OpenSSH

Ein Vortrag

über

„OpenSSH“

© BS-LUG 2020 by Marius Schwarz

OpenSSH

„Das Secure Shell (SSH) Protokoll ist ein Protokoll für sicheres Einloggen und andere sichere Netzwerkdienste über ein unsicheres Netzwerk.“

(RFC 4251)

© BS-LUG 2020 by Marius Schwarz

OpenSSH

Was kann SSH alles?

© BS-LUG 2020 by Marius Schwarz

OpenSSH

- per Shell in einen (entfernten) Server einloggen

© BS-LUG 2020 by Marius Schwarz

OpenSSH

- per Shell in einen (entfernten) Server einloggen
- per SFTP-Erweiterung Dateien auf oder von einem Server kopieren

© BS-LUG 2020 by Marius Schwarz

OpenSSH

- per Shell in einen (entfernten) Server einloggen
- per SFTP-Erweiterung Dateien auf oder von einem Server kopieren
- Ports eines entfernten Servers lokal zur Verfügung stellen

© BS-LUG 2020 by Marius Schwarz

OpenSSH

- per Shell in einen (entfernten) Server einloggen
- per SFTP-Erweiterung Dateien auf oder von einem Server kopieren
- Ports eines entfernten Servers lokal zur Verfügung stellen
- lokale Ports auf einem entfernten Server zur Verfügung stellen

© BS-LUG 2020 by Marius Schwarz

OpenSSH

- per Shell in einen (entfernten) Server einloggen
- per SFTP-Erweiterung Dateien auf oder von einem Server kopieren
- Ports eines entfernten Servers lokal zur Verfügung stellen
- lokale Ports auf einem entfernten Server zur Verfügung stellen
- ein verschlüsseltes VPN aufbauen

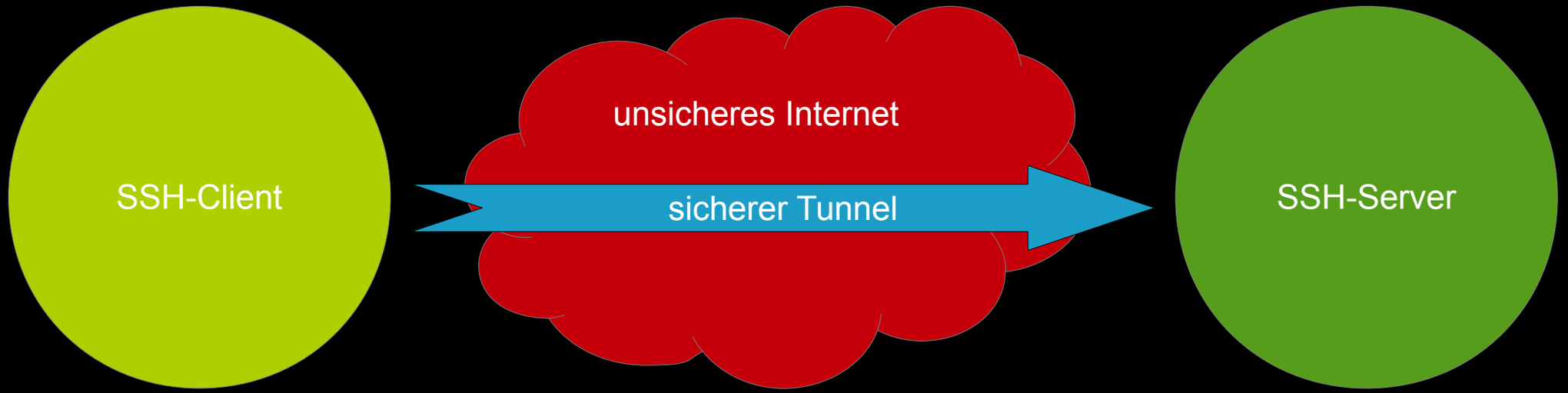
© BS-LUG 2020 by Marius Schwarz

OpenSSH

- per Shell in einen (entfernten) Server einloggen
- per SFTP-Erweiterung Dateien auf oder von einem Server kopieren
- Ports eines entfernten Servers lokal zur Verfügung stellen
- lokale Ports auf einem entfernten Server zur Verfügung stellen
- ein verschlüsseltes VPN aufbauen
- eine verschlüsselte Bridge zwischen Netzwerken bauen

© BS-LUG 2020 by Marius Schwarz

OpenSSH



© BS-LUG 2020 by Marius Schwarz

OpenSSH

Inbetriebnahme des Servers

© BS-LUG 2020 by Marius Schwarz

OpenSSH

Abhängig von der genutzten Distribution:

systemctl start sshd
+ „systemctl enable sshd“, damit es auch anbleibt.

