JUNTA DE ANDALUCIA
CONSEJERIA DE EDUCACION
CEP Lora del Río

Utilización elemental de ssh

Alberto Molina Coballes, José Domingo Muñoz Rodríguez y José Luis Rodríguez Rodríguez.

27 de abril de 2010

SSH es una aplicación muy extendida para el acceso remoto a un equipo GNU/Linux, ya que permite realizar operaciones en un equipo remoto como si se tratara de uno local y además todo el proceso se realiza de forma cifrada, lo que garantiza la seguridad del mismo.

En este documento se describe de forma breve la configuración del cliente y el servidor OpenSSH en Debian GNU/Linux (Lenny). Este documento forma parte del curso *Servicios en GNU/Linux. Portal Educativo*, organizado por el CEP de Lora del Río (Sevilla) en 2010.

Índice

1. Introdución

La Administración Remota es la capacidad algunos programas que permiten realizar ciertos tipos de acciones desde un equipo local y que las mismas se ejecuten en otro equipo remoto.

Una de las grandes ventajas que tienen los sistemas operativos que se manejan mediante un intérprete de comandos es la facilidad de manejarlos de forma remota desde otro equipo. En los sistemas UNIX esta función la ha realizado durante mucho tiempo la aplicación *telnet*, que hoy en día se ha quedado para usos marginales debido a la posibilidad de capturar la contraseña de acceso al realizar la conexión entre el cliente y el servidor de forma no cifrada. Para solucionar estos problemas de telnet, surgieron varias aplicaciones que incluían el cifrado de las comunicaciones, siendo ssh el estándar de facto hoy en día para la administración de sistemas UNIX.

En este documento vamos a presentar algunos de los usos más habituales de la aplicación de gestión remota ssh (Secure SHell, en español: intérprete de órdenes segura) que es el nombre de un protocolo y del programa que lo implementa, y sirve para acceder de forma segura a máquinas remotas a través de una red no segura. Las principales características de SSH son las siguientes:

- El cliente puede verificar que se está conectando a un determinado servidor.
- La información de autenticación es cifrada.
- Todo el tráfico de datos enviados y recibidos son cifrados.
- Es posible enviar aplicaciones gráficas lanzadas desde el intérprete de comandos (reenvío por X11 o X11 Forwarding).

Existen dos variedades de SSH actualmente (versión 1 y versión 2). La versión 1 de SSH hace uso de muchos algoritmos de encriptación patentados (sin embargo, algunas de estas patentes han expirado) y tiene problemas de seguridad que potencialmente permite a un intruso insertar datos en la corriente de comunicación. La versión 2 tiene un algoritmo de intercambio de claves mejorado que no es vulnerable a los problemas de seguridad software de la versión 1, obviamente la versión recomendada y que hoy en día se instala por defecto es la 2.

El paquete ssh que incluye Debian recoge OpenSSH, la implementación libre de SSH (versiones 1 y 2) que lleva a cabo principalmente OpenBSD. Dentro del paquete, se encuentra un conjunto de herramientas seguras que estudiaremos a continaución como son: ssh, scp, sftp, ...

2. Métodos de autentificación

Existen múltiples mecanismos de autenticación/autentificación y los más importantes que implementa ssh son los siguientes:

Contraseña Es el método utilizado por defecto, y en el que para conectarnos a un host, tenemos que indicar nombre de usuario y contraseña en dicho usuario en ese host. Todo el procedimiento de autenticación se realiza cifrado.

Clave pública/privada Basado en los mecanismos de cifrado asimétrico RSA y DSA. Este método, que estudiaremos más detalladamente a lo largo de este documento, nos da la posibilidad de conectar con el host sin indicar la contraseña del usuario en el host.

Kerberos Como en cualquier otro servicio basado en autenticación Kerberos, tanto el usuario como el servidor deben garantizar al servidor Kerberos su autenticidad, lo que en nuestro caso permite acceder a un equipo remoto sin necesidad de utilizar contraseña.

