemctl" command. Here's the command to check the status of a service:

systemctl status <service-name>

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

On a system which uses systemd, you can display the logs using the "journalctl" command. Here are some examples of how to use the command to display logs:

To display all logs:
journalctl

To display the logs for a specific service:
journalctl -u <service-name>

To display the logs for the current boot:
journalctl -b

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

To make a certain process or application a service in Linux, you can create a systemd unit file for that service. Here's how you can do it:

Create a new file in the /etc/systemd/system/ directory with a .service extension. For example, if you want to create a service for the Apache web server, you can create a file called apache.service.

In the unit file, define the service by specifying the service name, description, startup type, dependencies, and other parameters.

28. Troubleshooting and Debugging

Troubleshooting and debugging in Linux involves identifying and fixing problems that arise on the system. Here are some general steps that can be followed to troubleshoot and debug issues:

- *Identify the problem: Determine what the issue is by looking at error messages, system logs, and any other relevant information.*
- *Reproduce the problem: Try to reproduce the issue to ensure that it is not just a one-time occurrence.*

- *Gather information: Collect information such as system logs, configuration files, and output from relevant commands.*
- *Analyze the information: Look for patterns or commonalities in the information gathered to help identify the root cause of the issue.*
- *Test potential solutions: Try potential solutions to the problem, making sure to test each one thoroughly.*
- *Implement a solution: Once a solution has been found, implement it on the system.*
- *Verify the solution: Confirm that the solution has resolved the issue and that the system is functioning as expected.*

29. Where system logs are located?

System logs in Linux are typically stored in the /var/log directory. The exact location and names of the logs may vary depending on the specific distribution and configuration of the system. Some common log files in this directory include:

- */var/log/messages: Contains system messages and general system information.*
- */var/log/syslog: Contains messages from system programs and services.*
- */var/log/auth.log: Contains authentication-related messages, such as successful and failed login attempts.*
- */var/log/kernel.log: Contains messages from the kernel, including information about hardware and software errors*.

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

You can use the tail command with the -f option to follow a file's content as it is being appended. For example, to follow the content of the file example.log in real-time, you can use the following command:

tail -f example.log

This will display the last 10 lines of the file, and then update the display as new lines are added. You can use Ctrl + C to stop following the file.

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

There are several tools available in Linux that can be used for troubleshooting and debugging network issues, some of which include:

1. ***ping****: to check if a host is reachable and to measure the round-trip time for packets sent from the local host to a remote host.*

2. ***traceroute****: to trace the route taken by packets across an IP network to a remote host.*

3. ***netsta****t: to display active network connections, listening ports, and network statistics.*

4. ***tcpdump****: to capture and display packets on a network interface.*

5. ***nmap****: to perform network exploration and security auditing.*

6. ***wireshark****: to capture, analyze, and troubleshoot network traffic.*

7. ***ip****: to manage network interfaces, routing tables, and network protocols.*

These tools can help identify network connectivity issues, DNS resolution problems, routing errors, and other network-related problems.

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

There are several tools available in Linux that can be used for troubleshooting and debugging disk and file system issues, some of which include:

1. ***fdisk****: to view and modify disk partitioning.*

2. ***fsck****: to check and repair file system errors.*

3. ***dmesg****: to view kernel-level disk and file system error messages.*

4. ***smartctl****: to check and monitor the health of hard drives and solid-state drives.*

5. ***lsof****: to view open files and processes that are using them.*

6. ***strace****: to trace system calls and signals made by a program.*

7. ***grep****: to search for specific text or patterns in files.*

These tools can help identify disk errors, file system corruption, incorrect file permissions, and other disk and file system-related problems.

