

`ssh` - Zugriffe einrichten



Inhaltsverzeichnis

- `ssh` (Secure Shell) - Überblick
- `ssh` -Server und `ssh` -Client
- `ssh` -Software-Pakete unter Debian Linux
- Situation auf `debian` und `deb-srv` nach der Installation
- Namensauflösung
- OpenSSH-Server installieren auf `deb-srv`
- Erster `ssh` -Login von `debian` zu `deb-srv`

- `ssh` -Login ohne Passwort
- Aufgaben
- Komfort mit Aliasen
- Entfernte Kommando-Ausführung mit `ssh`
- Dateien übertragen mit `scp`

ssh (Secure Shell) - Überblick

- **ssh** ist ein Netzwerkprotokoll zur sicheren Datenübertragung
- Die Datenübertragung erfolgt verschlüsselt.
- Die **ssh**-Software erlaubt
 - die sichere Anmeldung an einem entfernten Rechner
 - die Ausführung von Befehlen auf einem entfernten Rechner
 - die Übertragung von Dateien zwischen zwei Rechnern
 - das Tunneln von (evtl. unverschlüsseltem) Netzwerkverkehr durch eine verschlüsselte **ssh**-Verbindung (Port-Forwarding) ähnlich wie VPN

ssh-Server und ssh-Client

- ssh-Server:
 - Der ssh-Server ist ein Dienst, der auf einem Rechner läuft und auf eingehende ssh-Verbindungen wartet.
 - Der ssh-Server "lauscht" (wartet auf eingehende Verbindungen) standardmäßig auf Port 22. Er lässt sich ggf. auf andere Ports umkonfigurieren.

- `ssh`-Client:
 - Der `ssh`-Client initiiert die Verbindung zum `ssh`-Server.
 - Es gibt mehrere `ssh`-Client-Tools für die Kommandozeile:
 - `ssh` (Secure Shell): Remote-Login und Befehlsausführung auf dem entfernten Rechner
 - `scp` (Secure Copy): Dateien über das Netzwerk kopieren (ähnlich wie `cp`)
 - `sftp` (Secure File Transfer Protocol): Dateien über das Netzwerk übertragen (ähnlich wie `ftp`)

- ○ `ssh-keygen`: Erzeugen von Schlüsselpaaren für die Authentifizierung
- `ssh-copy-id`: Kopieren des öffentlichen Schlüssels in einen Benutzer-Account auf einem entfernten Rechner

`ssh`-Software-Pakete unter Debian Linux

- `openssh-client` (`ssh`, `scp`, `sftp`, `ssh-keygen`, `ssh-copy-id`): ist bei Debian standardmäßig installiert
- `openssh-server` (`sshd`): ist bei Debian standardmäßig nicht installiert. Der `sshd`-Dienst muss installiert werden, es sei denn, dieses Paket wurde bei der Installation des Betriebssystems ausgewählt.

Situation auf `debian` und `deb-srv` nach der Installation

- `ping` zwischen beiden Systemen funktioniert.
- `debian`:
 - `ssh`-Client ist installiert.
 - `ssh`-Server ist installiert (schon bei der Installation ausgewählt).
- `deb-srv`:
 - `ssh`-Client ist installiert.
 - `ssh`-Server ist nicht installiert.

- Auf `debian`: `ssh <debian-user>@localhost` sollte schon funktionieren. (Bitte testen, ob das klappt!)
- Auf `dev-srv`: `ssh <debian-user>@debian` sollte schon funktionieren. (Bitte testen, ob das klappt!)
- Auf `dev-srv`: `ssh <deb-srv-user>@localhost` sollte nicht funktionieren. (Bitte trotzdem testen! Welche Fehlermeldung?)
- Auf `debian`: `ssh <deb-srv-user>@deb-srv` sollte nicht funktionieren. (Bitte trotzdem testen! Welche Fehlermeldung?)

