Sichere Datenübertragung mit SSH und GPG

Im folgenden Abschnitt sollen das Datenübertragungs-Protokoll SSH sowie die Dateiverschlüsselung GPG anhand von einigen praktischen Beispielen vorgestellt werden.

`ssh` -- Arbeiten auf entfernten Rechnern

SSH steht für "Secure Shell" und ermöglicht sichere, verschlüsselte Netzwerkverbindungen mit anderen Computern.

Um sich -- beispielsweise in einem Heim-Netzwerk -- wechselseitig auf einem entfernten Rechner einloggen zu können, sollten auf beiden Rechnern folgende Pakete installiert werden:

```
sudo aptitude install openssh-client openssh-server
```

Hat man hingegen mehrere Clients, die unter einem jeweiligen Benutzernamen Zugriff auf einen Server haben sollen, so genügt es, bei diesen `openssh-client` und auf dem Server `openssh-server` zu installieren. Durch die Installation wird automatisch der SSH-Hintergrunddienst `sshd` gestartet; nach jedem Neustart des Rechners ist dieser ebenfalls aktiv.

Um sich von einem Client aus in einem externen Rechner einzuloggen, kann man in einer Shell folgendes eingeben:

```
ssh benutzername@ip-adresse
```

Die ip-Adresse von Rechnern in einem lokalen Netzwerk erhält man beispielsweise, indem man in einer Shell beispielsweise `nmap -sP 192.168.1.0/24` oder (als Superuser) `nast -m` eingibt.

Existiert für den angegebenen Benutzername auf dem Rechner und wird das Passwort (des Benutzers auf dem externen Rechner) richtig eingegeben, so werden alle folgenden Anweisungen auf dem entfernten Rechner ausgeführt, nur die Bildschirmausgabe erfolgt lokal. Durch Eingabe von `exit` kann die SSH-Sitzung wieder verlassen werden.

`scp` und `ssh -X`

Mittels `scp` kann via SSH eine einzelne Datei (oder ein einzelnes Verzeichnis) auf den entfernten Rechner kopiert werden. Die Syntax dafür lautet:

```
scp lokale-datei benutzername@ip-adresse:zielpfad
```

Um mehrere Dateien auf einmal zu kopieren, können diese entweder mittels :ref:`tar <tar>` gebündelt oder in ein Verzeichnis kopiert werden; mittels `scp -r` kann dieses Verzeichnis dann (wie eine einzelne Datei) rekursiv kopiert werden. Manche Dateimanager wie :ref:`mc <mc>` nutzen ebenfalls intern `scp`, um Dateien auf einfache Weise vom lokalen auf einen externen Rechner oder umgekehrt zu kopieren.

Möchte man auf dem entfernten Rechner nicht nur Shell-Programme, sondern auch graphische Programme aufrufen, kann `ssh` mit der Option `-X` aufrufen werden, also mittels:

```
ssh -X benutzername@ip-adresse
```

In diesem Fall bekommt man nach einer richtigen Passworteingabe ebenfalls ein Shell-Fenster angezeigt, kann in diesem allerdings auch graphische Programme aufrufen.

Anmeldung mit Public Key anstelle eines Passworts

Sicherer als Passwörter sind für die Authentifizierung eines Benutzers so genannte Schlüsselpaare: Ein privater Schlüssel auf dem eigenen PC, und ein öffentlicher Schlüssel, der an beliebig viele andere Stellen kopiert werden kann. Ein Login ist damit nur noch dann möglich, wenn der private und der öffentliche Schlüssel zusammenpassen.

Ein Schlüsselpaar kann in einer Shell folgendermaßen erzeugt werden:

```
mkdir ~/.ssh

ssh-keygen -t rsa
```

Zu Übungszwecken kann bei der Nachfrage nach einem Passwort einfach `Enter` eingegeben werden; der private Schlüssel wird somit nicht mit einem Passwort versehen.

Durch den Aufruf von `ssh-keygen` werden im Verzeichnis `~/.ssh` zwei Dateien angelegt: Die Datei `id_rsa` enthält den privaten Schlüssel, der nicht in falsche Hände geraten sollte, und die Datei `id_rsa.pub` ("public") den öffentlichen Schlüssel, der auf alle Rechner kopiert werden kann, auf denen man sich via SSH einloggen will.

Zum Kopieren des öffentlichen Schlüssels kann folgendes in einer Schell eingegeben werden:

```
ssh-copy-id benutzername@ip-adresse
```

Hierbei muss nochmals das Passwort des Benutzers auf dem Zielsystem eingegeben werden. Durch den Aufruf von `ssh-copy-id` wird der Standard-Schlüssel (oder durch Angabe von `-i pfad` eine explizit angegebene Schlüsseldatei) auf dem Zielrechner der Datei `~/.ssh/authorized_keys` hinzugefügt.

Gibt man anschließend `ssh benutzername@ip-adresse` ein, so erfolgt das Einloggen via Schlüsselpaar anstelle der Eingabe eines Passworts. [1]

Für ein wechselseitiges Verbinden zweier Rechner mittels SSH muss das oben beschriebene Verfahren auf beiden Rechnern erfolgen.

Aliases für häufige Login-Adressen

In der Datei `~/.ssh/config` können Kurzbezeichnungen für häufig besuchte externe Rechner vergeben werden. Um beispielsweise auf einen "Server" im Home-Netzwerk mit der lokalen Netzwerkadresse `192.168.1.100` zuzugreifen, fügt man der Datei `~/.ssh/config` folgenden Eintrag hinzu:

```
Host server
    HostName 192.168.1.100
    User benutzername
    IdentityFile ~/.ssh/id_rsa
```

Anschließend muss man nicht mehr `ssh benutzername@192.168.1.100` eingeben, um sich mit dem Server zu verbinden: Von nun an genügt es `ssh server` einzugeben.

