

## **Aufgabe 2: Ports von SSH, HTTP und HTTPS**

Ein Port ist eine numerische Adresse in einem Computersystem, die zur Identifizierung eines bestimmten Dienstes oder einer Anwendung verwendet wird. Ports sind Teil des TCP/IP-Netzwerkprotokolls und ermöglichen es, Daten an den richtigen Ort auf einem Computer oder Server zu leiten.

- SSH (Secure Shell):  
SSH verwendet standardmäßig Port 22. SSH ist ein sicheres Protokoll zur Fernverwaltung von Computern über eine verschlüsselte Verbindung.
- HTTP (Hypertext Transfer Protocol):  
HTTP verwendet standardmäßig Port 80. HTTP ist das Protokoll, das für die Übertragung von Webseiten und anderen Ressourcen im World Wide Web verwendet wird.
- HTTPS (Hypertext Transfer Protocol Secure):  
HTTPS verwendet standardmäßig Port 443. HTTPS ist die sichere Version von HTTP und wird zur verschlüsselten Übertragung von Webseiten und sensiblen Daten wie Anmeldedaten und Kreditkarteninformationen verwendet.  
Der zusätzliche Buchstabe "S" steht für "sicher", und die Verschlüsselung erfolgt normalerweise mithilfe von SSL/TLS.

### **Aufgabe 3: Prinzip von einer Verbindung mit öffentlichen/privaten Schlüsseln (SSH)**

Eine SSH-Verbindung mit öffentlichen/privaten Schlüsseln basiert auf der asymmetrischen Verschlüsselung.

- Ein SSH-Schlüsselpaar besteht aus einem öffentlichen Schlüssel und einem privaten Schlüssel.
- Der öffentliche Schlüssel wird auf dem Server hinterlegt, zu dem du dich verbinden möchtest.
- Der private Schlüssel bleibt auf deinem Client.
- Bei der Verbindungsherstellung sendet der Client (der SSH-Client) seinen öffentlichen Schlüssel an den Server.
- Der Server überprüft, ob dieser öffentliche Schlüssel im System hinterlegt ist.
- Wenn der Server den öffentlichen Schlüssel erkennt, fordert er den Client auf, eine Signatur mit dem privaten Schlüssel zu erstellen und zu senden.
- Der Server überprüft dann die Signatur mit dem gespeicherten öffentlichen Schlüssel.
- Wenn die Überprüfung erfolgreich ist, wird die Verbindung hergestellt, und du bist authentifiziert, ohne ein Passwort eingeben zu müssen.

Erstellen eines SSH-Schlüsselpaars:

1. Öffne ein Terminal auf deinem Client-Computer.
2. Verwende den Befehl ``ssh-keygen`` zum Generieren eines Schlüsselpaares. Standardmäßig wird ein RSA-Schlüssel erstellt.
3. Folge den Anweisungen, um einen Speicherort für den Schlüssel und ein optionales Passwort festzulegen.
4. Der öffentliche Schlüssel wird normalerweise in der Datei ``~/.ssh/id_rsa.pub`` gespeichert.
5. Du kannst den Inhalt des öffentlichen Schlüssels kopieren und auf den Server hochladen, zu dem du dich verbinden möchtest, damit dieser deinen öffentlichen Schlüssel erkennt.

## Aufgabe 4: Research

```
losem@LTV: ~/LinuxPraxis
losem@LTV:/mnt/c/Windows/system32$ cd
losem@LTV:~$ cd Linuxpraxis
-bash: cd: Linuxpraxis: No such file or directory
losem@LTV:~$ cd
losem@LTV:~$ cd /
losem@LTV:/$ ls
bin  dev  home  lib  lib64  lost+found  mnt  proc  run  snap  sys  usr
boot  etc  init  lib32  libx32  media  opt  root  sbin  srv  tmp  var
losem@LTV:/$ cd
losem@LTV:~$ ls
'LinuxPraxis Kevin'  Techstarter  octave
losem@LTV:~$ cd LinuxPraxis Kevin
-bash: cd: too many arguments
losem@LTV:~$ cd "LinuxPraxis Kevin"
losem@LTV:~/LinuxPraxis Kevin$ cd
losem@LTV:~$ ls
'LinuxPraxis Kevin'  Techstarter  octave
losem@LTV:~$ mv "LinuxPraxis Kevin" LinuxPraxis
losem@LTV:~$ ls
LinuxPraxis  Techstarter  octave
losem@LTV:~$ cd LinuxPraxis
losem@LTV:~/LinuxPraxis$ echo "Apfel" > KevinSuche.txt
losem@LTV:~/LinuxPraxis$ echo "Banane" >> KevinSuche.txt
losem@LTV:~/LinuxPraxis$ echo "Kirsche" >> KevinSuche.txt
losem@LTV:~/LinuxPraxis$ echo "Dattel" >> KevinSuche.txt
losem@LTV:~/LinuxPraxis$ grep "Kirsche" KevinSuche.txt
Kirsche
losem@LTV:~/LinuxPraxis$ |
```

## Aufgabe 5: Zusatzaufgabe

### Setup SSH-Server

```
losem@LTV: /
Get:4 http://archive.ubuntu.com/ubuntu jammy-backports InRelease [109 kB]
Get:5 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 Packages [1103 kB]
Get:6 http://archive.ubuntu.com/ubuntu jammy-updates/universe amd64 c-n-f Metadata [22.0 kB]
Fetched 1463 kB in 2s (901 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
1 package can be upgraded. Run 'apt list --upgradable' to see it.
losem@LTV:/$ sudo apt install openssh-server
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
openssh-server is already the newest version (1:8.9p1-3ubuntu0.4).
0 upgraded, 0 newly installed, 0 to remove and 1 not upgraded.
losem@LTV:/$ ap addr
ap: command not found
losem@LTV:/$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1492 qdisc mq state UP group default qlen 1000
    link/ether 00:15:5d:be:6d:bc brd ff:ff:ff:ff:ff:ff
    inet 192.168.172.193/20 brd 192.168.175.255 scope global eth0
        valid_lft forever preferred_lft forever
    inet6 fe80::215:5dff:febe:6dbc/64 scope link
        valid_lft forever preferred_lft forever
losem@LTV:/$
```

### SSH Connection über PuTTY:

