LINUX

- Linux Commands
  - Linux Commands
- Linux Tests
  - 1. Benutzer & Zugriffsrechte
  - 1 b. Zusatzaufgaben zu Linux-Benutzern, sudo, Besitzrechten und Sonderrechten.
  - 2. Gemischt – Benutzer, Zugriffsrechte und Sonderrechte.
- A1_ssh

Linux Commands

Linux Commands

Basics

Ctrl+Shift+T	Terminal Öffnen	**Steuerung Terminal**
Ctrl+C	Abrechen / neue Befehlszeile	
Ctrl+D	Log out	
Ctrl+L	Verlauf löschen	
cd ..	Ordner zurück ⬅ □□	
cd ~	in home wechseln ➡ □□	
touch	Datei erstellen □ □□	
mkdir	Ordner erstellen □ □□	
rm -r	Löschen □ □□	
ls	Liste anzeigen □□+□□	□□ **Anzeigen mit LS**
ls -l /home	anzeigen aller /home Ordner /□□ /□□	
ls -l	Inhalt anzeigen. □□ □□ □□ □□	
ls -la (optional -h = KB, MB)	Inhalt anzeigen. (Rechte,User,Group,Größe,Datum+versteckte) . □□ □□ □□ □□ □□	

tree	tree struktur anzeigen □□└──	□□ **Anzeigen Struktur**
tree -L 1	Zeigt erstes unter level an □□├──	
tree -pug	tree mit □□ -pug(**Zugriffsrecht, User, Gruppe**) □□,□□,□□	
getent passwd	Alle user anzeigen □□□□□□	□□ **Anzeigen User & Group**
getent group	Gruppe anzeigen □□□□□□	
# getent shadow	Passwort-Hashes anzeigen Sudo oder aus Root #□ □□	
# useradd -m -s /bin/bash user	Benutzer erstellen + Home + Bash. □ □□□□ & (erstellt Standart Hauptgruppe mit user Namen)	**Benutzer &** **Gruppen erstellen**
# useradd -m -s /bin/bash -g HAUPT -G NEBENGRUPPE user	Benutzer erstellen + Home + Bash □ □□□□ +HauptG +NebenG □□ (□□)	-g Hauptgruppe -G Nebengruppe (alleinstehend ersetzt alle bisherigen nebengrupen) -aG Nebengruppe adden
# groupadd groupName	neue Gruppe erstellen □□ □	

# usermod -aG NebenGruppe user	User zur Nebengruppe hinzufügen. □□→□□	
# groupmod -n neuerName alterName	Gruppe umbenennen. □□ □□Name	
# chgrp -R gruppe /...	Gruppe ändern + ALLE Daten □□ + □□	**-R = Rekursiv (Alle Dateien)**
# usermod -s /bin/bash user	Bash nachträglich hinzufügen	
{ }	Kaskadieren	
cp	Kopie	**kopieren oder verschieben** Bsp. mv OldName NewName
mv	Verschiebt o. benennt Dateien	
getent passwd	Alle user anzeigen	
getent group	Gruppe anzeigen	
# passwd user	Passwort zuweisen	
su - user	Nutzer wechseln	
# chown user:group /home/OldUser	Besitzer übergeben □□ □□□□ □□	

# chgrp group datei	Ändert nur gruppe □□... □□ → □□	
# chown user:group dateiname	Ändert besitzer & gruppe. □□... □□□□ → □□	
# usermod -l oldUser newUser	Ändern des namen □□□□Name	
fdisk -L	Zeigt alle festplatten und formate an	
crontab -e	Um zeitlich gesteuerte dinge zu starten	

= sudo

CHMOD EINSTELLUNGEN:

