

## **INSTALACIÓN Y CONFIGURACIÓN BÁSICA DE OPENSSSH**

Algunas veces es necesario administrar de forma remota un servidor y para ello debemos establecer una comunicación segura entre dicho host y el sistema desde el cual establecemos la conexión. Las sesiones telnet no ofrecen mucha seguridad, ya que los datos viajan a través de la red sin ningún tipo de encriptamiento y son potencialmente susceptibles de ser interceptadas por un atacante, así que como mejor método posible a nuestro alcance tenemos OpenSSH.

Gracias a OpenSSH vamos a poder ejecutar un servidor de shell seguro y al mismo tiempo podemos conectar a él mediante un cliente también seguro, de forma que la comunicación quede blindada por ambos extremos.

Bien, para poder usar correctamente OpenSSH y que la comunicación quede totalmente cifrada, hemos de instalar primero OpenSSL, que proporciona las bibliotecas *Secure Socket Layer* o Capa de Conectores Segura (SSL). Podemos instalar OpenSSL desde los cds de la distribución o bien compilar su código fuente (representaremos la línea de comandos con el símbolo #):

```
# tar -zxvf openssl-númerodeversión.tar.gz
# cd openssl/
# ./configure
# make
# make install
```

Ya tenemos instalado OpenSSL, ahora le toca el turno a OpenSSH. Igualmente podemos instalar los paquetes precompilados que vienen con nuestra distribución o bien compilar las fuentes:

```
# tar -zxvf openssh-númerodeversión.tar.gz
# cd openssh/
# ./configure --with-ssl-dir=/usr/local/ssl
  ( nota: la ubicación de ssl puede variar en función de la distribución o de si se ha
  compilado o instalado mediante paquetes RPM )
# make
# make install
```

Ahora arrancamos el servidor SSH:

```
# /etc/init.d/ssh start
Starting SSH daemon                               done
```

E intentamos conectarnos al puerto 22 de nuestro sistema:

```
# ssh localhost
root@localhost's password:
Last login: Tue Nov 23 15:29:31 2004 from localhost
Have a lot of fun...
linux:~ #
```

Como podemos ver en el ejemplo nos hemos podido conectar como el usuario root, esto

es perfectamente seguro ya que la comunicación va totalmente encriptada, pero si quisiéramos cambiar esto deberíamos editar `/etc/ssh/sshd_config` y añadir:

```
PermitRootLogin no
```

Puede ser que ya exista en dicho archivo una línea que diga `PermitRootLogin yes`, si existe y está comentada, añadiremos la nuestra y si no lo está cambiaremos el `yes` por el `no`.

Este tipo de autenticación como podemos ver se basa en el intercambio de una pareja usuario-contraseña, pero podemos acrecentar todavía más la seguridad si añadimos un par de claves pública y privada, ¿con que fin?, muy sencillo, dicho mecanismo ofrece la seguridad al cliente ssh que se está conectando de que el servidor es el auténtico y no el de un posible intruso. Si ello ocurriera aparecería un mensaje en la pantalla advirtiéndonos de que no se puede comprobar la autenticidad del servidor y se nos pregunta si aceptamos continuar la sesión a pesar de dicha circunstancia.

```
# ssh localhost
```

```
The authenticity of host 'localhost (::1)' can't be established.
```

```
RSA key fingerprint is 74:4e:7c:2b:ee:18:ca:5a:e4:55:1c:bd:dc:72:dd:7e.
```

```
Are you sure you want to continue connecting (yes/no)?
```

Para generar las claves actuaremos de la manera siguiente como root:

```
# ssh-keygen -t rsa
```

```
Generating public/private rsa key pair.
```

```
Enter file in which to save the key (/root/.ssh/id_rsa):
```

```
Enter passphrase (empty for no passphrase):
```

```
Enter same passphrase again:
```

```
Your identification has been saved in /root/.ssh/id_rsa.
```

```
Your public key has been saved in /root/.ssh/id_rsa.pub.
```

```
The key fingerprint is:
```

```
fa:3a:7b:65:73:1b:20:ec:4e:3e:23:80:17:5c:52:60 root@linux
```

Los posibles valores de `-t` son `rsa1` para la versión 1 del protocolo y `rsa` y `dsa` para la 2. Elegiremos la versión 2 por ser más segura.

Cuando ya tenemos las claves solo hemos de proporcionar la clave pública al cliente y que este la añada a su fichero `known_hosts`, situado en el directorio `.ssh` de su `/home`.

```
# ssh localhost
```

```
usuario@localhost's password:
```

```
Last login: Thu Nov 25 15:01:06 2004 from localhost
```

```
Have a lot of fun...
```

Un último apunte, al intentar hacer login podemos indicar el usuario con el que queremos acceder al sistema:

```
# ssh -l root localhost
```

```
root@localhost's password:
```

```
Last login: Thu Nov 25 15:09:10 2004 from localhost
```

```
Have a lot of fun...
```

Y por supuesto, para salir de una sesión ssh y al igual que haríamos en la terminal de nuestro propio sistema, teclearíamos :

```
# exit
```