

Если вы видите что-то необычное, просто сообщите мне.

Проброс портов в туннеле ssh

Создаем постоянное подключение к удаленном серверу.

Требования:

1. Удаленный доступ должен иметь настроенный ssh сервер
2. Локально должен быть доступен ключ id_rsa
3. Публичная часть ключа должна быть на удаленном сервере

Настройки:

Создаем файл `/etc/systemd/system/SERVICENAME.service`
с содержанием вида

```
[Unit]
Description=SSH tunnel

[Service]
ExecStart=bash -c "ssh -i /PATH/TO/id_rsa -L 0.0.0.0:9999:localhost:8010 USER@REMOTE.HOST -N"
Restart=on-failure
EnvironmentFile=/etc/environment
```

```
[Install]
```

```
WantedBy=multi-user.target
```

Мы говорим, что на хосте `REMOTE.HOST` по адресу `localhost:8010` доступно приложение, и его мы мапируем на локальный компьютер на порт 9999.

Теперь удаленно приложение висящее на порту 8010, доступно локально, но на порту 9999. Любое подключение идущее на `localhost:9999` компьютера на котором запускается сервис будет уходить на `REMOTE.HOST` через ssh туннель.

`0.0.0.0:9999` - подключение будет доступно не только на localhost но и еще в локальной сети компьютера на котором запущен текущий сервис. `0.0.0.0` можно не писать.

Запускаем/останавливаем сервис:

```
systemctl start/stop SERVICENAME.service
```

`start/stop` - выберите что-то одно, или запускаем или останавливаем

Если демон просит перезагрузить конфиги - перезагружаем

Включаем сервис при загрузке:

```
systemctl enable SERVICENAME.service
```

Проверяем состояние сервиса на наличие ошибок:

```
systemctl status SERVICENAME.service
```

или

```
journalctl -u SERVICENAME
```

Немного теории

Описываю реальную ситуацию: есть удаленный сервер (назовем его `remoteserver`), на нем крутятся несколько контейнеров, один из них база данных (пусть будет `docker_mysql_1`). Разработчику (его комп пусть будет `developer`), находящемуся в одной локалке с девопсом (его хост - `devops`) понадобился доступ к порту `mysql` на том удаленном сервере. Из них двоих только у девопса есть доступ до сервера (`remoteserver`) - по `ssh`.

То есть, разработчик хочет запустить приложение (которое запросит хост, порт, логин, пароль), укажет этому приложению настройки и оно должно соединиться с базой данных, которая находится на удаленном сервере в контейнере, порт которого не проброшен.

Конечно, можно добавить проброс порта в `docker-compose` или перезапустить отдельный этот контейнер с нужными параметрами, но как сделать все это не трогая контейнер, не перезапуская и не устанавливая ни одной программы ни на одной машине.

Ответ - последняя команда этой страницы.

Теперь по порядку: `ssh` может создавать туннели, пробрасывая внутренние порты сервера на машину, с которой была запущена команда.

Например, следующая команда, если ее выполнит девопс, открыла бы на его машине (`devops`) порт `9999`, который был бы связан с портом `3306` (`mysql`) на сервере `remoteserver`. Этот порт был бы доступен только ему, так как открылся бы на интерфейсе `localhost`.

```
ssh -L 9999:localhost:3306 user@remoteserver
```

Если для подключения используется нестандартный порт, то не нужно забывать это указывать:

```
ssh -p 32323 -L 9999:localhost:3306 user@remoteserver
```

Но мне кажется, что хорошей практикой было бы включить такие настройки в `.ssh/config`.

Далее, если нам не нужен сам `ssh`-туннель, а только доступ к порту, то мы можем не запускать интерпретатор:

```
ssh -p 32323 -L 9999:localhost:3306 user@remoteserver -N
```

Тут мы вспомнили, что ломиться нужно не на localhost, а на определенный хост, доступный ему по сети (виртуальной или реальной - не важно). Смотрим ip-адрес нужного нам контейнера через

```
docker inspect docker_mysql_1
```

и затем указываем полученный ip-адрес в команде.

```
ssh -p 32323 -L 9999:172.31.0.2:3306 user@remoteserver -N
```

Теперь у нас все получилось и девопс выполняя команду `mysql -h localhost --port 9999` он попадает на порт mysql'a, который работает в контейнере `docker_mysql_1`, запущенном на `remoteserver`.

Осталось дать доступ разработчику. Следующая команда разрешает пользоваться данным портом через все интерфейсы, а значит он доступен не только девопсу, но и разрабу (команда уже будет `mysql -h devops --port 9999`):

```
ssh -p 32323 -L 0.0.0.0:9999:172.31.0.2:3306 user@remoteserver -N
```

Revision #3

Created 13 October 2022 13:53:43 by gasick

Updated 17 October 2022 10:55:38 by gasick