



# Túneles y agente SSH

Andrés Hernández

Universidad Nacional  
Autónoma de México



Linux  
Professional  
Institute



# hablando con experTUX



# Acerca de mí

2

- tonejito
- *Alma mater*: UNAM
- Trabajo como ingeniero DevOps
- He trabajado como SysAdmin, auditor de seguridad web y UNIX

## Me encanta el software libre

- Colaboro con el Laboratorio de Investigación y Desarrollo de Software Libre de la **Facultad de Ingeniería** de la **UNAM**

## Me encanta dar clases:

- Sistemas Operativos y Redes de Computadoras en la **Facultad de Ciencias** de la **UNAM**
- Administración y seguridad en Linux/UNIX en el *Plan de Becarios de Seguridad Informática* de **UNAM-CERT**
- Soy miembro del programa *AWS Educate Cloud Ambassadors* para la **UNAM**

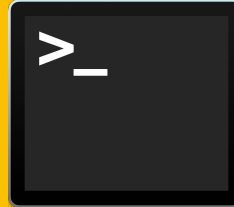
# Agenda

3

- SSH y OpenSSH
- Autenticación con llaves (IdentityFile)
- Agente SSH
  - Uso de ForwardAgent
- Túneles
  - Túnel normal (LocalForward)
  - Túnel inverso (RemoteForward)
  - GatewayPorts
- Proxy SOCKS (DynamicForward)
- Redirección de X11 (X11Forward)
- *Hardening* básico de OpenSSH
  - DebianVersion
  - PermitRootLogin
  - AllowGroups
  - fail2ban / denyhosts

# OpenSSH

- Derivado del proyecto OpenBSD
- Implementa el protocolo SSH
- Viene instalado de manera predeterminada en los equipos
- Se puede utilizar en otros sistemas operativos además de GNU/Linux



# Funcionalidades poco conocidas de OpenSSH

- SCP: Copia de archivos
- SFTP: Servidor de archivos
- Autenticación con llaves
  - Agente SSH
- Túneles TCP
  - Normal
  - Inverso
- Proxy SOCKS
- Redirección de X11
- ACL para usuarios y/o grupos
- Limitar el acceso a root
- Ejecución de "comandos forzados"
- Transporte a otros protocolos
  - rsync
- Integración con otras herramientas
  - fail2ban
  - denyhosts
  - HIDS

# Autenticación con llaves SSH

# Autenticación con llaves SSH (versión *express*)

7

## Cliente:

1. `ssh-keygen -t rsa -b 4096`
2. `ssh-copy-id -i ~/.ssh/id_rsa.pub usuario@servidor`
4. `ssh -i ~/.ssh/id_rsa usuario@servidor`

## Servidor:

3. Recibe la llave y la agrega a `~/.ssh/authorized_keys`
5. Revisa que la llave esté en `~/.ssh/authorized_keys` y realiza la autenticación con *challenge-response*

## Configuración:

### Línea de comandos:

- `-i ruta/hacia/llave_rsa` (privada)

### Configuración del cliente (`~/.ssh/config`):

- Host `example.com *.example.com`
- IdentityFile `ruta/hacia/llave_rsa` (privada)

# Autenticación con llaves - IdentityFile


8

Utilizar como mecanismo de autenticación algo que se tiene en lugar de algo que se conoce



①  
`ssh-keygen -t rsa -b 4096 -f ~/.ssh/id_rsa`

Privada: ~/.ssh/id\_rsa   Passphrase (opcional)

Pública: ~/.ssh/id\_rsa.pub 

②  
`ssh-copy-id -i ~/.ssh/id_rsa.pub usuario@servidor-remoto`

Contraseña



Pública



③

`cat id_rsa.pub >> ~/.ssh/authorized_keys`  
`chmod 0600 ~/.ssh/authorized_keys`

④  
`ssh -i ~/.ssh/id_rsa usuario@servidor-remoto`

Privada



Passphrase (opcional)



⑤

`~/.ssh/authorized_keys`

Pública





# Llave con frase clave

Generar una llave con *passphrase* (frase clave)

```
$ ssh-keygen -t rsa -b 4096
```

Generating public/private rsa key pair.

Enter file in which to save the key (/home/tonejito/.ssh/id\_rsa):

Created directory '/home/tonejito/.ssh'.