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

troubleshooting and debugging process issues:

- ***ps command**: This command is used to check the running processes on a Linux system.*
- ***top command**: This command is used to monitor system processes and resource usage in real-time.*
- ***strace command**: This command is used to trace system calls and signals made by a process, which can help diagnose issues.*
- ***lsof command**: This command is used to list open files and network connections, which can help identify issues with file or network access.*
- ***gdb debugger**: This is a powerful debugging tool that allows you to analyze and debug the behavior of programs running on a Linux system.*
- ***tcpdump command**: This command is used to capture network packets, which can help diagnose network-related issues.*
- ***journalctl command**: This command is used to view and analyze system logs, which can provide valuable information for troubleshooting issues.*
- ***dmesg command**: This command is used to display kernel messages, which can provide useful information for diagnosing hardware or driver issues.*

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

- *top command: This command is used to monitor system processes and resource usage, including CPU usage.*
- *htop command: This is an interactive process viewer that allows you to monitor system processes and resource usage, including CPU usage.*
- *perf command: This command is used to profile the performance of a Linux system and analyze CPU usage.*
- *strace command: This command is used to trace system calls and signals made by a process, which can help diagnose CPU-related issues.*
- *gdb debugger: This is a powerful debugging tool that allows you to analyze and debug the behavior of programs running on a Linux system, including CPU usage.*

- *sar command: This command is used to collect and report system resource utilization, including CPU usage.*

- *mpstat command: This command is used to report processor-related statistics, including CPU usage.*

- *vmstat command: This command is used to report virtual memory statistics, including CPU usage***.**

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

- *If I were a system administrator receiving a call from someone claiming that their system is slow, here are the steps I would take:*

- *Ask the user to provide more information about the problem. Specifically, I would ask them to describe the symptoms they are experiencing, such as which programs are running slow or if the entire system is slow.*

- *Check the system's resource usage. I would use tools such as top or htop to check the CPU and memory usage of the system. This would help me identify if there are any resource-intensive programs running that could be causing the slowness.*

- *Check the system logs. I would check system logs such as syslog or journalctl to see if there are any error messages that could indicate a problem with the system.*

- *Check for malware or other security issues. I would use antivirus software or other security tools to check for malware or other security issues that could be slowing down the system.*

- *Check the network connection. If the user is accessing resources over the network, I would check the network connection to make sure there are no issues that could be causing slow performance.*

- *Offer to connect remotely to the user's system. If the user agrees, I would use remote access software such as SSH or VNC to connect to the user's system and diagnose the problem directly.*

36. Explain iostat output

iostat is a command-line tool used to monitor system input/output (I/O) statistics, including CPU utilization, disk usage, and network activity

CPU statistics:

- *%user: The percentage of CPU time spent executing user processes.*

- *%nice: The percentage of CPU time spent executing processes with a "nice" priority level.*
- *%system: The percentage of CPU time spent executing kernel processes.*
- *%iowait: The percentage of CPU time spent waiting for I/O operations to complete.*
- *%steal: The percentage of CPU time stolen by a hypervisor, if running in a virtualized environment.*
- *%idle: The percentage of CPU time not being used.*

Device statistics:

- *Device: The name of the device.*
- *tps: The number of I/O operations per second that were issued to the device.*
- *kB_read/s: The number of kilobytes read from the device per second.*
- *kB_wrtn/s: The number of kilobytes written to the device per second.*
- *kB_read: The total number of kilobytes read from the device.*
- *kB_wrtn: The total number of kilobytes written to the device.*

37. How to debug binaries?

Here are the general steps you can take to debug a binary:

- *Compile the binary with debug information: The binary should be compiled with debug information, such as symbol tables, to enable the debugger to map machine code to source code.*

- *Choose a debugger: There are many different debuggers available, such as GDB, LLDB, and WinDbg. Choose a debugger that works with your platform and language.*

- *Start the debugger: Start the debugger and load the binary into it. Set any breakpoints you need to pause the execution of the binary at specific points.*

- *Run the binary: Start running the binary. When a breakpoint is reached, the debugger will pause the execution of the binary and give you control over its behavior.*

- *Inspect the state of the binary: At any point while the binary is paused, you can use the debugger to inspect the current state of the binary, such as the values of variables and registers.*

- *Modify the state of the binary: You can modify the state of the binary at runtime, such as changing the values of variables or registers, and then continue the execution of the binary.*

- *Analyze the behavior of the binary: Use the debugger to analyze the behavior of the binary and identify any issues or errors that are occurring.*

- *Repeat as necessary: If you identify an issue, make changes to the binary or its inputs, and then repeat the process until the issue is resolved.*

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

CPU load and utilization are related but different concepts.

