

# Introducción a OpenSSH



## Curso de Administración GNU/Linux NIVEL I

GUGLER

### Integrantes:

Churichi Maximiliano  
Gimenez Pablo  
Soliard Adrian  
Trossero Sebastian

mchurichi[at]gmail[dot]com  
pablo\_arg87[at]hotmail[dot]com  
a.soliard[at]gmail[dot]com  
sebat-hermetica[at]hotmail[dot]com

### Profesores:

Aramburu Exequiel  
Bonnet Federico

Copyright © 2009

Churichi Maximiliano, Gimenez Pablo, Soliard Adrian, Trossero Sebastian.  
Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

## **Indice:**

- Introducción a OpenSSH	4
- Uso básico de cada programa	5
ssh	5
scp	5
sftp	6
- Utilización de las herramientas	7
ssh-keygen	7
ssh-agent	7
ssh-add	7
ssh-keyscan	8
- Túneles con ssh	9
- Bibliografía consultada	10

## Introducción a OpenSSH

SSH (Secure Shell) es un protocolo el cual permite realizar conexiones remotas de computadoras. A diferencia de Telnet (Telecommunication Network), la información utilizada por SSH es cifrada, convirtiéndolo en una conexión segura.

OpenSSH es un paquete de herramientas para trabajar con este protocolo distribuido bajo licencia BSD, la cual es más permisiva con respecto a la GPL, permitiendo por ejemplo su utilización en software no libre.

Dentro del paquete se encuentran los siguientes programas:

- **SSH** (Secure Shell):  
*Permite el acceso a otra maquina otorgando una shell. Es el reemplazo de Telnet y rlogin.*
- **SCP** (Secure Copy):  
*Permite copiar rápidamente archivos entre equipos. Es el reemplazo de rcp.*
- **SFTP** (Secure File Transfer Protocol):  
*Es idéntico al File Transfer Protocol, pero con una mejor capa de seguridad.*

Además de las utilidades de configuración y funcionamiento:

- **SSHD** (Secure Shell Daemon):  
*Demonio del servidor SSH. Cuando se ejecuta, también lo hace sftp-server, que es el subsistema para el servidor sftp.*
- **SSH-KEYGEN**:  
*La herramienta para generar e inspeccionar claves RSA y DSA utilizadas por el protocolo para la autenticación.*
- **SSH-AGENT**:  
*Recuerda las claves que utiliza SSH, quedando estas listas para no tener que volver a autenticarse nuevamente.*
- **SSH-ADD**:  
*El SSH-AGENT no contiene ninguna clave al ejecutarse, para agregarlas se utiliza este comando.*
- **SSH-KEYSCAN**:  
*Escanea una lista de clientes y recolecta sus claves públicas, fue diseñado para construir y verificar la lista de equipos conocidos.*

## Uso básico de cada programa

- **SSH:**

Conexión básica:

```
ssh host
```

Es el uso mas común, conecta a "host" y nos brinda una consola remota si nos pudimos autenticar correctamente.

Conexión con previa especificación del usuario:

```
ssh usuario@host
```

Conecta al equipo "host" pero el usuario es definido con anterioridad, luego de conectarse pide la contraseña.

```
ssh host -l usuario
```

El parámetro "-l" también permite especificar el usuario con anterioridad.

Ejecución de comandos (Solo ejecuta un comando, no adquiere consola):

```
ssh host reboot
```

Se ejecuta el comando "reboot" al realizar la conexión.

```
ssh host /etc/init.d/apache2 restart
```

Este ejemplo es un poco mas útil, reinicia el demonio de apache.

El comando SSH tiene muchos mas parámetros y utilidades que podrían servir en distintas situaciones.

- **SCP:**

Copiar un archivo de un equipo remoto al equipo local:

```
scp user@host:/dir-remoto/archivo /dir-local/
```

Se conecta a "host" como "user" y copia el archivo "/dir-remoto/archivo" a "/dir-local/" del equipo local.

Copiar un archivo del equipo local a un equipo remoto:

```
scp /dir-local/archivo2 user@host:/dir-remoto/
```

Copia el archivo local "/dir-local/archivo2" al directorio "/dir-remoto/" de "host", autenticándose como "user".

Copiar un archivo de un equipo remoto a otro equipo remoto:

```
scp user@host:/dir-remoto/archivo user2@host2:/dir-remoto/
```

Este comando copia el archivo "dir-remoto/archivo" que se encuentra en el equipo "host" al directorio "/dir-remoto/" del equipo "host2"

Copiar un directorio recursivamente (completo):

```
scp -r user@host:/dir-remoto/ /dir-local/
```

Copia el contenido de "/dir-remoto/" recursivamente a "/dir-local", gracias al parámetro "-r".