Enter passphrase (empty for no passphrase): **Escriba aqui su *passphrase***

Enter same passphrase again: **Escriba aqui su *passphrase***

**Your identification has been saved in /home/tonejito/.ssh/id\_rsa.**

**Your public key has been saved in /home/tonejito/.ssh/id\_rsa.pub.**

The key fingerprint is:

SHA256:4uxf8ems8pnr2lG9e7r1cornyj2wq80Vb1+ZSB3aly4 tonejito@open-expo-lpi

# Uso de llave con frase clave

Si se especifica la frase clave (*passphrase*), la llave privada se guarda cifrada en el disco:

```
$ file ~/.ssh/id_rsa*
```

**/home/tonejito/.ssh/id\_rsa: OpenSSH private key (cifrada)**

**/home/tonejito/.ssh/id\_rsa.pub: OpenSSH RSA public key**

La frase clave se pide **cada vez que se quiere utilizar la llave privada** porque necesita descifrarse para ser utilizada:

```
$ ssh -i ~/.ssh/id_rsa usuario@example.com
```

Enter passphrase for key '**/home/tonejito/.ssh/id\_rsa**':

# Agente SSH

# Agente SSH - Ventajas y desventajas

- El agente SSH nos permite guardar las llaves privadas descifradas en memoria y mantenerlas listas para ser utilizadas cuando se quiera conectar a otro equipo.
- Las llaves privadas permanecen cifradas en disco (*encryption at rest*)
- Seguridad vs Usabilidad
  - Se puede obtener la versión en claro de las llaves públicas del agente SSH si se hace un volcado de memoria del equipo:

```
# insmod ./lime.ko "path=tcp:65535 format=lime"
```

# Uso del agente de SSH

Utilizar el agente de SSH local para autenticarse en equipos remotos

Cliente  
Externo  
192.0.2.111



Servidor  
remoto  
203.0.113.1



①

```
eval $(ssh-agent)  
export SSH_AUTH_SOCK  
export SSH_AGENT_PID
```

②

```
ssh-add ~/.ssh/keys/lave_rsa
```



Passphrase (opcional)

Privada: ~/.ssh/id\_rsa



③

```
ssh usuario@servidor-remoto  
Se comunica con ssh-agent  
a través de SSH_AUTH_SOCK  
Obtiene la llave privada descifrada
```



Pública



④



~/.ssh/authorized\_keys



# Manejo de llaves en el agente de SSH

Por suerte los ambientes gráficos como GNOME tienen una implementación o *wrapper* de ssh-agent como Seahorse o gnome-keyring para facilitarnos la vida

Para agregar una llave al agente SSH

```
$ ssh-add ~/.ssh/id_rsa
```

Enter passphrase for /home/tonejito/.ssh/id\_rsa:

Identity added: /home/tonejito/.ssh/id\_rsa (tonejito@open-expo-lpi)

Para listar las llaves que tenemos en el agente SSH

```
$ ssh-add -l
```

4096 SHA256:k+GvDfCLXTFQzvtGl4IbR61Cgg1YE/ARPeR2GFijRk tonejito@open-expo-lpi (RSA)

# Autenticación anidada a través de varios equipos

- Comúnmente se suele "saltar" a un equipo bastión para tener acceso a los demás equipos de una red
- Muchas veces se comete el error de copiar el par de llaves SSH al equipo bastión, lo que expone nuestro mecanismo de autenticación (y es peor si no ciframos las llaves privadas)
- Es posible redirigir el agente SSH entre la sesión local y el equipo bastión para conectarnos a los equipos internos
- Se utiliza la directiva **ForwardAgent** en [~/.ssh/config](#) o la opción **-A** en [línea de comandos](#)

