Reverse SSH-tunnel - Access your home server without router configuration

Release 1.1.0

Jens Getreu

Mar 16, 2017

Contents:

Accessing your home server from outside your local network is usually done by forwarding a port of your server through the router. This note describes a different approach allowing to establish a peer-to-peer connection between hosts on different private networks without having access to the router.

Routers support several technologies to provide access from the Internet to your local network. The most common solution consists of configuring various services from within the router requiring administrator access:

1. Assign a fix local IP address to your home server via DNS.
2. Forward a port on your router to the home server's port.
3. Subscribe to a free dynamic DNS service on the Internet and configure the router to use it.

The solution suggested in this note gets along without any router configuration! All you need is an external *OpenSSH* server (hereafter referred to as "gateway") with a public IP address somewhere on the Internet.

Technically, the home server establishes a reverse SSH tunnel to the gateway server. The mobile computer (hereafter called the “laptop”) connects to the gateway with an SSH tunnel. Finally both tunnels are interconnected.

The underlying technology is known as “hole punching.” and is one of the most effective methods of establishing peer-to-peer communication between hosts on different private networks. *[srisuresh2008]* documents the theoretical aspects of hole punching for both UDP and TCP, and details the crucial aspects of both application and NAT behavior that make hole punching work.

Network topology.:

```
[HomeServer:22]
    ↓
(NAT)[router]
    ↓
[Gateway:14321]
    ↑
[router](NAT)
    ↑
[Laptop:24321]
```

1 Configure the servers

1.1 Configure the *Gateway*-server

Gateway’s `OpenSSH`-server listens on port 22 and must have a static public IP or a DNS-domain-name, here `gateway.myownserver.org`. If the gateway has no static IP use some dynamic DNS service.

1. Login *Gateway*:

   ```
   Gateway# sudo nano /etc/ssh/sshd_config
   ```

2. Add to or change in `/etc/ssh/sshd_config`:

   ```
   ClientAliveInterval 30
   ClientAliveCountMax 99999
   GatewayPorts yes
   AllowTcpForwarding yes
   Port 22
   ```

   Note: The prompt `$` means the command can be executed as a normal user, the prompt `#` means the command must be executed as `root`.

3. Add user:

   ```
   Gateway# adduser sshgateway
   ```

4. Restart ssh server:

   ```
   Gateway# service ssh restart
   ```

1.2 Test tunnel from *HomeServer* to *Gateway*

HomeServer's ssh-server listens at port 22. *HomeServer* is behind a NAT.

Binds `[HomeServer:22]` → `[Gateway:14321]`

1. Open reverse tunnel:

```
HomeServer$ ssh -p 22 -fNC -R 14321:localhost:22 sshgateway@gateway.
↪myownserver.org
```

2. Check tunnel:

```
ClientBc$ ps x | grep autossh
839 ?        Ss     0:00 /usr/lib/autossh/autossh -p 22 -NC  -R␣
↪14321:localhost:22 sshgateway@gateway.myownserver.org

Gateway# netstat -a | grep 14321
tcp     0    0 *:14321         *:*         LISTEN
tcp6    0    0 [::]:14321      [::]:*      LISTEN
```

1.3 Configure the *HomeServer*-server

The following needs to be executed only once.

1. Install key on *Gateway*:

```
HomeServer$ apt-get install autossh
HomeServer$ ssh-keygen
HomeServer$ ssh-copy-id -i ~/.ssh/id_rsa.pub -p 22 sshgateway@gateway.
↪myownserver.org
```

2. Make tunnel persistent:

```
HomeServer$ crontab -e
```

Add the following line (all in one line)

crontab -e

```
@reboot  autossh -p 22 -fNC -R 14321:localhost:22 sshgateway@gateway.
↪myownserver.org
```

3. Reboot:

```
HomeServer# reboot
```

2 Establish tunnel from *Laptop* to *Gateway*

Laptop is behind a NAT.

Binds `[Gateway:14321]` ← `[Laptop:24321]`

1. Open tunnel:

```
Laptop$ ssh -p 22 -fNL  24321:localhost:14321 sshgateway@gateway.
↪myownserver.org
```

2. Check tunnel:

```
Gateway# netstat -a | grep 14321
...
tcp6    0    0 localhost:37109  localhost:14321   ESTABLISHED
tcp6    0    0 localhost:14321  localhost:37109   ESTABLISHED


Laptop# netstat -a | grep 24321
...
tcp     0    0 localhost:24321  *:*               LISTEN
tcp6    0    0 localhost:24321  [::]:*            LISTEN
tcp6    0    0 localhost:24321  localhost:48788   ESTABLISHED
tcp6    0    0 localhost:48788  localhost:24321   ESTABLISHED
```

3 Connect *Laptop* to *HomeServer*

Connects via `[HomeServer:22] ← [Laptop:24321]`:

```
Laptop$ slogin -l HOME_SERVER_USER -p 24321 localhost
```

4 Further reading

Reverse SSH Tunnel – Schritt für Schritt

Reverse SSH Tunneling

References

[srisuresh2008] P. Srisuresh, B. Ford, and D. Kegel, “State of peer-to-peer (P2P) communication across network address translators (NATs),” 2008.