Apéndice C. Secure Shell

"Los ejemplos son diez veces más útiles que los preceptos."

Charles James Fox

Secure Shell (SSH), desarrollado por Tatu Ylonen en la Universidad Tecnológica de Helsinki en Finlandia y OpenSSH, que nace del proyecto de un sistema operativo orientado con la filosofía de la seguridad en mente como lo es OpenBSD.Secure Shell y OpenSSH permiten realizar la comunicación y transferencia de información de forma cifrada proporcionando fuerte autenticación sobre el medio inseguro. [VIL00]

Secure Shell admite varios algoritmos de cifrado entre los cuales se incluyen:

- Blowfish
- 3DES
- IDEA
- RSA

Prevenciones

Debido a la promiscuidad de la interfaz ethernet, éste nos presenta una problemática sobre los servicios de red usados en la actualidad, tales como: telnet, ftp, http, rsh, rlogin, rexec

Ello representa un problema importante, ya que, incluso en un entorno de red cerrado, debe existir como mínimo un medio seguro para poder desplazar archivos, hacer copia de archivos, establecer permisos, ejecutar programas, scrips, etc, a través de medios seguros. [VIL00]

Por ello para evitar que determinadas personas capturen el trafico diario de la red, es conveniente instalar el Secure Shell(SSH)

Entre los ataques más comunes que nos previenen Secure Shell están:

- Sniffering(Captura de trafico)
- IP Spoofing
- MACpoofing
- DNS Spoofing
- Telnet Hickjacking
- ARP Spoofing
- IP Routing Spoofing
- ICMP Spoofing

Protocolos de Secure Shell

Existen actualmente dos protocolos desarrollados sobre ssh [VIL00]:

SSH1 : La ultima versión de ssh cliente/servidor para Unix que soporta este protocolo es la 1.2.31, esta puede ser utilizada libremente para propósitos no comerciales y es ampliamente usada en ambientes académicos.

SSH2 : Provee licencias más estrictas que SSH1 ya que es de carácter comercial. La ultima versión de ssh cliente/servidor para Unix con este protocolo es la 2.4.0 y puede ser utilizada libremente respetando la licencia expresa.

Actualmente existe un proyecto llamado OpenSSH, el cual fue desarrollado inicialmente dentro del proyecto OpenBSD. [VIL00]

OpenSSH es una versión libre de los protocolos SSH/SecSH bajo licencia BSD y es totalmente compatible con los protocolos SSH1 y SSH2. La ultima versión de OpenSSH cliente/servidor para Unix es la 2.3.0P1 (Liberada el 6 de Noviembre del 2000)

Instalación de Secure Shell cliente/servidor para Unix

Desafortunadamente los protocolos de Secure Shell (SSH1 y SSH2) no son compatibles uno con otro, por lo tanto, sí deseamos que exista compatibilidad debemos de instalar primero Secure Shell protocolo SSH1 y posteriormente Secure Shell protocolo SSH2.

Otra opción para mantener la compatibilidad con los dos protocolos sin problema alguno es instalar OpenSSH.

Secure Shell con protocolo SSH1 y SSH2 pueden instalarse exactamente igual tal como se muestra a continuación.

Después de obtener el programa de Secure Shell (SSH1 o SSH2) se procede a desempacarlo:

$ gunzip ssh−x.x.x.tar.gz

$ tar −xvf ssh−x.x.x.tar

En este punto obtendremos un directorio ssh−x.x.x sobre la ruta donde desempacamos.

A continuación se citan los pasos necesarios configurar y compilar Secure Shell. Estos pasos pueden realizarse sin ser root.

a. **Configuración del entorno de compilación**

A diferencia de otras herramientas ssh no requiere de editar el archivo Makefile, la configuración se realiza a través de los parámetros provistos al script llamado configure.

Dentro del directorio ssh−x.x.x se encuentra el script llamado configure, el cual tiene los siguientes argumentos validos:

- −−prefix=PREFIX
  Donde se instalarán los binarios por default /usr/local.
- −−exec_prefix=PREFIX
  Donde se instalarán los ejecutables por default es el mismo que la variable −−prefix.

- −−with−rsh=PATH
  Permitirá comandos rsh utilizando la estructura de ssh.
- −−without−idea
  No incluir IDEA
- −−with−tis=PATH
  Soporte a mecanismo de autenticación Tis authsrv.
- −−with−etcdir=PATH
  Ruta sobre la que obtendrá información sobre el sistema por default /etc.
- −−with−libwrap=[PATH]
  Usa libwrap (tcp_wrappers) y inetd.
- −−with−socks4[=PATH]
  Incluye soporte para SOCKS (Cruce de firewall).
- −−with−socks5[=PATH]
  Incluye soporte para SOCKS5.
- −−enable−warnigs
  Habilita la bandera −Wall al compilador gcc.

Sí la intención es lograr que tcp−wrapper lleve un control sobre los accesos realizados con ssh, se necesitará la bandera −−with−libwrap, además, sí se pretende realizar una autenticación de la máquina para ejecutar comandos rsh, se requiere entonces la bandera −−with−rsh.

Entonces se ejecuta el script tomando en cuenta que tenemos instalado TCP−Wrappers y con soporte para el uso de comandos rsh:

$ ./configure −−with−libwrap=/path/libwrap.a −−wit−rsh=/path/rsh.

