|  |  |  |  |
| --- | --- | --- | --- |
| **Übungsprotokoll**  **SYTB – Systemtechnik Betriebssysteme** | | | |
|  | **Übungsdatum:**  25. September 2023 | **Klasse:**  3BHIT | **Name:**  Gabriel Martin |
| **Abgabedatum:**  2. Oktober 2023 | **Gruppe:**  SYTB2 | **Note:** |
| **Leitung:**  Gerald Brandstetter | **Mitübende:**  None | | |
| **Übungsbezeichnung**:  UNIX Filesystem Part 2 / Knottenbelt Lecture 3 | | | |

Contents

[1 Aufgabenstellung 2](#_Toc146527399)

[2 Abstract 2](#_Toc146527400)

[3 Theoretical Foundaftion 3](#_Toc146527401)

[4 Übungsdurchführung 3](#_Toc146527402)

[4.1 File and Directory Permissions 3](#_Toc146527403)

[4.2 Inspecting file Content 4](#_Toc146527404)

[4.3 Finding Files 4](#_Toc146527405)

[4.4 Finding Text in Files 5](#_Toc146527406)

[4.5 File Compression and Archives 5](#_Toc146527407)

[4.6 Handling removable Media 6](#_Toc146527408)

[4.6.1 Mounting Removable Media 6](#_Toc146527409)

[4.6.2 Accessing Removable Media 6](#_Toc146527410)

[4.6.3 Ejecting Removable Media 6](#_Toc146527411)

[4.6.4 Automounting Removable Media 6](#_Toc146527412)

[5 Findings 7](#_Toc146527413)

[6 Code 7](#_Toc146527414)

[7 Comment 7](#_Toc146527415)

# Aufgabenstellung

Arbeiten Sie folgendes Knottenbelt-Kapitel durch:

[Introduction to UNIX and Linux: Lecture 3 (ic.ac.uk)](http://www.doc.ic.ac.uk/~wjk/unixintro/Lecture3.html)

Beantworten Sie danach folgende Fragen:

[Introduction to UNIX and Linux: Exercise Sheet 3 (ic.ac.uk)](http://www.doc.ic.ac.uk/~wjk/unixintro/Exercise3.html)

# Abstract

In this page I cover different parts of the UNIX filesystem, including permissions, finding files and file content, file sorting as well as file compression and archives. We will also look at how to handle removable media in UNIX.

# Theoretical Foundaftion

We will continue using the Virtual Machine created in lecture 1 and extend the knowledge we gained during the first introduction.

Requirements for this lecture:

Basic understanding of the UNIX operating system

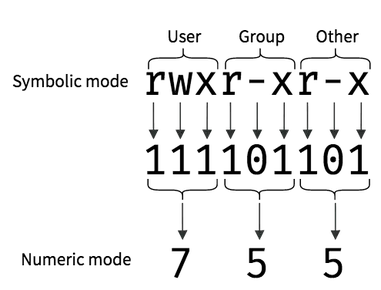
Some knowledge about number systems

Knowledge about regular expressions is recommended.

# Übungsdurchführung

## File and Directory Permissions

As we already saw during lecture one, we can view the permissions of files and directories by running the ls command with the -l option. Now we will go over what these characters actually mean.

All permissions under UNIX are separated in 3 parts: user permissions, group permissions and general permissions. (We can see the group and owner in column 3 and 4)

The group and owner of an entry can be changed with the commands chgrp and chown.

Permissions are stored as 8bit bitmasks that are represented by an octal number (0-7). Therefore, since we have 3 permissions, we can use 3 octal digits to represent the permissions of a file or directory. The second representation is using 3 characters to represent all possible combinations of an octal number, where “r” stands for read, “w” stands for write, “x” stands for execute and “- “stands for a not allowed operation.

We can ignore the first character and the 10-character string outputted by ls -l since it represents the file type and has nothing to do with the permissions. This we are left we the exact 9 characters we talked about above.

|  |  |
| --- | --- |
| --- | 0 |
| --x | 1 |
| -w- | 2 |
| -wx | 3 |
| r-- | 4 |
| r-x | 5 |
| rw- | 6 |
| rwx | 7 |

We can change the permissions of an item using the chmod command (chmod permissions filename).

For example: Lets make a file that can only be written to by the owner user but can be read by everyone.

Using the numeric code: “chmod 644 my\_file.txt”

Using symbolic code: “chmod u=rw,go=r my\_file.txt”

Using “+” and “-“: “chmod u+rw,go+r my\_file.txt”

The executable flag in UNIX is a bit special because UNIX does not care about file extensions the only way for the OS to know if a file is an executable file is the executable flag in the file permissions. Therefore, even the root user can not execute a file without the executable flag being set. This can easily be achieved by “chmod +X my\_executable”.

## Inspecting file Content

We already used the cat command to print out the entire content of a file to the console but there are also more advanced and specialized tools for printing out files.

The “file” command can detect common file types, like C code, HTML markup or simple text files.