oder /etc/init.d/sshd start

© BS-LUG 2020 by Marius Schwarz

OpenSSH

Der SSH Login

© BS-LUG 2020 by Marius Schwarz

OpenSSH

```
[user@pc1 ~]$ ssh root@s113
Last failed login: Mon Nov  9 10:36:58 CET 2020 from 81.68.230.252 on ssh:notty
There were 214 failed login attempts since the last successful login.
Last login: Sun Nov  8 19:31:48 2020 from 45.123.98.12
[root@server ~]# ls -la /tmp/
insgesamt 52
drwxrwxrwt  17 root root  560  9. Nov 10:41 .
dr-xr-xr-x. 22 root root 4096  5. Nov 11:22 ..
drwxrwxrwt   2 root root   40  5. Nov 11:32 .font-unix
drwxr-xr-x   2 root root   60  9. Nov 10:41 hsperfdata_root
drwxrwxrwt   2 root root   40  5. Nov 11:32 .ICE-unix
drwx------   3 root root   60  5. Nov 11:32 systemd-private-61e85821b18e4959b36a29afc6d7a387-chronyd.service-8QEAig
drwx------   3 root root   60  5. Nov 11:32 systemd-private-61e85821b18e4959b36a29afc6d7a387-dbus-broker.service-zfPu1e
drwx------   3 root root   60  5. Nov 11:32 systemd-private-61e85821b18e4959b36a29afc6d7a387-systemd-logind.service-vyMQRf
drwxrwxrwt   2 root root   40  5. Nov 11:32 .Test-unix
drwxrwxrwt   2 root root   40  5. Nov 11:32 .X11-unix
drwxrwxrwt   2 root root   40  5. Nov 11:32 .XIM-unix
```

© BS-LUG 2020 by Marius Schwarz

OpenSSH

Die SSH-Login Kennung:

Benutzername@Servername

© BS-LUG 2020 by Marius Schwarz

OpenSSH

```
[root@server ~]# cat /etc/ssh/sshd_config

Protocol 2

SyslogFacility AUTHPRIV

PasswordAuthentication yes
ChallengeResponseAuthentication no
GatewayPorts clientspecified
PermitTunnel yes

LogLevel info

GSSAPIAuthentication yes
GSSAPICleanupCredentials yes

UsePAM yes
PermitRootLogin without-password

AcceptEnv LANG LC_CTYPE LC_NUMERIC LC_TIME LC_COLLATE LC_MONETARY LC_MESSAGES
AcceptEnv LC_PAPER LC_NAME LC_ADDRESS LC_TELEPHONE LC_MEASUREMENT
AcceptEnv LC_IDENTIFICATION LC_ALL
X11Forwarding no

Subsystem       sftp    /usr/libexec/openssh/sftp-server
```

© BS-LUG 2020 by Marius Schwarz

OpenSSH

„PermitRootLogin without-password“

erlaubt den Root-Login nur mit einem OpenSSH-Schlüssel*

Das verhindert einen Brute-Force-Wörterbuchangriff auf den Rootuser

© BS-LUG 2020 by Marius Schwarz

OpenSSH

„GatewayPorts clientspecified
PermitTunnel yes“

erlaubt es Tunnel auf dem Server zu öffnen.

Das ist wichtig für VPN und Port-Forwarding.

© BS-LUG 2020 by Marius Schwarz

OpenSSH

SSH SFTP-Server

© BS-LUG 2020 by Marius Schwarz

OpenSSH: SFTP

```
[root@client ~]$ scp ¹545 - The Land That Time Forgot.txtª  root@server:/tmp/
545 - The Land That Time Forgot.txt                          100%  288KB   1.4MB/s   00:00

[root@client ~]$ ssh root@server "ls -la /tmp/*txt"
-rw-r--r-- 1 root root 294886  9. Nov 11:46 /tmp/545 - The Land That Time Forgot.txt
[root@client ~]$
```

Wie man im Beispiel erkennen kann,
wird der Pfad direkt an den Login angefügt.

© BS-LUG 2020 by Marius Schwarz

OpenSSH: SFTP

Allgemeine SSH-Loginform

Benutzername@server[:/Pfad/]

© BS-LUG 2020 by Marius Schwarz

OpenSSH: SFTP

Kommt das wem bekannt vor?

© BS-LUG 2020 by Marius Schwarz

OpenSSH: SFTP

Vielleicht jetzt?

