Betriebssysteme

Labor 2

Netzwerkprotokolle

- Um eine Verbindung zu einem Unix-Server herzustellen, kann ein Kommunikationsprotokoll verwendet werden (wie telnet oder ssh). Durch diese erhalten wir ein virtuelles Terminal zum Unix-Server.

- **Telnet** ist ein Netzwerkprotokoll, mit dem virtuell auf einen Computer zugegriffen und ein bidirektionaler, kollaborativer und textbasierter Kommunikationskanal zwischen zwei Computern bereitgestellt wird. Es funktioniert mit einem sogenannten virtuellen Terminalverbindungsemulator oder einer abstrakten Instanz einer Verbindung zu einem Computer, wobei Standardprotokolle verwendet werden, um sich wie ein physisches Terminal zu verhalten, das mit einer Maschine verbunden ist.

Netzwerkprotokolle

- **SSH** (Secure Shell) ist ein Netzwerkkommunikationsprotokoll, mit dem zwei Computer kommunizieren und Daten gemeinsam nutzen können. Ein Merkmal von SSH ist, dass die Kommunikation zwischen den beiden Computern verschlüsselt ist, was bedeutet, dass sie für die Verwendung in unsicheren Netzwerken geeignet ist.

- In unserem Netzwerk (*scs.ubbcluj.ro*) wird ausschließlich das SSH-Protokoll verwendet. Der am besten geeignete Verbindungsclient ist **putty**.

FTP Netzwerkprotokoll

- FTP = **File Transfer Protocol**
- auf dieser Weise kann man zwei Computer verbinden und Dateien zwischen ihnen verschieben.

- mit einem FTP-Client können wir die Datei auf einen Server hochladen, herunterladen, löschen, verschieben, umbenennen und kopieren. Wenn Sie Ihre Datei über FTP senden, führen Ihre Dateien hauptsächlich das Hoch- oder Herunterladen vom FTP-Server durch. Wenn Sie die Dateien hochladen, übertragen Sie die Dateien von Ihrem PC auf den Server, und wenn Sie die Datei herunterladen, übertragen Sie die Datei vom Server auf Ihren PC.

FTP Netzwerkprotokoll

- **Vorteile der Verwendung von FTP:**
  - Sie können mehrere Dateien und Ordner übertragen.
  - Die Datenübertragung ist schneller als HTTP.
  - Wenn die Verbindung unterbrochen wird, kann die Übertragung fortgesetzt werden.
  - Die Größe der zu übertragenden Datei ist nicht begrenzt.
- **Nachteile der Verwendung von FTP:**
  - Ist anfällig für Paketerfassung und andere Angriffe. FTP verschlüsselt den Datenverkehr nicht, sodass Daten leicht gelesen werden können, indem die Datenpakete erfasst werden, da sie während der Übertragung im Klartext gesendet werden.

- Die Dateiübertragung zwischen Unix und Windows erfolgt einfach über FTP. Ausgehend von ***Total Commander*** (Ctrl-N) oder ***WinScp***.

UNIX-Befehle

- **UNIX-Befehlsstruktur:**

```
befehl [optionen] [werte]
```

  - `befehl`: ist das erste Wort, das auf der Befehlszeile eingegeben wird, und ist ein Programm
  - `optionen`:
    - `short option`- ein einzelner Buchstabe, dem ein einzelner Bindestrich (−) vorangestellt ist
    - `long option`- ein Wort, dem ein doppelter Bindestrich (−−) vorangestellt ist
  - `werte`: kann zwingend, optional oder nicht vorhanden sein
    - die Abgrenzung der Reihenfolge der Optionen / Argumente erfolgt durch `LEERZEICHEN`
    - bei der Shell wird zwischen Groß- und Kleinschreibung unterschieden `case-sensitive`

UNIX-Befehle

- **Beispiele:**
  - nur der Befehl: `pwd, ls`
  - Befehl + Option: `ls -l, ls -a (ls --all)`
  - Befehl + Wert: `mkdir new_dir, touch new_file`
  - Befehl + Option + Wert: `ls -l /etc, cat -n hello.c`

Befehle

- **Navigationsbefehle:**
  - `pwd (print working directory)`: zeigt das aktuelle Verzeichnis an
  - `ls (list)`: zeigt den Inhalt des aktuellen Verzeichnisses an
  - `cd dir (change directories)`: ändert das aktuelle Verzeichnis in das angegebene
    - `cd ..` -> eine Ebene höher als der aktuelle Verzeichnis

- **Befehle zum Umgang mit Verzeichnise:**
  - `mkdir nume_dir (make directory)`: erstellt ein neues Verzeichnis
  - `rmdir nume_dir (remove directory)`: löscht das Verzeichnis, dessen Name als Argument angegeben wird
  - `mv dir_src dir_dest (move)`: schiebt das Verzeichnis *dir_src* in das Verzeichnis *dir_dest*
  - `cp dir_src dir_dest (copy)`: kopiert das Verzeichnis *dir_src* in das Verzeichnis *dir_dest*