We can use the head and tail command with the -l option specifying the number of lines we want to print out the first or last few lines of a file. The tail command also has a -f option that monitors the file and spits out all new content added to the file to the console. This can be useful for inspecting log files.

The objdump can be used to disassemble binary files. We can for example use “objdump -d /bin/ls” do disassemble the binaries of the ls command and view it as assembly instructions. There are also other options like -x to display the headers and -t for a symbol table where we can also see the binary format (in our case this should be elf64-x86-64) but may vary depending on your system.

Then there is the od command to view file contents in a variety of format. For example -x for hexadecimal representation of the data or -c to see all characters on their own including escape characters.

## Finding Files

The simplest way to find a file is using the “find” command with the filename we are looking for. We can also specify the directory to be searched. For example, searching the user directory for all text files: “find ~ -name “\*.txt” -print”. But keep in mind that searching a large file tree using find can take quite a long time.

We can also use the which command to search paths in our PATH environment variable. We can for example use this to find system binaries like ls using” which ls” should return something like “/bin/ls”.

## Finding Text in Files

"grep" is a powerful command-line tool in Linux used for searching text patterns within files. It stands for "Global Regular Expression Print" and is invaluable for quickly locating specific information in files.

The basic syntax of the "grep" command is as follows:

grep [options] pattern [file...]

"pattern": The text or regular expression to search for within specified files.

"file": One or more files in which you want to search for the pattern.

Example Usage: Suppose you have a directory with text files and want to find the word "example" in all of them. You can use "grep" like this: grep "example" file1.txt file2.txt file3.txt

Common Options:

-i or --ignore-case: Make the search case-insensitive. “grep -i "example" file1.txt”

-r or --recursive: Search for the pattern recursively in all files and subdirectories within a directory. “grep -r "example" /path/to/directory”

-l: Display only the names of files containing the pattern. “grep -l "example" \*”

-n: Display line numbers along with matching lines. “grep -n "example" file1.txt”

-v: Invert the search, displaying lines without the pattern. “grep -v "example" file1.txt”

-w: Match whole words only. “grep -w "example" file1.txt”

-E: Enable extended regular expressions for complex patterns. “grep -E "pattern1|pattern2" file1.txt”

-A, -B, -C: Display lines of context around matching lines. “grep -C 2 "example" file1.txt”

## File Compression and Archives

We can use the tar command to put many files including recursive subdirectory content into a single archive file. This is useful for creating backups and sharing multiple files in one. Other than .zip on Windows the tape archiver performs no compression on the archive created. For example: “tar -cf my\_archive.tar [files\_to\_archive]”

We can then use the gzip command to compress a file/archive. For example: “gzip -d my\_archive.tar”. Here we will be left of with a “.tar.gz” file that we see all over the place in the Linux world.

## Handling removable Media

### Mounting Removable Media

Mounting is the process of making the files on removable media accessible to your Linux system. When you plug in a USB drive, it's often automatically mounted, but you can also do it manually. Follow these steps:

Plug in the Media: Insert the USB drive or connect the external hard disk to your computer.

Check Device Status: To see a list of currently connected storage devices and their partitions, open a terminal and run the following command: lsblk

This command lists the devices, their sizes, and their mount points (if mounted).

Mount the Device: If your device is not automatically mounted, you can manually mount it. Create a mount point (a directory where you want the media to appear), and then use the mount command. Replace /dev/sdX with the appropriate device name, and /mnt/usb with your desired mount point: sudo mkdir /mnt/usb, sudo mount /dev/sdX /mnt/usb

Your media is now accessible in the /mnt/usb directory.

### Accessing Removable Media

Once your removable media is mounted, you can access its contents using a file manager or terminal. In a file manager like Nautilus (GNOME), Dolphin (KDE), or Nemo (Cinnamon), the mounted media should appear in the sidebar.

In the terminal, navigate to the mount point you specified earlier: cd /mnt/usb

You can now use standard file commands to interact with the files and folders on the removable media.

### Ejecting Removable Media

Before physically removing a removable device from your computer, it's essential to properly eject it to prevent data corruption. You can use the eject or umount command.

Eject Command: The eject command is used for optical media like CDs or DVDs. Simply specify the device: eject /dev/sr0

Replace /dev/sr0 with your optical drive device.

Unmount Command: For USB drives and external hard disks, you should use the umount command: sudo umount /mnt/usb

Replace /mnt/usb with the actual mount point of your device.

### Automounting Removable Media

If you prefer automatic mounting and unmounting of removable media, many Linux desktop environments provide this feature by default. You can plug in your USB drive, and it should appear on your desktop or in your file manager automatically. To disable or customize this behavior, consult your desktop environment's settings.

# Findings

Even though the UNIX filesystem might seem overwhelming, once you get to know the basics it suddenly seems a lot easier. After working with the UNIX filesystem for a few months I already consider myself an experienced user because most things are very simple under the hood once you start using them day by day.

# Code

None.

# Comment

I hope this paper provided a good overview of different parts of the UNIX filesystem, including permissions, finding files and file content, file sorting as well as file compression and archives. Now you should also have a solid understanding on how to handle removable media in Linux.