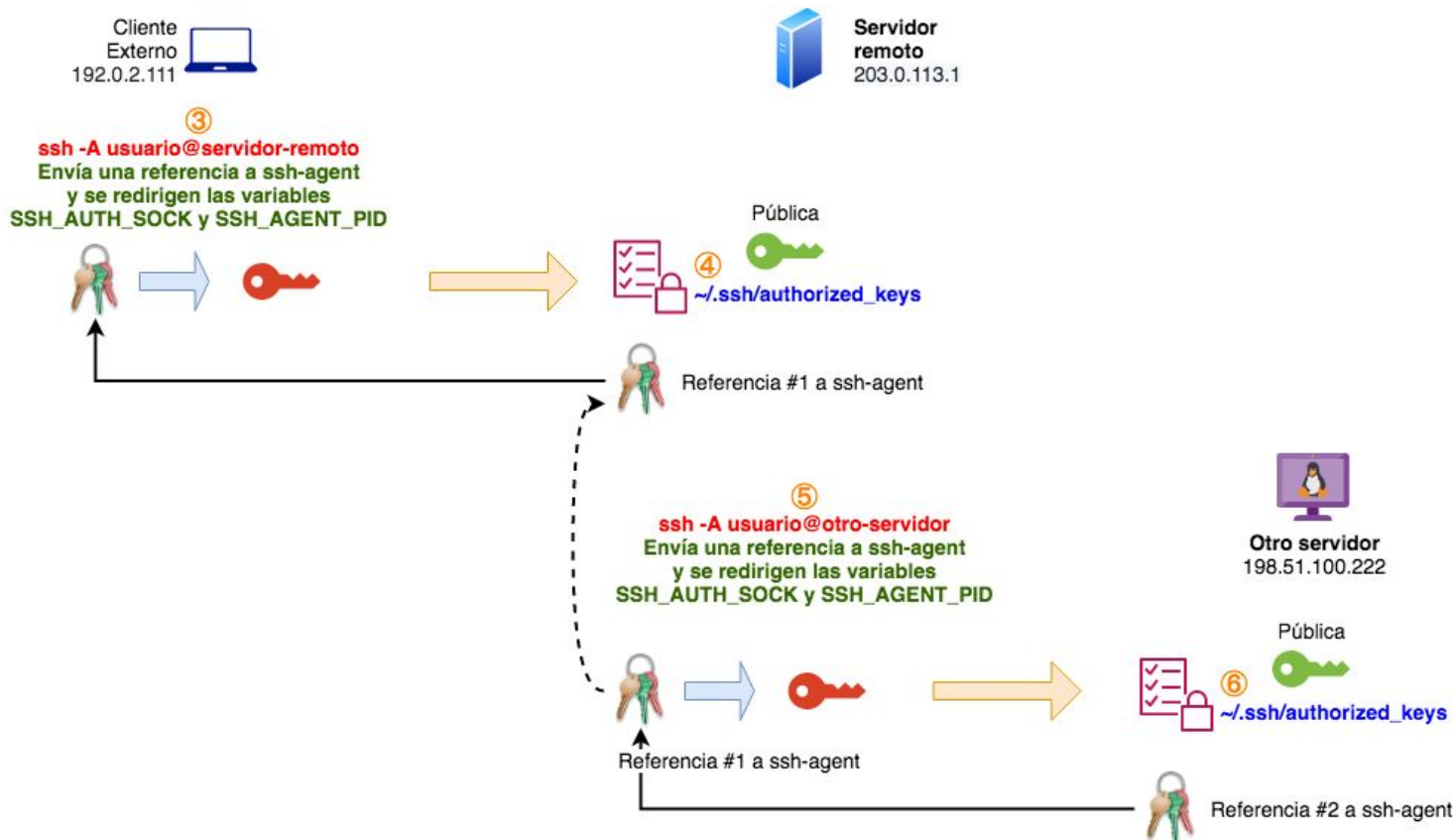


Linux  
Professional  
Institute



# Uso de ForwardAgent

Utilizar el agente de SSH local para autenticarse en equipos remotos





# Túneles SSH

# Túneles SSH (1)

- *Poor man's VPN*
- Se utilizan para alcanzar puertos locales o remotos a través de la conexión de SSH entre dos equipos
- **Únicamente** se pueden utilizar servicios de red que viajen por **TCP**
  - *Spoiler Alert:* DNS también puede viajar por el puerto 53 en TCP
- **Una sola conexión SSH** puede soportar muchos túneles locales y remotos, así como proxy SOCKS y redirección de X11
- Se puede deshabilitar el uso de túneles SSH en la configuración del servidor



Linux  
Professional  
Institute



# Túneles SSH (2) - Tipos

- **Túnel normal**

- Opción `LocalForward` en [~/.ssh/config](#)
- Opción `-L [bind_address:]port:host:hostport` en [línea de comandos](#)
- Acceder a servicios en el **equipo remoto** utilizando un puerto en el **equipo local**

- **Túnel inverso**

- Opción `RemoteForward` en [~/.ssh/config](#)
- Opción `-R [bind_address:]port:host:hostport` en [línea de comandos](#)
- Acceder a servicios en el **equipo local** a través de un puerto en el **equipo remoto**

- Es válido mezclar `LocalForward` (`-L`) y `RemoteForward` (`-R`) en una conexión SSH



# Túnel normal

# LocalForward

# Túnel normal de SSH con LocalForward

## Funcionamiento:

- Coloca un puerto en escucha en el equipo local (bind\_address:port)
  - Este apunta a un servicio en el equipo remoto (host:hostport)
  - Si no se especifica bind\_address, el valor predeterminado es **localhost**
  - Aplican las restricciones para puertos < 1024 en los usuarios normales
- El puerto escucha (port) debe estar libre
  - El puerto remoto (hostport) debe tener algún servicio asociado
  - Únicamente se aceptan conexiones TCP

## Línea de comandos:

- -L [bind\_address:]port:host:hostport

## Configuración del cliente (~/.ssh/config):

- Host example.com \*.example.com
- LocalForward [bind\_address:]port host:hostport