- **Löschen hängt vom Ordner ab, nicht von der Datei**
  → benötigt: `w + x` auf dem Ordner
- **Ordnerrechte ≠ Dateirechte**
  → Datei: `r=lesen`, `w=ändern`, `x=ausführen`
  → Ordner: `r=sehen`, `w=erstellen/löschen`, `x=betreten`
- **`x` ist Pflicht für Zugriff**
  → ohne `x`: kein `cd`, kein Öffnen, egal was sonst gesetzt ist
- **`r` ohne `x`**
  → sehen eingeschränkt möglich, aber nicht nutzen
- **`x` ohne `r`**
  → betreten möglich, aber kein `ls` (Blindzugriff)
- **`w` ohne `x`**
  → praktisch nutzlos (kein Zugriff auf den Ordner)
- **Verschieben (`mv`)**
  → benötigt `w + x` auf Quell- UND Zielordner
- **Datei löschen ohne Dateirechte möglich**
  → wenn Ordner `w + x` hat
- **root ignoriert Rechte weitgehend**
  → kann fast alles unabhängig von Permissions
- **`chmod 777`**
  → funktioniert immer, aber unsicher (jeder darf alles)
- **Sticky Bit (`t`) z. B. `/tmp`**
  → alle dürfen schreiben, aber nur eigene Dateien löschen

--

Linux Prompt (Befehlszeile) – komplett kompakt

Grundaufbau

`user@hostname:pfad$`
oder als Admin:
`root@hostname:pfad#`

Bestandteile

- `user` → aktueller Benutzer
  - `user` = normaler Benutzer
  - `root` = Administrator
- `hostname` → Rechner- oder Systemname
  - z. B. `server`, `NASSI`, `raspberrypi`, `container`
- `pfad` → aktueller Ordner
  - `~` = Home-Verzeichnis von `user` = `/home/user`
  - `/` = Root-Verzeichnis, also die oberste Ebene des Systems
  - `/etc`, `/var/log`, `/tmp` = absolute Pfade
  - `..` = eine Ebene zurück
- `$` / `#` → Rechte-Stufe
  - `$` = normaler Benutzer
  - `#` = root / Admin

Beispiele

`user@server:~$`
→ Benutzer `user`, Host `server`, im Home-Verzeichnis, normale Rechte

`root@server:~#`
→ Benutzer `root`, Host `server`, im Home-Verzeichnis von root (`/root`), volle Rechte

`user@server:/home/user/projects$`
→ Benutzer `user`, Host `server`, im Ordner `/home/user/projects`, normale Rechte

`root@server:/etc#`
→ Benutzer `root`, Host `server`, im Ordner `/etc`, volle Rechte

`user@NASSI:/var/log$`
→ Benutzer `user`, Host `NASSI`, im Log-Verzeichnis `/var/log`, normale Rechte

`root@container:/app#`
→ Benutzer `root`, Host `container`, im Ordner `/app`, volle Rechte

`user@raspberrypi:~$`
→ Benutzer `user`, Host `raspberrypi`, im Home-Verzeichnis, normale Rechte

`~/projects $`
→ verkürzter Prompt, nur Pfad sichtbar, aktueller Ordner ist `projects` im Home-Verzeichnis

`$`
→ minimaler Prompt, keine Infos zu Benutzer, Host oder Pfad sichtbar

`user@server:/tmp$`
→ Benutzer `user`, Host `server`, im temporären Ordner `/tmp`

`user@server:/$`
→ Benutzer `user`, Host `server`, im Root-Verzeichnis `/`

Wichtige Bedeutungen

- `~` = Home-Verzeichnis des aktuellen Benutzers
- `/` = oberste Ebene des gesamten Systems
- `..` = ein Verzeichnis zurück
- `user@host` = zeigt, **wer** du bist und **auf welchem System** du bist
- `pfad` = zeigt, **wo** du dich gerade befindest
- `$` oder `#` = zeigt, **mit welchen Rechten** du arbeitest

Typische Prüfungsfallen

- `#` bedeutet nicht Kommentar im Prompt, sondern **root-Rechte**
- `~` ist **nicht** dasselbe wie `/`
  - `~` = Benutzerbereich
  - `/` = gesamtes System
- Der Hostname ist wichtig, weil du damit erkennst, auf welchem Gerät du gerade arbeitest
- Ein Prompt mit `$` ist deutlich ungefährlicher als ein Prompt mit `#`