Passwortschutz für private Schlüssel

Gelangt der private Schlüssel an eine eigentlich unbefugte Person, so kann sich auch diese ebenso unmittelbar wie ungewollt auf dem Zielrechner einloggen. Um zu verhindern, dass der alleinige "Besitz" des privaten Schlüssels ausreicht, kann man diesen mit einem Passwort versehen; bevor er für das Einloggen verwendet werden kann, muss er erst mittels des Passworts freigegeben werden.

Üblicherweise werden passwortgeschützte SSH-Schlüssel in Verbindung mit `ssh-agent` genutzt. Dieses Programm wird im Allgemeinen automatisch mit dem X-Server und/oder zu Begin einer Login-Shell gestartet und bleibt aktiv, bis sich der Benutzer wieder abmeldet. Beim der erstmaligen Verwendung des Schlüssels in einer laufenden Sitzung muss das Schlüssel-Passwort eingegeben werden; alle weiteren Zugriffe auf den Schlüssel sind anschließend erlaubt. [2]

Ein passwortgeschützter Schlüssel, beispielsweise `~/.ssh/id_rsa` kann folgendermaßen zur Schlüsselverwaltung mittels `ssh-agent` hinzugefügt werden:

```
ssh-add ~/.ssh/id_rsa
```

Wird `ssh-agent` ohne die explizite Angabe eines Schlüsselpfads gestartet, so werden automatisch alle im Verzeichnis `~/.ssh` liegenden Schlüssel hinzugefügt. Mittels `ssh add -l` können die von `ssh-agent` verwalteten Schlüssel angezeigt werden.

gpg -- Verschlüsselung von Dateien

GPG steht für "GNU Privacy Guard" und ist die wohl am weitesten verbreitete Implementierung von PGP ("Pretty Good Privacy"). Letzteres stellt einen Standard dar, mit dem Emails sicher verschlüsselt verschickt werden können, sofern die Software sowohl beim Sender wie auch beim Empfänger installiert ist.

GPG gehört bei fast allen Linux-Distributionen zum Standard, muss also nicht extra installiert werden.

GPG nutzt -- ebenso wie SSH -- zum Verschlüsseln der Daten ein Schlüsselpaar: Der private Schlüssel bleibt auf dem eigenen Rechner und kann zum Entschlüsseln von Nachrichten verwendet werden; der öffentliche Schlüssel hingegen wird üblicherweise frei verteilt. Mit ihm können Nachrichten an den Eigentümer des Schlüssels verschlüsselt, jedoch nicht entschlüsselt werden.

Zum Erstellen eines Schlüsselpaars gibt man in einer Shell folgendes ein:

```
gpg --gen-key
```

Hierbei wird man zunächst nach dem gewünschten Verschlüsselungsverfahren gefragt, wobei die Vorgabe "RSA und RSA" mit `1` ausgewählt werden kann. Als Schlüssellänge sollte man einen möglichst großen Wert nehmen -- 2048 Bit sind ok, 4096 Bit sind sicherer und somit besser. Als letztes muss man angeben, wie lange der Schlüssel gültig bleiben soll. Hier sollte durchaus eine Zeitvorgabe, beispielsweise `1y` für "1 Jahr" eingegeben werden, da Schlüssel ohne Verfallsdatum auch dann im Umlauf bleiben, wenn beispielsweise die zugehörige Emailadresse nicht mehr existiert oder die Schlüssellänge durch immer schnellere Rechner zu klein geworden ist. Anschließend muss man als eindeutige Benutzerkennung noch den Namen und die Emailadresse angeben, zu dem der Schlüssel gehören soll.

Das Passwort, das man für den Schlüssel vergibt, sollte zwar gut merkbar, aber zugleich ausreichend sicher sein; längere Passwörter, die beispielsweise aus ganzen Sätzen bestehen, sind in der Regel sicherer als kurze, aber kryptische und damit schwer zu merkende Passwörter. Ohne Passwortschutz des privaten Schlüssels kann jede Person, die Zugriff auf die Schlüsseldatei bekommt, alle mit dem zugehörigen öffentlichen Schlüssel gesicherten Dateien öffnen. [3]

$$a^2 + b^2 = c^2$$

---

1 Man kann in einem Heim-Netzwerk sogar, um die Sicherheit zu erhöhen, das Anmelden mittels Passworteingabe komplett verbieten. Dazu müssen (mit Superuser-Rechten) in der Datei `/etc/ssh/sshd_config` folgende Einträge vorgenommen werden:

```
PasswordAuthentication no
UsePAM no
```

Damit können sich nur noch Benutzer einloggen, deren öffentliche Schlüssel in der jeweiligen `~/.ssh/authorized_keys`-Datei stehen.

In der gleichen Datei sollte zudem ein Login als Root unbedingt verboten werden:

```
PermitRootLogin no
```

Gegebenenfalls kann ein :ref:`Benutzer mit Superuser-Rechten <su>` immer noch mit `sudo` systemweite Änderungen vornehmen oder sich mit `sudo su root` dauerhaft Superuser-Rechte verschaffen.

2 Die Freigabe gilt auch für andere Programme, sofern diese in der laufenden Sitzung vom gleichen Benutzer gestartet wurden.

3 Zusätzlich ist eine Kopie des privaten Schlüssels auf einem :ref:`verschlüsselten <Partitions-Verschlüsselung>` USB-Stick oder einer verschlüsselten externen Festplatte empfehlenswert!