# GatewayPorts para Túneles SSH

- De manera predeterminada los puertos asociados a túneles normales e inversos escuchan en la interfaz *loopback*
- GatewayPorts permite que se pongan en escucha estos puertos en cualquier interfaz de red
- Con esto es posible que otros equipos se puedan conectar a los servicios del túnel que se configuró

## Configuración del servidor [ssh\\_config](#):

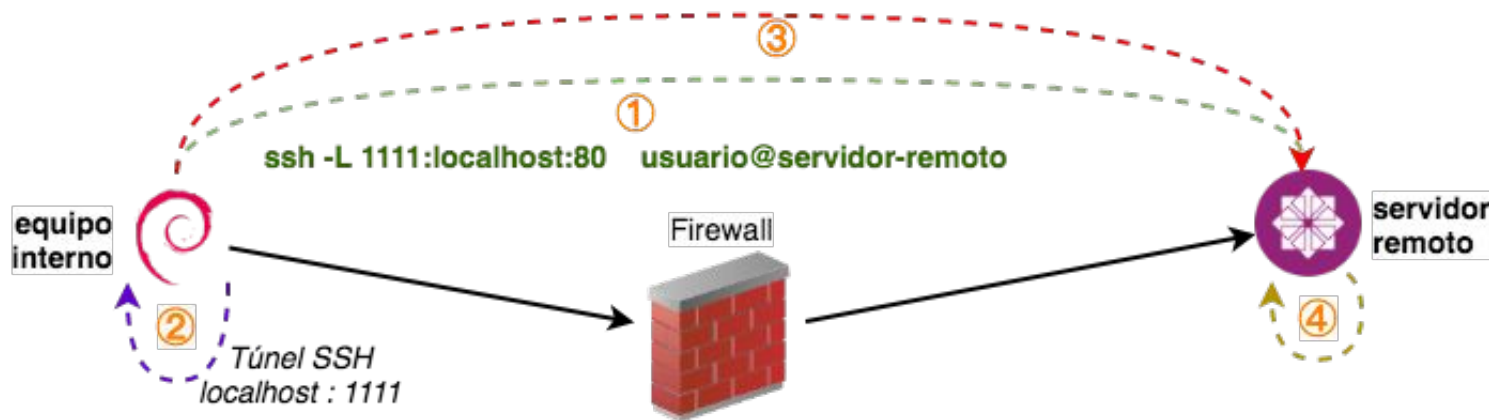
- GatewayPorts yes

## Consideraciones:

- Viene deshabilitado de manera predeterminada
- Considerar los beneficios y riesgos antes de habilitar

# Túnel SSH - LocalForward (1)

Exponer un puerto interno a través de una conexión SSH



① `ssh -L 1111:localhost:80 usuario@servidor-remoto`

② `elinks -dump 'http://localhost:1111/'`

③ Se redirige la petición a través del túnel SSH

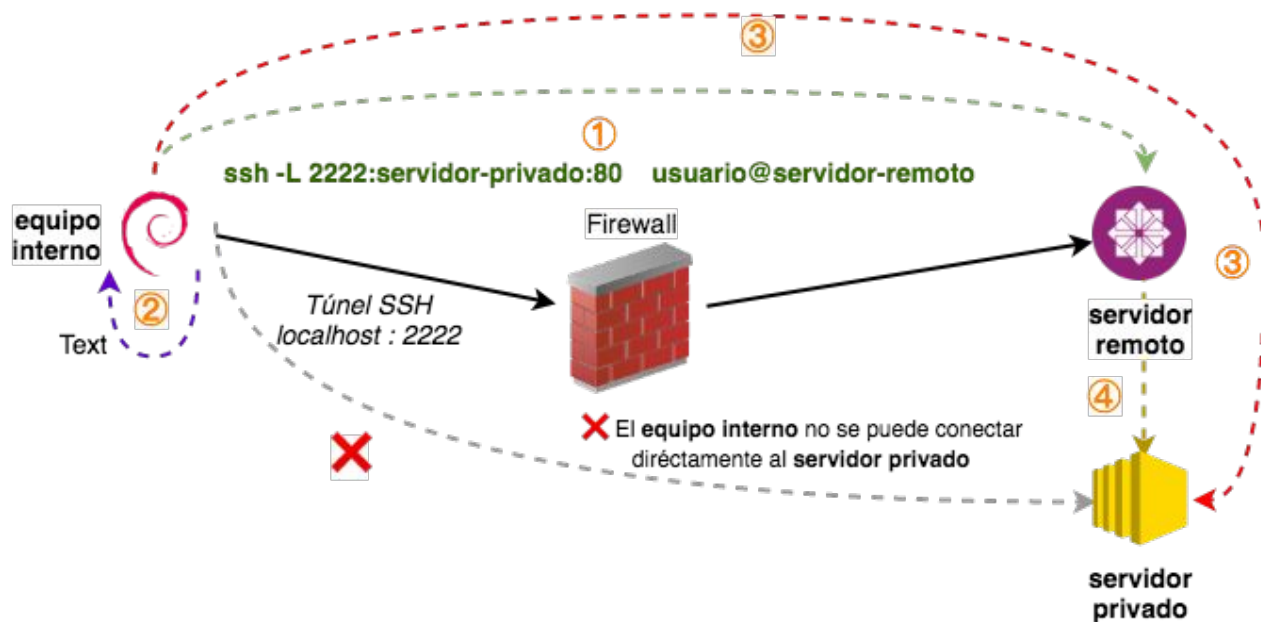
④ localhost se conectó al  
puerto 80 del servidor remoto