Beim der ersten erfolgreichen Verbindung zu einem neuen Server muss der Fingerabdruck des Servers bestätigt werden. Danach wird das Passwort des Benutzers auf dem Server abgefragt.

Namensauflösung

- Ein Server kann über seine IP-Adresse (IPv4 oder IPv6) oder über seinen Hostnamen angesprochen werden. Beides ist grundsätzlich möglich.
- Wir befinden uns in einem virtuellen Netzwerk, das von Hyper-V über den virtuellen "Default Switch" bereitgestellt wird.
- In diesem Netzwerk werden die **IP-Adressen** von einem **DHCP**-Server **dynamisch** vergeben. D.h. die IP-Adressen können sich nach jedem Neustart der virtuellen Maschinen ändern.

- Hyper-V stellt auch einen **DNS**-Server zur Verfügung, der die Namensauflösung im virtuellen Netzwerk übernimmt. Da sich die IP-Adressen nach jedem Reboot ändern können, ist es zweckmäßig, bei der Kommunikation der virtuellen Maschinen untereinander und mit dem Windows-Host die **Hostnamen** zu **verwenden**.
- Der Domain-Name des virtuellen Netzwerks ist `mshome.net`.
- Gelegentlich kann es vorkommen, dass die Namensauflösung nicht funktioniert. Nach meinen Erfahrungen hilft es, in diesen Fällen den voll qualifizierten Domainnamen (FQDN) zu verwenden, also `<hostname>.mshome.net`, z.B. `deb-srv.mshome.net`.

Erfolglose `ssh`-Verbindung von `debian` nach `deb-srv`

```
hermann@debian:~$ ssh hermann@deb-srv  
ssh: connect to host deb-srv port 22: Connection refused
```

oder

```
hermann@debian:~$ ssh hermann@deb-srv.mshome.net  
ssh: connect to host deb-srv port 22: Connection refused
```

OpenSSH-Server installieren auf **deb-srv**

```
hermann@deb-srv:~$ # update the system
hermann@deb-srv:~$ sudo apt update && sudo apt upgrade
...
hermann@deb-srv:~$ # install the OpenSSH server
hermann@deb-srv:~$ sudo apt install openssh-server
...
```

Erster **ssh**-Login zu **localhost**

```
hermann@debian:~$ ssh hermann@localhost
The authenticity of host 'localhost (127.0.0.1)' can't be established.
ED25519 key fingerprint is SHA256:9uPk1CbTc2H2FT0e9jLVBo2kPgKiqPHbBcI/5yfvzDU.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'localhost' (ED25519) to the list of known hosts.
hermann@localhost's password:
Linux localhost 6.1.0-28-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.119-1 (2024-11-22) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Sun Dec  1 00:52:23 2024
hermann@debian:~$
```

Erster `ssh`-Login von `debian` zu `deb-srv`

```
hermann@debian:~$ ssh hermann@deb-srv
The authenticity of host 'deb-srv (172.28.140.92)' can't be established.
ED25519 key fingerprint is SHA256:9uPk1CbTc2H2FT0e9jLVBo2kPgKiqPHbBcI/5yfvzDU.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'deb-srv' (ED25519) to the list of known hosts.
hermann@deb-srv's password:
Linux deb-srv 6.1.0-28-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.119-1 (2024-11-22) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Sun Dec  1 00:52:23 2024
hermann@deb-srv:~$
```


- Beim ersten Login wird der Fingerabdruck des Servers `deb-srv` abgefragt. Bestätigen Sie den Fingerabdruck mit `yes` (nicht `y`, sondern `yes` ausschreiben).
- Nach der Bestätigung des Fingerabdrucks wird dieser in der Datei `~/.ssh/known_hosts` auf dem `ssh`-Client `debian` gespeichert.
- Ist der Fingerprint gespeichert, unterbleibt die Abfrage bei zukünftigen Logins.
- Dann folgt die Abfrage des Passworts des Benutzers auf dem Server `deb-srv`.

