Servicios de acceso remoto II: SSH.

Autor: Enrique V. Bonet Esteban

Introducción.

Secure SHell es un protocolo que permite crear conexiones seguras entre dos ordenadores. Usando SSH, la máquina del cliente inicia una conexión con la máquina del servidor mediante una sesión cifrada, imposibilitando que alguien pueda obtener una contraseña o cualquier otro tipo de información que se envíe por la red[1].

SSH esta diseñado para reemplazar tanto los métodos comunes de acceso remoto a un interprete de comandos de otro ordenador, tales como rlogin o telnet, como otros programas diseñados para copiar ficheros entre ordenadores como por ejemplo rcp o ftp, pues estas aplicaciones no cifran las contraseñas entre el cliente y el servidor[2].

SSH proporciona los siguientes tipos de protección:

- Una vez se ha realizado una primera conexión a un servidor, el cliente puede verificar que se está conectando al mismo servidor durante posteriores sesiones.

- El cliente transmite su información de autenticación al servidor, como el nombre de usuario y la contraseña, en formato cifrado[3].

- Todos los datos enviados y recibidos durante la conexión se transfieren por medio de encriptación fuerte, lo cual los hacen extremamente difícil de descifrar y leer por alguien ajena a la comunicación.

- El cliente tiene la posibilidad de usar X11 desde aplicaciones lanzadas desde el intérprete de comandos. Esta técnica proporciona una interfaz gráfica segura[4] y un medio seguro para usar aplicaciones gráficas sobre una red.

Como el protocolo SSH cifra todo lo que envía y recibe, se puede usar para asegurar protocolos inseguros. El servidor SSH puede convertirse en un conducto para convertir en seguros los protocolos inseguros mediante el uso de una técnica llamada reenvío por puerto, como por ejemplo POP3[5], incrementando la seguridad del sistema y la privacidad de los datos.

Actualmente existen dos variedades diferentes de SSH, la versión 1 y la versión 2, siendo recomendable el uso de la versión 2, pues la versión 1 de SSH contiene

[1] El uso de SSH requiere el uso de Secure Socket Layer (SSL), que proporciona librerías criptográficas para establecer las comunicaciones encriptadas.
[2] Este reemplazo se realiza mediante los programa sftp o scp, los cuales utilizan SSH para su conexión.
[3] Actualmente, el cifrado utilizado es de 128 bits, lo que lo convierte en un cifrado robusto y muy difícil, por no decir imposible, de descifrar si no se conoce la clave.
[4] Esta técnica se conoce como reenvío de X11.
[5] POP3 es un protocolo de envío final de correo electrónico que veremos en temas posteriores.

algunos algoritmos de encriptación patentados[6] y un agujero de seguridad que potencialmente permitiría la inserción de datos falsos en el flujo de datos.

Establecimiento de una conexión mediante SSH.

El establecimiento de una conexión mediante SSH sucede en tres pasos sucesivos.

1. Creación de una capa de transporte segura (TLS) para que el cliente sepa que se está comunicando con el servidor correcto. Luego se cifra la comunicación entre el cliente y el servidor por medio de un código simétrico.

2. Autenticación del cliente ante el servidor, sin preocuparse de que la información de autenticación pueda ponerse en peligro.

3. Uso de la conexión establecida por los servicios que requieran una conexión segura, como pueden ser un interprete de comandos, una aplicación X11 o un túnel TCP/IP.

Creación de la capa de transporte segura.

El papel principal de la capa de transporte, que se ejecuta de forma normal sobre TCP/IP, es el de facilitar una comunicación segura entre los dos ordenadores tanto en el momento de la autenticación como con posterioridad.

Para ello, la capa de transporte verifica que el servidor sea la máquina correcta para la autenticación, se ocupa del cifrado y descifrado de la información y proporciona protección a la integridad de los paquetes de datos cuando son enviados y recibidos. Además, la capa de transporte también puede comprimir los datos a enviar, acelerando la transmisión de la información.

Al establecer la comunicación mediante SSH, un cliente negocia con el servidor varios aspectos que permitirán construir la capa de transporte correctamente. Esa negociación se produce en los siguientes pasos:

1. Intercambio inicial de claves.

2. Selección del algoritmo de clave pública utilizado.

3. Selección del algoritmo de encriptación simétrico utilizado.

4. Selección del algoritmo de autenticación de mensajes utilizado.

5. Selección del algoritmo de hash utilizado.

El servidor se identifica ante el cliente con una clave de ordenador durante el intercambio de claves. Obviamente, si el cliente nunca se había conectado con el servidor, la clave del servidor le resultará desconocida al cliente. SSH evita este problema permitiendo que el cliente acepte la clave de ordenador del servidor la primera

[6] Algunas de las patentes de esos algoritmos ya han caducado.

vez que se lleva a cabo una conexión SSH[7]. Luego la clave de ordenador servidor se puede verificar con la versión guardada en el cliente en las siguientes conexiones, proporcionando la confianza que el cliente está realmente comunicando con el servidor deseado[8].