CPU load is a measure of the amount of work that the CPU is currently handling. It is typically represented as a numerical value, where a value of 1.0 represents 100% utilization of a single CPU core. For systems with multiple CPU cores, a load value of 2.0 would represent 100% utilization across two CPU cores.

CPU utilization, on the other hand, is a measure of the percentage of time that the CPU is busy executing tasks compared to the amount of time it is idle. It is typically represented as a percentage, where 100% utilization means that the CPU is busy all the time, and 0% utilization means that the CPU is idle all the time.

In other words, CPU load measures the amount of work being done by the CPU, while CPU utilization measures the percentage of time that the CPU is busy doing that work.

*It is worth noting that CPU load and utilization are related but not equivalent. A system with a high CPU load may not necessarily have a high CPU utilization if the load is spread across multiple CPU cores. Similarly, a system with a high CPU utilization may not necessarily have a high CPU load if the work being done is not particularly demanding***.**

39. How you measure time execution of a program?

There are several ways to measure the execution time of a program, including:

- *Using the time command: In Linux and Unix systems, you can use the time command to measure the execution time of a program. Simply prefix the command with the time command, and the output will include the user time, system time, and real time taken by the command.*

- *Using a profiler: A profiler is a tool that can be used to measure the execution time of a program and identify performance bottlenecks. Profilers can be integrated into development environments or run as separate tools. They provide detailed information about the time spent in each function or method and can help identify areas where optimizations can be made.*

- *Using the clock function: The clock function in C/C++ can be used to measure the CPU time taken by a program. The clock function returns the number of CPU clock cycles used by the program, which can be converted to seconds or milliseconds using the CLOCKS_PER_SEC constant.*

- *Using a stopwatch: For simple programs, you can measure the execution time manually using a stopwatch or timer. Simply start the stopwatch when the program begins executing and stop it when it finishes, and then record the elapsed time.*

- *Using built-in timing functions: Many programming languages and frameworks have built-in timing functions that can be used to measure the execution time of a program. For example, Python has the timeit module, which can be used to time the execution of small code snippets.*

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 that is writing to a specific file and then kill it, you can follow these steps:

***Identify the process ID (PID) of the process writing to the file**: You can use the lsof command to list all open files and the processes that have them open. For example, to find the process writing to a file named example.log, you can run the command lsof | grep "example.log". This will return a list of processes with their PIDs that are accessing the file.*

***Kill the process**: Once you have identified the PID of the process writing to the file, you can use the kill command to terminate it. For example, if the PID of the process is 1234, you can run the command kill 1234 to kill the process.*

Kernel

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

A kernel is the central component of an operating system (OS). It is responsible for managing system resources, providing services to user-level processes, and acting as an interface between the hardware and the software components of the system.

Some of the main functions of a kernel include:

- *Memory management: The kernel allocates and deallocates memory for programs and manages the virtual memory space.*

- *Process management: The kernel schedules processes and manages their execution, including allocating CPU time and managing inter-process communication.*

- *Device management: The kernel controls access to hardware devices and manages input/output (I/O) operations.*

- *File system management: The kernel provides a file system interface to user-level processes and manages file access and storage.*

- *Network management: The kernel provides networking functionality, including managing network interfaces and implementing networking protocols.*

- *Security: The kernel enforces security policies, such as user permissions and access controls, and manages system-level security features like firewalls and encryption.*

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 in a terminal or command prompt. Simply open a terminal and type the following command:

uname -r

This will display the version of the running kernel on your system. The output will look something like this:

5.4.0-91-generic

This indicates that the system is running kernel version 5.4.0-91-generic. Depending on your system configuration, the output may differ slightly, but the format should be similar.

Alternatively, you can also use the cat command to view the contents of the /proc/version file, which contains information about the running kernel. For example, you can run the command:

cat /proc/version

This will display the version information for the kernel, including the release version, build date, and build number.