3. Instalación y configuración del servidor ssh

En Debian existe el metapaquete ssh, que incluye tanto el servidor (openssh-server) ssh como el cliente(openssh-client). Por lo tanto para su instalación ejecutamos:

```
avatar:~# aptitude install ssh
```

El fichero de configuración del sevidor openssh lo encontramos en /etc/ssh/sshd_config, y los parámetros más interesantes son:

AllowGroups Esta opción puede ir seguida de una lista de grupos de nombres, separados por espacios. Si se especifica, sólo se permite realizar un login a los usuarios cuyo grupo principal o suplementarios coincida con uno de los patrones establecidos. Se puede usar '*' y '?' como comodines en los patrones.

AllowUsers Esta opción puede ir seguida de una lista de usuarios, separados por espacios. Si se especifica, sólo se permite realizar login a los usuarios cuyo nombre concuerde con el patrón. Se pueden usar '*' y '?' para la construcción de patrones.

DenyGroups, DenyUsers Similares a los anteriores pero denegando el servicio.

KerberosAuthentication Especifica si el método de autentificación Kerberos está permitido. Esto puede llevarse a cabo mediante un ticket Kerberos o si la opción PasswordAuthentication está habilitada. En ese caso, la contraseña suministrada por el usuario puede ser validada contra un KDC Kerberos.

LoginGraceTime El servidor desconecta después de este tiempo a los usuarios que no se hayan validado correctamente. Si el valor es 0, no hay límite de tiempo. Por defecto, 120 (segundos).

PasswordAuthentication Especifica si la autentificación por password está admitida. Por defecto, "yes".

PermitRootLogin Especifica si el superusuario puede validar usando ssh. Los argumentos posibles son: "yes", "without-password", "forced-commands-only" o "no". El valor por defecto es "yes". Si a esta opción se le asigna "without-password", la autentificación por password se deshabilita para el usuario root.

Si a esta opción se le asigna "forced-commands-only", el login del superusuario con autentificación de clave pública será admitido, pero sólo si la opción del comando se ha especificado (lo que puede ser útil para realizar backups remotos incluso si el login de superusuario no está admitido normalmente). El resto de métodos de autentificación están vetados para el root. Si a la opción se le asigna "no", el root no puede hacer login.

Port Especifica el número de puerto al que escucha sshd. El puerto predeterminado es el `22/tcp`. Este parámetro admite opciones múltiples.

Protocol Especifica las versiones del protocolo que soporta sshd. Las posibilidades son "1" y "2". Las opciones múltiples se separan por comas. El valor por defecto es "2".

PubkeyAuthentication Especifica si se admite la autentificación mediante clave pública. Por defecto, "yes". Esta opción sólo se aplica a la versión 2 del protocolo.

StrictModes Especifica si sshd debe comprobar los permisos y propietarios de los ficheros del usuario y el directorio home antes de aceptar el login. Es recomendable habilitar esta opción pues los usuarios noveles a veces dejan accidentalmente su directorio o sus ficheros con permisos de escritura universales. El valor por defecto, "yes".

X11Forwarding Establece si se permite o no la ejecución remota de aplicaciones gráficas. Si se va a acceder hacia el servidor desde red local, este parámetro puede quedarse con el valor yes. Si se va a permitir el acceso hacia el servidor desde redes públicas, resultará prudente utilizar este parámetro con el valor no.

Después de cambiar la configuración del servidor tendremos que reinicar el servicio:

```
avatar:~# /etc/init.d/ssh restart
```

3.1. Configuración y uso del cliente openssh

El fichero de configuración del cliente lo encontramos en /etc/ssh/ssh_config, inicialmente no es necesario cambiar ningún parámetro, pero pueden verse las opciones en la pagina del manual correspondiente:

```
cliente:~$ man 5 ssh_config
```

3.2. Utilización de ssh

Para acceder a un ordenador remoto que tenga instalado un servidor ssh, desde otro ordenador con un cliente ssh, se utiliza la siguiente instrucción:

```
cliente:~$ ssh nombredelequipo
```

En este caso la conexión se realizará con un usuario con el mismo nombre que estas usando ahora en el cliente, en caso de que queramos especificar un usuario concreto, utilizaremos:

```
cliente:~$  ssh usuario@nombredelequipo
```

Por ejemplo desde nuestro cliente podemos acceder a avatar de la siguiente manera:

```
cliente:~$ ssh usuario@avatar.example.com
```