Merksatz

`user@host = WER und WO`
`pfad = an welcher Stelle im System`
`$ / # = wie viele Rechte du hast`

Linux Tests

1. Benutzer & Zugriffsrechte

Gegeben:

Objekt	Rechte	Eigentümer	Gruppe
/home/	rwx r-x r-x	root	root
/home/anton/	rwx r-x ---	anton	berliner
bz	rw- r-- r--	anton	berliner
caesar.jpg	r-- r-- r--	caesar	roemer
/home/berta/	rwx rwx r-x	berta	berliner
fahrplan	rw- rw- rw-	berta	berliner
BaS	rw- --- ---	anton	berliner
/home/caesar/	rwx r-x r-x	caesar	roemer
urteile.txt	rw- r-- ---	caesar	roemer
kino.pdf	rw- r-- r--	berta	berliner

---

1. Darf berta die Datei bz lesen?

Antwort anzeigen

Ja. Begründung: others hat Leserechte (`r--`).

---

2. Darf caesar die Datei bz lesen?

Antwort anzeigen

Ja. Begründung: others hat Leserechte (`r--`).

---

3. Wer darf die Datei urteile.txt lesen?

Antwort anzeigen

Anton und berta. Begründung: Die Gruppe hat Leserechte (`r--`).

---

4. Was ist zu tun, damit alle Benutzer die Datei urteile.txt lesen können?

Antwort anzeigen

`chmod o+r urteile.txt`

---

5. Darf caesar die Datei kino.pdf verändern?

Antwort anzeigen

Nein. Begründung: Er hat nur Leserechte.

---

6. Darf caesar die Datei kino.pdf löschen?

Antwort anzeigen

Ja. Begründung: Das Löschen hängt von den Rechten des Verzeichnisses ab, nicht von der Datei.

---

7. Darf anton die Datei BaS löschen?

Antwort anzeigen

Ja. Begründung: Entscheidend sind die Rechte des Verzeichnisses.

---

8. Wer darf die Datei fahrplan ändern?

Antwort anzeigen

Anton, berta und caesar. Begründung: Die Datei hat `rw- rw- rw-`.

---

9. Nur berta soll die Datei fahrplan ändern dürfen. Was ist zu tun?

Antwort anzeigen

`chmod 600 fahrplan`

---

10. Wie versperrt caesar den Mitgliedern der Gruppe berliner den Zugriff auf sein Heimatverzeichnis?

Antwort anzeigen

`chmod 700 /home/caesar`

---

11. Die berliner sollen im Heimatverzeichnis von berta keine Ordner und Dateien anlegen dürfen und alle anderen sollen keinen Zugriff haben. Was ist zu tun?

Antwort anzeigen

`chmod 750 /home/berta`

---

12. Welche Rechte müssen gesetzt werden, damit caesar sein Bild caesar.jpg in antons Heimatverzeichnis ablegen kann?

Antwort anzeigen

`/home/anton/  rwx r-x rwx`

Begründung: Zum Ablegen einer Datei in ein Verzeichnis braucht man `w + x`. Caesar ist weder Eigentümer noch in der Gruppe `berliner`, daher müssen diese Rechte bei `others` gesetzt sein.

---

13. Welche Rechte müssen gesetzt werden, damit berta die Datei kino.pdf in caesars Heimatverzeichnis ablegen kann?

Antwort anzeigen

`/home/caesar/  rwx r-x rwx`

Begründung: Berta ist weder Eigentümerin noch in der Gruppe `roemer`. Darum braucht `others` Schreib- und Betretungsrechte (`w + x`).

---

14. Wer darf die Zugriffsrechte von Dateien und Ordnern verändern?

Antwort anzeigen

Der Eigentümer und root.

1 b. Zusatzaufgaben zu Linux-Benutzern, sudo, Besitzrechten und Sonderrechten.