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 dynamically loaded and unloaded into the running kernel to provide additional functionality or to extend the kernel's capabilities. Kernel modules are typically used to add support for new hardware devices, file systems, network protocols, or other system features that were not present in the kernel at the time of its compilation.

To load a new kernel module, you can use the insmod command followed by the name of the module file. For example, to load a module named mymodule.ko, you can run the command:

sudo insmod mymodule.ko

This will insert the module into the kernel, making its functions and services available to the system.

To unload a kernel module, you can use the rmmod command followed by the name of the module. For example, to unload the mymodule module, you can run the command:

sudo rmmod mymodule

his will remove the module from the kernel and free up any resources that it was using.

Note that in order to load or unload kernel modules, you typically need root or administrative privileges on the system. Additionally, it is important to ensure that any modules you load or use are compatible with your kernel version and that they are obtained from a trusted source to avoid potential security risks.

44. Explain user space vs. kernel space

- ***User space*** *is the memory space that is allocated to user-level processes, such as applications, utilities, and services. User space processes run in a restricted environment and are not allowed direct access to the system's hardware or low-level system resources. Instead, they interact with the kernel through system calls, which are requests for services or information provided by the kernel.*

- ***Kernel space****, on the other hand, is the memory space that is reserved for the kernel and its associated system components. The kernel is responsible for managing system resources, including memory, CPU time, and hardware devices. It has unrestricted access to the system's hardware and low-level resources and can perform privileged operations that are not available to user space processes*.

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

The kernel configuration can be changed during different phases of the kernel lifecycle, depending on the type of configuration and the specific method used to apply it. Some of the common phases where the kernel configuration can be modified include:

- *During kernel compilation: The kernel configuration can be modified before the kernel is compiled, by editing the kernel configuration file (.config) using a text editor or a configuration tool. The configuration options can be set or unset depending on the system requirements, and the resulting kernel image will reflect the selected configuration.*

- *During kernel boot: The kernel configuration can also be modified during the boot process, by passing command-line parameters to the kernel or by using boot-time configuration files, such as grub.conf or menu.lst. These parameters can enable or disable kernel features, set hardware parameters, or specify system settings.*

- *At runtime: Some kernel configuration options can be changed at runtime, while the system is running. This can be done using kernel modules or sysctl, a system utility that allows users to modify kernel parameters during runtime. For example, the sysctl command can be used to change networking parameters, file system settings, or security options.*

46. Where can you find kernel's configuration?

The kernel configuration file can be found in the kernel source tree, usually in the root directory, and is named .config. This file contains a list of configuration options, either set or unset, that determine the features and behavior of the kernel when it is compiled.

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 depends on the specific boot loader used by the system. The two most common boot loaders used in Linux systems are GRUB (GRand Unified Bootloader) and systemd-boot (formerly known as gummiboot).

In the case of GRUB, the boot loader configuration file is typically located in the ***/boot/grub/grub.cfg*** *file. This file is automatically generated by the update-grub command and contains the configuration options for all bootable kernels and operating systems installed on the system.*

For systemd-boot, the boot loader configuration file is typically located in the ***/boot/loader/loader.conf*** *file. This file contains the global configuration options for the boot loader, such as the default boot entry and the timeout value.*

48. How to list kernel's runtime parameters?

You can list the kernel's runtime parameters, also known as kernel command line parameters, by checking the contents of the /proc/cmdline file. This file contains the arguments passed to the kernel at boot time by the boot loader.