ssh-Login ohne Passwort

- Der `ssh`-Login mit Passwort ist hergestellt. Er hat jedoch zwei Nachteile:
 - Das Passwort muss umständlich bei jedem Login eingegeben werden.
 - Passwörter sind grundsätzlich unsicher. Sie können evtl. erraten oder geknackt (z.B. brute force attack) werden.
- `ssh` unterstützt die Authentifizierung mit asymmetrischer Verschlüsselung. Dies ermöglicht den Login ohne Passwort. (Gewinn an Sicherheit und Komfort!)

Asymmetrische Verschlüsselung (nach Diffie-Hellman)

- Asymmetrische Verschlüsselung erfolgt mit Schlüsselpaaren:
 - Ein Schlüsselpaar besteht aus einem privaten und einem öffentlichen Schlüssel.
 - Der öffentliche Schlüssel wird auf dem Server gespeichert.
 - Der private Schlüssel bleibt (für andere Benutzer nicht lesbar) auf dem Client.
 - Der private Schlüssel kann durch eine Passphrase geschützt werden.
- Dieses Verfahren kann bei `ssh` statt Passwort-Abfrage zur Authentifizierung verwendet werden.

Wie geht man vor?

- Auf dem `ssh`-Client (`debian`) wird ein Schlüsselpaar (private key und public key) erzeugt. Dabei ist das Verschlüsselungsverfahren (RSA) und die Schlüssellänge (4096 Bits) anzugeben.
- Die beiden Schlüssel werden in zwei Dateien im Verzeichnis `~/.ssh` gespeichert: `id_rsa` (private key) und `id_rsa.pub` (public key).
- Dabei wird eine Passphrase abgefragt, die den privaten Schlüssel schützt. Diese Passphrase muss bei jedem Login eingegeben werden. Man kann sie auch leer lassen.

- Der öffentliche Schlüssel wird auf den Server (`deb-srv`) kopiert und dort in die Datei `~/.ssh/authorized_keys` eingefügt. Dabei wird ein letztes Mal das Passwort des Benutzers auf dem Server abgefragt.
- Der private Schlüssel bleibt auf dem Client (`debian`). Zum Schutz des privaten Schlüssels bleibt das Verzeichnis `~/.ssh` auf dem Client für andere Benutzer unzugänglich (Rechte: `rwX-----`).
- Nun ist der passwortlose `ssh` -Login möglich.
- Auch auf dem Server bleibt das Verzeichnis `~/.ssh` für andere Benutzer unzugänglich (Rechte: `rwX-----`).

Schlüsselpaar erzeugen auf **debian**

```
ermann@debian:~$ ssh-keygen -t rsa -b 4096 # --- use RSA algorithm for encryption with 4096 bits key length ---
Generating public/private rsa key pair.
Enter file in which to save the key (/home/hermann/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/hermann/.ssh/id_rsa
Your public key has been saved in /home/hermann/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:pdDY45eMo6jcGmirm0/AS8jxgizSYkRw/oz29G8XiaY hermann@debian
The key's randomart image is:
+---[RSA 4096]-----+
|. .                |
|.o      +         |
|. .  o + .        |
|.  +  o * .       |
| oo +  S.+       |
|B+oo o .ooo      |
|00o.o oo   .     |
|B=++  E.. .      |
|==*..  ...       |
+-----[SHA256]-----+
```

`~/ .ssh` -Verzeichnis auf `debian`

```
hermann@debian:~$ ls -ld .ssh
drwx----- 2 hermann hermann 4096 1. Dez 01:43 .ssh
hermann@debian:~$ ls -l .ssh/*
-rw----- 1 hermann hermann 3381 1. Dez 01:42 .ssh/id_rsa
-rw-r--r-- 1 hermann hermann 740 1. Dez 01:42 .ssh/id_rsa.pub
-rw----- 1 hermann hermann 1956 1. Dez 01:04 .ssh/known_hosts
-rw----- 1 hermann hermann 1120 1. Dez 01:04 .ssh/known_hosts.old
```

!!! WICHTIG !!! Die Rechte des Verzeichnisses `~/ .ssh` müssen alle Benutzerrechte außer Eigentümer aussperren (`rwX-----`), damit der private Schlüssel nicht von anderen Benutzern gelesen und kopiert werden kann.