Befehle

- **Befehle zum Umgang mit Dateien:**
  - `touch` *`nume_fisier`*: erstellt eine neue Datei, die keinen Inhalt hat (ist leer)
  - `cp` *`fis_src fis_dest`*: kopiert die Datei *`fis_src`* in die Datei *`fis_dest`*
  - `mv` *`fis_src fis_dest`*: schiebt die Datei *`fis_src`* in die Datei *`fis_dest`*
  - `cat` *`nume_fisier`*: zeigt den Inhalt der angegebenen Datei (als Argument) an
- **Andere Befehle:**
  - `help, history, clear, cut, file, grep, head, less, more, sort, tail, wc, who, whoami, users, passwd`
- **Sondertasten:**
  - `TAB` - automatische Vervollständigung der Kommandozeile (autocompletion)
  - `Pfeil nach oben` oder `Pfeil nach unten` - Navigation in der Befehlhistorie

Tastenkombinationen

- `Ctrl-C`: stoppt die Ausführung des aktuell ausgeführten Programms
- `Ctrl-Z`: unterbricht das aktuell laufende Programm
- `Ctrl-D`: loggt sich aus (in manchen Fällen gleichbedeutend mit EOF)

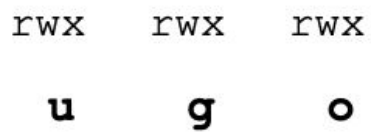
- `Ctrl-S`: sperrt die Konsole
- `Ctrl-Q`: entsperrt die Konsole

- `Ctrl-K`: schneidet den Text von der aktuellen Position bis zum Ende der Befehlszeile ab
- `Ctrl-Y`: fügt den zuvor zugeschnittenen Text ein
- `Ctrl-R`: sucht in die Befehlhistorie durch

- `Ctrl-A`: bewegt den Cursor an den Anfang der Befehlszeile
- `Ctrl-B`: bewegt den Cursor ein Zeichen zurück
- `Ctrl-F`: bewegt den Cursor um ein Zeichen vorwärts
- `Ctrl-E`: bewegt den Cursor an das Ende der Befehlszeile

Zugriffsrechte

- Zugriffsrechte (Berechtigungen) werden für jede Datei oder jedes Verzeichnis festgelegt (*permissions*)
- Berechtigungen für Dateien/Verzeichnisse anzeigen: `ls -l`



- symbolische Darstellung von Rollen:
  - **u (owner)** - der Eigentümer der Datei/des Verzeichnisses
  - **g (group)** - die Gruppe, zu der der Eigentümer der Datei/des Verzeichnisses gehört
  - **o (others)** - andere Benutzer (die nicht Teil der Gruppe sind, zu der der Eigentümer gehört)

Zugriffsrechte

- Symbolische Darstellung von Dateiberechtigungen:
  - **`r (read)`** - gewährt die Möglichkeit, den Inhalt der Datei zu untersuchen (anzuzeigen)
  - **`w (write)`** - gewährt die Möglichkeit, den Inhalt der Datei zu ändern oder zu entfernen
  - **`x (execute)`** - gewährt die Möglichkeit, die Datei als Programm auszuführen

- Symbolische Darstellung von Verzeichnisberechtigungen:
  - **`r (read)`** - gewährt die Möglichkeit, sich die Dateien/Unterverzeichnisse innerhalb des Verzeichnisses anzusehen
  - **`w (write)`** - gewährt die Möglichkeit, Dateien/Unterverzeichnisse aus dem Verzeichnis hinzuzufügen/zu löschen
  - **`x (execute)`** - gewährt die Möglichkeit, das Verzeichnis zu betreten (durchqueren)

Zugriffsrechte

- Numerische Darstellung der Zugriffsrechte:
  - **r (read)** = 4
  - **w (write)** = 2
  - **x (execute)** = 1

- Zugriffsrechte ändern:

```
chmod +x file
chmod 755 file
chmod 644 file
chmod g+r file
chmod u+rw, g+r-w, g+r file
chmod u=rwx, g=rw, o=r file
```

Texteditoren

- können zum Bearbeiten von Textdateien, Schreiben der Code, Aktualisieren von Benutzeranweisungsdateien usw. verwendet werden.
- Das Linux-System unterstützt mehrere Texteditoren.
- Beispiele:
    - `nano`
    - `joe`
    - `vi`
    - `mcedit`
    - `pico`
- Es ist kein spezieller Texteditor erforderlich. Wählt euch, was eurer Meinung nach am besten passt.

Standard-Streams-Umleitung.

