## 24.14 Exkurs: SSH-Server/SSH-Client – Installation, Konfiguration und Test

Wenn Sie bei einer TCP/IP-Verbindung von Ihrem Computer zu einem entfernten Computer auf die folgenden Eigenschaften wert legen:

- Authentifizierung der Gegenstelle,
- Verschlüsselung der Datenübertragung und
- Datenintegrität,

dann lohnt die Installation eines SSH-Servers auf dem entfernten Computer. Das SSH-Protokoll ermöglicht eine verschlüsselte Verbindung von Computern und die Ausführung von Terminal-Befehlen auf dem entfernten Computer.

**Inhaltsverzeichnis**



### 24.14.1 Hinweise

Die folgenden Hinweise und Anregungen sollten Sie beachten:

- In der vorliegenden Beschreibung wird die Installation des OpenSSH-Servers beschrieben – der freien Implementierung von 'Secure Shell'.
- Ein SSH-Client wird bei der Installation von allen Linux-Distributionen automatisch installiert!
- Für viele administrative Aufgaben und Tests wird in einer Konsole der OpenSSH-Client `ssh` eingesetzt.
- Auf die Dokumentation zum OpenSSH-Server wird nur verwiesen.
- Verschlüsselte Terminal-Verbindungen sind besonders dann von großem Nutzen, wenn ein Server 'headless' eingerichtet wurde – also weder über Tastatur, Bildschirm noch grafischer Benutzeroberfläche verfügt.
- Das beschriebene Vorgehen in diesem Exkurs eignet sich nur für Verwendung innerhalb eines Heimnetzes.

### 24.14.2 Installation OpenSSH-Client

So können Sie feststellen, welche OpenSSH-Pakete bereits auf dem entfernten Computer installiert sind. Eine Ausgabe könnte so aussehen:

```
hans@mint20:~$ dpkg --list | grep openssh
ii  openssh-client 1:8.2p1-4ubuntu0.3 amd64 secure shell (SSH) client, for secure access to remote machines
hans@mint20:~$
```

Gut – ein SSH-Client ist auf diesem System bereits installiert, was der bereits erwähnte Standard bei Linux ist. Sonst ist ein SSH-Client schnell installiert:

```
$ sudo apt-get update
$ sudo apt-get upgrade
$ sudo apt-get install openssh-client
```

### 24.14.3 Installation OpenSSH-Server

Ein OpenSSH-Server ist bei Ubuntu und Mint standardmäßig nicht installiert. Er kann über die Anwendungsverwaltung installiert werden oder Sie geben in einem Terminal nacheinander folgende Zeilen ein:

```
$ sudo apt-get update
$ sudo apt-get upgrade
$ sudo apt-get install openssh-server
```

### 24.14.4 Steuerung des OpenSSH-Servers

Achtung! Der SSH-Server wird bei jedem Systemstart *automatisch* gestartet (Standard). Das können Sie mit diesen beiden Befehlen ändern:

```
$ sudo systemctl disable openssh     -- OpenSSH aus der Autostart-Liste entfernen
$ sudo systemctl enable openssh      -- OpenSSH der Autostart-Liste wieder hinzufügen
```

Wenn der SSH-Server beim Systemstart nicht gestattet wird, dann können Sie ihn selbst starten.

Die folgenden Aufrufe in einem Terminal steuern den SSH-Server mit den in der Liste aufgeführten Parametern:

```
$ sudo systemctl parameter ssh    {start|stop|reload|force-reload|restart|try-restart|status}
$ sudo service ssh parameter      {start|stop|reload|force-reload|restart|try-restart|status}
```

Beispiele:

```
$ sudo service ssh stop
$ sudo systemctl stop ssh         -- Alternativ bei einem ubuntu-basierten System
```