Public Key übertragen auf `deb-srv`

```
hermann@debian:~$ ssh-copy-id hermann@deb-srv
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/hermann/.ssh/id_rsa.pub"
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any ...
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now ...
hermann@deb-srv's password:

Number of key(s) added: 1

Now try logging into the machine, with:  "ssh 'hermann@deb-srv'"
and check to make sure that only the key(s) you wanted were added.
```

Der öffentliche Schlüssel `id_rsa.pub` wird an den Benutzer `hermann` auf dem Server `deb-srv` übertragen und in `~/.ssh/authorized_keys` eingefügt. Ein letztes Mal wird das Passwort des Benutzers auf dem Server abgefragt.

ssh -Login ohne Passwort

```
hermann@debian:~$ ssh hermann@deb-srv
Linux deb-srv 6.1.0-28-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.119-1 (2024-11-22) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Sun Dec  1 01:44:53 2024 from 172.28.141.229
```

`~/ .ssh` -Verzeichnis auf `deb-srv`

```
hermann@deb-srv:~$ ls -ld .ssh
drwx----- 2 hermann hermann 4096  1. Dez 01:43 .ssh
hermann@deb-srv:~$ ls -l .ssh/*
-rw----- 1 hermann hermann  740  1. Dez 01:43 .ssh/authorized_keys
hermann@deb-srv:~$ exit
exit
hermann@debian:~$
```

Rollenverständnis

In der obigen Beschreibung haben wir die Rollen von `debian` als `ssh`-Client und `deb-srv` als `ssh`-Server angenommen. Es ist jedoch auch möglich, dass `deb-srv` als `ssh`-Client auf `debian` zugreift. In diesem Fall sind die Rollen vertauscht: `deb-srv` ist der `ssh`-Client und `debian` ist der `ssh`-Server.

Aufgaben

- Das passwortlose `ssh`-Login von `debian` zu `deb-srv` ist eingerichtet. Richten Sie nun auch das passwortlose `ssh`-Login von `deb-srv` zu `debian` ein. Die Rollen sind dabei vertauscht: `deb-srv` ist der `ssh`-Client und `debian` ist der `ssh`-Server.
- Fragen Sie zunächst das TTY Ihrer aktuellen Sitzung ab: `tty`.
- Versuchen Sie auf beiden Systemen den `ssh`-Login auf `localhost`: `ssh <user>@localhost`. Bildschirm-Ausgaben beachten!
- Fragen Sie das TTY Ihrer neuen Sitzung auf `localhost` ab. Was fällt auf?

Komfort mit Aliasen

Ein weiterer Komfortgewinn lässt sich mit Aliasen erzielen.

- Für den schnellen Login auf `deb-srv` definieren wir den Alias `srv` und analog für den schnellen Login auf `debian` und `localhost` die Aliase `deb` und `lh`.
- Alle drei Aliase werden auf `debian` definiert und an `~/.bash_aliases` angehängt.
- Die Datei `~/.bash_aliases` wird auf `deb-srv` kopiert. Damit stehen alle Aliase auch auf beiden Systemen zur Verfügung.

3 neue Aliase auf **debian** definieren (in der aktuellen Shell)

```
hermann@debian:~$ alias deb='ssh hermann@debian.mshome.net'  
hermann@debian:~$ alias srv='ssh hermann@dev-srv.mshome.net'  
hermann@debian:~$ alias lh='ssh hermann@localhost'
```

Passen Sie den Benutzernamen und die Hostnamen an Ihre Umgebung an.