El proceso de conexión detallado de un cliente a un servidor podemos verlo con la opción -v:

```
cliente:~$ ssh -v usuario@avatar.example.com
```

```
OpenSSH_5.1p1 Debian-5, OpenSSL 0.9.8g 19 Oct 2007
debug1: Reading configuration data /etc/ssh/ssh_config
debug1: Applying options for *
debug1: Connecting to avatar.example.com [192.168.2.1] port 22.
debug1: Connection established.
debug1: identity file /home/usuario/.ssh/identity type -1
debug1: identity file /home/usuario/.ssh/id_rsa type -1
debug1: identity file /home/usuario/.ssh/id_dsa type -1
debug1: Remote protocol version 2.0, remote software version OpenSSH_5\
.1p1 Debian-5
debug1: match: OpenSSH_5.1p1 Debian-5 pat OpenSSH*
debug1: Enabling compatibility mode for protocol 2.0
debug1: Local version string SSH-2.0-OpenSSH_5.1p1 Debian-5
debug1: SSH2_MSG_KEXINIT sent
debug1: SSH2_MSG_KEXINIT received
```

```
debug1: kex: server->client aes128-cbc hmac-md5 none
debug1: kex: client->server aes128-cbc hmac-md5 none
debug1: SSH2_MSG_KEX_DH_GEX_REQUEST(1024<1024<8192) sent
debug1: expecting SSH2_MSG_KEX_DH_GEX_GROUP
debug1: SSH2_MSG_KEX_DH_GEX_INIT sent
debug1: expecting SSH2_MSG_KEX_DH_GEX_REPLY
The authenticity of host 'avatar.example.com (192.168.2.1)' can't be e\
stablished.
RSA key fingerprint is 8b:ef:98:39:0d:95:ab:75:28:92:e8:ab:61:6e:95:f3.
Are you sure you want to continue connecting (yes/no)?
```

Donde el cliente ssh nos comunica que no reconoce la clave pública del servidor y nos pide permiso para confiar en ella y almacenarla en el fichero de claves públicas de servidores conocidas (/.ssh/known_hosts).

```
Warning: Permanently added 'avatar.example.com' (RSA) to the list of k\
nown hosts.
debug1: ssh_rsa_verify: signature correct
```

Verifica que la clave del servidor es correcto y ahora empieza el proceso de autenticación del cliente, donde se establecen una serie de métodos y se utiliza el primero que funcione, en este caso autenticación mediante contraseña.

```
debug1: SSH2_MSG_NEWKEYS sent
debug1: expecting SSH2_MSG_NEWKEYS
debug1: SSH2_MSG_NEWKEYS received
debug1: SSH2_MSG_SERVICE_REQUEST sent
debug1: SSH2_MSG_SERVICE_ACCEPT received
debug1: Authentications that can continue: publickey,password
debug1: Next authentication method: publickey
debug1: Trying private key: /home/usuario/.ssh/identity
debug1: Trying private key: /home/usuario/.ssh/id_rsa
debug1: Trying private key: /home/usuario/.ssh/id_dsa
debug1: Next authentication method: password
usuario@avatar.example.com's password:
```

Para ejecutar un comando en el ordenador remoto, si necesidad de abrir un terminal, podemos usar la siguiente expresión:

```
cliente:~$ ssh usuario@nombredeleuipo comando
```

Por ejemplo:

```
cliente:~$ ssh@usuario@avatar.example.com ifconfig
```

3.3. Transferir ficheros con ssh

Para copiar un fichero de un ordenador a otro de forma segura, encriptando la información, podemos usar la herramienta scp de la siguiente manera:

Copiar un fichero a un ordenador remoto

```
scp archivo_local usuario@servidor:/archivo_remoto
```

Copiar un fichero desde un ordenador remoto

```
scp usuario@servidor:/archivo_remoto archivo_local
```

Por ejemplo, para copiar un archivo que tengo en mi directorio personal del cliente, a un directorio del servidor:

```
cliente::$ scp /home/usuario/documento.odt usuario@avatar.example.com:
```

4. Autentificación basada en clave pública

Como hemos visto anteriormente openssh puede realizar la autentificación de usurios de varias maneras, si no se configura ningún otro método, se utiliza la autentificación basada en contraseña. En este apartado se va a estudiar la autentifcación basada en mecanismos de clave pública/privada. Lo primero que hacemos es configurar el servidor para que pueda recibir claves públicas, para ello tenemos que indicar los siguientes valores o verificar que están activos en los parámetros del archivo de configuración /etc/ssh/sshd_config:

```
#RSAAuthentication yes
PubkeyAuthentication yes
AuthorizedKeysFile %h/.ssh/authorized_keys
```

La primera opción (RSAAuthentication) sirve para indicar cuando se permitirá autenticación RSA, está opción esta habilitada por defecto, y en el ejemplo está comentada porque sólo sirve para la versión 1 del protocolo SSH.

