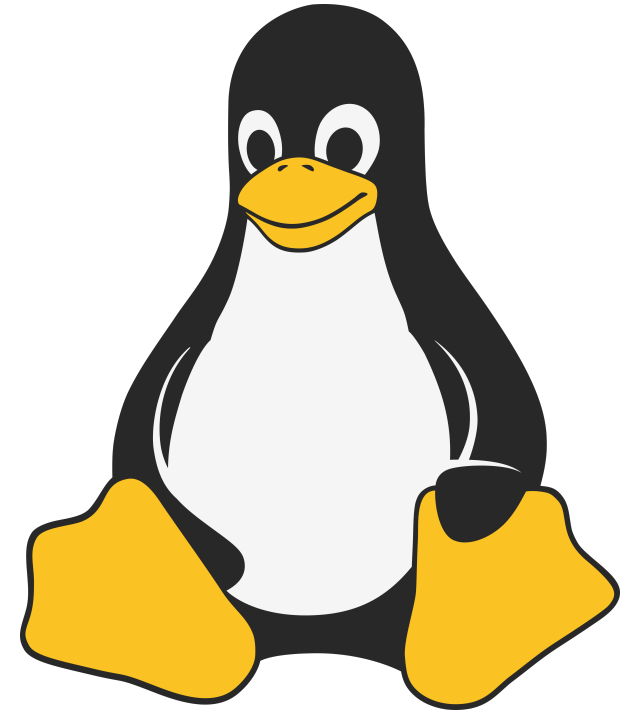


Passwortloses SSH-Login von **debian** zu **debsrv** und umgekehrt



Inhaltsverzeichnis

- [Voraussetzung](#)
- [Schlüssel-basiertes `ssh`-Login ohne Passwort](#)
- [Schlüsselpaar erzeugen mit `ssh-keygen`](#)
- [Public Key übertragen mit `ssh-copy-id`](#)
- [Rollenverständnis](#)

Voraussetzung

Wir können uns von `debian` auf `debsrv` und umgekehrt einloggen. Dazu müssen wir uns jedes Mal mit dem Passwort des Benutzers auf dem entfernten Host authentifizieren.

Schlüssel-basiertes `ssh`-Login ohne Passwort

- Das `ssh`-Login mit Passwort hat 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 das 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 (hier: RSA) und die Schlüssellänge (hier: 4096 Bits) anzugeben.
- Die beiden Schlüssel werden standardmäßig 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 (`debsrv`) übertragen und dort an die Datei `~/.ssh/authorized_keys` angehängt. 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-----`).
- Danach ist das passwortlose `ssh` -Login möglich.
- Auch auf dem Server bleibt das Verzeichnis `~/.ssh` für andere Benutzer unzugänglich (Rechte: `rwX-----`).

Schlüsselpaar erzeugen mit `ssh-keygen`

```
hermann@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 mit `ssh-copy-id`

- Der öffentliche Schlüssel `id_rsa.pub` wird an den Benutzer `hermann` auf dem Server `debsrv` übertragen und an `~/.ssh/authorized_keys` angehängt. Ein letztes Mal wird dabei das Passwort des Benutzers auf dem Server abgefragt.
- Bei der nächsten Anmeldung auf `debsrv` kann man sich ohne Passwort anmelden. Zur Authentifizierung wird dabei das Schlüsselpaar verwendet.

```
hermann@debian:~$ ssh-copy-id hermann@debsrv
/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@debsrv's password:
```

Number of key(s) added: 1

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

Schlüssel-basiertes **ssh**-Login ohne Passwort

```
hermann@debian:~$ ssh hermann@debsrv
Linux debsrv 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 `debsrv`

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

Rollenverständnis

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

Völlig analog zu obiger Beschreibung kann nun auch das passwortlose `ssh`-Login von `debsrv` auf `debian` eingerichtet werden.