Después del intercambio inicial de claves, y de la selección de los algoritmos a utilizar, se crean dos valores, un valor de hash usado para intercambios y un valor de secreto compartido, y los dos sistemas empiezan inmediatamente a calcular claves y algoritmos nuevos para proteger la autenticación y los datos que se enviarán a través de la conexión en el futuro.

Después que una cierta cantidad de datos ha sido transmitida con un determinado algoritmo y clave (la cantidad exacta depende de la ejecución de SSH), ocurre otro intercambio de claves, el cual genera otro conjunto de valores de hash y otro valor de secreto compartido. De esta manera aunque un agresor lograse determinar los valores de hash y de secreto compartido, deberá repetir todo el procedimiento después de un intercambio de claves nuevo para poder continuar descifrando la comunicación.

Autenticación.

Cuando la capa de transporte ha construido un túnel seguro para transmitir información entre los dos sistemas, el servidor informa al cliente de los diferentes métodos de autenticación soportados, como el uso de firmas privadas codificadas con claves o la inserción de una contraseña. El cliente entonces intentará autenticarse ante el servidor mediante el uso de cualquiera de los métodos soportados.

Ya que los servidores se pueden configurar para que concedan varios tipos de autenticación, este método proporciona a cada parte un control óptimo. Luego el servidor podrá decidir qué métodos de encriptación soportará basado en su pauta de seguridad, y el cliente puede elegir el orden en que intentará utilizar los métodos de autenticación entre las opciones a disposición. Gracias a la naturaleza segura de la capa de transporte de SSH, hasta métodos de autenticación que parecen inseguros, como la autenticación basada en el ordenador, son en realidad seguros.

La mayoría de los usuarios que requieren un intérprete de comandos seguro se autenticarán por medio de una contraseña. Ya que la contraseña está cifrada en el proceso de envío, se puede enviar fácilmente y de forma segura a través de la red.

Conexión.

Después de una autenticación exitosa sobre una capa de transporte SSH, se abren canales múltiples por medio de la multiplexión[9]. Cada canal se ocupa de la

[7] Generalmente, la aceptación se lleva a cabo preguntando al usuario si desea aceptar esa clave como identificación valida de ese servidor.

[8] Un agresor podría enmascararse como servidor SSH durante el contacto inicial ya que el sistema local no conoce la diferencia entre el servidor en cuestión y el falso configurado por un agresor. Pare evitar que esto ocurra debería verificar la integridad del nuevo servidor SSH contactando con el administrador del servidor antes de conectarse por primera vez.

[9] Una conexión multiplexada consiste en varias señales enviadas simultáneamente por un medio compartido. Con SSH, se envían varios canales en una conexión en común segura.

comunicación para sesiones entre terminales diferentes, el reenvío de información por X11 o cualquier servicio aparte que intente usar la conexión SSH.

Tanto el cliente como el servidor pueden crear un canal nuevo, asignando un número diferente para cada canal en cada lado. Cuando una parte intenta abrir un canal nuevo, el número asignado para el canal en esa parte se envía junto con la petición. Esta información es almacenada en la otra parte y se usa para dirigir un determinado tipo de comunicación de servicio a ese canal. Esto se lleva a cabo para que diferentes tipos de sesiones no interfieran entre ellas y que los canales puedan cerrarse sin interrumpir la conexión SSH principal entre los dos sistemas.

Los canales también soportan el control de flujo, el cual les permite enviar y recibir datos ordenadamente. De esta manera, un ordenador no envía datos a través del canal hasta que no haya recibido un mensaje avisando que el canal puede recibirlos.

Los canales son especialmente útiles para el reenvío por X11 y el reenvío por puerto TCP/IP con SSH. Se pueden configurar canales aparte en modo diferente, tal vez para usar un tamaño de paquete máximo diferente o para transferir un determinado tipo de datos. Esto permite que SSH sea flexible en su modo de encargarse de los diferentes tipos de conexiones remotas, como el acceso telefónico en redes públicas o enlaces LAN de alta velocidad, sin tener que cambiar la infraestructura básica del protocolo.

Configuración de un servidor SSH.

