Assignment 2

1. **In Linux FHS (Filesystem Hierarchy Standard) what is the /?**
   In Linux FHS, the / (root) directory is the highest level of the filesystem hierarchy. It is the starting point of the file system, and all other directories and files are organized under it. All other directories and files are mounted onto the root directory, either directly or indirectly.

2. **What is stored in each of the following paths?**
   a. /**bin**- This directory contains executable files that are essential to the system. These files are required for booting, repairing, and recovering the system.
   b. /**sbin**- This directory contains system binaries that are essential for system administration.
   c. /**usr**- This directory contains user-related programs, libraries, documentation, and configuration files.
   d. /**bin**- This directory contains executable files that are essential to the system. These files are required for booting, repairing, and recovering the system.
   e. /**usr**- This directory contains user-related programs, libraries, documentation, and configuration files.
   f. /**sbin**- This directory contains system binaries that are essential for system administration.
   g. /**etc**- This directory contains configuration files that are used by the system and various applications.
   h. /**home**- This directory contains the home directories of all users on the system. Each user has a subdirectory within /home that is named after their username.
   i. /**var**- This directory contains variable files that change frequently, such as system logs, spool directories, and temporary files.
   j. /**tmp**- This directory contains temporary files that are created by various programs and processes on the system.

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

The /tmp directory is special because it is designated for temporary files. Unlike other directories, the contents of /tmp are not preserved across reboots. The files in /tmp can be deleted at any time by the system or the administrator, and users should not rely on them for long-term storage.

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

   The /proc directory provides a virtual filesystem that exposes information about running processes and system resources. One can find information about system hardware, kernel parameters, system statistics, and more.

5. **What makes /proc different from other filesystems?**

   The /proc filesystem is different from other filesystems in that it is not a real filesystem, but rather a virtual filesystem that provides information about system resources and processes. It is a special filesystem that is dynamically generated by the kernel on the fly and does not exist on a physical storage device.

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

   False. Any user can read information from /proc, but only privileged users like root can modify the contents of /proc.

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

   The /proc/cmdline file contains the command-line arguments that were passed to the kernel at boot time. This includes information about the kernel version, boot loader, and system parameters.

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

   The system devices, including block storage devices, can be found under the /dev directory. These devices are represented as special files in the /dev directory, such as /dev/sda for a hard disk or /dev/ttyUSB0 for a USB serial port.

Permissions

1. **How to change the permissions of a file?**

   The permissions of a file can be changed using the chmod command. For example, to add write permission for the owner of the file, the command would be:

   chmod u+w file.txt

2. **What do the following permissions mean?**

   777- This means that the file or directory has read, write, and execute permissions for the owner, group, and all other users.

   644- This means that the file has read and write permissions for the owner, and read-only permissions for the group and all other users.

   750- This means that the file or directory has read, write, and execute permissions for the owner, read and execute permissions for the group, and no permissions for all other users.

3. **What this command does? chmod +x some_file**

   This command adds executable permissions to the file named "some_file" for the owner, group, and all other users.

4. **Explain what is setgid and setuid**

   Setgid and setuid are special permissions that can be set on a file or directory. Setgid causes any files or directories created within the directory to inherit the group ownership of the directory, while setuid causes any programs executed by a user to run with the privileges of the file owner.

5. **What is the purpose of sticky bit?**

   The sticky bit is a special permission that can be set on a directory to prevent users from deleting or renaming files that they do not own within the directory.

6. **What do the following commands do?**

   Chmod- This command is used to change the permissions of a file or directory.

   Chown- This command is used to change the owner of a file or directory.

   Chgrp- This command is used to change the group ownership of a file or directory.

7. **What is sudo? How do you set it up?**

Sudo is a command that allows users to run commands with the privileges of another user, typically the root user. It is used to perform tasks that require administrative privileges. To set up sudo, the user must be added to the sudoers file, which is typically located in the /etc directory.

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

True. Installing packages typically requires administrative privileges, so the user must either be the root user or use the sudo command to install packages.

9. **Explain what ACLs are. For what use cases would you recommend using them?**

ACLs (Access Control Lists) are a mechanism for setting permissions on files and directories that allows more fine-grained control over who can access the file or directory. ACLs allow multiple users or groups to be granted different levels of access to the same file or directory. They can be useful in cases where simple Unix permissions are not sufficient, such as in complex file-sharing environments or in situations where different users or groups require different levels of access to a file or directory.

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

Some possible reasons why creating a file could fail include:

- Lack of permission to write to the directory where the file is being created.
- Running out of disk space on the filesystem.
- Attempting to create a file with a filename that is not valid or contains invalid characters.

11. **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 would remove the execute permission from the `chmod` command, making it impossible to modify file permissions using the `chmod` command.

Scenarios

1. **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` command. The syntax for `scp` is: `scp /path/to/local/file username@remotehost:/path/to/destination`.