Aliase an die Datei `~/.bash_aliases` anhängen

```
hermann@debian:~$ alias deb srv lh
alias deb='ssh hermann@debian'
alias srv='ssh hermann@deb-srv'
alias lh='ssh hermann@localhost'
hermann@debian:~$ alias deb srv lh >> ~/.bash_aliases
```

Inhalt von `~/.bash_aliases` kontrollieren

```
hermann@debian:~$ tail -3 ~/.bash_aliases
alias deb='ssh hermann@debian'
alias srv='ssh hermann@dev-srv'
alias lh='ssh hermann@localhost'
```

~/ .bash_aliases von **debian** auf **deb-srv** bereitstellen

```
hermann@debian:~$ scp .bash_aliases hermann@deb-srv.mshome.net:.  
.bash_aliases          100% 313   154.2KB/s   00:00
```


~/ .bash_aliases auf deb-srv anzeigen

```
hermann@debian:~$ ssh hermann@deb-srv.mshome.net 'ls -l .bash_aliases'
-rwxr-xr-x 1 hermann hermann 313  1. Dez 01:57 .bash_aliases
hermann@debian:~$ ssh hermann@deb-srv.mshome.net 'nl -ba .bash_aliases'
     1  alias a=alias
     2  alias cl=clear
     3  alias h=history
...
    11  alias deb='ssh hermann@debian.mshome.net'
    12  alias srv='ssh hermann@dev-srv.mshome.net'
    13  alias lh='ssh hermann@localhost'
hermann@debian:~$
```

Entfernte Kommando-Ausführung mit `ssh`

- Mit `ssh` können auch Kommandos auf einem entfernten Rechner ausgeführt werden.
- Die Ausgabe des Kommandos wird auf dem lokalen Rechner angezeigt.
- Syntax: `ssh <user>@<host> '<command> [arg ...]'`
- Beispiel: `ssh hermann@deb-srv 'ls -l'`
- Das aktuelle Verzeichnis auf dem entfernten Rechner ist das Heimatverzeichnis des Benutzers. Alle relativen Pfade beziehen sich auf dieses Verzeichnis.

- Sonderzeichen in Kommandos müssen ggf. maskiert werden, damit sie nicht von der lokalen Shell interpretiert werden.
- Aus diesem Grund wird das Kommando meist in einfache Anführungszeichen gesetzt.
- Auch doppelte Anführungszeichen sind möglich, wenn Variablen-Substitution oder Kommando-Substitution auf dem lokalen Rechner erfolgen soll.
- **Beachte:** Die entfernten Kommandos werden zunächst von der lokalen Shell interpretiert, dann an den entfernten Rechner übertragen und dort von der entfernten Shell nochmals interpretiert und ausgeführt.

Kommandos auf dem entfernten Rechner ausführen

```
hermann@debian:~/my-tests$ pwd
/home/hermann/my-tests
hermann@debian:~/my-tests$ ssh hermann@deb-srv.mshome.net 'pwd'
/home/hermann
hermann@debian:~/my-tests$ ssh hermann@deb-srv.mshome.net 'ls -l'
insgesamt 0
hermann@debian:~/my-tests$ ssh hermann@deb-srv.mshome.net 'ls -al'
insgesamt 32
drwx----- 3 hermann hermann 4096  4. Dez 11:56 .
drwxr-xr-x  4 root      root    4096  1. Dez 00:51 ..
-rwxr-xr-x  1 hermann hermann  313  4. Dez 11:53 .bash_aliases
-rw-----  1 hermann hermann  572  4. Dez 12:01 .bash_history
-rw-r--r--  1 hermann hermann  220  1. Dez 00:51 .bash_logout
-rw-r--r--  1 hermann hermann 3680  4. Dez 11:56 .bashrc
-rw-r--r--  1 hermann hermann  807  1. Dez 00:51 .profile
drwx----- 2 hermann hermann 4096  1. Dez 22:30 .ssh
```

