

Inhoud

Hoofdstuk 3: Beheren van Bestanden, Mappen en Tekst (NL)	2
Chapter 3: Managing Files, Directories, and Tekst (EN)	7

Hoofdstuk 3: Beheren van Bestanden, Mappen en Tekst (NL)

In dit hoofdstuk worden de essentiële vaardigheden voor het beheren van bestanden en mappen op een Linux-systeem behandeld. Het benadrukt het belang van de commandoregelinterface en biedt een overzicht van de meest gebruikte commando's die je nodig hebt voor dagelijks gebruik.

1. Bestanden en Mappen Beheren

- **Virtuele Directorystructuur:** In Linux worden bestanden opgeslagen in een hiërarchische structuur die begint bij de rootdirectory (/). Deze structuur maakt het mogelijk om bestanden georganiseerd en toegankelijk te houden.
- **Navigeren in de Directory:**
 - Het `cd`-commando (change directory) wordt gebruikt om tussen directories te navigeren. Bijvoorbeeld, om naar de map Documents te gaan, typ je:
`cd ~/Documents`
 - Om naar de bovenliggende directory te gaan, gebruik je:
`cd ..`

2. Bestanden Bekijken en Aanmaken

- **Bestanden Bekijken:**
 - Het `ls`-commando toont de bestanden en mappen in de huidige directory. Het kan worden uitgebreid met verschillende opties:
`ls -l`
Deze optie toont gedetailleerde informatie over elk bestand, zoals machtigingen, eigenaar, grootte en wijzigingsdatum.

Opties voor het `ls`-commando:

- `-l` | Lange lijstweergave met details
- `-a` | Toont ook verborgen bestanden
- `-h` | Toont bestandsgrootte in leesbare vorm (bijv. KB, MB)
- **Bestanden Aanmaken:**
 - Het `touch`-commando maakt een nieuw, leeg bestand aan. Bijvoorbeeld:
`touch nieuw_bestand.txt`
 - Om een nieuwe map aan te maken, gebruik je het `mkdir`-commando:
`mkdir nieuwe_map`

3. Bestanden Kopiëren en Verplaatsen

- **Kopiëren van Bestanden:**

- Het cp-commando wordt gebruikt om bestanden te kopiëren. Bijvoorbeeld, om bestand.txt te kopiëren naar kopie_bestand.txt:
cp bestand.txt kopie_bestand.txt
- Voor het recursief kopiëren van een map gebruik je de -R optie:
cp -R map_naam/ nieuwe_map/

Opties voor het cp-commando:

- -R | Recursief kopiëren van mappen
- -i | Vraagt om bevestiging voordat bestanden worden overschreven
- -u | Kopieert alleen als de bron nieuwer is dan de bestemming
- **Verplaatsen van Bestanden:**
 - Het mv-commando verplaatst bestanden en mappen. Het kan ook worden gebruikt om een bestand te hernoemen:
mv oud_bestand.txt nieuw_bestand.txt

Opties voor het mv-commando:

- -i | Vraagt om bevestiging voordat bestanden worden overschreven

4. Bestanden Verwijderen

- **Verwijderen van Bestanden:**

- Met het rm-commando kun je bestanden verwijderen. Wees voorzichtig, want dit is permanent:
rm bestand.txt
- Voor het verwijderen van een map die niet leeg is, gebruik je:
rm -r map_naam/

Opties voor het rm-commando:

- -r | Recursief verwijderen van mappen
- -i | Vraagt om bevestiging voordat bestanden worden verwijderd
- -f | Dwingt het verwijderen zonder bevestiging
- **Lege Mappen Verwijderen:**
 - Het rmdir-commando kan worden gebruikt om lege mappen te verwijderen:
rmdir lege_map/