El servidor de SSH se es el programa */usr/sbin/sshd*, siendo arrancado el servicio de SSH mediante el comando:

```
systemctl start sshd.service
```

La configuración del servidor se realiza en el fichero */etc/ssh/sshd_config*, si bien el fichero de configuración a utilizar puede modificarse mediante la opción *-f*[10]. Un aspecto a destacar es que si el fichero de configuración por defecto no existe, o bien no existe el fichero especificado mediante la opción *-f*, el servicio de SSH no se ejecuta, mostrando un mensaje de error que indica la falta de un fichero de configuración.

Las opciones principales de configuración del servidor de SSH se encuentran en la siguientes tablas. En ellas, todas las opciones que se encuentran marcadas por el símbolo * indican que solo están disponibles para la versión 2 del protocolo SSH.

Opciones generales

Opción	Descripción	Valor por defecto
AcceptEnv*	Indica las variables de ambiente enviadas por el cliente que serán aceptadas por el servidor.	Ninguna variable es aceptada.
AddressFamily	Especifica la familia de direcciones IP aceptadas por el servidor, los valores pueden ser *any*, *inet* ó *inet6*.	any
AllowTcpForwarding	Autoriza el reenvío de puertos.	Yes
GatewayPorts	Especifica si ordenadores remotos están autorizados a utilizar puertos reenviados a otros clientes. Los valores posibles son *no*, *yes* y *clientspecified*.	No

[10] Existen muchas más opciones que permiten modificar el funcionamiento del servidor. Si se desean conocer consultar la página de manual mediante *man sshd*.

Opciones generales

Opción	Descripción	Valor por defecto
ListenAddress	Dirección IP local que escucha las conexiones entrantes. Pueden especificarse varias entradas para indicar varias direcciones de red.	Todas las direcciones.
Port	Puerto en que permanece a la escucha el servidor en espera de conexiones. Pueden especificarse varias entradas para especificar varios puertos distintos.	22 TCP.
Protocol	Versión de los protocolos SSH soportados por el servidor y orden de preferencia.	Versión 2.
TCPKeepAlive	Indica si deben enviarse paquetes para comprobar si la conexión con el cliente se encuentra activa.	Yes
UseDNS	Indica si se debe realizar una comprobación inversa de la identidad del cliente.	Yes
UsePrivilegeSeparation	Indica si SSH creará un proceso hijo sin privilegios una vez el usuario ha accedido al sistema.	Yes

Opciones de configuración de acceso

Opción	Descripción	Valor por defecto
AuthorizedKeysFile	Fichero con las claves públicas usadas para autenticación.	~/.ssh/authorized_keys.
ChallengeResponseAuthentication	Indica si el intercambio de respuestas de autenticación es permitido.	Yes
Ciphers*	Indica los cifrados permitidos por el protocolo.	Todos.
GSSAPIAuthentication*	Especifica si la autenticación basada en GSSAPI es permitida.	No
GSSAPICleanupCredentials*	Especifica si las credenciales son automáticamente destruidas cuando termina la sesión.	Yes
HostbasedAuthentication*	Autoriza el acceso mediante clave pública de usuarios de los ordenadores indicados en *rhosts* o en */etc/hosts.equiv*.	No
HostKey	Especifica el fichero que contiene la clave privada del servidor. Sus valores por defecto son */etc/ssh/ssh_host_key* para la version 1 y */etc/ssh/ssh_host_rsa_key* y */etc/ssh/ssh_host_dsa_key* para la versión 2.	Ver descripción.
IgnoreRhosts	Deniega el uso de los ficheros *.rhosts* y *.shosts* en el acceso remoto.	Yes
IgnoreUserKnowHosts	Deniega el uso del fichero *~/.ssh/know_hosts* para encontrar los ordenadores conocidos.	No
LoginGraceTime	Tiempo, en segundos, antes de que se cierre la sesión de autenticación.	120 segundos.
LogLevel	Información que se escribirá en los accesos. Sus valores posibles son, de menor a mayor información QUIET, FATAL, ERROR, INFO, VERBOSE, DEBUG, DEBUG1, DEBUG2 y DEBUG3.	INFO
MaxAuthTries	Número máximo de intentos de autenticación por conexión.	6
MaxStartups	Número máximo de conexiones simultaneas en estado de autenticación.	10
PasswordAuthentication	Permite la autenticación mediante contraseña.	Yes
PermitEmptyPasswords	Permite el acceso a usuarios sin contraseña.	No
PermitRootLogin	Permite el acceso de root mediante SSH.	Yes

