CS 3307-01 Operating Systems 2

Instructor: Professor Olawale Omoyeni

Name: Anonymos

Written Assignment Unit 3

1. **Introduction**

This document aims to help junior developers understand the fundamental concepts and mechanisms of file systems, prompts, remote procedure calls (RPC), and Unix/Linux security. These concepts are critical for ensuring efficient operations and secure environments when working with Unix/Linux systems. Mastery of these topics not only enhances development efficiency but also ensures robust and secure operations in real-world development environments.

The target audience includes junior developers who are beginning to work with Unix/Linux systems. This document combines theoretical explanations with practical examples to deepen understanding.

The following sections cover these topics in detail:

* Basic concepts and functions of file systems
* The role and customization of prompts
* The mechanism of RPC
* Unix/Linux security

1. **File Systems**

**2.1 Definition and Basic Concepts**

A file system is a mechanism that an operating system uses to store and manage data. It creates a hierarchical structure of files and directories (folders) that allows efficient access to data on storage devices. In Linux systems, the file system is represented as a single tree structure, and external storage devices or network drives are mounted into this tree.

For example, Linux uses the /etc/fstab file to manage automatic device mounting settings. This file defines which devices are mounted at boot time and where, streamlining device management and ensuring reliability during the boot process (Shotts, 2019). Linux supports various file system types, such as ext4, xfs, and btrfs, which offer unique features like journaling and snapshot capabilities.

**2.2 Key Functions**

The primary functions of a file system include:

* Data Storage: Managing the hierarchical structure of files and directories for efficient data storage.
* Access Control: Setting permissions (read, write, execute) to manage user-specific access rights.
* Recovery Tools: Providing tools like fsck to detect and repair file system errors (Lavarian, 2022).

**2.3 File System Operations**

2.3.1 Displaying File Systems

Commands like mount or df -h can display the current file systems. Below is an example output of df -h:

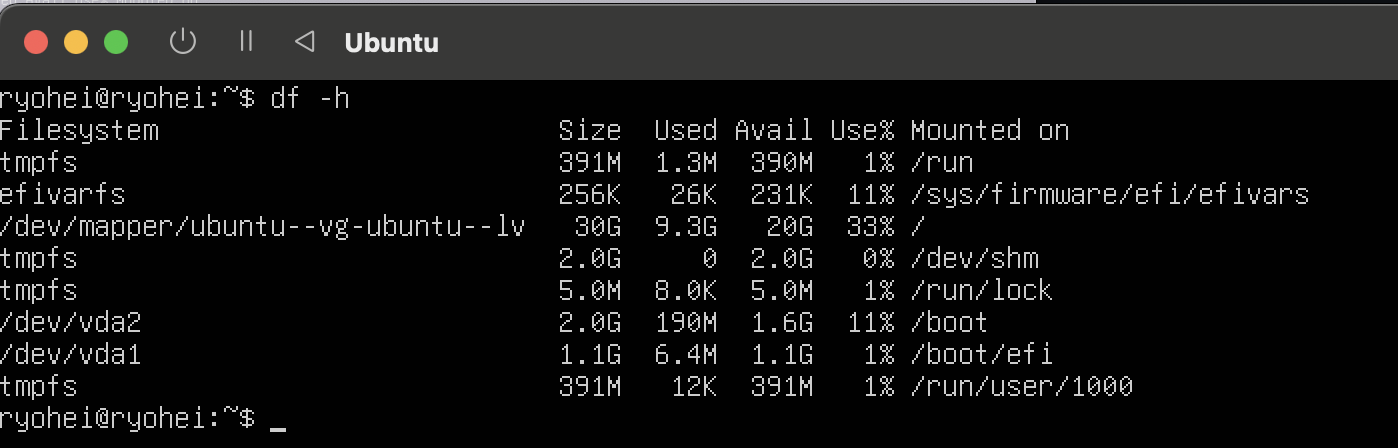


Figure 2.3.1-1 output of `df -h`

2.3.2 Mounting and Unmounting Devices

When connecting new devices, such as USB drives, to the OS, a mounting operation is required. For instance:

sudo mount /dev/sda4 /mnt

To remove a mounted device:

sudo umount /mnt

2.3.3 Creating a File System

To create a file system on a new partition, use the mkfs command:

sudo mkfs.ext4 /dev/sda4

1. **Prompts**

**3.1 Definition and Role**

A prompt is a text-based indicator that provides an interface for users to input commands in the command-line interface (CLI). In Linux, Bash (Bourne Again Shell) is commonly used by default, and prompts typically take the following format:

[username@hostname currentDirectory]$

Prompts provide basic information that helps users understand the current state of the computer and execute commands effectively (Shotts, 2019).