1. Der Benutzer `hund` soll Superuser-/Root-Rechte erhalten. Was ist zu tun?

Antwort anzeigen

Der Benutzer muss zur Gruppe der Administratoren (sudo-Gruppe) hinzugefügt werden:

```
sudo usermod -aG sudo hund
```

Danach muss sich der Benutzer neu anmelden, damit die Gruppenrechte aktiv werden.

---

2. Der Benutzer `hund` kann mit `sudo <befehl>` arbeiten und muss dabei sein Passwort eingeben. Gibt es eine Möglichkeit, ohne Passwortabfrage zu arbeiten?

Antwort anzeigen

Ja, dies ist über die Konfigurationsdatei `/etc/sudoers` möglich.

Ein entsprechender Eintrag lautet:

```
hund ALL=(ALL:ALL) NOPASSWD: ALL
```

Die Datei sollte aus Sicherheitsgründen nicht direkt bearbeitet werden, sondern mit:

```
sudo visudo
```

---

3. Der `tagesplan` soll in den Besitz des Benutzers `hund` übergehen und in seinen Heimatordner verschoben werden. Was ist zu tun?

Antwort anzeigen

Zuerst wird der Besitzer der Datei geändert:

```
sudo chown hund:hund tagesplan
```

Danach wird die Datei in das Heimatverzeichnis verschoben:

```
mv tagesplan /home/hund/
```

Je nach Speicherort der Datei muss der Pfad angepasst werden.

---

4. Beim Anlegen von Benutzern sind manchmal schon Dateien oder Ordner vorhanden, zum Beispiel unter `ubuntu` die Datei `examples.desktop`. Woher kommen diese?

Antwort anzeigen

Diese Dateien stammen aus dem Vorlagenverzeichnis für neue Benutzer:

```
/etc/skel
```

Beim Anlegen eines neuen Benutzers wird dieses Verzeichnis in das Home-Verzeichnis kopiert.

---

5. Was ist der Nutzen solcher bereits vorhandenen Dateien und Ordner im Home-Verzeichnis?

Antwort anzeigen

Diese Dateien stellen eine Grundausstattung für neue Benutzer bereit.

Dazu gehören beispielsweise:

- Beispiel-Dateien
- voreingestellte Konfigurationen
- vorbereitete Ordnerstrukturen

Dadurch kann der Benutzer sofort mit einer sinnvollen Umgebung arbeiten.

---

6. Welche Aufgabe hat das Set-UID-Recht (SUID-Bit)?

Antwort anzeigen

Ein Programm wird mit den Rechten des Dateibesitzers ausgeführt und nicht mit den Rechten des Benutzers, der es startet.

Dadurch können Programme kurzfristig mit erweiterten Rechten ausgeführt werden.

---

7. Welche Aufgabe hat das Set-GID-Recht (SGID-Bit)?

Antwort anzeigen

Bei Dateien wird ein Programm mit den Rechten der Gruppe ausgeführt.

Bei Verzeichnissen sorgt das SGID-Bit dafür, dass neu angelegte Dateien und Ordner automatisch die Gruppe des Verzeichnisses übernehmen.

---

8. Welche Aufgabe hat das Sticky-Bit?

Antwort anzeigen

In einem Verzeichnis mit gesetztem Sticky-Bit dürfen Dateien nur vom jeweiligen Eigentümer, vom Verzeichnisbesitzer oder von root gelöscht oder umbenannt werden.

Ein typisches Beispiel ist das Verzeichnis:

`/tmp`

---

9. Wie werden diese Sonderrechte gesetzt und dargestellt?

Antwort anzeigen

Die Sonderrechte werden mit dem Befehl `chmod` gesetzt:

SUID setzen:
`chmod u+s datei`

SGID setzen:
`chmod g+s datei`

Sticky-Bit setzen:
`chmod +t verzeichnis`

Die Darstellung erfolgt mit `ls -l`:

- SUID: `s` oder `S` im Benutzer-Bereich
- SGID: `s` oder `S` im Gruppen-Bereich
- Sticky-Bit: `t` oder `T` im Other-Bereich

Kleinbuchstaben bedeuten, dass zusätzlich das Ausführungsrecht gesetzt ist.
Großbuchstaben bedeuten, dass das Ausführungsrecht fehlt.

