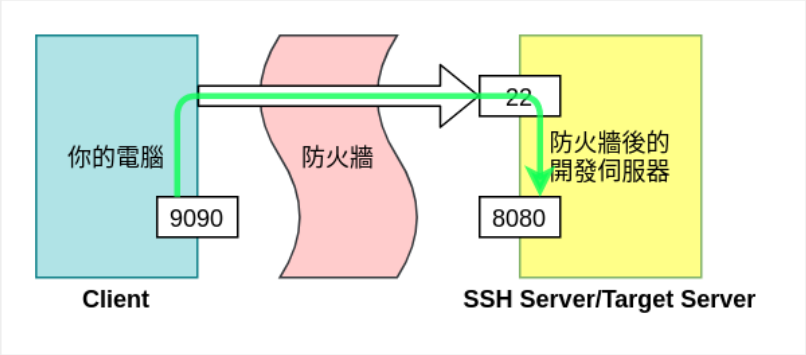
SSH Tunneling (Port Forwarding)

Local Port Forwarding:

```
ssh -L [bind_address:]<port>:<host>:<host_port> <SSH Server>
```

ssh -L [bind_address:]<port>:<host>:<host_port> <SSH Server>



Client

- 你的電腦

SSH Server

- 防火牆後的伺服器
- SSH Destination： `johnliu@my-server`

Target Server

- 防火牆後的伺服器

SSH 指令：

```
ssh -L 9090:localhost:8080 johnliu@my-server
```

這邊的 `localhost` 是相對於 `johnliu@my-server` ，指的就是防火牆後的伺服器本身。

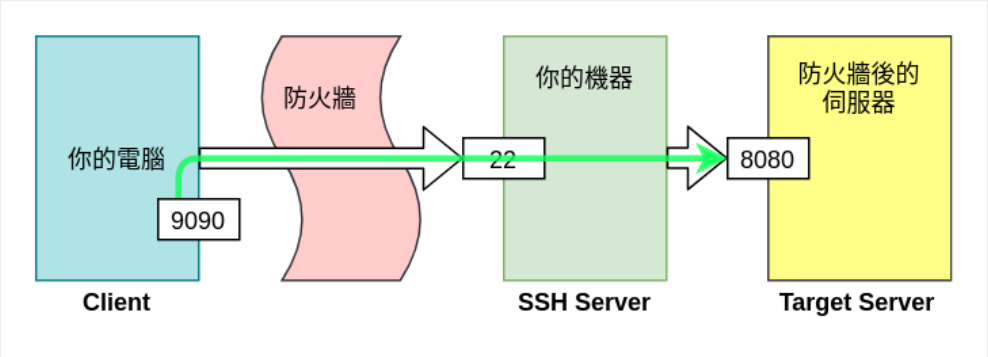
註釋

- 你完全可以在你的電腦上用相同的 Port number 來做 Port Forwarding，這邊用 `9090` 只是為了避免混淆：

  ```
  ssh -L 8080:localhost:8080 johnliu@my-server
  ```

- 如果你沒有給 `bind_address` ，預設會 Bind 在 `localhost` 上。如果你想把 Port `9090` 開放給所有人用：

  ```
  ssh -L 0.0.0.0:9090:localhost:8080 johnliu@my-server
  ```



Client

- 你的電腦

SSH Server

- 防火牆後你的機器
- SSH Destination： `johnliu@my-server`

Target Server

- 防火牆後的伺服器
- `192.168.1.101:8080`

SSH 指令：

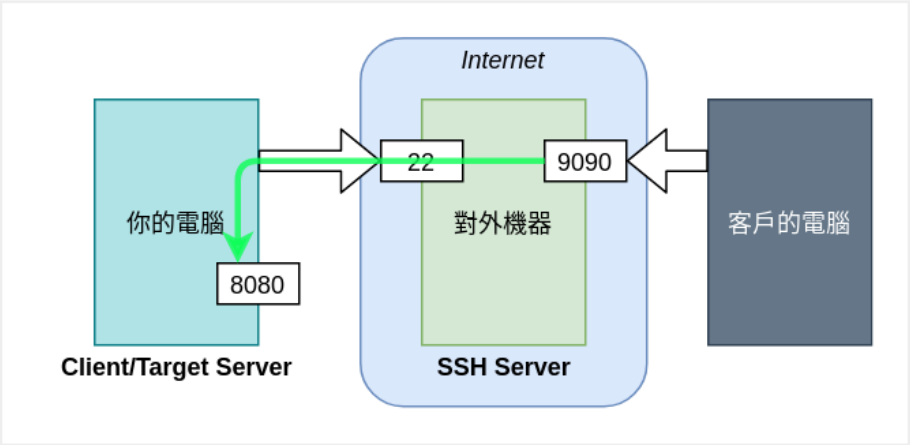
```
ssh -L 9090:192.168.1.101:8080 johnliu@my-server
```

