SSH

Secure Shell

Grundlagen

- SSH → Secure Shell
- Verschlüsseltes Netzwerkprotokoll (vgl. http, https, ftp, ...)
- Ermöglicht u.a. Fernzugriff auf entfernte Rechner, Dateitransfer

Bestandteile

- SSH – Client
- SSHD – OpenSSH-Daemon (Server)

Zweck

- **Verbindung zwischen zwei Rechnern mit Ende-zu-Ende-Verschlüsselung**
- Verwaltung von Computern, auf die man nicht lokal zugreifen kann
- Sichere Übermittlung von Dateien
- Sicheres Erstellen von Backups
- Nutzung als VPN-Tunnel

Installation des Clients

- Debian-basiert: `sudo apt install openssh-client`
- Arch-basiert: `sudo pacman –S openssh`
- Windows: integrierter Client oder PuTTY

Installation des Servers

- Debian o.ä.: `sudo apt install openssh-server` und `sudo apt install openssh-sftp-server`
- Arch o.ä.: `sudo pacman -S openssh`

- Starten des Servers mit systemd: `sudo systemctl start sshd`
- Autostart des Servers mit systemd: `sudo systemctl enable sshd`

Konfiguration des Servers

- Schlüssel werden automatisch generiert
- Konfigurationsdatei: `/etc/ssh/sshd_config`
- Standardkonfiguration ist zwar ausreichend, kann aber für den Produktivbetrieb verbessert werden:

Konfiguration des Servers

`AllowUsers user1 user2`
erlaubt Zugriff nur für best. Nutzer

`AllowGroups group1 group2`
erlaubt Zugriff nur für best. Gruppen

`Port 4269`
ändert den Port, auf dem der Server lauscht

`PermitRootLogin no`

Anmeldung von Root über SSH verweigern

`chmod 600 ~/.ssh/authorized_keys`

Datei vor unbefugten Zugriffen schützen

Konfiguration des Servers

Um Brute-Force Angriffen und schwachen Kennwörtern entgegenzuwirken, kann man die passwortbasierte Authentifizierung abschalten:

- sicherstellen, dass SSH-Schlüssel mit ausreichenden Rechten dem Server bekannt sind
- in `/etc/sshd_config` folgende Optionen setzen

```
PasswordAuthentication no
PubkeyAuthentication yes
ChallengeResponseAuthentication no
```

- Konfiguration neu laden: `sudo systemctl reload sshd`

Konfiguration des Servers

Konfiguration des Clients

```
$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key
  (C:\Users\User/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in
  C:\Users\User/.ssh/id_rsa.
Your public key has been saved in
  C:\Users\User/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:mstyW1phoEPqbMT3stzYUapfT6V/rBpMMyEHMmjiTp0
  user@DESKTOP-EPNUPVN
The key's randomart image is:
```

```
+---[RSA 2048]----+
|        .+       |
|     . o. .      |
|    ..+... o     |
| . oo.E. o .     |
|  +o+   S + .    |
| + ..o * + =     |
|  + . * + =  .   |
| . ..O.B o o  o  |
|    =+O.  o.oo   |
+----[SHA256]-----+
```

Konfiguration des Clients

- Kopieren des öff. Schlüssels auf den Server (insofern passwortbasiertes Auth aktiviert ist)

```
cat ~/.ssh/id_rsa.pub | ssh pi@raspberry ”mkdir
-p ~/.ssh && chmod 700 ~/.ssh && cat
>>  ~/.ssh/authorized_keys”
```

- Darüber hinaus keine besondere Konfiguration notwendig.

Dateitransfer am Beispiel FileZilla

- Server: `sftp://fqdn.oder.ip`
- Benutzer: auf dem Server existierender Nutzer eingeben
- Passwort analog
- Port: SSH-Port vom Server

im Falle Schlüsselbasiertes Auth:

- Filezilla kann automatisch Putty-Agent “Pageant” nutzen
- alternativ Pfad des priv. Schlüssels unter `Bearbeiten - Einstellungen - SFTP`

Dateitransfer mit scp (Commandline)

- Vom Localhost zum Server: `scp virus.exe nutzer@fqdn.oder.ip:/ziel/pfad/`
- vom Server zum Localhost: `scp nutzer@fqdn.oder.ip:/quell/pfad/virus.exe /ziel/pfad/`

Absicherung mit Fail2Ban

- nach best. Anzahl Fehlversuchen wird IP-Adresse in Firewall gesperrt
- Installation:

```
sudo apt install fail2ban
```

- Konfiguration:

Config: /etc/fail2ban/jail.conf

```
ignoreip = 127.0.0.1/8
bantime = 600
findtime = 600
maxretry = 3
destemail = root@localhost
sendername = Fail2Ban
mta = sendmail
```