Trabajo final del curso de administración de sistemas y servicios telemáticos

Acceso remoto cifrado: SSH

Nombre: Santiago Iglesias Pradas
D.N.I.: 05427189

Índice

¿Qué es SSH?

Secure SHell o *SSH* es un programa que permite la conexión y ejecución de comandos en máquinas remotas de forma segura. Pretende reemplazar a *rlogin* y *rsh*, proporcionando comunicaciones seguras sobre canales que no lo son.

¿Cómo se usa SSH?

ssh [-l login_name] hostname [opciones]

Donde *hostname* es el nombre del servidor remoto al que se conecta el cliente. Si el usuario con el que se desea realizar la conexión remota no coincide con el de la sesión abierta en la máquina cliente, se utiliza la opción *–l login_maquina_remota*.

Otras opciones interesantes que permite, entre otras muchas, son:

-c *algoritmo*: Selecciona el algoritmo a utilizar para encriptar la sesión.
-i fichero: selecciona el fichero de donde se lee la clave privada.
-p puerto: puerto de conexión a la máquina remota.
-v: muestra mensajes (útil para la depuración), etc.

¿Por qué se utiliza SSH?

Con la popularización de Internet, también se han popularizado los *sniffers*, programas que analizan el tráfico que circula por la red y pueden reconstruir las conexiones. Estos programas permiten "ver" los nombres de usuario y las contraseñas de dichos usuarios. Una vez que un intruso tiene el nombre de usuario y su contraseña, tiene acceso al sistema, poniendo en peligro la integridad de este.

¿Cómo funciona SSH? Métodos de autentificación que soporta SSH.

SSH tiene una arquitectura cliente-servidor, y soporta distintos métodos de autentificación.

Fichero .rhosts. La máquina cliente debe estar en el fichero */etc/hosts.equiv* o en el */etc/ssh/shosts.equiv* de la máquina remota o servidor. Además, los nombres de usuario deben ser los mismos en ambas máquinas.
Si existe el fichero *.rhosts* o *.shosts* en el *home directory* del usuario en la máquina remota y contiene una línea con el nombre de la máquina cliente y el usuario en esa máquina, también permite la conexión.
No obstante, esta forma de autentificación no suele permitirla el servidor porque no es segura.

Método combinado entre *.rhosts* o *hosts.equiv* y autentificación basada clave pública/clave privada. Mediante este método, sólo se permite el acceso si se cumple alguna de las condiciones del método anterior y si se verifica la clave pública del cliente, que debe estar en el fichero *$HOME/.ssh/known_hosts* o en */etc/ssh/shosts.equiv*

Método basado en clave pública/clave privada. El esquema está basado en una clave pública, que se utiliza para encriptar los datos y una clave privada para descifrarlos.
Cada usuario genera una pareja de claves pública/privada. El servidor sólo conoce la pública, que se lista en el fichero *$HOME/.ssh/authorized_keys*. Sólo las claves públicas que aparezcan en este fichero podrán acceder al servidor. El proceso es el siguiente:

1. Cuando un usuario se conecta, el servicio de acceso remoto le dice al servidor con qué clave se va a autentificar (su clave pública)

2. El servidor comprueba que la clave es válida y manda un número aleatorio al cliente (SSH del usuario remoto) encriptado con la clave pública.
3. El usuario debe descifrar el mensaje enviado con la clave privada, que en ningún momento se baja del servidor. Si tiene éxito la operación se establece la conexión.

Todo este proceso lo implementa SSH automáticamente. El usuario genera su pareja de claves ejecutando *ssh-keygen*, que deja la clave pública en *$HOME/.ssh/identity.pub* y la privada en *$HOME/.ssh/identity*. A continuación, se debe copiar la primera en su *$HOME/.ssh/authorized_keys* de la máquina remota. De este modo, el usuario se podrá conectar sin ninguna contraseña de una forma mucho más segura que con el fichero *.rhosts*.
La manera más conveniente de utilizar este método es mediante un agente de autentificación (*ssh-agent*)

Método basado en servidores de autentificación TIS. El servicio SSH también puede autentificar contra a los usuarios contra un servidor TIS, para ello mapea los nombres de usuarios locales con los de la base de datos TIS en el fichero */etc/ssh/sshd_tis.map*. Si no encuentra la entrada, supone los nombres iguales.

