Port Forwarding & Tunneling (cliente)

tunneling

- puerto local (3306) mapeado a puerto de <server> (3306) a través de SSH:

```
ssh <user>@<SERVER> -N -L 3306:127.0.0.1:3306
```

  - -N : no ejecuta comandos, se queda abierto
  - -L : es el encargado de establecer la conexión
  - se puede usar **-f** para dejar SSH en segundo plano
  - si la conexión a <SERVER> está definida en ~/.ssh/config, usar directamente el identificador

-
```
Host MasterServerSQL
    Hostname x
    User y
    IdentityFile ~/.ssh/id_ed25519
    LocalForward 3306 localhost:3306
    ControlPath ~/.ssh/control-%h-%p-%r
```

  - si usamos **ControlPath**, podemos conectar como `ssh -MNf MasterServerSQL`, y desconectar con `ssh -O exit MasterServerSQL` (mirar MasterControl + MasterPath para más detalles, ControlMaster no deja de ser un método de conexión compartida)

-
```
Host Dreamhost-mysql-tunnel
    Hostname x
    User y
    IdentityFile ~/.ssh/id_ed25519
    LocalForward 3310 <remote_server>:3306
    ControlPath ~/.ssh/control/control-%h-%p-%r
```

  - mapeamos en local el puerto 3310 a remote_server:3306, que está accesible desde el servidor al que conectamos (y@x)

Port Forwarding + Tunneling

- conectar via SSH desde máquina local a <SERVER_B> pasando por <SERVER_C> (no accesible desde local):

```
ssh <user>@<SERVER_C> -L localport:<SERVER_B>:remoteport
```

- conectar a <SERVER_B> pasando por <SERVER_C>:

```
ssh <user>@<SERVER_C> -N -L localport:<SERVER_B>:remoteport
```

  - deja conexión abierta para usar otro programa (atacando a localhost:localport)

From:
https://matebcn.eu/ - **miguel angel torres egea**

Permanent link:
https://matebcn.eu/linux:ssh:localportforwarding

Last update: **21/11/2023 10:38**