En realidad es el **equipo interno**  
quién se conectó



# Túnel SSH - LocalForward (2)

Redirigir un puerto de otro equipo a través de una conexión SSH



① `ssh -L 2222:servidor-privado:80 usuario@servidor-remoto`

② `elinks -dump 'http://localhost:2222/'`

③ Se redirige la petición a través del túnel SSH

④

servidor-remoto se conectó al  
puerto 80 del servidor-privado

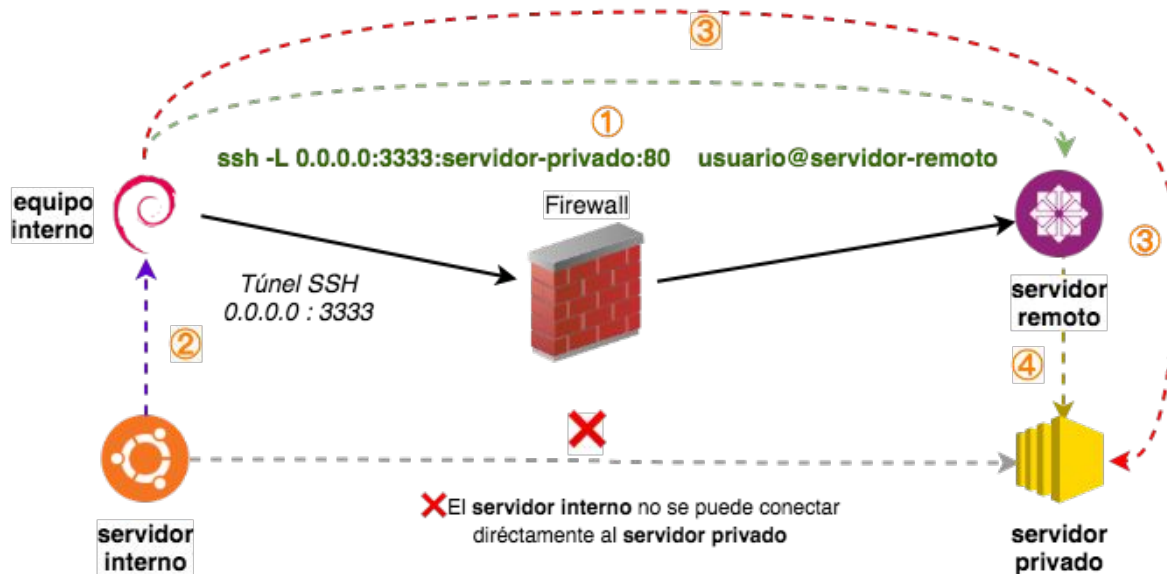
En realidad es el equipo-interno  
quién se conectó





# Túnel SSH - LocalForward (3)

Acceder con otro equipo a la redirección un puerto de otro equipo a través de una conexión SSH



① El equipo interno configura GatewayPorts en sshd\_config

② `ssh -L 0.0.0.0:3333:servidor-privado:80 usuario@servidor-remoto`

③ `elinks -dump 'http://equipo-interno:3333/'` (en el servidor interno)

④ Se redirige la petición a través del túnel SSH

④ servidor-remoto se conectó al puerto 80 del servidor-privado

En realidad es el servidor-interno quién se conectó



# Túnel inverso

## RemoteForward



Linux  
Professional  
Institute



# Túnel inverso de SSH con RemoteForward

## Funcionamiento:

- Coloca un puerto en escucha en el equipo remoto (bind\_address:port)
  - Este apunta a un servicio en el equipo local (host:hostport)
  - Si no se especifica bind\_address, el valor predeterminado es **localhost**
  - Aplican las restricciones para puertos < 1024 en los usuarios normales
- El puerto escucha (port) debe estar libre
  - El puerto local (hostport) debe tener algún servicio asociado
  - Únicamente se aceptan conexiones TCP