---

10. Welchen Vorteil hat das Sticky-Bit gegenüber dem normalen Schreibrecht auf einen Ordner?

Antwort anzeigen

Ohne Sticky-Bit kann jeder Benutzer mit Schreibrechten in einem Verzeichnis auch Dateien anderer Benutzer löschen.

Mit gesetztem Sticky-Bit wird dies verhindert, sodass Benutzer nur ihre eigenen Dateien löschen können.

Dadurch wird die Sicherheit in gemeinsam genutzten Verzeichnissen erhöht.

2. Gemischt – Benutzer, Zugriffsrechte und Sonderrechte.

Gegeben: Linux – Benutzer, Zugriffsrechte und Sonderrechte.

---

1. Darf berta die Datei bz lesen?

Antwort anzeigen

Ja.
Begründung: Für „others“ sind Leserechte (`r--`) gesetzt.

---

2. Darf caesar die Datei bz lesen?

Antwort anzeigen

Ja.
Begründung: Auch „others“ besitzen Leserechte (`r--`).

---

3. Wer darf die Datei urteile.txt lesen?

Antwort anzeigen

Anton und berta.
Begründung: Die Gruppe besitzt Leserechte (`r--`).

---

4. Was ist zu tun, damit alle Benutzer die Datei urteile.txt lesen können?

Antwort anzeigen

Die Leserechte für „others“ müssen gesetzt werden:

```
chmod o+r urteile.txt
```

5. Darf caesar die Datei kino.pdf verändern?

Antwort anzeigen

Nein.
Begründung: Er besitzt nur Leserechte.

6. Darf caesar die Datei kino.pdf löschen?

Antwort anzeigen

Ja.
Begründung: Das Löschen hängt von den Rechten des Verzeichnisses ab, nicht von der Datei.

7. Wer darf die Datei fahrplan ändern?

Antwort anzeigen

Anton, berta und caesar.
Begründung: Die Datei besitzt die Rechte `rw- rw- rw-`.

8. Nur berta soll die Datei fahrplan ändern dürfen. Was ist zu tun?

Antwort anzeigen

Die Rechte müssen eingeschränkt werden:

```
chmod 600 fahrplan
```

9. Wie versperrt caesar den Mitgliedern der Gruppe berliner den Zugriff auf sein Heimatverzeichnis?

Antwort anzeigen

Die Rechte des Verzeichnisses müssen angepasst werden:

```
chmod 700 /home/caesar
```

---

10. Welche Rechte müssen gesetzt werden, damit caesar eine Datei in antons Heimatverzeichnis ablegen kann?

Antwort anzeigen

Das Verzeichnis muss Schreib- und Betretungsrechte für andere besitzen:

```
/home/anton/  rwx r-x rwx
```

Begründung: Zum Ablegen einer Datei werden `w` und `x` benötigt.

---

11. Der Benutzer `hund` soll Superuser-/Root-Rechte erhalten. Was ist zu tun?

Antwort anzeigen

Der Benutzer muss zur sudo-Gruppe hinzugefügt werden:

```
sudo usermod -aG sudo hund
```

---

12. Der Benutzer `hund` kann mit `sudo <befehl>` arbeiten und muss dabei sein Passwort eingeben. Gibt es eine Möglichkeit, ohne Passwortabfrage zu arbeiten?

Antwort anzeigen

Ja, über die Datei `/etc/sudoers`:

```
hund ALL=(ALL:ALL) NOPASSWD: ALL
```

Bearbeitung mit:

```
sudo visudo
```

---

13. Der `tagesplan` soll in den Besitz des Benutzers `hund` übergehen und in seinen Heimatordner verschoben werden. Was ist zu tun?