Opciones de configuración de acceso

Opción	Descripción	Valor por defecto
PermitUserEnvironment	Especifica si las variables de ambiente del usuario serán procesadas por SSH.	No
PubkeyAuthentication*	Permite la autenticación mediante clave pública.	Yes
RhostsRSAAuthentication	Indica si se permite el uso de *rhost* o */etc/hosts.equiv* en la autenticación mediante RSA. Solo aplicable a la versión 1 del protocolo.	No
RSAAuthentication	Permite la autenticación mediante RSA. Solo aplicable a la versión 1 del protocolo.	Yes
UseLogin	Indica si se utiliza login para comprobar el acceso de los usuarios.	No
UsePAM	Indica si se utiliza PAM para comprobar el acceso de los usuarios.	No

Opciones de usuarios y grupos

Opción	Descripción	Valor por defecto
AllowGroups	Lista de nombres de grupos, separados por espacios, cuyos miembros, sea como grupo primario o grupo suplementario, tienen permitido el acceso al sistema mediante SSH. Pueden utilizarse los caracteres comodín * e ?.	Todos los grupos.
AllowUsers	Lista de nombres de usuarios, separados por espacios, cuyo acceso al sistema esta permitido por SSH. Puede tomar la forma usuario@ordenador, comprobando entonces tanto el nombre del usuario como el nombre del ordenador desde el que intenta el acceso. Pueden utilizarse los caracteres comodín * e ?.	Todos los usuarios.
DenyGroups	Lista de nombres de grupos, separados por espacios, cuyos miembros, sea como grupo primario o grupo suplementario, no tienen permitido el acceso al sistema mediante SSH. Pueden utilizarse los caracteres comodín * e ?.	Ningún grupo.
DenyUsers	Lista de nombres de usuarios, separados por espacios, cuyo acceso al sistema no esta permitido por SSH. Puede tomar la forma usuario@ordenador, comprobando entonces tanto el nombre del usuario como el nombre del ordenador desde el que intenta el acceso. Pueden utilizarse los caracteres comodín * e ?.	Ningún usuario.

El orden de procesamiento de las opciones de usuarios y grupos es: *DenyUsers*, *AllowUsers*, *DenyGroups*, *AllowGroups*. Si se especifican entradas en *DenyUsers* o *DenyGroups*, los usuarios o grupos especificados tienen denegado el acceso por SSH, mientras que el resto de usuarios o grupos pueden tener acceso. Si se especifican entradas en *AllowUsers* o *AllowGroups*, solo los usuarios o grupos especificados tienen permitido el acceso por SSH al sistema, teniendo el resto de usuarios o grupos no especificados denegado el acceso al sistema por SSH.

Opciones de reenvío de conexiones X11

Opción	Descripción	Valor por defecto
X11DisplayOffset	Indica el primer identificador de pantalla que utilizará SSH en sus conexiones X11 para no interferir con los identificadores locales X11.	10
X11Forwarding	Permite el reenvío de conexiones X11.	No
X11UseLocalhost	Indica si SSH escucha las conexiones X11 en el interfaz de loopback o en los otros interfaz de red existentes.	Yes
XAuthLocation	Indica la localización del programa de autorización de acceso mediante X11.	/usr/bin/xauth

Otras opciones de configuración

Opción	Descripción	Valor por defecto
Banner*	Muestra un mensaje antes de acceder al servidor de SSH.	Sin ningún mensaje.
ClientAliveCountMax*	Número de paquetes de comprobación sin responder que se espera antes de cerrar la conexión por no obtener respuesta del cliente.	3
ClientAliveInterval*	Intervalo de inactividad, en segundos, que el servidor espera antes de enviar un mensaje al cliente solicitando una respuesta.	No activado (valor 0).
Compression	Especifica si la compresión es permitida o retrasada hasta que el usuario se ha autenticado correctamente. Sus valores son yes, no o delayed.	Delayed.
ForceCommand	Fuerza la ejecución del comando especificado.	Ninguno.
PrintLastLog	Especifica si al acceder mediante SSH se mostrará la información del último acceso sucedido.	Yes
PrintMotd	Especifica si SSH mostrará el mensaje del día indicado en */etc/motd*.	Yes
StrictModes	Especifica si SSH debe chequear el modo y propietario de los ficheros en el directorio raíz del usuario antes de permitir su acceso.	Yes
Subsystem*	Configura un subsistema externo, por ejemplo el *sftp-server*.	Ninguno.

Además del fichero anterior, existen una serie de ficheros en el directorio */etc/ssh* que son utilizados por el servidor de SSH. Estos ficheros son:

- *moduli*: Contiene grupos Diffie-Hellman usados para el intercambio de la clave mediante el algoritmo de Diffie-Hellman. Este intercambio de claves es imprescindible para la construcción de una capa de transporte segura. Cuando se intercambian las claves al inicio de una sesión SSH, se crea un valor secreto y compartido que no puede ser determinado por ambas partes a la misma vez. Este valor se usa para proporcionar la autentificación del ordenador.