2. **How to generate a random string?**
   To generate a random string in Linux, you can use the `tr`, `dev`, and `urandom` commands together. The `tr` command can be used to translate or delete characters, while `dev/urandom` is a pseudo-random number generator that outputs random bytes. The command to generate a random string of characters is:
   `tr -dc 'a-zA-Z0-9' < /dev/urandom | head -c 20`

3. **How to generate a random string of 7 characters?**
   To generate a random string of 7 characters, you can modify the command to output only 7 characters:
   `tr -dc 'a-zA-Z0-9' < /dev/urandom | head -c 7`

Systemd

1. **What is systemd?**
   `systemd` is a system and service manager for Linux operating systems. It is responsible for starting and stopping services, managing system resources, and providing logging and monitoring functionality.

2. **How to start or stop a service?**
   To start or stop a service using systemd, you can use the `systemctl` command.
   The syntax to start a service is: `sudo systemctl start service_name`
   The syntax to stop a service is: `sudo systemctl stop service_name`

3. **How to check the status of a service?**
   `systemctl status service_name`
   This will display the current status of the service, whether it is running or not, and any error messages or warnings.

4. **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.
   To display all logs, use the following command: **sudo journalctl**
   To display logs for a specific service, use: `sudo journalctl -u service_name`

5. **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. A unit file is a configuration file that describes how systemd should manage the service. You can create a unit file in the /etc/systemd/system/ directory. The file should have a .service extension and follow the systemd unit file format. Once the unit file is created, you can start and stop the service using systemctl.

6. **Troubleshooting and debugging.**
   Troubleshooting and debugging involves identifying and resolving issues that arise in a system. This can involve checking logs, running diagnostic tools, analyzing system performance, and identifying potential issues with hardware or software components.

7. **Where are system logs located?**
   System logs are in the /var/log/ directory. Each log file contains messages and information related to a specific component or service.

8. **How to follow file's content as it being appended without opening the file every time?**
   To follow the content of a file as it is being appended without opening the file every time, you can use the tail command with the -f option. For example: tail -f /var/log/syslog. This will display the content of the syslog file as it is being written to.

9. **What are you using for troubleshooting and debugging network issues?**
   For troubleshooting and debugging network issues, you can use tools such as ping, traceroute, netstat, and tcpdump. These tools can help you identify issues with network connectivity, traffic routing, and firewall rules.

10. **What are you using for troubleshooting and debugging disk & file system issues?**
    For troubleshooting and debugging disk and file system issues, you can use tools such as df, lsblk, mount, and fsck. These tools can help you identify issues with disk space, disk usage, and file system errors.
11. **What are you using for troubleshooting and debugging process issues?**
    For troubleshooting and debugging process issues, you can use tools such as ps, top, kill, and strace. These tools can help you identify issues with process performance, memory usage, and system resource allocation.

12. **What are you using for debugging CPU related issues?**
    For debugging CPU related issues, you can use tools such as mpstat, perf, and sar. These tools can help you identify issues with CPU usage, CPU load, and system performance.

13. **You get a call from someone claiming, "my system is SLOW". What do you do?**
    To troubleshoot a slow system, you can start by checking the system logs for any error messages or warnings. You can also check the resource usage using tools such as top, htop, or vmstat. Checking disk usage, network connectivity, and CPU usage can also provide clues to the cause of the slow performance.

14. **Explain iostat output.**
    iostat is a system monitoring tool that reports input/output (I/O) statistics for storage devices and partitions. It provides information on the number of reads and writes per second, the amount of data transferred, and the average response.

15. **How to debug binaries?**
    Debugging binaries can involve using various tools such as gdb, strace, and ltrace. Gdb is a popular debugger tool that allows you to step through the code, set breakpoints, and examine memory and variables. Strace is a tool that traces system calls and signals, while ltrace is used to trace library calls. These tools can help you diagnose issues with a binary, identify memory leaks, and track down problems with system calls and libraries.

16. **What is the difference between CPU load and utilization?**
    CPU load and CPU utilization are related concepts but are not the same thing. CPU load refers to the number of processes waiting to be executed or in a runnable state, while CPU utilization is a measure of the percentage of time that the CPU is busy processing tasks.

17. **How do you measure the time execution of a program?**
    There are various ways to measure the execution time of a program, such as using the time command or using performance profiling tools like gprof or valgrind. The time command allows you to measure the execution time of a program by running it and displaying the real, user, and system time taken by the program. Performance profiling tools like gprof or valgrind allow you to measure the time spent in each function and can help you identify performance bottlenecks in your program.

Scenarios

1. **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 kill a process that is writing to a specific file, you can use the lsof command to find the process ID (PID) associated with the file and then use the kill command to terminate the process.**
    - **First, use the lsof command with the file path as an argument to find the PID of the process writing to the file: `lsof /path/to/file`.**
    - **This will list all processes that have the file open, along with their PIDs.**

- **Identify the PID of the process that you want to kill from the output of the lsof command.**
- **Use the kill command with the PID to terminate the process: kill PID.**
- **This will send a signal to the process asking it to terminate gracefully. If the process does not respond, you can use the -9 option to force kill the process: kill -9 PID.**

Kernel

1. **What is a kernel, and what does it do?**
   The kernel is the core component of the operating system that interfaces with the hardware and provides essential services to other software running on the system. It manages system resources such as memory, CPU time, and input/output operations. It also handles security, process management, and device drivers.