**3.2 Key Operations**

The following basic operations can be executed using a prompt:

3.2.1 Checking the Current Directory (`pwd`)

Use pwd to display the current working directory.

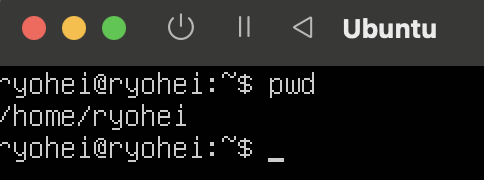


Figure 3.2.1-1 output of `pwd`

3.2.2 Navigating Between Directories (`cd`)

Use cd /path/to/directory to move to another directory.

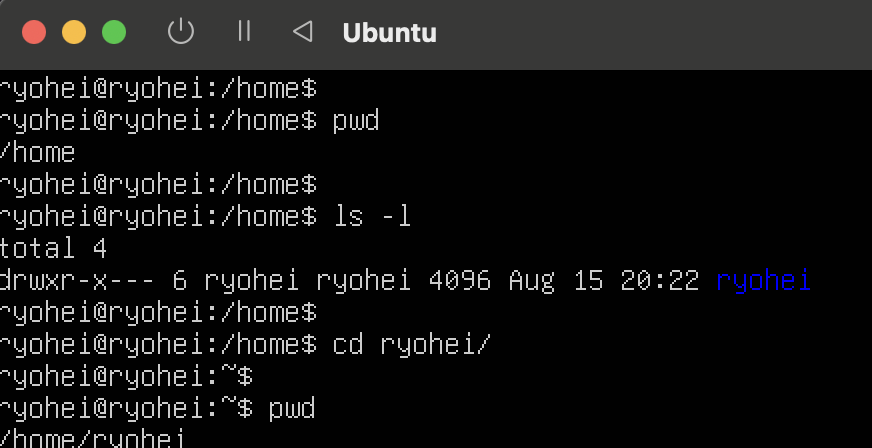


Figure 3.2.2-1 output of `cd`

3.2.3 Displaying Files and Directories (`ls`)

Use ls or ls -l to list files and directories in the current directory.

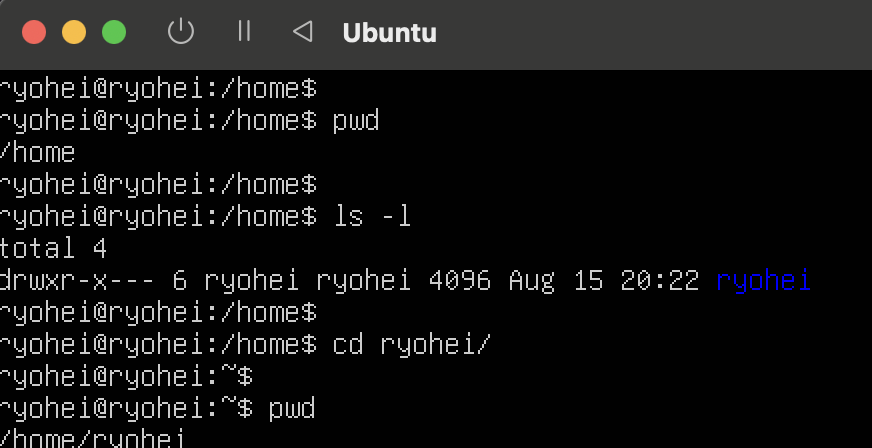


Figure 3.2.3-1 output of `ls`

**3.3 Customizing Prompts**

Prompt customization can enhance work efficiency. In Bash, you can change the prompt by setting the PS1 variable. For example:

PS1="[\u@\h \W]\$ "

The following is an example of a prompt that has actually been customised:

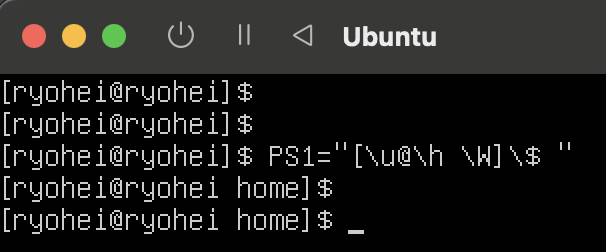


Figure 3.3-1 output of customizing prompt

To make this change permanent, add the setting to the `.bashrc` file.