- *ssh_host_key*: La clave privada RSA usada por sshd en la versión 1.

- *ssh_host_key.pub*: La clave pública RSA usada por sshd en la versión 1.

- *ssh_host_dsa_key*: La clave privada DSA usada por sshd en la versión 2.

- *ssh_host_dsa_key.pub*: La clave pública DSA usada por sshd en la versión 2.

- *ssh_host_rsa_key*: La clave privada RSA usada por sshd en la versión 2.

- *ssh_host_rsa_key.pub*: La clave pública RSA usada por sshd en la versión 2.

Cada vez que un servidor de SSH es instalado, se crean un conjunto nuevo de claves de identificación, con lo que cualquier cliente que ya conozca la clave de identificación del servidor mostrará un mensaje como:

```
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@    WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED!     @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
Someone could be eavesdropping on you right now (man-in-the-middle attack)!
```

```
It is also possible that the RSA host key has just been changed.
```

Para evitar esto, el servidor debe mantener las claves antiguas del sistema y restaurarlas con posterioridad. Este proceso es tan sencillo como realizar una copia de seguridad de los ficheros */etc/ssh/ssh_host*key** y restaurarlos una vez ha sido reinstalado el servidor de SSH.

Creación de un par de claves pública/privada para acceso remoto.

Como hemos podido ver, existe la posibilidad de permitir que un usuario acceda a un servidor de SSH utilizando autenticación basada en un par de claves pública/privada. La creación de este par de claves pública/privada se realiza mediante el comando:

```
ssh-keygen -t <tipo>
```

Donde <*tipo*> debe ser *rsa1* para RSA de la versión 1 del protocolo y *rsa* ó *dsa* para RSA ó DSA de la versión 2 del protocolo.

En todos los casos, se nos solicitará una contraseña de acceso[11] a la clave privada, la cual debe dejarse vacía si no se desea tener que introducirla cada vez.

Los ficheros creados en cada caso se encuentran en la tabla siguiente[12]:

Fichero	Descripción
~/.ssh/identity	Contiene la clave privada RSA para la versión 1 del protocolo SSH.
~/.ssh/identity.pub	Contiene la clave pública RSA para la versión 1 del protocolo SSH.
~/.ssh/id_dsa	Contiene la clave privada DSA para la versión 2 del protocolo SSH.
~/.ssh/id_dsa.pub	Contiene la clave pública DSA para la versión 2 del protocolo SSH.
~/.ssh/id_rsa	Contiene la clave privada RSA para la versión 2 del protocolo SSH.
~/.ssh/id_rsa.pub	Contiene la clave pública RSA para la versión 2 del protocolo SSH.

Cuando deseamos acceder a un ordenador remoto mediante este método, además de que el servidor remoto debe tener permitida la opción, debemos añadir al fichero *~/.ssh/authorized_keys* en el usuario del servidor remoto la clave pública con la que deseamos acceder.

Configuración de un cliente SSH.

El cliente de SSH existente en Linux es */usr/bin/ssh*. La configuración por defecto del mismo se realiza mediante el fichero */etc/ssh/ssh_config*, siendo sus principales opciones las expuestas en la tabla siguiente:

Opción	Descripción	Valor por defecto
Host	Restringe las declaraciones que se hagan a continuación a los ordenadores identificados hasta la siguiente aparición de Host. Pueden usarse los caracteres comodín * e ?.	Ninguno.

[11] La contraseña es cifrada utilizando DES3.
[12] Recuérdese que el símbolo ~ hace referencia al directorio raíz de cada usuario.

Opción	Descripción	Valor por defecto
AddressFamily	Especifica la familia de direcciones IP aceptadas por el servidor, los valores pueden ser *any*, *inet* ó *inet6*.	any
BindAddress	Dirección IP a enviar en ordenadores con múltiples interfaces de red.	Ninguno.
CheckHostIP	Chequea el nombre del ordenador en la lista de ordenadores conocidos.	Yes
EscapeChar	Asigna el carácter de escape de la conexión	~
ForwardAgent	Permite o no el redireccionamiento de la autenticación a un ordenador remoto.	No
ForwardX11	Permite el redireccionamiento de la conexión X11 por la conexión SSH.	No
GatewayPorts	Permite la conexión de ordenadores remotos a los puertos reasignados a una conexión segura.	No.
LocalForward	Especifica el puerto local que puede ser redireccionado a una conexión segura de otro ordenador y puerto.	Ninguno.
LogLevel	Información que se escribirá en los accesos. Sus valores posibles son, de menor a mayor información QUIET, FATAL, ERROR, INFO, VERBOSE, DEBUG, DEBUG1, DEBUG2 y DEBUG3.	INFO
NumberOfPasswordPrompts	Número de veces que se solicita la contraseña antes de cerrar la comunicación.	3
Protocol	Versión de los protocolos SSH soportados por el servidor que será aceptada y orden de preferencia.	Versión 2.
RemoteForward	Especifica que el ordenador y puerto remoto serán redireccionados a una conexión segura en un puerto del ordenador.	Ninguno.