To view the contents of the /proc/cmdline file, open a terminal window and run the following command:

cat /proc/cmdline

Some common kernel command line parameters include:

- *root: specifies the root file system for the kernel to mount during boot*
- *quiet: suppresses most of the kernel boot messages*
- *debug: enables kernel debugging output*
- *mem: specifies the amount of memory to reserve for the kernel*
- *acpi: enables or disables Advanced Configuration and Power Interface (ACPI) features*

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

*Yes, running **sysctl -a** as a regular user versus as root will produce different results. The **sysctl** command is used to view or modify kernel parameters at runtime. However, many kernel parameters are only accessible by the root user for security reasons.*

*When you run **sysctl -a** as a regular user, you will only see the subset of kernel parameters that are accessible to that user. This may include only a few non-sensitive parameters related to the user's own processes and environment.*

*When you run **sysctl -a** as root, you will see the full list of kernel parameters that are accessible to the root user. This includes sensitive parameters related to system security, networking, and hardware configuration.*

In general, it is recommended to only run sysctl commands as root when necessary, and to avoid making changes to kernel parameters unless you fully understand their implications and have a specific need to modify them.

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

- *To enable IPv4 forwarding in the kernel on a Linux system, you can use the sysctl command to modify the net.ipv4.ip_forward parameter:*

- *Open a terminal window and log in as root or a user with sudo privileges.*
  Run the following command to enable IPv4 forwarding:

  sysctl net.ipv4.ip_forward=1

  This will immediately enable IPv4 forwarding in the kernel. To make the change persistent across reboots, you can add the following line to the /etc/sysctl.conf file:

  net.ipv4.ip_forward=1

*This will ensure that the **net.ipv4.ip_forward** parameter is set to 1 every time the system boots.*

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 parameter, the change takes effect immediately in the running kernel. This is because the sysctl command uses the sysctl() system call to modify the corresponding system variable in the kernel.

The sysctl() system call allows a user-space program, like the sysctl command, to query and modify various system parameters at runtime. When the sysctl command is run, it invokes the sysctl() system call with the appropriate arguments to read or write the specified system variable.

After the sysctl() system call returns, the kernel updates the corresponding parameter with the new value, and the change takes effect immediately. This means that any processes or services that rely on that kernel parameter will immediately begin using the new value.

Note that some kernel parameters may not be modifiable at runtime, or may only be modifiable by the root user for security reasons. In these cases, attempting to modify the parameter using sysctl may result in an error.

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

To make changes to kernel runtime parameters persistent across reboots, you can modify the system's sysctl.conf configuration file.

The sysctl.conf file is a text file located in the /etc directory that contains configuration settings for various kernel parameters. To make a change to a kernel parameter persistent, you can add a line to the sysctl.conf file that sets the desired value for that parameter.

For example, if you want to enable IPv4 forwarding in the kernel, you can add the following line to /etc/sysctl.conf:

net.ipv4.ip_forward=1

When the system boots, the sysctl program reads the /etc/sysctl.conf file and sets the corresponding kernel parameters to the values specified in the file. This means that any changes you make to the sysctl.conf file will be applied automatically on the next system boot.

Note that changes to kernel parameters made through /etc/sysctl.conf will only take effect after the system reboots. If you need to apply the changes immediately, you can use the sysctl command to modify the parameter at runtime

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, the changes you make to kernel parameters inside a container will not affect the kernel parameters of the host system.

Containers are designed to provide an isolated environment for running applications, which includes their own filesystem, networking stack, and process namespace. While the container shares the same kernel as the host system, it runs in its own namespace, which means that changes made to kernel parameters inside the container only affect the container's view of the kernel, not the host's.

Similarly, changes made to kernel parameters on the host system will not affect the kernel parameters inside the container. Each container has its own view of the kernel and its own set of kernel parameters, which are independent of the host system.

It's worth noting, however, that some kernel parameters, such as those related to networking, may have a global effect on the system and can affect all containers running on that system. In general, it's best to avoid making changes to kernel parameters inside a container unless you have a specific reason to do so, and to carefully consider the potential impact of any changes you make.

SSH

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

SSH (Secure Shell) is a protocol for secure remote login and file transfer over an unsecured network. It provides a secure channel over an unsecured

network in a client-server architecture, allowing users to log in and execute commands on a remote machine as if they were physically present at the console.

To check if a Linux server is running SSH, you can try to connect to the server using the SSH client on your local machine. If the server is running SSH, you should be able to establish a secure connection using the SSH client.