Si los métodos expuestos fallan, SSH pide al usuario una contraseña, que es enviada a la máquina remota para su comprobación. No obstante, mejora las condiciones de seguridad respecto al *telnet* normal ya que ésta va cifrada. Una vez se acepta la conexión con el servidor, todas las comunicaciones se cifran.

¿En qué ordenadores se puede utitilizar SSH?

El servidor está disponible en dominio público para casi todos los tipos de UNIX (*Solaris, SunOS, HP-UX, Linux*, etc.)
El cliente esta para esos mismos UNIX y hay una versión para Windows.

¿Cómo se configura el servicio?

SSH obtiene los datos de configuración en el siguiente orden:

1. - Opciones de la línea de comando.
2. - Fichero de configuración del usuario: ***$HOME/.ssh/config***.
3. - Fichero de configuración del sistema: ***/etc/ssh/ssh_config***.

El servicio buscará cada parámetro en este orden y el primer valor que encuentre lo tomará como válido, desechando los que pueda encontrar posteriormente. Dentro de estos ficheros de configuración, se pueden especificar valores concretos dependiendo del *host* remoto al que se conecte. Debido a que, como se ha comentado antes, el primer valor que encuentra es el válido, estos valores concretos para cada *host* deben emplazarse al principio del archivo y los valores que se desean por defecto al final.

El formato del fichero de configuración es el siguiente:
Las líneas vacías y que comienzan con el carácter '#' son comentarios. Las demás líneas son del tipo "argumentos de los parámetros" o "parámetro=argumento". Un ejemplo del *$HOME/.ssh/config*, se muestra a continuación[1]:

```
Host *panix.com
        User pepe
        Compression no
```

[1] Es importante destacar que los ficheros de configuración son sensibles a las mayúsculas o minúsculas, mientras que los nombres de los parámetros no lo son.

*Host *gw.com*
FallBackToRsh no

*Host * #Opción por defecto, si el host no coincide con los anteriores*
Compression yes
CompressionLevel 9
FallBackToRsh yes
KeepAlive no

Algunos de los parámetros más importantes que se incluyen en este fichero son:

Host. Restringe las declaraciones que le siguen a las máquinas que coincidan con el argumento. Si en el argumento se utilizan caracteres como '*' o '?', las sentencias que le siguen pueden referirse a varias máquinas.

Batchmode. Los argumentos pueden ser "yes" o "no". Si el argumento es "yes" no pide contraseña. Esta opción es útil para su uso en *scripts*.

Cipher. Especifica el algoritmo de codificación con el que se encriptará la sesión. En la actualidad, soporta *idea* (opción por defecto), *des*, *3des*, *blowfish arcfour* y *none*. Con esta última opción no se encripta la comunicación, haciéndola insegura y, por tanto, útil únicamente para depuración.

Compression. Especifica el uso de compresión mediante los parámetros "yes" o "no".

CompressionLevel. Especifica el nivel de compresión cuando ésta está activada. El argumento es un número entero comprendido en un rango entre 1 y 9, primando en 1 la rapidez y en 9 el menor tamaño en la compresión, aunque en este último caso pueda ralentizarse la comunicación. El valor por defecto es 6.

ConnectionAttemps. Especifica el número de intentos fallidos que permite el servicio antes de interrumpir la ejecución del servicio.

FallBackToRsh. Especifica si la conexión es rechazada por el servidor al no tener el servicio activo. Se utiliza automáticamente *rsh*, mandando un mensaje de avisando al usuario de que la comunicación no está cifrada. El argumento es "yes" o "no".

GlobalKnownHostsFile. Especifica un fichero para ser utilizado en vez del que utiliza por defecto, que es */etc/ssh/ssh_known_hosts*.

IdentityFile. Especifica el fichero de donde se lee la identidad del usuario para autentificarse por el método RSA. El que se usa por defecto es *$HOME/.ssh/identity*.