5. Koppelingen van Bestanden en Mappen

- **Hard Links:**

- Een hard link is een andere naam voor hetzelfde bestand en deelt dezelfde inode. Om een hard link te maken:
In bestand_bestand.txt hard_link.txt

```
$ touch OriginalFile.txt
$
$ ls OriginalFile.txt
$
$ ln OriginalFile.txt HardLinkFile.txt
$
$ ls HardLinkFile.txt OriginalFile.txt
$
$ ls -li 2101459 HardLinkFile.txt 2101459 OriginalFile.txt
```

Als je een gelinkt bestand wilt verwijderen, maar niet het originele bestand, gebruik dan het commando `unlink`. Typ simpelweg `unlink` op de commandoregel en voeg de naam van het gelinkte bestand als argument toe.

Bij het maken en gebruiken van harde links zijn er een paar belangrijke zaken om te onthouden:

- Het originele bestand moet bestaan voordat je het `ln` commando uitvoert.
- Het tweede bestand dat in het `ln` commando wordt genoemd, mag niet bestaan voordat je het commando uitvoert.
- Een origineel bestand en zijn harde links delen hetzelfde inode-nummer.
- Een origineel bestand en zijn harde links delen dezelfde data.
- Een origineel bestand en al zijn harde links kunnen in verschillende directories staan.
- Een origineel bestand en zijn harde links moeten op hetzelfde bestandssysteem staan.

- **Symbolische Links:**

- Een symbolische link is een verwijzing naar een bestand op een andere locatie. Maak een symbolische link met:

`ln -s /pad/naar/bestand.txt symbolische_link.txt`

```
$ touch OriginalSFile.txt
```

```
$
```

```
$ ls OriginalSFile.txt
```

```
$
```

```
$ ln -s OriginalSFile.txt SoftLinkFile.txt
```

```
$
```

```
$ ls -li 2101456 OriginalSFile.txt 2101468 SoftLinkFile.txt
```

```
$
```

```
$ ls -og total 0 -rw-rw-r--. 1 0 Aug 24 19:04 OriginalSFile.txt lrwxrwxrwx. 1 17 Aug 24 19:04 SoftLinkFile.txt -> OriginalSFile.txt $
```

Als je een gelinkt bestand wilt verwijderen, maar niet het originele bestand, gebruik dan het commando `unlink`. Typ simpelweg `unlink` op de commandoregel en voeg de naam van het gelinkte bestand als argument toe.

Bij het maken en gebruiken van harde links zijn er een paar belangrijke zaken om te onthouden:

- Het originele bestand moet bestaan voordat je het `ln` commando uitvoert.
- Het tweede bestand dat in het `ln` commando wordt genoemd, mag niet bestaan voordat je het commando uitvoert.
- Een origineel bestand en zijn harde links delen hetzelfde inode-nummer.
- Een origineel bestand en zijn harde links delen dezelfde data.
- Een origineel bestand en al zijn harde links kunnen in verschillende directories staan.
- Een origineel bestand en zijn harde links moeten op hetzelfde bestandssysteem staan.

6. Bestanden Lezen

- **Inhoud van Bestanden Bekijken:**

- Met het cat-commando kun je de inhoud van een bestand bekijken:
cat bestand.txt
- Voor het bekijken van de eerste of laatste lijnen van een bestand, gebruik head of tail:
head bestand.txt
tail bestand.txt

Opties voor het head en tail-commando:

- -n | Aantal regels weergeven (bijv. head -n 5 bestand.txt)
- **Pagineren van Inhoud:**
 - Als het bestand te groot is, gebruik dan less of more om door de inhoud te bladeren:
less bestand.txt

7. Informatie Vinden

- **Metadata van Bestanden:**

- Het file-commando toont het type bestand:
file bestand.txt
- Het stat-commando geeft gedetailleerde informatie over een bestand, zoals laatst gewijzigd, grootte en toegangsmachtigingen:
stat bestand.txt

- **Bestanden Vergelijken:**

- Gebruik het diff-commando om de verschillen tussen twee bestanden te bekijken:
diff bestand1.txt bestand2.txt

- **Bestanden Vinden:**

- Het which-commando toont het pad naar een uitvoerbaar bestand:
which python
- Met whereis kun je andere gerelateerde bestanden vinden:
whereis python
- Het locate-commando, dat snel bestanden doorzoekt, vereist dat de database up-to-date is:
locate bestand.txt
 - Zorg ervoor dat de locate-database up-to-date is door regelmatig updatedb uit te voeren.