- **SFTP:**

Conectarse a un servidor remoto para adquirir una consola interactiva:

```
sftp host
```

Conecta al servidor "host" y pide autenticación.

```
sftp user@host
```

Se conecta al servidor "host" como "user" y luego pide la contraseña del usuario.

Una vez conseguida la consola interactiva, se utilizan los comandos especiales, muy similares a los comandos de ftp común.

Descargar un archivo de un servidor:

```
sftp host:archivo
```

Realiza la conexión a "host" únicamente para descargar "archivo" al equipo local, requiere autenticación.

Conectarse a un servidor remoto en un determinado directorio:

```
sftp host:/directorio/
```

Conecta al servidor "host" y se ubica automáticamente en "/directorio".

## Utilización de las herramientas

- **SSH-KEYGEN:**

Generar clave RSA:

*ssh-keygen*

Genera una clave solicitando la nueva contraseña. Por defecto produce una de tipo RSA.

*ssh-keygen -b 2048*

Especifica la fortaleza de la clave a generar, en este caso "2048" bits. Las claves dsa DEBEN ser de 1024.

*ssh-keygen -f archivo*

Genera una clave RSA y la almacena en "archivo".

*ssh-keygen -t dsa*

Permite elegir el tipo de clave a generar, los posibles valores son "rsa1" (protocolo versión 1), "rsa" y "dsa" (protocolo versión 2).

- **SSH-AGENT:**

Este programa recuerda las claves privadas y sus datos de autenticación (usuario y contraseña) en el equipo local, automatizando el proceso de autenticación en un equipo remoto. ssh-agent solo almacena las claves durante su ejecución, una vez muerto el proceso los datos son perdidos. Su uso más común es iniciar junto con las X para permanecer ejecutado durante toda la sesión.

- **SSH-ADD:**

Uso básico del comando:

*ssh-add*

Si se ejecuta sin argumentos intenta añadir los archivos "~/.ssh/id\_rsa", "~/.ssh/id\_dsa" y "~/.ssh/identity". Solicita la contraseña de la clave privada.

Añadir los datos de un archivo específico:

*ssh-add archivo*

Añade las claves almacenadas en "archivo" al agente.

Limpiar el repositorio de claves del agente:

```
ssh-add -D
```

Elimina las claves almacenadas en el agente.

- **SSH-KEYSCAN:**

Uso básico del comando:

```
ssh-keyscan host
```

Se conecta al equipo "host" y colecta su clave pública.

```
ssh-keyscan host host1 host2
```

Colecta las claves de los equipos "host", "host1" y "host2".

Leer un archivo para colectar las claves públicas de los equipos listados:

```
ssh-keyscan -f archivo
```

Lee el contenido de "archivo" y colecta las claves publicas de los equipos listados.

## Túneles con ssh

### ¿Qué es un túnel SSH?

La idea en la que se basa este procedimiento es la de hacer un túnel por el cual viajarán los datos de manera segura ("tunneling"). En cada uno de los extremos del túnel están las aplicaciones estándar y la comunicación se asegura haciendo uso de toda la potencia criptográfica de SSH.

SSH recoge los datos que el cliente quiere enviar y los reenvía por el túnel o canal seguro, al otro lado del túnel se recogen los datos y se reenvían al servidor conveniente, esto se denomina "port-forwarding".

Existen dos tipos de túneles bajo SSH:

- **Locales:**

En estos se redirecciona un puerto de la máquina local (cliente) hacia un puerto en una máquina remota a la que el servidor tenga acceso. Se pueden crear túneles locales con OpenSSH con el parámetro "-L", de la manera siguiente:

```
ssh -L puerto_local:host_remoto:puerto_remoto
```

Escucha en "puerto\_local" y redirecciona a "puerto\_remoto" de "host\_remoto".

- **Remotos:**

En este caso, lo que se hace es redireccionar un puerto desde una máquina remota hacia un puerto de la máquina local. Esto es posible utilizando el parámetro "-R", que tiene la siguiente sintaxis:

```
ssh -R puerto_escucha:host_remoto:puerto_remoto
```

Manda una instrucción a "host\_remoto", que le indica que escuche en su propio puerto "puerto\_escucha" y redirija el tráfico hacia el "puerto\_remoto" en nuestro equipo.

## **Bibliografía consultada:**

<http://www.openssh.com/>

[http://es.wikipedia.org/wiki/Secure\\_Shell](http://es.wikipedia.org/wiki/Secure_Shell)

<http://es.wikipedia.org/wiki/OpenSSH>

<http://es.wikipedia.org/wiki/SSH-Agent>

<http://www.lodemenos.net/Como-se-crea-un-tunel-ssh-entre-tu.html>

[http://www.vilecha.com/Hellquest/ssh\\_tuneles.asp](http://www.vilecha.com/Hellquest/ssh_tuneles.asp)

manuales específicos de cada comando