KeepAlive. La opción por defecto es "yes", y así el usuario es avisado si se cae la red o se muere la máquina remota. Si no se quiere utilizar, se debe especificar en los ficheros de configuración tanto del cliente como del servidor.

PasswordAuthentication. Especifica si se usa contraseña o no.

Port. Especifica el puerto del servidor al que se desea conectarse. El utilizado por defecto es el 22.

UsePrivilegePort. Especifica si se usa un puerto protegido en el servidor para establecer la conexión. La opción por defecto es "yes".

User. Especifica el nombre del usuario con el que se establece la conexión. Es útil si se tienen distintos nombres de usuarios en las máquinas entre las que se establece la conexión.

UserKnownHostsFile. Especifica el fichero a utilizar en lugar del *$HOME/.ssh/known_hosts*.

UseRsh. Obliga a utilizar *rlogin/rsh* en el *host* donde se activa (opción "yes"), el resto de los argumentos de los parámetros (dentro del mismo *host*, ya que no afecta al parámetro *host*) quedan inhabilitados.

Existen a su vez distintas opciones que no se han descrito por ser específicas y muy concretas (bien de cada método de autentificación, de redireccionamientos, etc.)

¿Cuáles son las variables de entorno cuyo valor modifica el servicio?

Entre las variables de entorno más importantes que se van afectadas por el servicio SSH, se encuentran las siguientes:

DISPLAY. Que indica la localización del servidor X11. El servicio actualiza automáticamente su valor a "*nombre_host:n*", donde *nombre_host* indica el host donde corre la shell y n es un entero mayor o igual a 1.

HOME. Apunta al *home directory* del usuario.

LOGNAME. Sinónimo de *USER*, se establece para mantener la compatibilidad con los sistemas que utilizan esta variable de entorno.

MAIL. Apunta al *mailbox* del usuario.

PATH. Apunta al path por defecto, como se especifica cuando se lanza SSH.

USER. Se le asigna el valor del usuario que se conecta.

Adicionalmente, SSH lee de los ficheros */etc/enviroment* y *$HOME/.ssh/environment* y añade variables en el formato *nombre_variable=valor*. Algunos sistemas, como *Solaris*, utilizan mecanismos alternativos como el fichero */etc/default/login*.

¿Cuáles son los ficheros donde se guarda información relativa al servicio?

Como se ha ido comentando en los capítulos anteriores, SSH guarda información relacionada con distintos aspectos del servicio en varios ficheros. El administrador del sistema se encargará de que la información almacenada en ellos sea la adecuada para su correcto funcionamiento. Estos ficheros son:

$HOME/.ssh/identity. Contiene la identidad del usuario (clave privada) para el uso del método de autentificación basado en RSA. Esta información es delicada y no debe estar accesible para el resto de los usuarios. Se puede codificar su contenido para lograr un mayor grado de confidencialidad.

$HOME/.ssh/identity.pub. Contiene las claves públicas para la autentificación por RSA. El contenido de este fichero debe ser añadido al *$HOME/.ssh/authorized_keys* de todas las máquinas (servidores) a las cuales se quiera realizar la conexión remota.

$HOME/.ssh/config. Fichero de configuración individual de cada usuario, que es utilizado por el cliente de SSH. El formato se ha descrito en el punto anterior. Se recomienda que no sea accesible al resto de los usuarios.

$HOME/.ssh/authorized_keys. Contiene las claves públicas de las máquinas desde las que se puede conectar el usuario.

/etc/ssh/ssh_known_hosts. Contiene todas las máquinas conocidas por el sistema. Este fichero debe prepararlo el administrador para que contenga las claves públicas de todas las máquinas de la organización. Debe ser accesible para todos los usuarios. El formato es el siguiente:
system_name number_of_bits_in_modulus public_exponent modulus [comments]

/etc/ssh/ssh_config. Fichero de configuración del sistema, especifica parámetros para las máquinas que no existen en los ficheros de cada usuario. Debe ser accesible para todo el mundo.