2. **How do you find out which Kernel version your system is using?**
   You can find out which kernel version your system is using by running the following command in a terminal: uname -r.

3. **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 into the kernel at runtime to add new functionality or modify existing functionality. You can load a new module using the modprobe command, for example: sudo modprobe my_module.

4. **Explain user space vs. kernel space.**
   User space is the area of the system memory where user applications and processes run. Kernel space is the area of the system memory where the kernel and device drivers run. User space programs interact with the kernel through system calls, which provide an interface to the kernel's functionality.

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

You can change the configuration of the kernel during the compilation phase or at runtime by modifying the kernel's runtime parameters.

6. **Where can you find kernel configuration?**
   Kernel configuration can be found in the /proc/config.gz file, which contains the configuration used to build the currently running kernel.

7. **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 typically /boot/grub/grub.cfg or /boot/grub2/grub.cfg, depending on the version of GRUB used.

8. **How to list kernel's runtime parameters?**
   You can list the kernel's runtime parameters by running the sysctl -a command.

9. **Will running sysctl -a as a regular user vs. root produce a different result?**
   Running sysctl -a as a regular user vs. root will produce a different result because some of the kernel's runtime parameters are only accessible to the root user.

10. **You would like to enable IPv4 forwarding in the kernel, how would you do it?**
    To enable IPv4 forwarding in the kernel, you can set the value of the /proc/sys/net/ipv4/ip_forward parameter to 1 by running the following command:
    echo 1 | sudo tee /proc/sys/net/ipv4/ip_forward.

11. **How sysctl applies the changes to kernel's runtime parameters the moment you run sysctl command?**
    Sysctl applies the changes to kernel's runtime parameters by writing the new values to the appropriate files in the /proc/sys directory.

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

Changes to kernel runtime parameters can persist across reboots by modifying the system's configuration files, such as /etc/sysctl.conf or /etc/sysctl.d/.

13. **Do the changes you make to kernel parameters in a container affects also the kernel parameters of the host on which the container runs?**
    Changes to kernel parameters made inside a container generally do not affect the kernel parameters of the host system, unless the container is running in privileged mode and has access to the host's kernel.

SSH

1. **What is SSH? How to check if a Linux server is running SSH?**
   SSH stands for Secure Shell, it is a network protocol used to securely access a remote system over an unsecured network. To check if a Linux server is running SSH, you can try to connect to the server using an SSH client, such as OpenSSH or PuTTY, by specifying the server's IP address or hostname along with the SSH port (usually 22). If the connection is successful and prompts you for login credentials, then the server is running SSH.

2. **Why SSH is considered better than telnet?**
   SSH is considered better than telnet because it uses encryption and provides secure communication between the client and the server, whereas telnet sends all data in plaintext, making it vulnerable to interception and eavesdropping.

3. **What is stored in ~/.ssh/known_hosts?**
   ~/.ssh/known_hosts are a file that stores the public key fingerprints of remote hosts that the user has connected to using SSH. When the user tries to connect to a remote host, SSH checks the public key fingerprint of the remote host against the one stored in the known_hosts file to verify the host's identity and prevent man-in-the-middle attacks.

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

"Host key verification failed" means that the SSH client cannot verify the authenticity of the remote host's public key, which can be caused by several reasons such as the remote host's key has changed, or a man-in-the-middle attack is being attempted. To resolve this issue, you can manually remove the old key from the known_hosts file and reconnect to the remote host to obtain the new key and store it in the known_hosts file.

5. **What is the difference between SSH and SSL?**
   SSH and SSL are both protocols used for secure communication over a network, but they differ in several ways. SSH is primarily used for secure remote access to a system, whereas SSL is used for secure communication between a client and a server, typically used in web applications. SSH uses public-key cryptography to authenticate the client and server and to establish a secure channel, whereas SSL uses a combination of public-key and symmetric-key cryptography. Additionally, SSH runs on port 22 by default, while SSL typically uses ports 443 or 8443.

6. **What ssh-keygen is used for?**
   ssh-keygen is a command-line tool used to generate, manage, and convert SSH keys. It can be used to create public and private key pairs, convert keys between different formats, and manage authorized keys for SSH authentication.

7. **What is SSH port forwarding?**
   SSH port forwarding, also known as SSH tunneling, is a feature of SSH that allows a user to securely forward network traffic from one system to another over an encrypted SSH connection. This can be used to encrypt and secure communication between two systems, or to access services on a remote system that are not directly accessible over the network. SSH port forwarding can be configured to forward traffic to specific ports on specific systems, and can be used with TCP, UDP, or even UNIX sockets.