這邊的 `192.168.1.101` 是相對於 `johnliu@my-server` ，所以是防火牆後的伺服器的 IP 位址。

Remote Port Forwarding:

```
ssh -R [bind_address:]<port>:<host>:<host_port> <SSH Server>
```

ssh -R [bind_address:]<port>:<host>:<host_port> <SSH Server>



Client

- 你的電腦

SSH Server

- 對外機器
- SSH Destination： `johnliu@external-server`

Target Server

- 你的電腦

SSH 指令：

```
ssh -R 0.0.0.0:9090:localhost:8080 johnliu@external-server
```

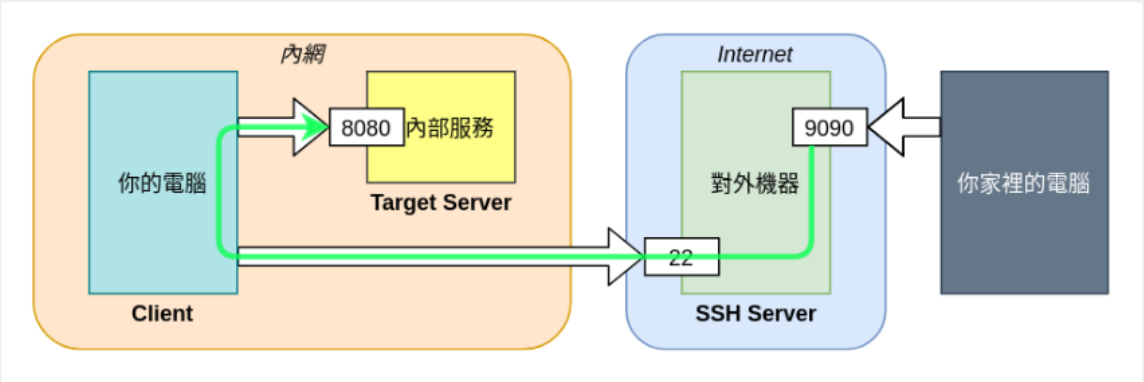
這邊的 `localhost` 是相對於 **Client** ，指的就是你的電腦本身。

⚠ 警告

基於安全考量， **Remote Forwarding 預設都只能夠 bind 在 SSH Server 的 localhost 上**，所以單靠以上指令是無法讓 Port 9090 開放給外部連線的。你必須調整 SSH Server 上的 SSH 服務的設定檔（一般在 `/etc/ssh/sshd_config` ）加入 `GatewayPorts` 設定，才能讓所有人都連到：

```
GatewayPorts yes
```

這邊有三個選項：預設為 `no` ，也就是唯一指定 localhost；設定為 `yes` 可以唯一指定為 wildcard（ `0.0.0.0` ）；設定為 `clientspecified` 可以讓啟動 Remote Forwarding 的 Client 自行指定。



Client

- 你的電腦

SSH Server

- 對外機器
- SSH Destination： `johnliu@external-server`

Target Server

- 內部服務
- `192.168.1.100:8080`

SSH 指令：

```
ssh -R 0.0.0.0:9090:192.168.1.100:8080 johnliu@external-server
```

常用的 SSH 指令參數

`-N`

不要執行任何遠端指令。沒有加這個參數時，建立 Port Forwarding 的同時也會開啟 Remote Shell，讓你可以對 SSH Server 下指令，而這個參數可以讓 Remote Shell 不要打開。

`-f`

讓 `ssh` 指令在背景執行，讓你可以繼續用 Shell 做事情。通常會搭上面的 `-N` 使用。

常用的 SSH Client 端設定

> **註釋**
>
> 設定檔通常在 `~/.ssh/config` 或是 `/etc/ssh/ssh_config` 。

`ServerAliveInterval`

設定一段時間，如果 Client 在這段時間內都沒從 SSH Server 收到資料，就發出一段訊息請 SSH Server 回應。這會讓連線不會呈現閒置狀態，避免防火牆或 Router 切斷你的連線。預設為 `0` ，不會發出任何訊息。

`ServerAliveCountMax`

設定在 SSH Server 沒回應的情況下，Client 最多要送幾次請求回應的訊息（上面提到的那個）。達到此次數後，Client 就會切斷與 SSH Server 之間的連線。這個主要是避免在 SSH Server 已經無法連線後，Client 還不斷送出請求回應的情況。預設為 `3` 。

autossh：自動重啟 SSH 連線

autossh 是一支可以幫你監控 SSH 連線狀態並自動重連的程式。如果你的網路狀況很糟糕，或是防火牆會三不五時把你斷線，他可以幫你自動重啟連線。

Fail2Ban：阻擋不明連線

Fail2Ban 可以幫你阻擋不明連線，原理就是去監看 SSH 服務的 log 來偵測登入失敗的 IP，然後在這些 IP 的失敗次數達到一定值時，利用防火牆來暫時停止該 IP 的連線請求，過一定時間後再恢復。可以拿來擋掉最基本的暴力攻擊。

如果你租了線上主機來玩，建議最少要裝 Fail2Ban 來保護你的 SSH Server。

Port Knocking：有條件的開啟 SSH Port

Port Knocking 指的是 Client 必須用特殊的順序來對 SSH Server 上的某些 Port 發出連線請求後，SSH Server 才會開放 Client 連線的技巧（比如依序對 Port 1000、2000、3000 發出請求，才會對你開放 Port 22）。這樣的好處是平時 Port 22 就會是關閉的狀態，讓攻擊者以為 SSH 沒有開放，減少被攻擊的機會。我沒用過，但看起來會搭配其他服務（像 knockd ）一起用。