Además del fichero anterior, existen una serie de ficheros, en el directorio *~/.ssh* que permiten modificar la configuración por defecto. De estos ficheros, ya hemos visto con anterioridad los ficheros que contienen los pares de claves públicas/privadas (*identity*, *identity.pub*, *id_dsa*, *id_dsa.pub*, *id_rsa* e *id_rsa.pub*) y el fichero con las claves públicas conocidas y autorizadas (*authorized_keys*). Los otros dos ficheros son:

- *config*: Fichero de configuración que se utiliza, si existe, en lugar del fichero de configuración por defecto del cliente.

- *known_hosts*: Este fichero contiene las claves DSA de los servidores SSH a los que se ha accedido desde este usuario. Este fichero inicialmente esta vacío, insertándose en el mismo las líneas de los servidores que son aceptados la primera vez que el usuario se conecta al mismo.

  La inserción en el mismo de las claves la primera vez se realiza, generalmente, de forma manual, recibiendo esta primera vez el siguiente mensaje:

  ```
  The authenticity of host 'glup.uv.es' can't be established.
  RSA key fingerprint is 94:68:3a:3a:bc:f3:9a:9b:01:5d:b3:07:38:e2:11:0c.
  Are you sure you want to continue connecting (yes/no)?
  ```

  Si respondemos de forma afirmativa, la clave RSA de ese servidor se copiara al fichero junto con la identificación del ordenador, de forma que en posteriores conexiones podremos verificar su identidad y, en caso de que no coincida, el cliente rechazara la conexión.

En el caso de que las claves del servidor están alteradas legítimamente debido a una reinstalación del sistema, para conectarse al servidor, el usuario debe abrir el fichero *known_hosts* con un editor de textos y borrar la clave para ese ordenador. Esto permite que el cliente SSH cree una nueva clave para ese ordenador.

Uso del cliente de SSH.

El cliente de SSH posee un gran número de opciones. Sin embargo, para iniciar una sesión mediante el cliente de SSH en un servidor de SSH las opciones básicas de la línea de comandos son:

```
ssh {[-l usuario] ,[usuario@]}<servidor> [comando]
```

Donde *usuario* indica el nombre del usuario con el que deseamos acceder al servidor[13], *servidor* es el nombre del servidor y *comando* es el comando que deseamos ejecutar, obteniendo una shell del sistema si no se especifica el comando.

Por ejemplo, para acceder con el nombre del usuario actual al servidor *glup.uv.es*, basta con escribir en el intérprete de comandos:

```
ssh glup.uv.es
```

La primera vez que un usuario ejecuta ssh contra un servidor remoto, aparece un mensaje similar al siguiente:

```
The authenticity of host 'glup.uv.es' can't be established.
RSA key fingerprint is 94:68:3a:3a:bc:f3:9a:9b:01:5d:b3:07:38:e2:11:0c.
Are you sure you want to continue connecting (yes/no)?
```

Si se escribe *yes*, se añadirá el servidor a la lista de ordenadores conocidos por el usuario (*~/.ssh/known_host*), mostrándose el siguiente mensaje:

```
Warning: Permanently added 'glup.uv.es,147.156.222.65' (RSA) to the
list of known hosts.
```

A continuación, aparece un indicador de comandos preguntando la contraseña desde el ordenador remoto. Después de introducir la contraseña, se ejecutará un interprete de comandos en el ordenador remoto.

Si deseamos especificar un nombre de usuario diferente, por ejemplo *root*, podemos escribir cualquiera de las dos opciones siguientes:

```
ssh -l root glup.uv.es
ssh root@glup.uv.es
```

Y si deseamos ejecutar un comando, por ejemplo *ls /usr/share/doc* en el ordenador *glup.uv.es*, como usuario *root*, la línea de comandos es:

```
ssh root@glup.uv.es ls /usr/share/doc
```

[13] Si no se especifica ningún usuario, el cliente envía al servidor el nombre del usuario que somos en el ordenador cliente.

Una vez que introduzca la contraseña correcta, se ejecutará el comando y se mostrará el contenido de */usr/share/doc*, cerrando a continuación la conexión.