$HOME/.rhosts. Fichero utilizado en el método de autentificación basado en el fichero *.rhosts* para definir la pareja de máquina-usuario a los cuales se les permite la conexión ("invitados") El formato es:
nombre_maquina_invitada usuario_invitado
Este fichero debe tener permisos sólo para cada usuario particular, la recomendación es lectura/escritura para el usuario y no accesible para los demás.

$HOME/.shosts. Es utilizado de la misma forma que el anterior. El motivo de su mantenimiento es permitir la autentificación SSH basada en *.rhosts* sin permitir la conexión con *rlogin* o *rsh*.

/etc/hosts.equiv. Se usa con la autentificación basada en *.rhosts*. Contiene los nombres canónicos de las máquinas, uno por cada línea. Si la máquina cliente se encuentra en este fichero, la conexión se permite automáticamente. Debe ser modificable únicamente por el usuario *root*.

/etc/ssh/shosts.equiv. Es utilizado de la misma forma que el anterior. El motivo de su mantenimiento es permitir la autentificación SSH sin permitir la conexión con *rlogin* o *rsh*.

¿Cuáles son los requisitos de la instalación?

SSH se instala normalmente como *root*. Sólo se necesitan permisos de *root* para la autentificación mediante *rhosts*, ya que requiere que la conexión proceda de un puerto de acceso restringido para el resto de usuarios. También se necesita permisos de lectura del fichero */etc/ssh/ssh_host_key* para desarrollar la autentificación basada en clave pública/clave privada.
Se puede utilizar SSH sin permisos de *root*, pero la autentificación mediante *rhosts* queda deshabilitada. SSH elimina el resto de privilegios extras inmediatamente después de que se haya establecido la conexión.

¿Qué actividades diarias conlleva la administración del servicio?

No acarrea actividades más complejas de las que se han ido enumerando a lo largo del documento. Las actividades principales serían las motivadas por el mantenimiento de los ficheros, como por ejemplo el *$HOME/.ssh/known_hosts*, que es donde se almacenan las claves de todas las máquinas del sistema. Los esfuerzos se deben centrar en una buena configuración inicial ya que la seguridad es un factor crítico en la administración de todo sistema informático y es bueno no dejar huecos por donde puedan acceder personas no deseadas.

Temas relacionados y pruebas del servicio

Sshd. Es el demonio del servicio SSH, permanece a la escucha en un puerto del servidor de las posibles conexiones de las máquinas clientes. Normalmente se inicializa en el arranque desde el fichero */etc/rc.local* y genera un proceso hijo por cada conexión entrante.
El fichero de donde lee la configuración es /etc/ssh/sshd_config (a no ser que se le especifique la opción -f en la línea de comandos)
Tiene multitud de parámetros de configuración que no se detallan por su extensión, como se puede apreciar en el siguiente ejemplo:

 # Ejemplo de /etc/ssh/sshd_config.

```
Port 22
ListenAddress 0.0.0.0
HostKey /etc/ssh_host_key
RandomSeed /etc/ssh_random_seed
ServerKeyBits 768
LoginGraceTime 600
KeyRegenerationInterval 3600
PermitRootLogin yes
IgnoreRhosts no
StrictModes yes
QuietMode no
X11Forwarding yes
X11DisplayOffset 10
FascistLogging no
PrintMotd yes
KeepAlive yes
SyslogFacility DAEMON
RhostsAuthentication no
RhostsRSAAuthentication yes
RSAAuthentication yes
PasswordAuthentication yes
PermitEmptyPasswords yes
UseLogin no
# AllowHosts *.our.com friend.other.com
# DenyHosts lowsecurity.theirs.com *.evil.org evil.org
# Umask 022
# SilentDeny on
AllowHosts *.acme 10.0.*
```