## Línea de comandos:

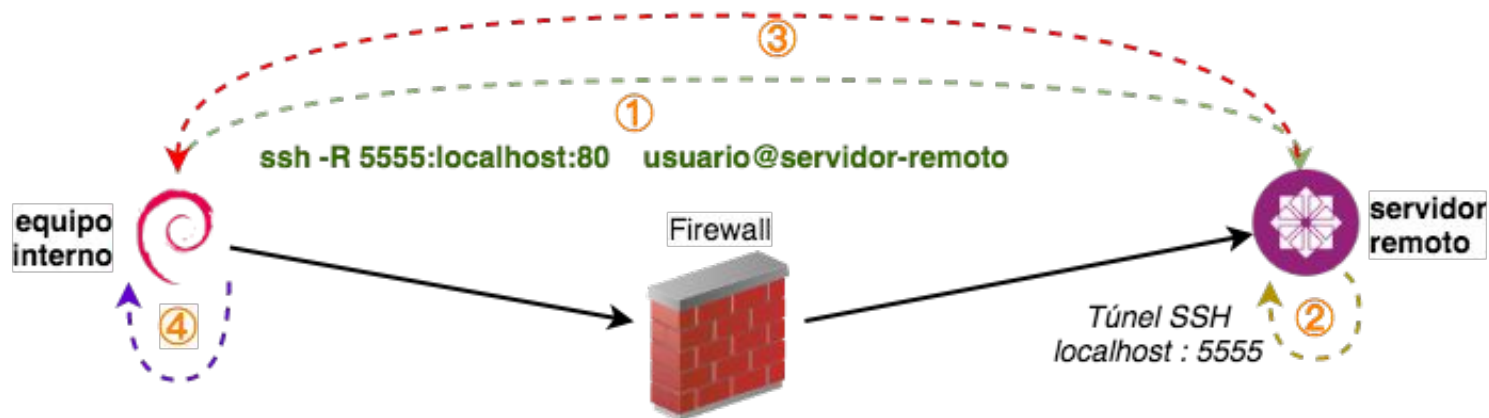
- -R [bind\_address:]port:host:hostport

## Configuración del cliente (~/.ssh/config):

- Host example.com \*.example.com
- RemoteForward [bind\_address:]port host:hostport

# Túnel SSH - RemoteForward (1)

Alcanzar un servicio local en el equipo remoto a través de un túnel inverso de SSH



④ **localhost** se conectó al  
puerto 80 del **equipo interno**

En realidad es el **servidor remoto**  
quién se conectó

① **ssh -R 5555:localhost:80 usuario@servidor-remoto**

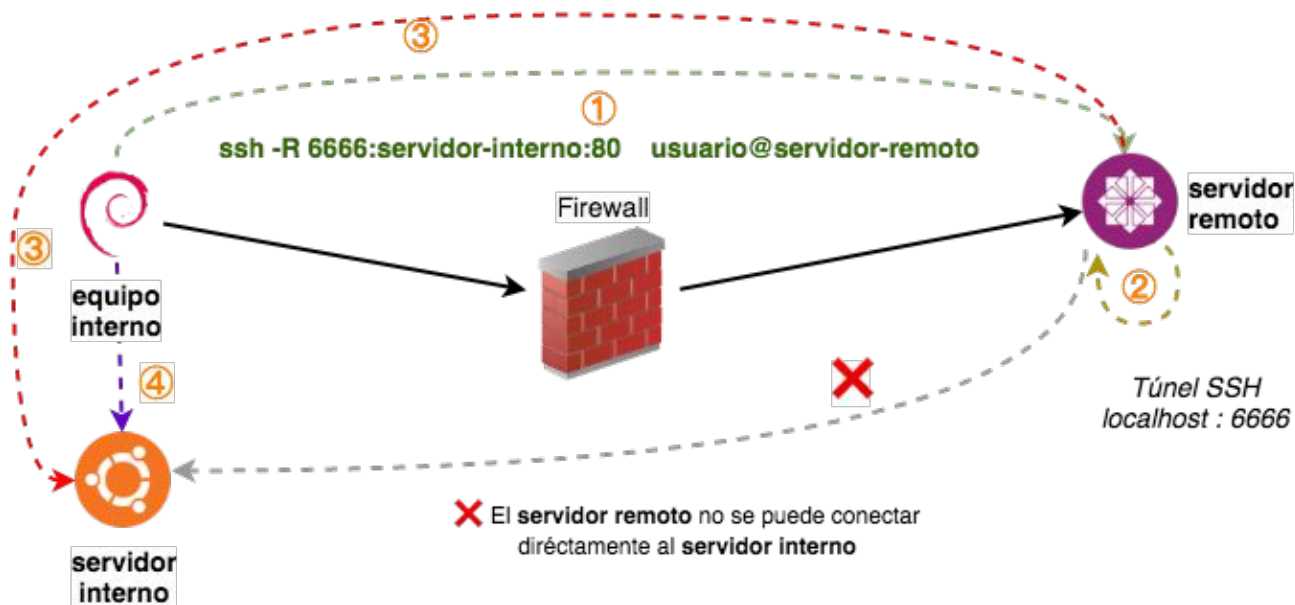
② **elinks -dump 'http://localhost:5555/'** (en el **servidor remoto**)