Transferencia de archivos mediante SSH.

La transferencia de archivos entre dos ordenadores mediante SSH se puede realizar mediante dos programas, *scp* y *sftp*. El programa *scp* es similar a *rcp*[14], mientras que *sftp* es similar a *ftp* y pueden considerarse las versiones seguras de esos dos programas inseguros de transferencia de ficheros por la red.

El programa */usr/bin/scp* se utiliza para transferir ficheros entre ordenadores de forma segura y encriptada. La sintaxis general, para transferir un fichero local a un sistema remoto es la siguiente:

```
scp fichero_local usuario@ordenador_destino:/fichero_remoto
```

Donde *fichero_local* especifica el origen, y *usuario@ordenador_destino:/fichero_remoto* especifica el destino. Por ejemplo, para transferir el fichero local *enviar.txt* a nuestra cuenta en *glup.uv.es*, el comando es:

```
scp enviar.txt usuario@glup.uv.es:/home/usuario
```

Esto trasferirá el fichero local *enviar.txt* a */home/usuario/enviar.txt* en el ordenador *aglup.uv.es*.

De forma similar, la sintaxis general para transferir un fichero remoto a un sistema local es:

```
scp usuario@ordenador_origen:/fichero_remoto fichero_local
```

Donde *fichero_remoto* especifica el origen, y *fichero_local* especifica el destino.

Se pueden transferir varios ficheros diferentes del origen al destino. Por ejemplo, para transferir el contenido del directorio */descargar* a un directorio existente llamado */descargados* en el ordenador remoto *robotica.uv.es*, el comando es:

```
scp /descargar/* usuario@glup.uv.es:/descargados/
```

Por su parte, el programa */usr/bin/sftp* permite abrir una conexión segura e interactiva de FTP, pudiendo ejecutar cualquier comando disponible en FTP a través de esta conexión encriptada y por tanto segura.

```
sftp usuario@ordenador
```

Siendo solicitada la contraseña del usuario y, una vez aceptada dicha contraseña, ejecutar cualquier comando de FTP valido.

[14] Se puede considerar scp como una versión simplificada de FTP que vimos con anterioridad.

Reenvío de conexiones mediante SSH.

Además de proporcionar una interfaz de línea de comandos segura, SSH permite redirigir conexiones X11[15] y puertos inseguros por un canal SSH previamente abierto.

Reenvío de X11.

En primer lugar, para poder realizar un reenvío de X11, el cliente de SSH debe estar ejecutando un servidor de X, pues en caso contrario las ordenes enviadas por la aplicación X que se ejecuta en el ordenador remoto no podrán ejecutarse al no encontrarse el servidor X que las interprete. Además, tanto el cliente como el servidor deben tener habilitada en su configuración la posibilidad de permitir el reenvío de X11.

Cuando un programa X se ejecuta desde un intérprete de comandos seguro, el cliente y el servidor SSH crean un nuevo canal seguro dentro de la conexión SSH actual, y los datos del programa X se envían a través de ese canal a la máquina del cliente como si se estuviese conectado al servidor X por medio de una terminal local.

El reenvío de X11 es muy útil pues, por ejemplo, se puede usar el reenvío por X11 para crear una sesión segura e interactiva con el programa xclock[16] en el servidor para visualizar su hora. Para hacer esto, simplemente hay que conectarse al servidor mediante SSH y ejecutar:

```
xclock &
```

Y se ejecutará el programa X en el ordenador cliente.

Para poder ejecutar de forma automática el reenvío de una conexión X11 sin necesidad de modificar ficheros de autorización del servidor X, es necesario ejecutar el cliente de SSH con la opción -Y, de forma que la conexión se realiza con el comando:

```
ssh -Y {[-l usuario] ,[usuario@]}<servidor>
```

Por ejemplo:

```
ssh -Y root@robotica.uv.es
```

Reenvío de un puerto.

SSH permite además asegurar los protocolos TCP/IP a través del reenvío de un puerto. Mediante esta técnica, el servidor SSH se convierte en un conducto cifrado para el cliente SSH.

El reenvío de un puerto funciona mediante la redirección de un puerto local en el cliente en un puerto remoto del servidor. SSH permite redirigir cualquier puerto desde el servidor a cualquier puerto en el cliente, sin que los números de puerto deban coincidir para que funcione.

[15] La redirección de una conexión X11 requiere un ancho de banda de transmisión elevado.
[16] Mostrar un reloj gráfico del servidor en la ventana del cliente es un ejemplo sencillo, simple y de poco uso, pero téngase en cuenta que cualquier aplicación X puede ser mostrada en el cliente.