HTTP://Benutzername@server:Port/Pfad/

© BS-LUG 2020 by Marius Schwarz

OpenSSH: SFTP

```
[root@client ~]$ scp ¹545 - The Land That Time Forgot.txtª  root@server:/tmp/
545 - The Land That Time Forgot.txt                          100%  288KB   1.4MB/s   00:00

[root@client ~]$ ssh root@server "ls -la /tmp/*txt"
-rw-r--r-- 1 root root 294886  9. Nov 11:46 /tmp/545 - The Land That Time Forgot.txt
[root@client ~]$
```

Merke: eine nicht-interaktive Befehlsausführung ist direkt möglich.

© BS-LUG 2020 by Marius Schwarz

OpenSSH

SSH Port-Forwarding

© BS-LUG 2020 by Marius Schwarz

OpenSSH: Port-Forwarding

Prämisse:

Wir möchten einen Server *sicher* kontaktieren,
der nicht direkt vom eigenen PC erreichbar ist,
aber von einem anderen Server erreicht werden kann.

© BS-LUG 2020 by Marius Schwarz

OpenSSH: Port-Forwarding

Hier im Beispiel soll dies der Webserver von heise.de sein.

© BS-LUG 2020 by Marius Schwarz

OpenSSH: Port-Forwarding

Die Serverseite:

```
[root@client ~]# $ ssh -L 4443:193.99.144.80:443 root@server
Last login: Mon Nov  9 11:18:08 2020 from 3.15.1.26
[root@server ~]# $
```

heise.de hat u.a. die IP: 193.99.144.80

© BS-LUG 2020 by Marius Schwarz

OpenSSH: Port-Forwarding

Die Clientseite:

```
[root@client ~]# $ netstat -lnap|grep 443
(Es konnten nicht alle Prozesse identifiziert werden; Informationen über
nicht-eigene Processe werden nicht angezeigt; Root kann sie anzeigen.)
tcp        0      0 127.0.0.1:4443          0.0.0.0:*               LISTEN      10871/ssh

[root@client ~]# $ curl --insecure  -I https://127.0.0.1:4443/
HTTP/1.1 301 Moved Permanently
Server: nginx
Date: Mon, 09 Nov 2020 10:23:52 GMT
Content-Type: text/html; charset=iso-8859-1
Connection: keep-alive
X-Cobbler: servo65.heise.de
X-Pect: The Spanish Inquisition
X-Clacks-Overhead: GNU Terry Pratchett
X-42: DON©T PANIC
Location: https://www.heise.de/
[root@client ~]#
```

© BS-LUG 2020 by Marius Schwarz

OpenSSH: Port-Forwarding

Dem lokalen Klienten PC steht jetzt eine verschlüsselte Verbindung über den SSH-Server zum eigentlichen Ziel zur Verfügung.

© BS-LUG 2020 by Marius Schwarz

OpenSSH

SSH REVERSE-Port-Forwarding

© BS-LUG 2020 by Marius Schwarz

OpenSSH: REVERSE-Port-Forwarding

ssh **-R** 4443:zielip:4443 root@Server

Öffne auf dem Server einen Port 4443
und verbinde diesen mit dem lokalen Port 4443 auf der IP ZielIP.

Wenn sich jemand zu dem Server auf Port 4443 verbindet,
landet diese Verbindung auf dem lokalen PC.

Die Option „-g“ erlaubt dann den Zugriff auch von außerhalb des Servernetzes.

© BS-LUG 2020 by Marius Schwarz

OpenSSH

SSH TUNNEL

© BS-LUG 2020 by Marius Schwarz

OpenSSH: Tunnel

Um einen Tunnel im Sinne des Kernels aufbauen zu können,

muß man der anderen Seite sagen,

welche Tunnel-Kanäle benutzt werden sollen.

© BS-LUG 2020 by Marius Schwarz

OpenSSH: Tunnel

Der SSH-Server muß entsprechend konfiguriert sein.

(siehe Seite 14 uf.)

© BS-LUG 2020 by Marius Schwarz

OpenSSH: Tunnel

```
ssh -NTCf -w 0:0 root@server
```

© BS-LUG 2020 by Marius Schwarz

OpenSSH: Tunnel

“-NTCf“

-N Keinen Befehl ausführen
-T kein Terminal anbinden
-C Kompression aktivieren
-f SSH in den Hintergrund schicken

“-w 0:0“ benutze Tunnel ID 0 hier und Tunnel ID 0 dort

Damit lassen sich mehrere verschiedene Tunnel zum Server erzeugen.