Esto genera los cambios necesarios en los archivos de código fuente para su correcta compilación. Para obtener la ruta del comando rsh utilizar el comando whereis.

b. **Compilar los binarios de Secure Shell (se requiere el compilados GCC)** . Para esto basta con ejecutar:

$ make

Nota importante: Para poder ejecutar la instalación es necesario estar dentro de la cuenta de root.

c. **Instalación de Secure Shell en el sistema y obtener las llaves del host.**

$ make install

Los archivos de configuración de Secure Shell (ssh_host_key y sshd_config) quedan localizados en el directorio "/etc", los programas clientes (ssh y scp) quedan en "/usr/local/bin". Finalmente el programa servidor o daemon de Secure Shell (sshd) queda localizado en "/usr/local/sbin".

d. Ya que se realizó la instalación en el equipo se procede a configurar el sistema para poder ejecutar el daemon del servidor de ssh y permitir accesos por el puerto por default de Secure Shell (port 22).

Editar el archivo **/etc/inetd.conf** e incluir la siguiente línea:

(Sí no tiene habilitado TCP−Wrapper)

ssh tcp root nowait /usr/local/sbin/sshd /usr/local/sbin/sshd −i

(Sí se esta usando TCP−Wrapper)

ssh tcp root nowait /usr/local/etc/tcpd /usr/local/sbin/sshd −i

Editar el archivo /etc/services y habilitar el puerto para Secure Shell usando la siguiente línea:

ssh 22/tcp Secure Shell

ssh 22/udp Secure Shell

Por último reiniciar el daemon de inetd.

Obtener el número de proceso del daemon inetd.

$ ps −fea | grep inetd

Enviar la señal HUP al proceso del inetd.

$ kill −HUP procesid

En este punto el sistema debe responder a las peticiones de conexión por Secure Shell.

Sesión entre máquinas en UNIX

Ssh permite mantener sesiones interactivas con una máquina remota de la forma como lo hace el comando telnet.

ssh [−a] [−c idea|blowfish|des|3des|arcfour|none] [−e esc] [−i identity_file] [−l login_name] [−n]

[−k] [−V] [−o] [−p port] [−q] [−P] [−t] [−v] [−x] [−X] [−C] [−g] [−L port:host:hostport] [−R

port:host:hostport] hostname [command]

A continuación se describen brevemente las principales opciones del comando ssh.

−a

Deshabilita el agente de autenticación.

−c

idea|des|3des|...

Selecciona el algoritmo de cifrado utilizado para cifrar la sesión.

−e

Habilita el carácter de escape para una determinada sesión (default:~).

−i identity−file

Selecciona el archivo donde leerán la llave de autenticación. Por default

utiliza .ssh/identity dentro del home del usuario.

−l login−name

Especifica la cuenta remota por medio de la cual se desea tener acceso.

−p port

Puerto remoto de conexión.

−q

Causa que todas las advertencias y mensajes de diagnostico sean suprimidos, solo los mensajes de errores fatales son desplegados.

−v

Causa que ssh despliegue mensajes de depuración acerca del proceso de

conexión.

−x

Deshabilita el reenvio (forwarding) de X11.

−C

Compresión de todos los datos transmitidos durante la conexión

Transferencia de archivos

Secure Shell proporciona una herramienta que permite realizar transferencia de archivos entre distintos hosts, haciendo uso de las características de ssh, este comando es scp, el cual cuenta con las siguientes opciones:

scp [−aAqrvBCL1] [−S path−to−ssh] [−o ssh−options] [−P port] [−c cipher] [−i identity]

[[user@]host1:]filename1,... [[user@]host2:]filename2

−a

Habilita la estadística de transferencia de cada archivo que se transfiere.

−A

Deshabilita el mensaje de estadística de transferencia

−c

cipher Selecciona el algoritmo de cifrado a utilizar en la transferencia de datos.

−i

identity Selecciona el archivo donde se tendrá acceso a la llave RSA.

−L

Uso de puerto no privilegiado. Esta opción posee algunas restricciones como son la imposibilidad de usar rhosts o autenticación rsarhosts

−1

Forzar a scp a usar el comando scp1 por parte de host remoto, Esto puede ser necesario en el caso que el host remoto utilice scp2.

−o ssh−options

Opciones que se pasarán al ssh.

−p

Preserva las distintas fechas y horas de acceso, modificación y atributos del archivo original

−q

Deshabilita el despliegue de estadísticas de transferencia

−r

Copia recursiva de directorios completos

−v

Causa que scp y ssh desplieguen mensajes de estado relacionados con el proceso de conexión, etc. Adecuado para depurar errores existentes en las conexiones realizadas.

Uso de .shosts

Los archivos .shosts al igual que los archivos .rhosts permite realizar conexiones entre hosts sobre mecanismo de confianza, en el caso de .shosts este mecanismo de confianza se habilita sobre las características que proporciona Secure Shell, por esto, en aquellas máquinas en las cuales el uso de .rhosts es indispensable sugerimos la sustitución de todos los .rhosts por .shosts.

Para esto se requiere de realizar la siguiente serie de pasos:

1. En la máquina cliente realizar una conexión hacia la máquina servidor que se va a hacer uso de .shosts.
   $ ssh −l cuenta maquina.cliente.de.shosts
   Esto es para que la máquina a la que se entrará utilizando mecanismo .shosts obtenga la llave pública de la máquina cliente.
2. En la máquina servidor debemos de crear sobre el home del usuario el archivo .shosts conteniendo el nombre de la máquina o la IP cliente.
   $ more .shots
   maquina.cliente.de.shosts
3. Desde la máquina cliente realizamos el enlace.
   $ ssh −l cuenta maquina.con.shosts
   NOTA: Este tipo de conexión no solicitará el password.
   Este tipo de conexión presenta un riesgo, sin embargo, introduce mecanismos propios de ssh, un método con mayores niveles de seguridad se puede obtener utilizando llave RSA.

TIPS

Sí no se lleva a cabo la conexión debemos de verificar los permisos del archivo .shosts

$ chmod 644 .shosts

índice figuras introducción 1 2 3 4 5 A B C D referencias