Antwort anzeigen

Besitzer ändern:

```
sudo chown hund:hund tagesplan
```

Danach verschieben:

```
mv tagesplan /home/hund/
```

---

14. Woher stammen Standard-Dateien beim Anlegen eines neuen Benutzers?

Antwort anzeigen

Sie stammen aus dem Verzeichnis:

```
/etc/skel
```

Dieses wird in das Home-Verzeichnis kopiert.

---

15. Welche Aufgabe hat das Set-UID-Recht (SUID-Bit)?

Antwort anzeigen

Programme werden mit den Rechten des Dateibesitzers ausgeführt.

---

16. Welche Aufgabe hat das Set-GID-Recht (SGID-Bit)?

Antwort anzeigen

Programme laufen mit Gruppenrechten,
und in Verzeichnissen wird die Gruppe vererbt.

---

17. Welche Aufgabe hat das Sticky-Bit?

Antwort anzeigen

Nur Eigentümer dürfen Dateien löschen oder umbenennen.

---

18. Wie werden Sonderrechte gesetzt?

Antwort anzeigen

SUID: `chmod u+s`
SGID: `chmod g+s`
Sticky: `chmod +t`

---

19. Welchen Vorteil hat das Sticky-Bit?

Antwort anzeigen

Es verhindert das Löschen fremder Dateien in gemeinsam genutzten Verzeichnissen.

Selbstbewertung

0–6 richtig → □ wiederholen
7–12 richtig → ⚠ unsicher
13–17 richtig → □ gut vorbereitet
18–19 richtig → □□ prüfungsbereit

A1_ssh

ssh - Secure Shell zur sicheren remote-Administration

Eine der bekanntesten Anwendungen für asymmetrische Schlüsselverfahren ist die Secure Shell, z.B. openssh.

Abb1. Zur Erinnerung: Bei symmetrischen Verfahren (oben) dient ein Key sowohl zum Ver- als auch zum Entschlüsseln. Die asymmetrische Varianten (unten) benutzen dafür zwei verschiedene Schlüssel, einen privaten und einen öffentlichen.

Jeder Host benötigt ein eigenes Schlüsselpaar aus privatem und öffentlichem Schlüssel, das meist schon bei der Installation der Software generiert wird. Bei dem einleitenden Handshake authentifiziert sich der Zielhost (ssh-Server) mit seinem privaten Schlüssel.

Der Client generiert anschließend einen Sitzungsschlüssel, verschlüsselt ihn mit dem öffentlichen Key des Servers und sendet ihn an den Server. Damit öffnet er einen sicheren Tunnel (siehe Abb.2), über den anschließend die Datenkommunikation abläuft. Die Benutzer-Authentifizierung erfolgt danach über einen Passwortdialog oder ebenfalls mittels asymmetrischer Kryptographie. Beide Versionen laufen über die gesicherte Verbindung des ssh-Tunnels und sind somit für Außenstehende nicht entzifferbar. Nach der Authentifizierung stellt der Server dem Client eine Shell zur Verfügung. Das ssh-Protokoll hat noch viele weitere nützliche Funktionen, z.b. scp zum sicheren Datentransport (siehe ganz unten).

Abb.2 Die Secure Shell verwendet asymmetrische Kryptographie, um den Server gegen den Client und den Benutzer gegen den Server zu authentifizieren. Beide Partner brauchen dazu ein eigenes Schlüsselpaar.

ssh - Secure Shell zur sicheren remote-Administration

Anmelden mit Benutzernamen und Passwort auf srvYZ

userXY@pcXY:~$ ssh [userYZ@]srvYZ

- Beachte ! Benutzerangabe [userYZ@] nur bei abweichendem
- Benutzer auf dem Server.
- Bei der erstmaligen Anmeldung wird ein 'fingerprint' des öffentlichen Schlüssels des Remote-PC/Servers errechnet und bei Bestätigung* (Authentizität) wird der Public-Key des Servers in die .ssh/known_hosts geschrieben.

