Introduction to Linux

3rd semester @ Erhvervsakademi København

GitHub

Goal:

Be comfortable using Linux commands and using the terminal for development tasks and server management.

What is Linux?

> Linux is an open-source operating system kernel that serves as the foundation for various operating systems, commonly referred to as Linux distributions. It was created by Linus Torvalds in 1991 and has since become one of the most widely used operating systems in the world, particularly in server environments, embedded systems, and increasingly on desktops.

Terminal vs. GUI

> Using GUI is user-friendly but can be limited for certain tasks.

> Using the terminal can be more efficient and powerful, but has a steeper learning curve.

Terminal, CLI and Shell

> You will often here the term "terminal", CLI (Command Line Interface) or "shell" used interchangeably.

> A terminal is a **text-based interface** that allows you to **interact with the operating system** by typing commands. It **does not interpret the commands** but passes them to the shell for execution.

> **The shell is a program that takes the commands you type in the terminal and executes them**. There exist **several different shells**, such as `sh` (Bourne shell), `bash` (Bourne Again SHell) and `zsh` (Z shell). Each shell has its own features and syntax, but they all serve the same basic purpose of interpreting and executing commands.

Running a Linux docker container

> You can run a Linux docker container using the `docker run` command. For example, to run an Ubuntu container, you can use:

```
docker run -it ubuntu bash
```

winpty on windows: If you're using Docker on Windows, you may need to use `winpty` to run interactive commands in the terminal. For example:

```
winpty docker run -it ubuntu bash
```

Navigating the filesystem

> The terminal allows you to navigate the filesystem using commands like `cd` (change directory), `ls` (list files), and `pwd` (print working directory).

```
cd /path/to/directory  # Change to a specific directory
ls                     # List files and directories
pwd                    # Print the current directory
```

Managing files and directories

> You can create, move, copy, and delete files and directories using commands like `mkdir` (make directory), `touch` (create an empty file), `mv` (move or rename), `cp` (copy), and `rm` (remove).

```
mkdir new_directory   # Create a new directory
touch new_file.txt    # Create an empty file
mv old_name.txt new_name.txt  # Rename a file
cp file.txt copy_of_file.txt   # Copy a file
rm file.txt           # Remove a file
rm -r directory_name  # Remove a directory and its contents
```

Information about file content

> You can view the content of files using commands like `cat` (concatenate and display), `less` (view file content one page at a time), and `head` / `tail` (view the beginning or end of a file).

```
echo "Hello, World!" > file.txt  # Create a file with some content
touch file.txt                   # Create an empty file
cat file.txt                     # Display the content of a file
less file.txt                    # View file content one page at a time
head file.txt                    # View the first 10 lines of a file
tail file.txt                    # View the last 10 lines of a file
```

File permissions

> Linux uses a permission system to control access to files and directories. Each file has three types of permissions: read (r), write (w), and execute (x). These permissions can be set for the owner of the file, the group, and others. You can view and modify permissions using the `ls -l` command to view permissions and `chmod` to change them.

```
-rw-r--r-- 1 user group 0 Jan 1 00:00 file.txt
```

- User (owner) has read and write permissions (`rw-`)
- Group has read permissions (`r--`)
- Others have read permissions (`r--`)

Changing file permissions with `chmod`

> The `chmod` command allows you to change the permissions of a file or directory.

```
ls -l file.txt          # View file permissions
chmod 755 file.txt      # Set permissions to rwxr-xr-x
chmod 644 file.txt      # Set permissions to rw-r--r--
```

- `1` - execute permission
- `2` - write permission
- `4` - read permission
- `7` - read, write, and execute permissions (4 + 2 + 1)

Bitmasks and file permissions

> A bitmask is a way to represent some set of options or permissions using a single integer value.

Bits: Each permission (read, write, execute) can be represented as a bit in a binary number. For example:

- Read (r) = 4 (100 in binary)
- Write (w) = 2 (010 in binary)
- Execute (x) = 1 (001 in binary)

Combining permissions: To set multiple permissions, you can add their values together. For example:

- Read and write (rw-) = 4 + 2 = 6 (110 in binary)

Editing files

> You can edit files using command-line text editors like `nano` or `vim`.

```
nano file.txt  # Open file.txt in the nano text editor
```

- `nano` is a simple, user-friendly text editor that is easy to learn for beginners
- `vim` is a more powerful and feature-rich text editor, but has a steeper learning curve.

```
vim file.txt   # Open file.txt in the vim text editor
```

- Exiting vim: Press `Esc`, then type `:q` to quit, `:wq` to save and quit, or `:q!` to quit without saving.

Searching for files and content

> You can search for files and content using commands like `find` (search for files) and `grep` (search for text within files).

```
find /path/to/search -name "filename.txt"  # Search for a file by name
grep "search term" file.txt                # Search for a term within a file
grep -r "search term" /path/to/search      #
```

Environment variables

> Environment variables are a variable that is available in the shell and can affect the behavior of processes and applications. Y

View the current environment variables with `printenv` or `env`, and set new ones with `export`:

```
printenv
```

Networking and remote access

> You can use commands like `ssh` to securely connect to remote servers, and `scp` to copy files between local and remote machines.

```
ssh user@hostname  # Connect to a remote server using SSH
scp file.txt user@hostname:/path/to/destination  # Copy a file to a remote server
```

Package management

> Linux distributions use package managers to install, update, and manage software.
>
> For example, Debian-based distributions use `apt`.

```
sudo apt update              # Update the package list
sudo apt install package_name  # Install a package
sudo apt upgrade             # Upgrade installed packages
```

Installing nginx:

```
sudo apt install nginx
```

`$PATH` and executing commands

> The `$PATH` environment variable is a list of directories that the shell searches for executable files.

```
echo $PATH
```

To add a new directory to your `$PATH`, you can use the `export` command:

```
export PATH=$PATH:/new/directory
```

`.bashrc` and shell configuration

> The `.bashrc` file is a script that is executed whenever a new terminal session is started. You can use it to set environment variables, create aliases, and customize your shell environment.

```
# Example .bashrc content
export PATH=$PATH:/new/directory  # Add a directory to PATH
alias ll='ls -la'                 # Create an alias for a command
```

Basic shell script

> You can create a simple shell script to automate tasks. For example, create a file called `hello.sh` with the following content:

```
#!/bin/bash
echo "Hello, World!"
```

Make the script executable and run it:

```
chmod +x hello.sh  # Make the script executable
./hello.sh         # Run the script
```