

P3

☰ Comentario	2 mal, 5 y 6 sin hacer
☰ Nota	
☰ Tags	A medias

1. Tomando como base de trabajo el SSH pruebe sus diversas utilidades:

▼ a. Abra un shell remoto sobre SSH y analice el proceso que se realiza.

Configure su fichero `ssh_known_hosts` para dar soporte a la clave pública del servidor.

Abrir sesión con `ssh -v lsi@10.11.48.48`

EXPLICACIÓN DE LÍNEAS

1. Establecer conexión:

El cliente lee los ficheros de configuración y establece conexión con el servidor. Se comprueba la compatibilidad de los protocolos SSH se mira el fichero `sshconfig` y se abre conexión al puerto 22.

2. Seleccionar versión a utilizar y autenticación:

Ambos extremos conversan para saber que algoritmo de cifrado conocen ambos y se ponen de acuerdo para utilizar uno, comprobando el cliente si tiene o no una serie de ficheros. También se selecciona la versión de ssh a utilizar(v1 o v2), en caso de v1 mejor no conectarse.

3. Negociación de parámetros de la conexión:

El cliente recibe la clave pública del servidor, comprueba con esta la autenticidad de la máquina (en los `ssh_known_hosts` tanto de `etc/ssh` como del

`$HOME/.ssh/known_hosts`) y cifra también con ella la clave de sesión. Esta clave de sesión cifrada será enviada al servidor más

tarde(paso 5) ya que es el único que puede descifrarla al tener la clave privada. Se negocian los algoritmos que vamos a utilizar en la conexión, se sacan de los ficheros de configuración, cliente y servidor miran sus lineas y se ponen de

acuerdo. (algoritmo de autentificación, algoritmo de intercambio de clave, algoritmo de cifrado, algoritmo de MAC para integridad y algoritmo de compresión).

4. Autenticación del servidor:

Se inicia el intercambio de la clave de sesión, preguntando si te fías de la clave en caso de ser la primera vez. (Utilizando ecdsa, algoritmo de curva elíptica)

5. Establecimiento de la clave de sesión.

(utilizando la pública del servidor). Se genera una clave de sesión y se cifra con la pública del servidor y se envía. Dicha clave se utiliza en toda la sesión para cifrar las conexiones y se autentifica el servidor, para tener la certeza de que no le están suplantando)

6. Autenticación del usuario:

Primero este lo intenta mediante la clave pública, es decir, el servidor mira en el fichero `$/home/ssh/authorized_keys` (2) si está registrada la pública del cliente (puede probar con varias: dsa,rsa,ecdsa,ed25519...). En caso de no estar la clave registrada solicita contraseña (esto se puede cambiar en `sshd_config`, haciendo que solo se pueda entrar por clave pública para evitar ataques password guessing).

7. Termina debug:

Una vez llegados a este punto la máquina ya está lista para usar.

CONFIGURACIÓN FICHERO `SSH_KNOWN_HOSTS`

Nuestra clave pública está en: `cat /etc/ssh/ssh_host_rsa_key.pub`

Se la enviamos a nuestro compi con el siguiente comando:

```
ssh-keyscan 10.11.48.45 >> /etc/ssh/ssh_known_hosts
```

→ Comprobar conexión:

1. Borramos fichero `/home/lsi/.ssh/known_hosts` con `rm`
2. Lanzamos sesión con `ssh -X lsi@10.11.48.45`

De esta manera ahora no va a asegurarse si realmente nuestro servidor es el que le indicamos, pues al tener su clave pública

| no desconfía

▼ **b. Haga una copia remota de un fichero utilizando un algoritmo de cifrado determinado. Analice el proceso que se realiza.**

Creamos un fichero:

```
cd /home/lsi  
nano P3_1b_sab
```

COMPI 1:

-Usaremos el algoritmo **aes256-ctr**.

-Enviamos el fichero a mi compi:

```
scp -v -c aes256-c P3_1b_sab lsi@10.11.48.45:/home/lsi/
```

| ¿Cuántos algoritmos de cifrado hay en nuestra máquina?

```
ssh -Q cipher
```

COMPI 2:

-Comprobamos ver como el archivo que nos envió está cifrado:

```
cd /home/lsi  
cat P3_1b
```

▼ **c) Configure su cliente y servidor para permitir conexiones basadas en un esquema de autenticación de usuario de clave pública.**

CLIENTE

Generamos la clave publica y privada con: `ssh-keygen -t rsa -b 4096`

Esto generará en `/home/lsi/.ssh` los ficheros **id_rsa** e **id_rsa.pub**

```
ssh-copy-id lsi@10.11.48.45
```

→ Tener activo apache2. (systemctl enable/start apache2)

▼ d) Mediante túneles SSH securice algún servicio no seguro.