Der SSH-Server-Dienst sollte nach Abschluss der Installation bereits aktiv sein. So können Sie das nachprüfen:

```
$ service ssh status       -- Nur bei der Statusabfrage ohne erhöhte Berechtigung
● ssh.service - OpenBSD Secure Shell server
     Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: enabled)
     Active: active (running) since Mon 2021-11-29 15:28:49 CET; 2h 56min ago
       Docs: man:sshd(8)
             man:sshd_config(5)
    Process: 682 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)
   Main PID: 733 (sshd)
      Tasks: 1 (limit: 18939)
     Memory: 3.2M
     CGroup: /system.slice/ssh.service
             └─733 sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups
Nov 29 15:28:49 pc-mint20 systemd[1]: Starting OpenBSD Secure Shell server...
Nov 29 15:28:49 pc-mint20 sshd[733]: Server listening on 0.0.0.0 port 22.
Nov 29 15:28:49 pc-mint20 sshd[733]: Server listening on :: port 22.
Nov 29 15:28:49 pc-mint20 systemd[1]: Started OpenBSD Secure Shell server.
```

### 24.14.5 Dokumentation

Eine umfangreiche Dokumentation zu OpenSSH finden Sie unter:

```
https://www.openssh.com/manual.html
```

### 24.14.6 Log-Dateien

Log-Einträge des OpenSSH-Servers finden Sie in der Datei /var/log/auth.log. Hier ein Auszug:

```
Nov 21 10:36:46 mint20 sshd[7016]: Server listening on 0.0.0.0 port 22.
Nov 21 10:36:46 mint20 sshd[7016]: Server listening on :: port 22.
```

### 24.14.7 Verbindungsdaten

Wenn Sie eine SSH-Verbindung von Ihrem Computer mit einem SSH-Client zu einem entfernten Com-

puter mit einem SSH-Server aufbauen wollen, dann benötigen Sie folgende Daten:

- IP-Adresse des SSH-Server-Hosts oder in einer DHCP-Umgebung mit DNS-Server dessen Hostnamen und
- die Konto-Daten (Name, Passwort) eines System-Benutzers auf dem SSH-Server-Hotsts oder
- ein Schlüsselpaar, bestehend aus einem öffentlichen und einem privaten Schlüssel (Option).

Mit folgenden Befehlen ermitteln Sie auf dem SSH-Server-Host oder SSH-Client-Host die IPv4-Adresse und die IPv6-Adressen

```
hans@pc-mint20:~$ hostname -I | grep -oE '([[:digit:]]{1,3}\.){3}[[:digit:]]{1,3}'     # IPv4
192.168.0.2
hans@pc-mint20:~$ hostname -I | grep -oE '([a-fA-F0-9]{1,4}\:){7}[a-fA-F0-9]{1,4}'     # IPv6
2a02:8109:b8bf:f2ec:2010:2d22:8ecb:31d0
2a02:8109:b8bf:f2ec:e101:fa83:9d1:7b1c
2a02:8109:b8bf:f2ec:bfb0:6aab:b6f5:f9ea
```

Einen Überblick über alle IP-Adressen und Host-Namen in Ihrem (Heim-)Netzwerk erhalten Sie, wenn Sie einen Blick in die Netzwerk-Konfiguration Ihres Routers werfen.

### 24.14.8 SSH-Verbindung mit Passwort-Authentifizierung

Eine Verbindung zum SSH-Server kann etwa mit folgendem Befehl aufgebaut werden, wobei die IP-Adresse des Servers und ein System-Benutzername auf diesem – hier `hans` – verwendet wird:

```
honsek@laptop-mint20:~$ ssh hans@192.168.0.2
The authenticity of host '192.168.0.2 (192.168.0.2)' can't be established.
ECDSA key fingerprint is SHA256:HqdkscKY8DwNMCqw3Xj2ga5pWQ3VyGpD5rzLttRLLsQ.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
```

Beim allerersten Login wird der Fingerprint des Public Keys des SSH-Servers angezeigt – hier als blau markierter SHA-256-Hash des Public-ECDSA-Keys. Wenn sich der entfernte Computer mit dem SSH-Server in Ihrem Heimnetz befindet, können Sie in diesem Fall auf eine Prüfung des Fingerprints verzichten und den Dialog mit 'yes' fortsetzen. Danach werden Sie zur Eingabe des System-Benutzerpassworts auf dem entfernten Computer aufgefordert:

```
honsek@laptop-mint20:~$ ssh 192.168.0.2
The authenticity of host '192.168.0.2 (192.168.0.2)' can't be established.
ECDSA key fingerprint is SHA256:HqdkscKY8DwNMCqw3Xj2ga5pWQ3VyGpD5rzLttRLLsQ.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.0.2' (ECDSA) to the list of known hosts.
hans@192.168.0.2's password:
hans@pc-mint20:~$
```

Beachten Sie die letzte Zeile – Sie sehen den Prompt des Servers!