Anmelden mittels Public-Key-Verfahren

1. Schlüssel erzeugen:

- Syntax: ssh-keygen [-f KEY_FILE] [-C COMMENT] [-t TYP (rsa|dsa)] [-b KEY_LENGTH (1024|2048)]

userXY@pcXY:~$ ssh-keygen

- Standard ist ein RSA-Schlüsselpaar mit 2048 bit Schlüssellänge
- Eine Passphrase (Passwortsatz) dient der Sicherheit, oft im Intranet aus Komfortgründen weggelassen (für automatisierte
- Jobs)
- Das Schlüsselpaar liegt in ~/.ssh.
- Der private Schlüssel in der Datei id_rsa/id_dsa (default) Der public-key in der Datei id_rsa.pub/id_dsa.pub (default)

2. öffentlichen Schlüssel auf remote-Server ablegen

userXY@pcXY:~$ ssh-copy-id -i ~/.ssh/id_rsa.pub [userYZ]@srvYZ

- Kopieren des public-keys automatisch auf den remote-Server in die Datei 'authorized_keys' mit dem Befehl 'ssh-copy-id'

zwei kompliziertere Alternativen dazu:

userXY@pcXY :~$ scp .ssh/id_rsa.pub [userYZ@]srvYZ:.ssh/authorized_keys

- Kopieren des ersten public-keys auf den remote-Server und umbenennen in die Datei authorized_keys

userYZ@srvYZ:~/.ssh$ cat .../usbstick/id_rsa.pub >> authorized_keys

- public-key über einen sicheren Weg (usbstick) direkt auf dem Server an die Datei authorized_keys angehängen (neben anderen public-Keys)

3. Anmelden nun ohne Passwort möglich

userXY@pcXY:~$ ssh [userYZ@]srvYZ

Tip: bei Fehlermeldung 'Agent admitted failure to sign using the key' neuen privatkey mit dem Befehl 'ssh-add' hinzufügen!

ssh - Secure Shell zur sicheren remote-Administration

weitere Beispiel für die Secure Shell:

Kopieren eines Verzeichnisses: z.B. ~/linux nach srvYZ:

userXY@pcXY:~$ scp -rv ~/linux [userYZ@]srvYZ:

Synchronisieren von Daten mit rsync über eine ssh-Verbindung:

userXY@pcXY:~$ rsync -av ~~/linux/ [userYZ@]srvYZ:~~/linux/

Absetzen eines remote-Kommandos:

userXY@pcXY:~$ ssh user@pcXY 'ls -l'

Port-Forwarding:

Port-Weiterleitung eines lokalen Ports an einen entfernten

Zielport durch eine verschlüsselte ssh-Verbindung (Tunnel)

ssh user@ssh-srv -L 1025:localhost:80

oder (Verbindung dauerhaft):

ssh -f -N -L 1025:localhost:80 user@ssh-srv

ftp via ssh → sftp:

userXY@pcXY:~$ sftp user@pcXY

Connected to pcXY

sftp>

für Konsolenpuristen!

userXY@pcXY:~$ cat .ssh/id_rsa.pub | ssh [userYZ@]srvYZ 'cat >> .ssh/authorized_keys'

Der Inhalt des public-keys ' id_rsa.pub ' wird in einen Puffer|Pipe gesteckt

und nach dem Anmelden auf dem Server ' ssh userYZ@srvYZ ' an das Schlüsselbund angehängt, mit dem übergebenen Befehl ' 'cat >> .ssh/authorized_keys''

- Fingerprint des ECDSA-pubkey auf dem Server anzeigen und zur Bestätigung vergleichen:

```
userYZ@srvYZ:~$ ssh-keygen -l -f /etc/ssh/ssh_host_ecdsa_key.pub
```

Unterlagen:

- wget ftp://ftp.itm.bfw/pub/unterricht/linux/netzwerk/09_sicherheit/03_openssh/

Internet:

- http://wiki.ubuntuusers.de/SSH
- http://www.youtube.com/watch?v=Hxsl-jj2Bq0

- http://www.lug-ottobrunn.de/wiki/Remote_Desktop_mit_X2GO

Quellen

- Dokument: A1_ssh
- ID: 138