© BS-LUG 2020 by Marius Schwarz

OpenSSH: Tunnel

Die Serverseite:

```
[root@server ~]# modprobe tun
[root@server ~]# ip link set tun0 up
[root@server ~]# ip addr add 10.0.1.1/32 peer 10.0.1.2 dev tun0
[root@server ~]# echo 1 > /proc/sys/net/ipv4/ip_forward
[root@server ~]# iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
[root@server ~]#
```

© BS-LUG 2020 by Marius Schwarz

OpenSSH: Tunnel

„modprobe tun“

Lade TUN Kernelmodul

„ip link set tun0 up“

Fahre ein Netzwerkdevice für Tunnel 0 hoch

„ip addr add 10.0.1.1/32 peer 10.0.1.2 dev tun0“

Setze eine nicht-öffentliche IP und sage dem Tunnel, wer sein Partner ist.

„echo 1 > /proc/sys/net/ipv4/ip_forward“

Erlaube dem Kernel, IPv4-Pakete von einem Interface zum Anderen zu transportieren.

„iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE“

Aktivieren NAT (Network-Address-Translation), damit IPs aus dem Quellnetz zu IPs des Servers umgesetzt werden. Die Antworten kommen sonst nicht an.

© BS-LUG 2020 by Marius Schwarz

OpenSSH: Tunnel

Die Clientseite:

```
[root@client ~]# echo "1" >/proc/sys/net/ipv6/conf/all/disable_ipv6
[root@client ~]# modprobe tun
[root@client ~]# ip link set tun0 up
[root@client ~]# ip addr add 10.0.1.2/32 peer 10.0.1.1 dev tun0
[root@client ~]# route add SSH-SERVERIP gw GATEWAY-IP
[root@client ~]# route del default gw GATEWAY-IP
[root@client ~]# route add default gw 10.0.1.1 dev tun0
```

© BS-LUG 2020 by Marius Schwarz

OpenSSH: Tunnel

„echo "1" >/proc/sys/net/ipv6/conf/all/disable_ipv6“

Schalte IPv6 aus, weil sonst IPv6 Traffic nicht durch den Tunnel geht. Alternativ: zusätzlich IPv6 Tunnel aufbauen!

„modprobe tun
ip link set tun0 up;
ip addr add 10.0.1.2/32 peer 10.0.1.1 dev tun0;“

siehe Server, aber mit vertauschten IPs!

„route add SSH-SERVERIP gw GATEWAY-IP“

der VPN-Servertraffic darf natürlich nicht durch den Tunnel gehen. Die IPs hängen von Euren Gegebenheiten ab!
(!! Bei Nichtbeachtung droht Kollaps der Wellenfunktion des Universums !!)

„route del default gw GATEWAY-IP“

die normale Route zum Internet kappen!

„route add default gw 10.0.1.1 dev tun0“

durch den Tunnel neue Route zum Internet setzen!

© BS-LUG 2020 by Marius Schwarz

OpenSSH: Tunnel

Glückwunsch!

© BS-LUG 2020 by Marius Schwarz

OpenSSH: Tunnel

Ihr abhörsicheres VPN steht jetzt zur Verfügung!

© BS-LUG 2020 by Marius Schwarz

OpenSSH

Die Netzwerk-Brücke

© BS-LUG 2020 by Marius Schwarz

OpenSSH

© BS-LUG 2020 by Marius Schwarz

OpenSSH

Die Netzwerk-Brücke ist eigentlich eine Vorstufe des VPNs.

Alles was man machen muß ist, den Gatewaywechsel zu lassen.

© BS-LUG 2020 by Marius Schwarz

OpenSSH: Tunnel

Die Serverseite:

```
[root@server ~]# modprobe tun
[root@server ~]# ip link set tun0 up
[root@server ~]# ip addr add 10.0.1.1/32 peer 10.0.1.2 dev tun0
[root@server ~]# echo 1 > /proc/sys/net/ipv4/ip_forward
[root@server ~]# iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
[root@server ~]#
```

© BS-LUG 2020 by Marius Schwarz

OpenSSH: Tunnel

Die Clientseite:

```
[root@client ~]# echo "1" >/proc/sys/net/ipv6/conf/all/disable_ipv6
[root@client ~]# modprobe tun
[root@client ~]# ip link set tun0 up
[root@client ~]# ip addr add 10.0.1.2/32 peer 10.0.1.1 dev tun0
[root@client ~]# route add -net N.E.T.Z/24 dev tun0
[root@client ~]#
```

© BS-LUG 2020 by Marius Schwarz

OpenSSH

Das war es schon.

© BS-LUG 2020 by Marius Schwarz

OpenSSH

Fragen?

© BS-LUG 2020 by Marius Schwarz