Para depurar los posibles errores, se puede ejecutar con la opción de *debug* (-d). La salida por pantalla es la que se muestra a continuación:

```
[root]# sshd -d
    debug: sshd version 1.2.22 [i586-unknown-linux]
    debug: Initializing random number generator; seed file
    /etc/ssh_random_seed
    log: Server listening on port 22.
    log: Generating 768 bit RSA key.
    Generating p:  ............................++ (distance 590)
    Generating q:  .............++ (distance 178)
    Computing the keys...
    Testing the keys...
    Key generation complete.
    log: RSA key generation complete.
    debug: Server will not fork when running in debugging mode.
    log: Connection from 10.0.2.1 port 1023
    debug: Client protocol version 1.5; client software version 1.2.22
```

debug: Sent 768 bit public key and 1024 bit host key.
debug: Encryption type: idea
debug: Received session key; encryption turned on.
debug: Attempting authentication for pepe.
debug: Trying rhosts with RSA host authentication for pepe
debug: RhostsRSA authentication failed for 'pepe', remote 'pepe', host 'pepe_host'.
log: Password authentication for pepe accepted.
debug: Allocating pty.
debug: Received request for X11 forwarding with auth spoofing.
debug: Allocated channel 0 of type 1.
debug: Forking shell.
debug: Setting controlling tty using TIOCSCTTY.
debug: Entering interactive session.
debug: Received SIGCHLD.
debug: End of interactive session; stdin 1, stdout (read 561, sent 561), stderr 0 bytes.
debug: pty_cleanup_proc called
debug: Command exited with status 0.
debug: Received exit confirmation.
log: Closing connection to 10.0.2.1
[root]#

Permisos en los directorios y en los ficheros. Uno de los posibles fallos en el acceso puede venir ocasionado por los permisos de los siguientes ficheros:
- El propio *home directory*.
- *$HOME /.ssh*
- *$HOME/.ssh/authorized_keys*

Sólo debe tener permisos de escritura el propietario. Este ejemplo muestra los permisos más amplios posibles que se podrían dar a estos ficheros:

sip% cd
sip% ls -ld . .ssh .ssh/authorized_keys
drwxr-xr-x 36 kim kim 4096 Jul 25 02:24 .
drwxr-xr-x   2 kim kim   512 Apr 10 02:30 .ssh
-rw-r--r--   1 kim kim 1674 Apr 10 02:29 .ssh/authorized_keys

Es necesario realizar esta operación en todas las máquinas a las que se desee establecer conexiones.

Copia de ficheros remotos: *scp (secure copy)*. Para especificar el archivo de la máquina remota, se le añade un prefijo con el nombre de la máquina seguido de ':', como se muestra en el ejemplo.[2]

sip% scp guanchi:aliases .

Make-ssh-known-hosts. Es un *script* de *perl* que ayuda a crear el fichero /etc/ssh/ssh_known_hosts, que es donde se guardan las claves de todas las máquinas conocidas. Sirve por tanto de gran ayuda al administrador del servicio.

Ssh-keygen. Comando que genera y gestiona las claves de autentificación para SSH. Su utilización se muestra a continuación (las entradas al sistema están en negrita).

sip% ***ssh-keygen***

[2] Los *paths* relativos de los ficheros se resuelven de distinta forma en el cliente que en el servidor. Mientras que en el primero se asume el directorio actual, como en la ejecución normal, en el segundo caso son relativos al *home directory* del usuario

Initializing random number generator...
Generating p: .++ (distance 6)
Generating q: ....++ (distance 110)
Computing the keys...
Testing the keys...
Key generation complete.
Enter file in which to save the key ($HOME/.ssh/identity): ***[RETURN]***
Enter passphrase (empty for no passphrase): ***esta es mi clave***
Enter same passphrase again: ***esta es mi clave***
Your identification has been saved in /users/sip/.ssh/identity.
Your public key is:
1024 37 [lots of numbers]
Your public key has been saved in /users/sip/.ssh/identity.pub

Se puede cambiar la clave con la opción -p del comando.

Ssh-agent. Es un programa que mantiene las claves privadas del usuario. La idea es que el agente se inicie al comienzo de una sesión (consola o X) y las demás pantallas o programas se inicien como hijos del agente SSH. Los programas invocados bajo el agente heredan su conexión, encargándose este mismo de la autentificación.

sip% ssh-agent startx &

Ssh-add. Añade identidades al agente SSH. Este agente debe estar ejecutándose y debe ser el proceso padre para que *ssh-add* funcione correctamente.