La segunda opción (PubkeyAuthentication) es la que especifica si se podrán usar claves públicas para demostrar la autenticidad de un usuario. Si su valor es yes como en el ejemplo, entonces se podrán emplear las claves públicas, si por el contrario su valor es no, entonces el uso de claves públicas quedará prohibido.

La tercera opción (AuthorizedKeysFile) especifica el archivo que contiene las claves públicas empleadas para la autenticación de los usuarios. Por defecto suele ser el archivo .ssh/authorized_keys, dentro de la carpeta personal de cada usuario.

Para poder crear nuevas claves privadas y sus correspondientes claves públicas, OpenSSH dispone de una herramienta llamada ssh-keygen, esta herramienta puede crear claves RSA para el protocolo SSH versión 1, cuyo uso se desaconseja y aquí no se va a hablar de ellas, por otro lado, también puede generar claves RSA o DSA para el protocolo SSH versión 2. Para especificar que tipo de clave crear, se emplea el parámetro -t seguido del tipo de clave: "rsa" o "dsa". Por ejemplo:

```
cliente:~$ ssh-keygen -t rsa
```

```
Generating public/private rsa key pair.
Enter file in which to save the key (/home/usuario/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/usuario/.ssh/id_rsa.
Your public key has been saved in /home/usuario/.ssh/id_rsa.pub.
The key fingerprint is:
e5:34:91:ec:53:d1:dc:58:7c:9d:70:27:2a:5d:f8:a1 usuario@cliente
The key's randomart image is:
+--[ RSA 2048]----+
|          ....*+*=|
|           oooo*o*|
|          .=.oo ..|
|          +ooE .  |
|         S ..     |
```

```
|                   |
|                   |
|                   |
|                   |
+-----------------+
```

Los anteriores comandos piden los mismos datos, el primero, en donde salvar la clave privada, si simplemente pulsamos intro, se salvará en la ruta por defecto (la indicada entre paréntesis), a no ser que emplees varias claves privadas, la ruta por defecto es una buena opción, ya que el cliente la usará sin necesidad de que el usuario tenga que especificarla.

Las siguientes dos cosas que pide son la frase clave con la que encriptar la clave privada y una petición de que se repita la frase para cerciorarse de que no se han cometido errores al escribirla la primera vez. Para la frase con la que encriptar la clave privada se puede emplear cualquier carácter (letras, numeros, signos de puntuación, espacios), en principio el tamaño de la frase puede ser arbitrario, pero ssh-keygen se quejará si es menor de cuatro carácteres. En caso de no introducir ninguna frase, la clave privada quedará sin encriptar, lo cual podria ser útil para realizar algunas automatizaciones como podría ser el realizar copias de seguridad, pero, para uso habitual, se desaconseja dejar las claves privadas sin encriptar por el peligro que puede representar que estas sean robadas.

A continuación el ssh-keygen muestra donde se ha salvado la clave privada (/.ssh/id_rsa) y donde está la clave pública que le corresponde (/.ssh/id_rsa.pub), que no es más que el mismo nombre de archivo con la extensión .pub añadida al final.

En la última línea imprime una huella dactilar que sirve para identificar la clave que acabamos de crear, seguida de un comentario que puede servir para identificar la clave pública.

Por último lo único que hay que hacer es añadir el contenido de la clave pública, /.ssh/id_rsa.pub al fichero /.ssh/authorized_keys de la carpeta de usuario del ordenador remoto con el que queremos acceder de forma directa con scp o mediante la instrucción ssh-copy-id. Si no existe el directorio .ssh en el directorio raíz del usuario deberas crearlo y darle permisos de lectura y escritura adecuados.