Ejecutar un túnel inverso mediante SSH tiene ciertas utilidades. Entre ellas:

PERMITIR QUE EL SERVIDOR SSH REMOTO ACCEDA A UN SERVICIO DE NUESTRO ORDENADOR/SERVIDOR

PAGINA WEB

Si estamos sirviendo una página web en nuestro ordenador y queremos que el ordenador con un servidor SSH remoto pueda acceder a esa web como si estuviera publicada en el puerto 23456 de sí mismo, ejecutaríamos desde el servidor donde está corriendo nuestro servicio web:

```
ssh -R localhost:23456:localhost:80 root@dominioquenoesnuestro.com
```

Lógicamente, si el servidor SSH remoto está corriendo en un puerto diferente al 22, ejecutaríamos:

```
ssh -R localhost:23456:localhost:80 root@dominioquenoesnuestro.com -p 11122
```

De esta forma, el ordenador que esté corriendo el servidor SSH remoto, podrá acceder a la web que tengamos publicadas en nuestro puerto 80 ingresando la siguiente URL en un navegador:

```
localhost:23456
```

ESCRITORIO REMOTO

Si queremos que la persona que administra un servidor SSH remoto se conecte por RDP a nuestro escritorio, podemos ejecutar:

```
ssh -R 63389:localhost:3389 root@dominioquenoesnuestro.com
```

Lógicamente, si el servidor SSH remoto está corriendo en un puerto diferente al 22, ejecutaríamos:

```
ssh -R 63389:localhost:3389 root@dominioquenoesnuestro.com -p 11122
```

De esta forma, el ordenador que esté corriendo el servidor SSH remoto, podrá acceder y administrar nuestro escritorio abriendo su cliente de RDP y conectándose a la dirección:

```
localhost:63389
```

PERMITIR QUE EL SERVIDOR SSH REMOTO ACCEDA A UN SERVICIO EN OTRO ORDENADOR DE NUESTRA SUBRED

La publicación de arriba es posible hacerla, incluso desde un ordenador diferente al que sirve la página web, pero que esté en la misma subred. Por ejemplo, si el servidor que sirve la página web está en la IP 192.168.0.10 y nosotros tenemos un portátil con la IP 192.168.0.11, entonces ejecutaríamos desde nuestro portátil:

```
ssh -R localhost:23456:192.168.0.10:80 root@dominioquenoesnuestro.com -p 11122
```

De esa forma, daríamos acceso a la página web alojada en el servidor de nuestra subred a un tercero que navegaría por ella como si estuviera conectado mediante la IP de nuestro portátil.

DAR ACCESO A UN TERCERO PARA QUE HAGA CAMBIOS EN NUESTRO ROUTER

De forma similar a los pasos anteriores, podemos conectarnos al servidor SSH de un amigo para darle acceso a la web de configuración de nuestro router. Lo haríamos ejecutando en nuestro ordenador:

```
ssh -R 11111:192.168.0.1:80 usuario@88.88.88.88
```

...donde **192.168.1.1** es la dirección IP de nuestro router, **88.88.88.88** es la dirección IP pública de la casa de nuestro amigo y 11111 es el puerto mediante el cual va a poder acceder a la web del router. Es decir, para acceder a nuestro router, nuestro amigo, en su ordenador, tendrá que ingresar la URL localhost:11111.

En este ejemplo vemos que eralmente no hace falta poner el primer localhost. Es decir, la orden:

```
ssh -R 11111:192.168.0.1:80 usuario@88.88.88.88
```

...sería equivalente a la orden:

```
ssh -R localhost:11111:192.168.0.1:80 usuario@88.88.88.88
```

Y eso es así para todos los otros ejemplos.

PUBLICAR UN SERVICIO PROPIO EN UNA IP WAN DIFERENTE A LA NUESTRA

Esto es un poco complicado de entender, pero vamos a intentarlo:

Imaginemos que tenemos en casa un servidor de Minecraft Java Edition (que utiliza el puerto 25565) y queremos que sea accesible mediante el nombre de dominio **servidoresgaming.com** sin que esté realmente en la IP del servidor de ese dominio o en cualquiera de las IPs de los ordenadores/servidores de su subred. Podríamos hacerlo si tenemos acceso SSH al servidor de servidoresgaming.com. Para ello, desde el servidor donde esté corriendo Minecraft Server, crearíamos el primer túnel, ejecutando:

```
ssh -R [bind_address:]port:host:hostport usuario@servidoresgaming.com
```

...y una vez dentro de ese túnel, y en la terminal remota (es decir, la del servidor servidoresgaming.com), ejecutaríamos:

```
ssh -R localhost:12345:localhost:25565 usuario@servidoresgaming.com
```

En este caso, cualquier persona que quiera conectarse a nuestro servidor de minecraft podría hacerlo indicando el dominio servidoresgaming.com y el puerto 12345.

Si el servidor SSH de servidoresgaming.com está en un puerto diferente al 22, el comando sería algo así:

```
ssh -R localhost:12345:localhost:25565 usuario@servidoresgaming.com -p 11122
```

EXTRA

Hay dos argumentos interesantes que se le puede pasar al comando SSH que, en este caso nos puede interesar:

- -N: no ejecuta /bin/bash o cualquiera sea la consola que el usuario tenga configurada por defecto. Esto hace que el comando no obtenga ningún prompt, lo que puede ser interesante para que el usuario no enrede con nuestro sistema de archivos. Lógicamente, si el usuario elimina el parámetro, todavía podrá seguir accediendo a la consola pero, para ello,

siempre podemos crear un usuario con una consola nula.

- -f: Manda la ejecución al segundo plano, de forma que no corramos el riesgo de cerrar la ventana y romper la conexión.