Mit der Eingabe des Benutzernamens in hans@192.168.0.2 und des Passworts haben Sie Ihre Identität nachgewiesen, die der SSH-Server anhand der ihm bekannten Konto-Daten (Benutzername, Passwort) prüft und danach Zugang zum gesicherten Bereich gewährt. Damit ist eine SSH-Verbindung zwischen dem SSH-Server und dem SSH-Client hergestellt.

Sie können den SSH-Server jetzt so administrieren, als hätten Sie dort eine Konsole geöffnet. Das wird sofort getestet, indem exemplarisch ein find-Befehl auf dem entfernten Computer ausgeführt wird:

```
hans@pc-mint20:~$ find /etc/ssh -type f -name '*conf*'
/etc/ssh/ssh_config
/etc/ssh/sshd_config
/etc/ssh/sshd_config.d/sshd.conf
hans@pc-mint20:~$ exit
Abgemeldet
Connection to 192.168.0.2 closed.
honsek@laptop-mint20:~$
```

Zum Beenden der SSH-Verbindung geben Sie `exit` ein und befinden sich danach wieder in der Konsole auf dem SSH-Client-Host. Bei allen weiteren Verbindungsversuchen vom gleichen Client verzichtet der SSH-Server auf Warnungen, da der Client den Host bei der ersten SSH-Verbindung in der Datei ~/.ssh/known_hosts als bekannt registriert.

Da man es in einem Heimnetz oft mit einer DHCP-Umgebung zu tun hat, in der sich IP-Adressen ändern können, bietet sich für ein Login auf dem entfernten Computer alternativ zur IP-Adresse die Verwendung des Hostnamen an. Ein Login mit dem Hostnamen sieht dann beispielsweise so aus:

```
honsek@laptop-mint20:~$ ssh hans@pc-mint20
```

### 24.14.9 Überprüfung des SSH-Server-Fingerprints

Wenn Sie sich mit dem SSH-Client-Host an einem anderen Ort oder in einer anderen Netzwerk-Domäne befinden, dann sollten Sie zur Sicherheit und vor dem Aufbau der ersten SSH-Verbindung einen prüfenden Blick auf den Fingerprint des Public-Keys des SSH-Servers werfen. Nur so können Sie sicherstellen, dass Sie sich mit dem richtigen SSH-Server verbinden.

Den Fingerprint des Public-Server-Keys ermitteln Sie vorort und direkt auf dem SSH-Server-Host mit folgendem Konsolen-Befehl:

```
hans@pc-mint20:~$ ssh-keygen -l -f /etc/ssh/ssh_host_ecdsa_key.pub
256 SHA256:HqdkscKY8DwNMCqw3Xj2ga5pWQ3VyGpD5rzLttRLLsQ root@mint20 (ECDSA)
hans@pc-mint20:~$
```

Besteht keine direkte lokale Zugriffsmöglichkeit zum SSH-Server-Host, dann verbinden Sie sich – wie oben beschrieben – vorort über eine SSH-Verbindung und notieren Sie sich für den Fall einer Erstverbindung den angezeigten Fingerprint oder geben Sie den o.a. Konsolen-Befehl in einer Konsole auf dem Client ein.

SSH-Verbindungen über das Internet können ebenfalls mit hoher Sicherheit betrieben werden, allerdings ist die Einrichtung kritisch und schließt auch die Firewall mit ein. Auf eine Beschreibung dieser Variante wird hier verzichtet.