- Het find-commando zoekt naar bestanden en mappen binnen een specifieke directory en subdirectories, met de mogelijkheid om te filteren op naam, grootte, datum, enz.:
`find /pad/naar/directory -name bestand.txt`

Belangrijke Leerdoelen

- **Effectief Bestandsbeheer:** Dit hoofdstuk leert je hoe je bestanden en mappen op een efficiënte manier kunt beheren met behulp van verschillende commando's.
- **Inzicht in Koppelingen:** Je leert het verschil tussen hard en symbolische links en hoe deze kunnen worden gebruikt om bestanden te organiseren.
- **Bestanden Lezen en Informatie Vinden:** Het hoofdstuk biedt handvatten voor het lezen van bestanden en het vinden van belangrijke informatie over bestanden en systemen.

Dit hoofdstuk is essentieel voor iedereen die met Linux werkt, of het nu voor persoonlijke projecten of professionele doeleinden is. Het helpt je om de basisvaardigheden te ontwikkelen die nodig zijn voor het efficiënt beheren van een Linux-omgeving, en is bovendien nuttig voor het behalen van de CompTIA Linux+ certificering.

Chapter 3: Managing Files, Directories, and Tekst (EN)

This chapter covers the essential skills for managing files and directories on a Linux system. It emphasizes the importance of the command line interface and provides an overview of the most commonly used commands needed for daily use.

1. Managing Files and Directories

- **Virtual Directory Structure:** In Linux, files are stored in a hierarchical structure that starts at the root directory (/). This structure allows for organized and accessible file management.
- **Navigating the Directory:**
 - The `cd` command (change directory) is used to navigate between directories. For example, to go to the Documents folder, type:
`cd ~/Documents`
 - To go to the parent directory, use:
`cd ..`

2. Viewing and Creating Files

- **Viewing Files:**

- The ls command displays the files and directories in the current directory. It can be extended with various options:

ls -l

This option shows detailed information about each file, such as permissions, owner, size, and modification date.

Options for the ls command:

- -l | Long listing format with details
- -a | Also shows hidden files
- -h | Displays file size in a human-readable format (e.g., KB, MB)

- **Creating Files:**

- The touch command creates a new, empty file. For example:
touch new_file.txt
- To create a new directory, use the mkdir command:
mkdir new_directory
-

3. Copying and Moving Files

- **Copying Files:**

- The cp command is used to copy files. For example, to copy file.txt to copy_file.txt:
cp file.txt copy_file.txt
- To recursively copy a directory, use the -R option:
cp -R directory_name/ new_directory/

Options for the cp command:

- -R | Recursively copy directories
- -i | Prompts for confirmation before overwriting files
- -u | Copies only if the source is newer than the destination

- **Moving Files:**

- The mv command moves files and directories. It can also be used to rename a file:
mv old_file.txt new_file.txt

Options for the mv command:

- -i | Prompts for confirmation before overwriting files

4. Deleting Files

- **Deleting Files:**
 - The `rm` command allows you to delete files. Be careful, as this is permanent:
`rm file.txt`
 - To remove a non-empty directory, use:
`rm -r directory_name/`

Options for the `rm` command:

- `-r` | Recursively remove directories
- `-i` | Prompts for confirmation before deleting files
- `-f` | Forces deletion without confirmation
- **Removing Empty Directories:**
 - The `rmdir` command can be used to remove empty directories:
`rmdir empty_directory/`

5. Links to Files and Directories

- **Hard Links:**
 - A hard link is another name for the same file and shares the same inode. To create a hard link:
`ln existing_file.txt hard_link.txt`

```
$ touch OriginalFile.txt $
$ ls OriginalFile.txt $
$ ln OriginalFile.txt HardLinkFile.txt $
$ ls HardLinkFile.txt OriginalFile.txt $
$ ls -li 2101459 HardLinkFile.txt 2101459 OriginalFile.txt
```

If you want to remove a linked file but not the original file, use the `unlink` command. Just type `unlink` at the command line and include the linked filename as an argument.