Wildcards - Werden diese lokal oder remote interpretiert?

```
hermann@debian:~/my-tests$ ssh hermann@deb-srv.mshome.net 'mkdir my-remote-dir'
hermann@debian:~/my-tests$ ssh hermann@deb-srv.mshome.net 'ls -l'
insgesamt 4
drwxr-xr-x 2 hermann hermann 4096  4. Dez 12:53 my-remote-dir
hermann@debian:~/my-tests$ ssh hermann@deb-srv.mshome.net 'ls -l my-remote-dir'
insgesamt 0
hermann@debian:~/my-tests$ ssh hermann@deb-srv.mshome.net ls -ld my-*
ls: Zugriff auf 'my-etc' nicht möglich: Datei oder Verzeichnis nicht gefunden
hermann@debian:~/my-tests$ ssh hermann@deb-srv.mshome.net ls -ld my-*
ls: Zugriff auf 'my-etc' nicht möglich: Datei oder Verzeichnis nicht gefunden
hermann@debian:~/my-tests$ ssh hermann@deb-srv.mshome.net 'ls -ld my-*'
drwxr-xr-x 2 hermann hermann 4096  4. Dez 12:53 my-remote-dir
```

Variablen-Substitution - lokal oder remote interpretiert?

```
hermann@debian:~/my-tests$ pwd
/home/hermann/my-tests
hermann@debian:~/my-tests$ ssh hermann@deb-srv.mshome.net echo $PWD
/home/hermann/my-tests
hermann@debian:~/my-tests$ ssh hermann@deb-srv.mshome.net "echo $PWD"
/home/hermann/my-tests
hermann@debian:~/my-tests$ ssh hermann@deb-srv.mshome.net 'echo $PWD'
/home/hermann
hermann@debian:~/my-tests$ ssh hermann@deb-srv.mshome.net \
> "echo local workdir: $PWD, remote workdir: \"$PWD\""
local workdir: /home/hermann/my-tests, remote workdir: /home/hermann
```

Kommando-Substitution - lokal oder remote interpretiert?

```
hermann@debian:~/my-tests$ ssh hermann@deb-srv.mshome.net echo $(pwd)
/home/hermann/my-tests
hermann@debian:~/my-tests$ ssh hermann@deb-srv.mshome.net "echo $(pwd)"
/home/hermann/my-tests
hermann@debian:~/my-tests$ ssh hermann@deb-srv.mshome.net 'echo $(pwd)'
/home/hermann
hermann@debian:~/my-tests$ ssh hermann@deb-srv.mshome.net \
> "echo local workdir: $(pwd), remote workdir: \$(pwd)"
local workdir: /home/hermann/my-tests, remote workdir: /home/hermann
hermann@debian:~/my-tests$ ssh hermann@deb-srv.mshome.net 'cd my-rem*; echo $(pwd)'
/home/hermann/my-remote-dir
```

Alias-Benutzung

Alle obigen Beispiele hätten wir auch mit dem Alias `srv` durchführen können. Hier ein paar Beispiele:

```
hermann@debian:~/my-tests$ srv echo $(pwd)
/home/hermann/my-tests
hermann@debian:~/my-tests$ srv "echo $(pwd)"
/home/hermann/my-tests
hermann@debian:~/my-tests$ srv 'echo $(pwd)'
/home/hermann
hermann@debian:~/my-tests$ srv 'cd my-rem*; echo $(pwd)'
/home/hermann/my-remote-dir
```


Dateien übertragen mit `scp`

- `scp` (Secure Copy) ist ein Kommandozeilen-Tool, das Dateien über das Netzwerk kopiert.
- Die Syntax von `scp` gleicht der von `cp`. Quelle und Ziel können Pfade auf dem lokalen Rechner oder auf einem entfernten Rechner sein.
- Lokale Pfade werden wie bei `cp` spezifiziert.
- Syntax für entfernte Pfade: `<user>@<host>:<path>`
 - Relative Pfade beziehen sich auf das Heimatverzeichnis des Benutzers auf dem entfernten Rechner.