### 24.14.10 SSH-Verbindung mit Public-Key-Authentifizierung

Alternativ zur SSH-Verbindung mit Passwort-Autentifizierung, kann man den Server und Client auch so einrichten, dass die Verbindung ohne Eingabe eines Passworts erfolgt. Dies gilt allgemein als die sicherere Form und wird durch die Bekanntmachung von öffentlichen Schlüsseln oder Public Keys und entsprechend angepassten Konfigurationen erreicht. Der folgende Abschnitt baut auf den oben beschriebenen Einrichtungen von SSH-Server und SSH-Client auf und verwendet die damit erworbenen Fähigkeiten.

#### 24.14.10.1 Erzeugung eines Schlüssel-Paares (Private-Key und Public-Key)

Um Ihre Identität gegenüber dem SSH-Server nachweisen zu können, erzeugen Sie zunächst auf dem SSH-Client ein Schlüssel-Paar. Wählen Sie im Dialog eine geeignete Passphrase, die Sie sich merken müssen! Dies stellt sicher, dass die erzeugten Schlüssel verschlüsselt gespeichert werden.

```
honsek@laptop-mint20:~$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/hans/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/hans/.ssh/id_rsa
Your public key has been saved in /home/hans/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:/ouoLhznowzUXi6tMeMhR0uv3h4XsEz3LYVc2y+2Oxg honsek@laptop-mint20
The key's randomart image is:
+---[RSA 3072]----+
|           .     |
|         . o o   |
|     o . o o .   |
|  . o + . o   .  |
| . + + .So . o . |
|. = B  .. .E. o  |
|.o & = ..   o.   |
| o* % o. o . ..  |
|  +O+=. . o. ..  |
+----[SHA256]-----+
honsek@laptop-mint20:~$
```

Danach schränken Sie den Zugriff auf den privaten Schlüssel auf den aktuellen System-Benutzer ein:

```
honsek@laptop-mint20:~$ chmod 700 .ssh/id_rsa
```

### 24.14.10.2 Anpassung der SSH-Server-Konfiguration

Stellen Sie eine SSH-Verbindung per Passwort-Authentifizierung zum SSH-Server her und starten dort in der Konsole mit erhöhten Rechten den Editor `sudo nano /etc/ssh/sshd_config` mit der Konfigurationsdatei als Parameter.

Prüfen Sie mit

```
sudo nano /etc/ssh/sshd_config
```

, ob in der Konfigurationsdatei /etc/ssh/sshd_config die Zeile

```
PubkeyAuthentication yes
```

ohne führendes # steht. Sonst ändern Sie diese Zeile. Vergessen Sie nach dem Speichern der geänderten SSH-Server-Konfiguration nicht, den SSH-Server neu zu starten. Schließen Sie die bestehende SSH-Verbindung.

```
honsek@laptop-mint20:~$ ssh hans@192.168.0.2
hans@192.168.0.2's password:
Last login: Tue Nov 30 17:56:50 2021 from 192.168.0.3
hans@pc-mint20:~$ sudo nano /etc/ssh/sshd_config
hans@pc-mint20:~$ sudo service ssh restart
hans@pc-mint20:~$ exit
honsek@laptop-mint20:~$
```

### 24.14.10.3 Übertragung des öffentlichen Schlüssels zum SSH-Server

So kopieren Sie den öffentlichen Schlüssel von Ihrem SSH-Client-Host zum SSH-Server in das angegebene versteckte Verzeichnis:

```
honsek@laptop-mint20:~$ ssh-copy-id -i ~/.ssh/id_rsa.pub hans@192.168.0.2
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/hans/.ssh/id_rsa.pub"
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new keys
hans@192.168.0.2's password:

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh 'hans@192.168.0.2'"
and check to make sure that only the key(s) you wanted were added.

honsek@laptop-mint20:~$
```

Nach der Eingabe des Passwortes des Systembenutzers wird ein Fenster eingeblendet. Hier müssen Sie die Passphrase eingeben, die Sie beim Erzeugen des Schlüssel-Paares festgelegt hatten. Außerdem können Sie festlegen, ob das Passwort permanent entsperrt werden soll. Die Aufforderung, sich jetzt testweise zum SSH-Server zu verbinden, setzen wir um:

```
honsek@laptop-mint20:~$ ssh hans@192.168.0.2
Last login: Tue Nov 30 17:56:50 2021 from 192.168.0.3
hans@pc-mint20:~$ exit
Abgemeldet
Connection to 192.168.0.2 closed.
honsek@laptop-mint20:~$
```

Wie Sie feststellen werden, ist zum Nachweis Ihrer Identität auf dem Server kein Passwort mehr erforderlich, da jetzt auf den hinterlegten Schlüssel zurückgegriffen wird.

