# SSH-туннели — пробрасываем порт

Настройка Linux

Не всегда есть возможность, да и не всегда надо, строить полноценный туннель с интерфейсной парой адресов. Иногда нам нужно лишь «прокинуть» вполне определённые порты.

Тут важно понимать, что туннель можно организовать как изнутри сети, к ресурсам которой вы хотите получить доступ, на внешний ssh-сервер. Также можно организовать туннель с хоста в Интернете на пограничный ssh-сервер сети, чтобы получить доступ к внутренним ресурсам.

Итак. По-порядку.

**Строим туннель из сети в мир.**

```
$ ssh -f -N -R 2222:10.11.12.13:22 username@99.88.77.66
```

теперь введя на хосте 99.88.77.66:

```
$ ssh -p2222 localhost
```

мы попадём на хост 10.11.12.13.

Таким-же образом можно получить доступ к любому другому ресурсу, например:

```
$ ssh -f -N -R 2080:10.11.12.14:80 username@99.88.77.66
```

Введя на хосте 99.88.77.66:

```
$ w3m -dump http://localhost:2080
```

получим дамп web-ресурса на 10.11.12.14.

**Строим туннель из мира в сеть.**

```
$ ssh -f -N -L 4080:192.168.0.10:80 nameuser@88.77.66.55
```

Аналогично, вводим на своём хосте:

```
$ w3m -dump http://localhost:4080
```

и получаем доступ к web-ресурсу узла 192.168.0.10, который находится за хостом 88.77.66.55.

Поддерживаем туннели в поднятом состоянии
Ни для кого не секрет, что связь иногда обрывается, туннели при этом будут отваливаться по таймауту.
Чтобы не утруждать себя дополнительным монотонным вбиванием команды на поднятие туннеля и мониторингом этого процесса, автоматизируем его. Смело вводим:

$ crontab -e

и создаём расписание примерно следующего вида:

```
TUNCMD1='ssh -f -N -R 2222:10.11.12.13:22 username@99.88.77.66'
TUNCMD2='ssh -f -N -R 2080:10.11.12.14:80 username@99.88.77.66'

*/5 * * * * pgrep -f "$TUNCMD1" &>/dev/null || $TUNCMD1
*/5 * * * * pgrep -f "$TUNCMD2" &>/dev/null || $TUNCMD2
```

Сохраняемся. Проверяем по

```
$ crontab -l
```

что расписание принято.

Это лишь ещё один момент особой админской магии… Надеюсь, что лишних вопросов не должно водникнуть. С дополнительными опциями ssh можно ознакомиться в

```
$ man 1 ssh
```

*По практическому опыту — cron-задания на перезапуск абсолютно недостаточно. Разве что соединение абсолютно стабильно. В реальной жизни встречается в 0% случаев.*

*Даже соединённые напрямую кабелем две сетевые карты легко могут потерять n-ное количество пакетов и tcp-соединение «упадёт».*
*Клиент и сервер будут пребывать в святой уверенности, что всё в порядке, просто вторая сторона ничего не передаёт.*
*Нужен keepalive.*
*Примерно так:*

```
TCPKeepAlive yes
ServerAliveInterval 300
ServerAliveCountMax 3
```

*Интервал и счётчик — по вкусу.*
*Добавлять их надо либо в /etc/ssh_config, либо в ~/.ssh/config, либо прямо в команде через опцию -o.*
*В принципе, судя по man ssh_config, первую из опций можно и опустить. но, на всякий случай, пусть будет.*