To do this, open a terminal or command prompt on your local machine and enter the following command:

ssh username@server_ip_address

55. Why SSH is considered better than telnet?

SSH is considered better than Telnet for several reasons, including:

- ***Security**: SSH provides encryption and secure authentication, whereas Telnet sends data in plaintext and is vulnerable to interception and tampering by attackers.*
- ***Portability**: SSH can be used on virtually any platform or operating system, whereas Telnet is typically only available on legacy systems.*
- ***Functionality**: SSH provides a range of additional features and capabilities beyond those of Telnet, including file transfer, X11 forwarding, and tunneling of other protocols.*
- ***Flexibility**: SSH allows for flexible key management and authentication options, including the use of public and private key pairs, whereas Telnet relies on a password-based authentication system that is less secure.*
- ***Compliance**: SSH is often required for compliance with security regulations and standards, such as PCI DSS, HIPAA, and SOX, whereas Telnet may not meet these requirements.*

Overall, SSH provides a more secure, versatile, and reliable means of remote access and communication than Telnet, and is the preferred choice for most modern systems and networks.

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

*The **~/.ssh/known_hosts** file contains a list of public keys for remote hosts that the user has previously connected to via SSH. When the user connects to a remote host for the first time, the host's public key is added to this file. On subsequent connections, the client will check this file to verify the authenticity of the remote host's public key. If the public key presented by the remote host*

*matches the one stored in **known_hosts**, the connection will be allowed to proceed. If the public key has changed (e.g., due to a man-in-the-middle attack), the user will be prompted to confirm whether to continue with the connection or abort it.*

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

*The **~/.ssh/known_hosts** file contains a list of public keys for remote hosts that the user has previously connected to via SSH. When the user connects to a remote host for the first time, the host's public key is added to this file. On subsequent connections, the client will check this file to verify the authenticity of the remote host's public key. If the public key presented by the remote host matches the one stored in **known_hosts**, the connection will be allowed to proceed. If the public key has changed (e.g., due to a man-in-the-middle attack), the user will be prompted to confirm whether to continue with the connection or abort it.*

58. What is the difference between SSH and SSL?

SSH (Secure Shell) and SSL (Secure Sockets Layer) are both cryptographic protocols that provide secure communication over a network, but they are used for different purposes and operate at different layers of the network stack.

- *SSH is primarily used for secure remote access to a server, allowing users to log in and execute commands securely over an insecure network. SSH operates at the application layer of the network stack and uses its own transport protocol.*

- *SSL, now known as TLS (Transport Layer Security), is primarily used to provide secure communication between a client and a server, typically for web-based applications. SSL/TLS operates at the transport layer of the network stack and is used to secure traffic between a client and server over the internet.*

Both protocols use encryption to protect the confidentiality and integrity of the communication, but SSH provides a more complete solution for remote access, while SSL/TLS provides a more flexible and widely adopted solution for secure communication between clients and servers.

59. What ssh-keygen is used for?

***ssh-keygen** is a tool that is used to generate, manage, and convert SSH authentication keys. It can be used to generate a public-private key pair that can be used for authentication when connecting to an SSH server. The private key is kept on the client machine, and the public key is added to the authorized keys file on the server.*

60. What is SSH port forwarding?

SSH port forwarding, also known as SSH tunneling, is a mechanism that allows you to forward network traffic from one network port to another over an encrypted SSH connection. This can be useful in a number of scenarios, such as when you need to access a remote service that is only accessible from a certain network, or when you need to securely transfer data between two networks.

There are two types of port forwarding in SSH: local port forwarding and remote port forwarding.

- ***Local port forwarding**: This forwards traffic from a local port on your machine to a remote host and port. It allows you to connect to a remote service as if it were running on your local machine. For example, you can use local port forwarding to connect to a database running on a remote server from your local machine.*

- ***Remote port forwarding**: This forwards traffic from a remote port on a remote machine to a local host and port. It allows you to make a service running on your local machine accessible to a remote machine. For example, you can use remote port forwarding to make a web application running on your local machine accessible to a remote user.*

SSH port forwarding can be configured using the -L option for local port forwarding, and the -R option for remote port forwarding, when connecting to an SSH server.