③ Se redirige la petición a través del túnel SSH



# Túnel SSH - RemoteForward (2)

Alcanzar un destino local en el equipo remoto a través de un túnel inverso de SSH



El equipo interno se conectó al puerto 80 del servidor interno

En realidad es el servidor remoto quién se conectó

**1** `ssh -R 6666:servidor-interno:80 usuario@servidor-remoto`

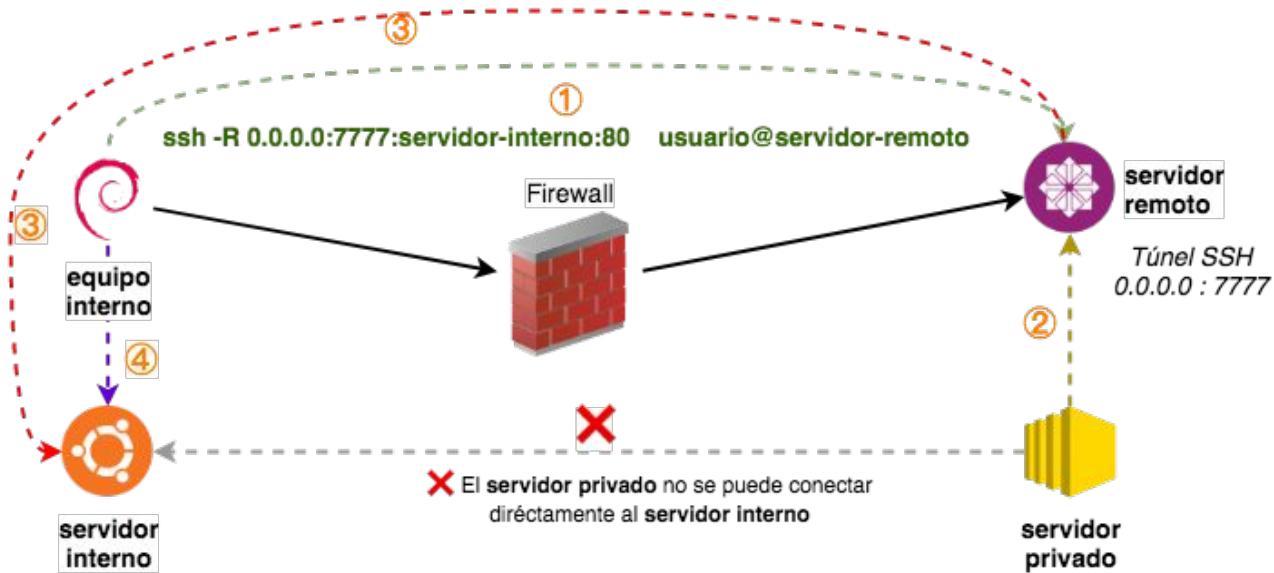
**2** `elinks -dump 'http://localhost:6666/'` (en el servidor remoto)

**3** Se redirige la petición a través del túnel SSH



# Túnel SSH - RemoteForward (3)

Permitir que un tercer equipo remoto acceda a un equipo local a través del túnel inverso de SSH



④ El **equipo interno** se conectó al puerto 80 del **servidor interno**

En realidad es el **servidor privado** quién se conectó

① El **equipo interno** y el **servidor remoto** configuran **GatewayPorts** en **sshd\_config**

① `ssh -R 0.0.0.0:7777:servidor-interno:80 usuario@servidor-remoto`

② `elinks -dump 'http://servidor-remoto:7777/'` (en el **servidor privado**)

③ Se redirige la petición a través del túnel SSH



# Proxy SOCKS

## DynamicForward



Linux  
Professional  
Institute



# Proxy SOCKS con DynamicForward

## Funcionamiento:

- El proxy SOCKS permite que se redirijan múltiples conexiones TCP
- No es necesario especificar LocalForward para cada conexión
- Útil para alcanzar múltiples destinos remotos