Para crear un canal de reenvío de puerto TCP/IP que escucha conexiones del ordenador local, el comando es[17]:

```
ssh -f -N -L puerto_local:ordenador_remoto:puerto_remoto
usuario@ordenador_servidor
```

Un ejemplo es el reenvío del correo del servidor *glup.uv.es* que utiliza POP3 a través de una conexión cifrada utilizando como servidor de SSH el propio ordenador. El comando a ejecutar es:

```
ssh -f -N -L 1100:glup.uv.es:110 glup.uv.es
```

Una vez que el canal de reenvío de puerto entre las dos máquinas se establece, cualquier petición enviada al puerto 1100 en el ordenador local será dirigida al servidor de POP3 de *glup.uv.es* mediante una conexión segura[18].

Si *glup.uv.es* no está ejecutando un demonio del servidor SSH, pero tenemos la posibilidad de acceder mediante SSH a otro ordenador, por ejemplo, *amparo.uv.es*, es posible usar SSH para asegurar la parte de la conexión POP3 mediante el comando:

```
ssh -f -N -L 1100:glup.uv.es:110 amparo.uv.es
```

En este ejemplo, se está reenviando la petición POP3 desde el puerto 1100 en el ordenador a través de una conexión SSH en el puerto 22 a *amparo.uv.es*, el cual conecta a su vez con el puerto 110 de *glup.uv.es* para permitir que se compruebe el correo. Mediante el uso de esta técnica, sólo es segura la conexión entre el ordenador local y *amparo.uv.es*.[19]

El reenvío del puerto se puede usar para obtener información segura a través de los cortafuegos de red, de forma que si un cortafuegos está configurado para permitir el tráfico SSH a través del puerto estándar (puerto 22), pero bloquea el acceso a través de otros puertos, es posible todavía una conexión entre dos ordenadores usando los puertos bloqueados al redirigir la comunicación sobre una conexión SSH establecida[20].

Ejercicios.

1- Configurar un servidor de SSH de forma que permita tan solo el uso de la versión 2 del protocolo y el acceso de los usuarios mediante la introducción de su usuario y contraseña, impidiendo el uso de cualquier otro método alternativo.

[17] La configuración del reenvío de un puerto para que escuche puertos privilegiados (puerto inferiores al 1024) requiere acceso de administrador (root).

[18] Por tanto, bastaría con indicarle a nuestro cliente de POP3 que utilizase el puerto 1110 local como puerto del servidor de POP3.

[19] Este uso tiene sentido si *amparo.uv.es* se encuentra en la misma subred que *glup.uv.es*, pues permite asegurar la transmisión a través de toda la red Internet hasta llegar a esa subred.

[20] Mediante el uso del reenvío de puerto para reenviar conexiones de este modo permiten al usuario en el sistema cliente conectarse al servicio al que se están reenviando las conexiones. Si el sistema del cliente se convierte en un sistema comprometido, el agresor tendrá acceso a los servicios reenviados. Si se desea deshabilitar el reenvío del puerto, basta especificar el valor No en la línea *AllowTcpForwarding* en el fichero de configuración del servidor de SSH.

2- Configurar un servidor de SSH que permita el uso de las versiones 2 y 1 del protocolo en este orden. El servidor deberá permitir el acceso a los usuarios del grupo “administración”, denegando el acceso al resto de usuarios.

3- Un ordenador posee dos interfaces de red, uno público de IP 147.156.222.65 y otro privado de IP 192.168.0.1. Deseamos configurar el servidor de SSH de forma que escuche tan solo el interface de red público, debiendo permitir tan solo el uso de la versión 2 del protocolo y permitir el reenvío de conexiones X11.

4- Un cliente de SSH desea conectarse a un servidor de forma que pueda reenviar conexiones X11. Indicar las configuraciones necesarias tanto del cliente como del servidor.

5- Deseamos configurar un cliente y un servidor de SSH de forma que permitan el reenvío de puertos. Indicar las configuraciones necesarias en los ficheros.

6- Deseamos que nuestro cliente de SSH pueda acceder a un servidor de SSH utilizando un par de claves pública/privada. Indicar tanto los ficheros de configuración del cliente y el servidor como los comandos necesarios para crear y manejar las claves.

7- Un servidor, de nombre *servidor.uv.es*, dispone de un servicio no seguro ejecutándose en su puerto 13 TCP. Deseamos crear un canal seguro entre nuestro ordenador y el servidor para acceder a dicho servicio. Indicar el comando necesario si el servidor ejecuta el servicio de SSH y ambos ordenadores se encuentran configurados para permitir el reenvío de puertos.