### 24.14.11 Anpassung der SSH-Client-Konfiguration

Legen Sie auf Ihrem Computer die Konfigurationsdatei config an

```
$ sudo xed ~/.ssh/config
```

und geben Sie den folgenden, beispielhaften Inhalt ein. Zeilen, die mit einem '#' beginnen sind deaktiviert. Wie Sie feststellen, ist es möglich mehrere Hosts zu konfigurieren. Beachten Sie die fett markierte Zeile, in der Sie einen beliebigen Bezeichner für den Host festlegen können. Dieser kann stellvertre-

tend beim Verbindungsaufbau verwendet werden. Nur in einer DHCP-Umgebung darf beim Parameter 'HostName' statt der IP-Adresse der Hostname eintragen werden. Wenn feste IP-Adressen verwendet werden, ist dort unbedingt die IP-Adresse anzugeben!

```
# 2021-11-20
#
# Aus c't 2018, Heft 14, S. 144 ff
# Inhalt von 'config' in ~/.ssh
#
Host h150
     HostName pc-mint20
     Port 22
     User claus
     IdentityFile ~/.ssh/id_rsa
#
#Other Hosts
#Host ems
#    HostName 192.168.1.10
#    Port 22
#    User pi
#    IdentityFile ~/.ssh/id_rsa
#
#Host NAS
#     HostName 192.168.1.2
#     Port 22
#     User admin
#     IdentityFile ~/.ssh/id_rsa
```

#### 24.14.11.1 Test

In Zukunft sollte Ihr Login mit diesem Befehl in einer Konsole gelingen:

```
$ ssh h150
```

Alternativ können Sie auch noch die bekannte Syntax verwenden, wobei Sie allerdings nicht mehr nach einem Passwort gefragt werden:

```
$ ssh hans@192.168.0.2
$ ssh hans@pc-mint20
```

### 24.14.12 Erweiterung – Dateiübertragung

Das Protokoll SSH sorgt zum Beispiel nicht nur für eine sichere Übertragung von Konto-Daten (Benutzer-Name, Passwort), sondern auch für eine sicherere Datenübertragung. Das Konsolen-Programm für die sichere Dateiübertragung vom einem System auf ein entferntes Ziel-System per SSH heißt `scp` (SecureCopy) und wird auf dem SSH-Server ausgeführt.

Es nutzt die folgende Syntax zum sicheren Kopieren einer Datei:

```
$ scp client-pfad-zur-orginal-datei [ziel-user-name]@[ziel-ip/hostname]:pfad-auf-dem-entfernten-ziel-system
```

Im folgenden Beispiel kopieren Sie als System-Benutzer `honsek` vom SSH-Client-Host mit dem Hostnamen `laptop-mint20` die Text-Datei `~/Schreibtisch/fernzugriff_via_scp.txt` auf ein entferntes Ziel-System mit der IP-Adresse `192.168.0.2` in dessen (Home-)Verzeichnis `/home/hans`.

```
honsek@laptop-mint20:~$ scp /home/honsek/Schreibtisch/fernzugriff_via_scp.txt hans@192.168.0.2:/home/hans
fernzugriff_via_scp.txt                                     100% 2131   517.2KB/s   00:00
honsek@laptop-mint20:~$
```

Bei voll eingerichteter Public-Key-Authentication gelingt das auch mit:

```
honsek@laptop-mint20:~$ scp /home/honsek/Schreibtisch/fernzugriff_via_scp.txt h150:/home/hans
fernzugriff_via_scp.txt                                     100% 2131   517.2KB/s   00:00
honsek@laptop-mint20:~$
```

Das sichere Kopieren funktioniert nur dann, wenn Sie auch ein Konto auf dem SSH-Server besitzen, da das Passwort des System-Benutzers dort abgefragt wird oder ohne Passwort, wenn Public-Key-Authentication eingerichtet wurde.

