Conexión Remota Usando OpenSSH Con Claves Publicas

Integrante: Felix Taborda.

Indice:

Introducción:

OpenSSH es una versión “Libre” del paquete de herramientas de comunicación segura del protocolo SSH/SecSH para redes, una solución de seguridad que está ganando la confianza de un número cada vez mayor de usuarios de Internet. Muchos usuarios que utilizan otros programas de conexión remota, no se dan cuenta que sus contraseñas se están transmitiendo sin cifrar a través de la red. OpenSSH cifra todo el tráfico (incluidas las contraseñas) para eliminar de un modo efectivo las “escuchas”, los secuestros de las conexiones y otros ataques a nivel de red. Además, OpenSSH ofrece amplias posibilidades para la creación de túneles seguros, aparte de una variedad de métodos de autenticación.

OpenSSH dispone de varios métodos para verificar la identidad de un usuario remoto, uno de ellos es el uso de contraseñas de usuarios, pero otro de los métodos se basa en la autenticación RSA, en donde se dispone de un juego de llaves privada/pública que garantiza la identidad de un usuario intentando conectarse al equipo remoto. El juego de llaves tiene la propiedad de que lo que se encripta con una, sólo se puede desencriptar con la otra, pero sin embargo, a partir de la llave pública no se puede derivar la llave privada.

Para asegurarse de que un usuario es quien dice ser por medio de criptografía pública OpenSSH emplea el siguiente procedimiento: el servidor genera un número aleatorio que encripta con la llave pública del usuario y le envía el resultado al cliente, si el usuario es quien dice ser entonces no tendrá ningún problema en desencriptarlo. El cliente aplicará una transformación predefinida a dicho número y lo enviará de vuelta al servidor, el cual, revisará dicho resultado y de ser correcto, dará acceso al cliente.

Configuración del Servidor o Host:

Para comenzar tenemos que tener instalado la aplicación servidor de ssh:

```
sudo apt-get install openssh-server
```

Para poder trabajar con llaves públicas, lo primero que tendremos que hacer será configurar el servidor de SSH para que las acepte. Los archivos de configuración de OpenSSH se ubican en la carpeta */etc/ssh*, en nuestro caso, el archivo que nos interesa es */etc/ssh/sshd_config*, lo abrimos con un editor de textos y modificamos los valores de las siguientes opciones relacionadas con las llaves públicas:

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

La opción "RSAAuthentication" sirve para indicar cuando se permitirá autenticación RSA, está opción esta habilitada por defecto.

La opción "PubkeyAuthentication" es la que especifica si se podrán usar llaves públicas para demostrar la autenticidad de un usuario o no.

La opción "AuthorizedKeysFile" especifica el archivo que contiene las llaves públicas empleadas para la autenticación de los usuarios. Por defecto suele ser el archivo .ssh/autorized_keys, dentro de la carpeta personal de cada usuario.

Finalmente, para que los cambios sean efectivos, será necesario reiniciar el demonio ssh.

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

Configuración del Cliente:

En el cliente se debe instalar la aplicación de cliente ssh, en muchas distros viene instalada por defecto.

```
sudo apt-get install openssh-clients
```

La cual nos permite conectarnos al servidor, copiar archivos, redireccionar la **X11** , etc.
Un ejemplo de uso de los comandos seria:

Ejecutar una aplicación gráfica del servidor en el cliente:

```
ssh -X user@SERVER-REMOTO
xclock
```

Copiar archivo usando scp:

```
scp foto.gif user@SERVER-REMOTO:~/
```

Crear el par de llaves publica/privada:

Para generar el par de llaves publicas/privadas se utiliza el comando “ssh-keygen -t rsa”, siendo -t el tipo de llave empleada “rsa” o “dsa”, si se omite se toma por defecto rsa.

```
ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/user/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/hell/.ssh/id_rsa.
Your public key has been saved in /home/hell/.ssh/id_rsa.pub.
The key fingerprint is:
5f:c3:50:81:e8:d9:b8:66:a3:a4:5f:af:42:9f:d1:27 user@casa
```

A continuación se nos pedirá en donde salvar la llave privada, si pulsamos enter, se salvará en la ruta por defecto (indicada entre paréntesis).

Luego lo que nos pide, son la frase clave con la que se puede encriptar la llave privada y una petición de que se repita la frase. En caso de no introducir ninguna frase, la llave privada quedará sin encriptar, lo cual podría ser útil para realizar algunas automatizaciones como podría ser el realizar copias de seguridad, pero, para uso habitual, se desaconseja dejar las llaves privadas sin encriptar por el peligro que puede representar que estas sean robadas.

Después de haber creado un juego de llaves, lo primero que tenemos que hacer es mantener la llave privada secreta, por defecto ssh-keygen establecerá los permisos del archivo con esta llave para que sólo el dueño pueda leerla y modificarla. Por el contrario la llave pública podrá ser leída por cualquier usuario.

Para poder autenticarnos en un equipo remoto con nuestra llave publica, lo único que tendremos que hacer es añadir nuestra llave pública en el archivo .ssh/authorized_keys dentro del directorio personal del usuario al que queremos acceder.

Para pasar al servidor la llave, podemos usar un comando que incorpora ssh:

```
ssh-copy-id user@SERVER-REMOTO
```

A continuación , pide la contraseña del usuario “user”, y copia la llave publica al servidor
Otra manera seria copiando el archivo “**id_rsa.pub**” creado anteriormente y copiarlo al servidor manualmente:

```
scp ~/.ssh/id_rsa.pub user@SERVER-REMOTO
cat id_rsa.pub >> .ssh/authorized_keys
shred -v --remove id_rsa.pub
```

A partir de este momento, si nos queremos conectar al servidor, en vez de pedirnos la contraseña de usuario nos pide la contraseña con la que fue encriptada la llave privada.

Una conexion mas segura:

Para aumentar la seguridad se puede modificar algunas lineas en el archivo de configuración en el servidor que se encuentra en la ruta */etc/ssh/sshd_config*

```
PasswordAuthentication no
```

No permite que se realice una conexion con la contraseñas de usuarios.

```
PermitRootLogin no
```

No se permite la conexion como administrador.

```
Port 5432
```

Cambiando el puerto de conexion por defecto se previene de ataques específicos

```
ListenAddress 192.168.1.190
```

Por defecto, SSH responderá peticiones a través de todas las interfaces del sistema. Para limitar las interfaces de escucha, puede establecerse lo siguiente, considerando que el servidor a configurar posee la IP **192.168.1.190**

```
AllowUsers Juan@10.0.2.4 Pedro@192.168.1.200
```

También se puede restringir el acceso por usuario y anfitrión desde el cual pueden hacerlo.

```
X11Forwarding Yes
```

Con esta linea controlamos si se permiten o no la ejecución remota de aplicaciones gráficas.

Bibliografia:

- www.openssh.com
- http://www.vilecha.com/Hellguest/ssh_llaves_publicas.asp
- http://linuxcode.wordpress.com
- http://crysol.org/es/ssh-public-key
- http://doc.ubuntu-es.org/Servidor_OpenSSH