- Die UNIX-Shells verwenden drei Standard-Streams:
  - `0 = standard input (STDIN)`
  - `1 = standard output (STDOUT)`
  - `2 = standard error (STDERR)`

- verwendete Symbole:
  - Umleitung der Standardeingabe: <
  - Umleitung der Standardausgabe/Standardfehler:
    - > (wenn die Datei existiert, wird der Dateiinhalt überschrieben)
    - >> (wenn die Datei vorhanden ist, fügen Sie die Ausgabe an den vorhandenen Dateiinhalt an)

Beispiele

- Standardeingabe umleiten:

```
cat    sort       sort <users.txt
```

- Standardausgabe in eine Datei umleiten:

```
ls -l >list.txt oder ls -l 1>list.txt
ls -l >>list.txt oder ls -l 1>>list.txt
```

- Standardfehler in eine Datei umleiten:

```
ls -l /bonus >error.log oder ls -l /bonus 2>error.log
ls -l /bonus >>error.log oder ls -l /bonus 2>>error.log
```

- Standardausgabe und Standardfehler in dieselbe Datei umleiten:

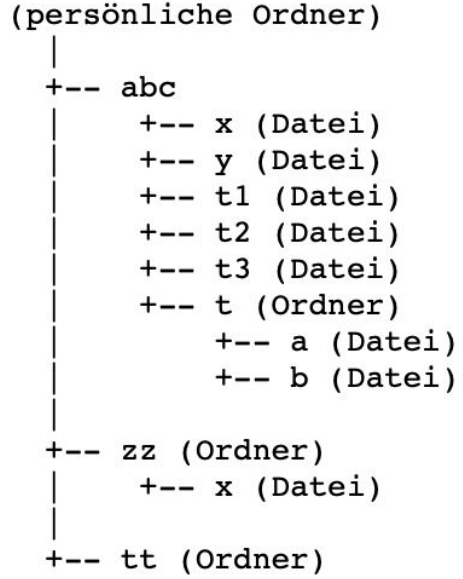
```
ls -l /bonus >output.log 1>&2
ls -l /bonus >>output.log 1>>&2
```

Verbindungsbefehle mit Pipes

```
who | sort
who | wc -l
sort users.txt | head -n 5
sort users.txt | tail -n 5
```

Aufgaben

1. Erstellen Sie in euren persönlichen Ordner die folgende Struktur von Ordner und Dateien:



Aufgaben

2. Kopieren Sie das ***abc***-Verzeichnis mit seinem gesamten Inhalt (rekursiv) als einem Unterverzeichnis von ***zz*** (-> es wird ein Unterverzeichnis/Unterordner ***abc*** in ***zz*** resultieren).

3. Erstellen Sie ein Verzeichnis, für das Sie den Recht **x** (Ausführen) angeben, aber ohne den Recht **r** (Lesen) zu haben. Erstellen Sie eine Datei darin. Was bemerken Sie? Geben Sie danach den Recht **r** und nehmen Sie dem Recht **x**. Was bemerken Sie jetzt?

Aufgaben

4. Geben Sie die richtigen Rechte ein, damit die folgenden Aussagen gleichzeitig erfüllt werden:
   - jeder darf den Inhalt der Ordner `abc` und `abc/t` sehen
   - jeder darf Dateien in `abc/t` hinzufügen
   - jeder darf die Dateien `x, y, t1, t2, t3` von `abc` lesen
   - jeder darf **`nicht`** die Dateien `a` und `b` aus `abc/t` lesen

5. Listen Sie im Langformat `t, t1, t2, t3` von `abc` auf. (Es sollen nur die Zugriffsrechte von `t` sichtbar sein und nicht auf die darin enthaltenen Dateien)

Aufgaben

6. Der Befehl ***cp /dev/zero /dev/null*** ist eine Art Endlosschleife (“unendlicher Zyklus” -> es endet nicht). Starten Sie den Befehl und verschieben Sie er in den Hintergrund. Listen Sie die aktiven Prozesse auf und beenden Sie den Befehl. (in beiden Fällen: im Vordergrund bewegt und mit ^C gestoppt oder mit kill).
   Vorsicht: Der Befehl verbraucht viele Systemressourcen. Lassen Sie es nicht länger laufen, als Sie es versuchen müsst. Beenden Sie es vor dem Abschluss der Sitzung.

7. Erstellen Sie in ***tt*** einen symbolischen Link (Verbindung) mit dem Namen ***c*** zu ***abc***. Entdecken Sie seine Funktionalität durch die Anwendung der **cd**- und **pwd**-Befehle.

Ressourcen

- Filesystem navigation:

  `http://www.ee.surrey.ac.uk/Teaching/Unix/unix1.html`
- Directory/file handling:

  `http://www.ee.surrey.ac.uk/Teaching/Unix/unix2.html`
- Redirecting:

  `http://www.ee.surrey.ac.uk/Teaching/Unix/unix3.html`
- Permissions:

  `http://www.ee.surrey.ac.uk/Teaching/Unix/unix5.html`
- Other UNIX commands:

  `http://www.ee.surrey.ac.uk/Teaching/Unix/unix6.html`