Alternative 1
Mit dieser Zielort-Eingabe im Datei-Manager `nemo` von Linux Mint

```
ssh://192.168.0.2/home/hans/dbt
sftp://192.168.0.2/home/hans/dbt
```

stellen Sie eine sichere FTP-Verbindung (SFTP) zu einem SSH-Server mit der Adresse `192.168.0.2` her und sehen den Inhalt vom Verzeichnis `/home/hans/dbt`. Jetzt können Sie dort eine Datei markieren, kopieren und im eigenen Home-Verzeichnis in einem ausgewählten Verzeichnis ablegen oder umgekehrt.

Alternative 2
Starten Sie im Dateimanager unter 'Datei/Mit Server verbinden' den folgenden Dialog und tragen Sie alle relevanten Daten ein, wobei Sie bei 'Server:' dessen IP-Adresse, Hostnamen oder den Alias-Namen aus der Datei ~/.ssh/config eingeben können:

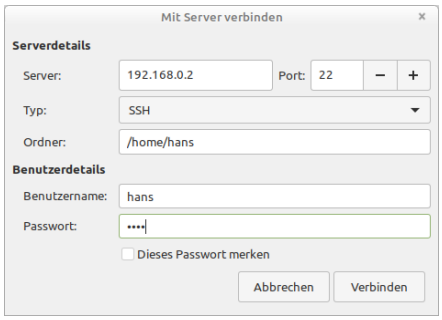


Abbildung 24.14.12.1: Dialog – Mit Server verbinden

Nach dem erfolgreichen Verbinden sehen Sie im Dateimanager das über das Netzwerk eingehängte (Remote-)Home-Verzeichnis. Nun können Sie wechselseitig Dateien sicher kopieren. Nach dem Kopieren hängen Sie den entfernten Datenträger wieder aus oder legen diese SSH-Verbindung als Lesezeichen ab.

### 24.14.13 Erweiterung – X11-Forwarding

Ein SSH-Server stellt nicht nur eine Konsole auf dem entfernten Computer zur Verfügung, sondern auch die grafische Benutzeroberfläche über die Konsole gestarteter Programme. Die Display-Umleitung oder X11-Forwarding übernimmt der SSH-Server, wenn er entsprechend konfiguriert wurde. In der Konfigurationsdatei /etc/ssh/sshd_config muss X11 Forwarding durch folgende Zeile erlaubt werden:

```
X11Forwarding yes
```

Nach einer Änderung der Konfigurationsdatei muss ein Restart des SSH-Servers erfolgen:

```
honsek@laptop-mint20:~$ ssh hans@192.168.0.2
Last login: Tue Nov 30 17:56:50 2021 from 192.168.0.3
hans@pc-mint20:~$ sudo nano /etc/ssh/sshd_config
[sudo] Passwort für hans:
hans@pc-mint20:~$ sudo service ssh restart
hans@pc-mint20:~$ exit
```

Entscheidend für die Aktivierung der Display-Umleitung ist die Angabe des Parameters -X bei der Verbindungsaufnahme. Sie könnten dann zum Beispiel die Gambas-IDE auf dem entfernten Rechner starten und dort aus der Ferne programmieren. Das Gambas-Fenster sehen Sie jedoch nahtlos auf Ihrem lokalen Desktop.

```
honsek@laptop-mint20:~$ ssh -X 192.168.0.2
hans@192.168.0.2's password:
Last login: Thu Nov 25 18:41:52 2021 from 192.168.0.3
hans@pc-mint20:~$ gambas3
hans@pc-mint20:~$ xed
hans@pc-mint20:~$ thunderbird
hans@pc-mint20:~$ beaver       # Datenbank-Manager
hans@pc-mint20:~$ exit
Abgemeldet
Connection to 192.168.0.2 closed.
honsek@laptop-mint20:~$
```

Wie Sie sehen, wurden auch weitere Programme auf dem entfernten Computer gestartet, von denen bekannt ist, dass sie dort installiert sind. Beachten Sie jedoch, dass diese Form der Verbindung keine Sound-Signale unterstützt.