1. **Remote Procedure Calls**

**4.1 Definition and Basic Concepts**

A Remote Procedure Call (RPC) is a technology that enables communication between processes on different computers over a network. RPC allows a remote server's function to be executed with the same ease as a local function call, simplifying distributed application design (GeeksforGeeks, 2022).

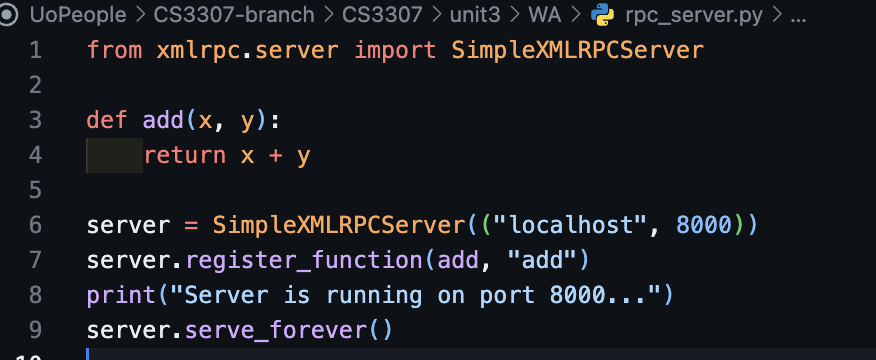
RPC Process:

1. The client program calls the remote procedure.
2. The RPC library converts the call into a network message and sends it to the remote server.
3. The server program receives the message and executes the specified procedure.
4. The execution result is returned to the client as a network message.

**4.2 Example Processes and Use Cases**

4.2.1 Preparing the Server Program

Define and prepare a remote executable procedure. For example, using Python's `xmlrpc.server` library:



4.2.2 Creating the Client Program

Here is a simple client program to call the remote procedure:



4.2.3 Verifying Communication

Keep the server program running and connect via the client. The client program will display the result of calling the remote procedure.



Figure 4.2.3-1 example of RPC execution

**4.3 Advantages and Disadvantages**

Advantages:

* Abstracts network details to improve development efficiency.
* Simplifies distributed system design.

Disadvantages:

* Potential performance degradation due to network delays.
* Risk of system-wide impact from network failures.

1. **Unix/Linux Security**

**5.1 Definition and Importance**

Unix/Linux security is essential for ensuring system stability and data protection. Security features include user and group management, file permission settings, and privilege escalation (sudo). These measures help prevent unauthorized access and mitigate significant system damage (Shotts, 2019).

**5.2 Key Security Features**

5.2.1 User and Group Management:

Manage users and groups using /etc/passwd and /etc/group.

5.2.2 File Permission Settings:

Use commands like chmod to modify permissions.

5.2.3 Privilege Escalation:

Grant temporary administrative privileges using sudo. Edit permissions via:

sudo visudo

5.2.4 Security Tools:

Configure firewalls using:

sudo ufw allow 22

**Conclusion**

This document introduced foundational knowledge for understanding critical technical concepts in Unix/Linux systems. By applying the information provided, readers can enhance the efficiency and security of their system operations. Hands-on practice with real Linux systems will further improve these skills.

Word Count: 916

References

1. GeeksforGeeks. (2022). *Remote procedure call (RPC) in operating system.* Retrieved March 7, 2023, from https://www.geeksforgeeks.org/remote-procedure-call-rpc-in-operating-system/
2. Lavarian, R. (2022). *What is a file system? Types of computer file systems and how they work – explained with examples.* FreeCodeCamp. Retrieved from https://www.freecodecamp.org/news/file-systems-architecture-explained/
3. Shotts, W. (2019). *The Linux command line: A complete introduction* (5th ed.). No Starch Press.