Ich habe nach einer Lösung gesucht, hinter einer Firewall mit gesperrtem SSH-Port administrativ auf einen SSH-Server zuzugreifen.

Die Lösung ist relative einfach. Man verbindet sich über einen Port, der garantiert nicht geschlossen ist. Dieser ist Port 443(SSL). Der Server muss jetzt nur noch unterscheiden, ob die Anfrage eine SSL oder SSH-Anfrage ist und auf den entsprechenden internen Port weiterleiten.

Vorbereitung des Servers

Da wir hier mit verschiedenen Ports herumspielen, sollten die betroffenen Serverdienste gestoppt werden.

```
/etc/init.d/apache2 stop
/etc/init.d/ssh stop
```

Installation

```
aptitude install sslh
```

sslh einrichten

In der Datei `/etc/default/sslh` müssen ein paar Änderungen vorgenommen werden.

Damit der sslh-Daemon überhaupt startet, muss man die Zeile

```
RUN=yes
```

eintragen.

des weiteren muss man die DAEMON_OPTS Zeile an die eigenen Bedürftnisse anpassen. Hinter dem Parameter `-p` muss man den Namen des Servers eintragen, unter dem er aus dem Internet erreichbar ist.

Wenn alles erledigt ist, kann man sslh starten

```
/etc/init.de/sslh start
```

Ob der Daemon wirklich läuft, kann man per `netstat -ltunp` herausfinden. Dort müsste etwas wie

```
tcp        0      0 85.214.76.238:443       0.0.0.0:*               LISTEN
31017/sslh
```

stehen

SSH Konfigurieren

Man muss sicherstellen, dass der SSH-Daemon auf dem Port 127.0.0.1 lauscht. Deshalb muss man in der Datei `/etc/ssh/sshd_config` folgendes zusätzlich eintragen.

```
ListenAddress=127.0.0.1
```

Will man über mehrere Adressen auf den SSH-Server zugreifen, einfach die Zeile ListenAddress mehrfach einfügen.

Jetzt noch den SSH-Daemon starten und testen, ob man per SSH über den Port 443 auf den Rechner zugreifen kann.

Apache2 konfigurieren

in der Datei `/etc/apache2/ports.conf` muss man alle SSL-Verbindungen auf den localhost umleiten.

ports.conf

```
<IfModule mod_ssl.c>
    Listen 127.0.0.1:443
</IfModule>
```

Jetzt muss man noch den Apache-Server starten und alles funktioniert wieder wie gehabt.

Über `netstat -ltunp` kann man jetzt lesen, wie alles funktioniert.

```
tcp        0      0 85.214.76.238:443       0.0.0.0:*               LISTEN
31017/sslh
tcp        0      0 127.0.0.1:22            0.0.0.0:*               LISTEN
30834/sshd
tcp        0      0 127.0.0.1:443           0.0.0.0:*               LISTEN
30983/apache2
```

Quellen

- http://wandzeitung.informations-compagnie.de/2011/09/02/sslh-let-https-ssh-and-openvpn-share-a-single-port/
- http://forum.ubuntuusers.de/topic/sslh/

- http://bernaerts.dyndns.org/linux/210-debian-sslh

From:
https://wiki.da-checka.de/ - **PSwiki**

Permanent link:
https://wiki.da-checka.de/doku.php/wiki/dienste/sslh

Last update: **2012/11/12 08:09**