- ○ Fehlt die Pfadangabe, dann wird `.` angenommen.
- Sonderzeichen in Pfaden müssen ggf. maskiert werden, wenn sie nicht von der lokalen Shell interpretiert werden sollen.

Man kann Dateien...

- von einem entfernten Rechner auf den lokalen Rechner kopieren,
- vom lokalen Rechner auf einen entfernten Rechner kopieren.
- von einem entfernten Rechner auf einen anderen entfernten Rechner kopieren.
- Auch können Quelle und Ziel auf dem gleichen Rechner liegen.
Dafür verwendet man jedoch einfacher das `cp`-Kommando.

Datei kopieren

```
hermann@debian:~/my-tests$ ls -l
insgesamt 8
drwxr-xr-x 2 hermann hermann 4096  2. Dez 18:31 my-etc
-rw-r--r-- 1 hermann hermann  219  2. Dez 18:32 users.txt
hermann@debian:~/my-tests$ scp users.txt hermann@deb-srv.mshome.net:my-remote-dir
users.txt                                100% 219    52.5KB/s   00:00
hermann@debian:~/my-tests$ scp 'hermann@deb-srv.mshome.net:my-remote-dir/use*.txt' \
users-from-remote.txt
users.txt                                100% 219    55.9KB/s   00:00
hermann@debian:~/my-tests$ ls -l
insgesamt 12
drwxr-xr-x 2 hermann hermann 4096  2. Dez 18:31 my-etc
-rw-r--r-- 1 hermann hermann  219  4. Dez 16:12 users-from-remote.txt
-rw-r--r-- 1 hermann hermann  219  2. Dez 18:32 users.txt
```

Dateibaum kopieren

```
hermann@debian:~/my-tests$ scp -r my-etc/apt hermann@deb-srv.mshome.net:my-remote-dir
20listchanges                                100% 307    140.6KB/s   00:00
00aptitude                                  100%  49     24.9KB/s   00:00
50apt-file.conf                             100% 2164    1.0MB/s    00:00
01autoremove                               100% 399    252.8KB/s   00:00
00CDMountPoint                             100%  82     42.9KB/s   00:00
00trustcdrom                               100%  40     25.2KB/s   00:00
70debconf                                  100% 182    110.2KB/s   00:00
99synaptic                                  100%  32     15.7KB/s   00:00
debian-archive-bookworm-security-automatic.asc 100% 12KB    4.9MB/s    00:00
debian-archive-buster-automatic.asc          100% 11KB    5.9MB/s    00:00
debian-archive-bullseye-automatic.asc        100% 12KB    5.9MB/s    00:00
debian-archive-bookworm-stable.asc           100% 461    233.8KB/s   00:00
debian-archive-bullseye-stable.asc           100% 3403    1.6MB/s    00:00
debian-archive-buster-stable.asc             100% 1704    911.4KB/s   00:00
debian-archive-buster-security-automatic.asc  100% 11KB    6.6MB/s    00:00
debian-archive-bullseye-security-automatic.asc 100% 12KB    5.4MB/s    00:00
debian-archive-bookworm-automatic.asc        100% 12KB    5.4MB/s    00:00
listchanges.conf                             100% 150     73.2KB/s   00:00
sources.list~                               100%   0      0.0KB/s    00:00
sources.list                                100% 1054    597.0KB/s   00:00
```

```
hermann@debian:~/my-tests$ ssh hermann@deb-srv.mshome.net 'ls -l my-remote-dir'
insgesamt 8
drwxr-xr-x 9 hermann hermann 4096  4. Dez 16:13 apt
-rw-r--r-- 1 hermann hermann  219  4. Dez 16:04 users.txt
```