When creating and using hard links, there are a few important things to remember:

- The original file must exist before you issue the `ln` command.
- The second file listed in the `ln` command must not exist prior to issuing the command.
- An original file and its hard links share the same inode number.
- An original file and its hard links share the same data.
- An original file and any of its hard links can exist in different directories.
- An original file and its hard links must exist on the same file system.

- **Symbolic Links:**

- A symbolic link is a reference to a file in another location. Create a symbolic link with:
`ln -s /path/to/file.txt symbolic_link.txt`

```
$ touch OriginalFile.txt
```

```
$
```

```
$ ls OriginalFile.txt
```

```
$
```

```
$ ln -s OriginalFile.txt SoftLinkFile.txt
```

```
$
```

```
$ ls -li 2101456 OriginalFile.txt 2101468 SoftLinkFile.txt
```

```
$
```

```
$ ls -og total 0 -rw-rw-r--. 1 0 Aug 24 19:04 OriginalFile.txt lrwxrwxrwx. 1 17 Aug 24 19:04  
SoftLinkFile.txt -> OriginalFile.txt $
```

Use the `unlink` command. Simply type `unlink` on the command line followed by the name of the linked file you want to remove.

Important points to remember when creating and using hard links:

- The original file must exist before you create a hard link to it using the `ln` command.
- The name you give to the new hard link file must not already exist before you run the `ln` command.
- The original file and all its hard links share the same inode number, which is a unique identifier within the file system.
- Because they share the same inode number, the original file and its hard links all point to the exact same data on the disk. Any changes made to the data through any of the links will be reflected in all the other links and the original file.
- Hard links can be located in different directories, but they must all reside within the same file system.

6. Reading Files

- **Viewing File Contents:**

- The `cat` command allows you to view the contents of a file:
`cat file.txt`
- To view the first or last lines of a file, use `head` or `tail`:
`head file.txt`
`tail file.txt`

Options for the `head` and `tail` commands:

- `-n` | Number of lines to display (e.g., `head -n 5 file.txt`)
- **Paging Through Content:**
 - If the file is too large, use `less` or `more` to paginate through the content:
`less file.txt`

7. Finding Information

- **File Metadata:**
 - The file command shows the type of a file:
file file.txt
 - The stat command provides detailed information about a file, such as last modified date, size, and access permissions:
stat file.txt
- **Comparing Files:**
 - Use the diff command to view the differences between two files:
diff file1.txt file2.txt
- **Finding Files:**
 - The which command shows the path to an executable file:
which python
 - Use whereis to find other related files:
whereis python
 - The locate command, which quickly searches for files, requires the database to be up-to-date:
locate file.txt
 - Ensure the locate database is up-to-date by regularly running updatedb.
 - The find command searches for files and directories within a specific directory and subdirectories, with the ability to filter by name, size, date, etc.: find
/path/to/directory -name filename.txt

Important Learning Objectives

- **Effective File Management:** This chapter teaches you how to efficiently manage files and directories using various commands.
- **Understanding Links:** You will learn the difference between hard and symbolic links and how they can be used to organize files.
- **Reading Files and Finding Information:** The chapter provides guidelines for reading files and finding important information about files and systems.

This chapter is essential for anyone working with Linux, whether for personal projects or professional purposes. It helps you develop the foundational skills needed to efficiently manage a Linux environment and is also useful for obtaining the CompTIA Linux+ certification.