## Línea de comandos:

- -D [bind\_address:]port

## Configuración del cliente (~/.ssh/config):

- Host example.com \*.example.com
- DynamicForward [bind\_address:]port

## Clientes:

- Firefox
  - Configurar Proxy SOCKS
    - localhost:1080
  - Habilitar la redirección del DNS a través del proxy
    - DNS puede viajar por TCP
- Otras aplicaciones
  - [tsocks](#)
  - Encapsula las llamadas al sistema de red y las redirige a un proxy SOCKS



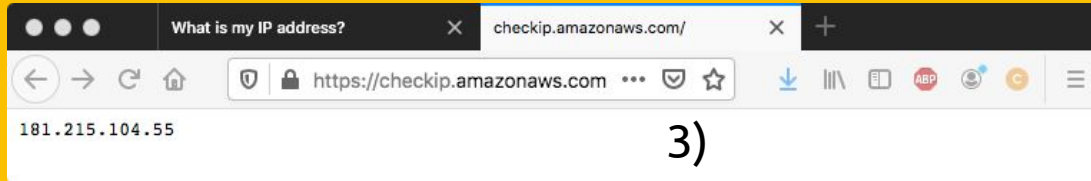
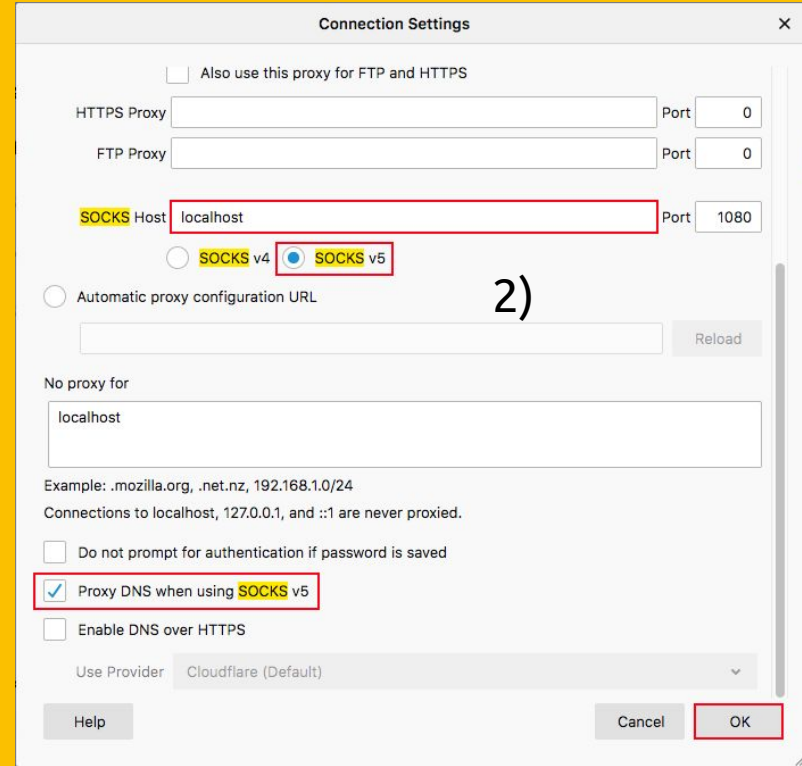
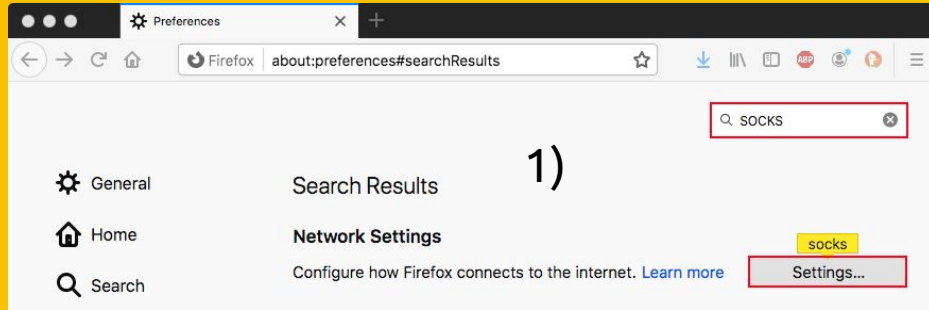
Linux  
Professional  
Institute





# Configuración en Firefox

33





Linux  
Professional  
Institute

Andrés Hernández

= ^ . ^ =

tonejito

[tonejito@comunidad.unam.mx](mailto:tonejito@comunidad.unam.mx)

[www.unam.mx](http://www.unam.mx)

[www.ingenieria.unam.mx](http://www.ingenieria.unam.mx)

[www.fcencias.unam.mx](http://www.fcencias.unam.mx)

[www.cert.unam.mx](http://www.cert.unam.mx)

[lidsol.org](http://lidsol.org)

[t.me/LIDSoL](https://t.me/LIDSoL)



# ¡Gracias!



## LIDSOL