CREAR TÚNEL:

```
ssh -L 8080:10.11.48.48:80 lsi@10.11.48.48
```

→ Este comando permite crear un túnel seguro desde el puerto local hasta el puerto del otro host → El otro host esta escuchando al puerto local, así, cuando el puerto local reciba una conexión esta se retransmite a través del canal seguro y la conexión se hace al host:port del host especificado.

→ Aquí establecemos un túnel desde el puerto 8080 del cliente al puerto 80 del servidor.

→ Para que el cliente se conecte al servidor simplemente tendrá que conectarse a su puerto 8080.

→ De esta forma estamos accediendo al servidor http de forma segura mediante un túnel ssh.

COMPROBACIÓN.

```
w3m http://localhost:8080/
```

▼ e) “Exporte” un directorio y “móntelo” de forma remota sobre un túnel SSH.

1. Creamos nuevo directorio: `mkdir /home/lsi/dir`
2. Hacemos `sshfs lsi@10.11.48.45 :/home/lsi/ /home/lsi/dir`
3. Ahora si creamos algo en **/home/lsi/dir** le aparecerá a andrea en su **/home/lsi**
4. Ponemos `fusermount -u /home/lsi/dir` si queremos desvincular los directorios.

▼ f) PARA PLANTEAR DE FORMA TEÓRICA.: Securice su servidor considerando que únicamente dará servicio ssh para sesiones de usuario desde determinadas IPs.

Para securizar mi servidor tendríamos que deshabilitar todos los servicios menos el ssh, tanto por tcp como por udp.

→ Editaríamos el fichero `/etc/sysctl.conf` y añadir: `net.ipv4.icm_echo_ignore_all=1`

→ En el servidor sshd en `/etc/ssh/sshd_config` poner: **PermitRootLogin no.**

→ Al final del `/etc/ssh/sshd_config`

ponemos: **AllowUsers lsi-ssh y Banner /etc/issue.net: Para limitar el acceso a un solo usuario.**

→ Pondremos un puerto de escucha alto, para securizarlo.

→ Configurar iptables y los wrappers. Permitir conexiones entrantes al puerto en el que

está nuestro ssh a todos o a las ips permitidas. O por ejemplo permitir únicamente direcciones de España. Denegar todo lo demás.

2. Tomando como base de trabajo el servidor Apache2:

▼ a) Configure una Autoridad Certificadora en su equipo.

→ Primero instalamos **openssl**.

```
apt-get install openssl
```

→ Ejecutamos el script CA.pl

```
cd /usr/lib/ssl/misc  
./CA.pl-newca
```

-newca: Crea el certificado de la Autoridad Certificadora que contiene la clave pública, y también crea la clave privada. Va a pedir frase de paso, la cual hay que meter ya que con ella se cifra la clave privada usando un algoritmo simétrico.

→ Mi frase de paso es **lsi48**

- **newreq.pem**: tiene la petición de certificado para firmar.

Ahora le damos permisos de lectura a newkey.pem y le pasamos los archivos al servidor:

```
chmod +r newkey.pem
scp newkey.pem lsi@10.11.48.48:/home/lsi/newkey.pem
scp newcert.pem lsi@10.11.48.48:/home/lsi/newcert.pem
scp demoCA/cacert.pem lsi@10.11.48.48:/home/lsi/cacert.pem
```

Ahora movemos los ficheros a los correspondientes directorios:

```
mv newkey.pem /etc/ssl/private/lsicompany.com.key
mv newcert.pem /etc/ssl/certs/lsicompany.com.crt
cd demoCA
mv cacert.pem /etc/ssl/certs/lsiauthority.crt
```

Entonces al servidor le enviamos el certificado generado y firmado, la clave privada del certificado y el certificado con la clave pública de la Autoridad Certificadora

Ahora con el certificado de la CA realizamos su hash y le creamos un link:

```
cd /etc/ssl/certsc
openssl x509 -in lsiauthority.pem -noout -hash
(8bbaf8f2)
ln -s lsiauthority.pem
```

▼ **c) Configure su Apache para que únicamente proporcione acceso a un determinado directorio del árbol web bajo la condición del uso de SSL. Considere que si su la clave privada está cifrada en el proceso de arranque su máquina le solicitará la correspondiente frase de paso, pudiendo dejarla inalcanzable para su sesión ssh de trabajo.**

▼ 3. Tomando como base de trabajo el openVPN deberá configurar una VPN entre dos equipos virtuales del laboratorio que garanticen la confidencialidad entre sus comunicaciones.

INSTALACIÓN:

```
apt-get install openvpn
```

CLIENTE:

```
lsmod | grep tun  
modprobe tun  
lsmod | grep tun
```

→ Así creamos un tun nuevo.

→ Ahora creamos unos interfaces de red tun con:

```
echo tun >> /etc/modules  
cd /etc/openvpn  
nano tunel.conf
```

→ Generamos la clave y se la pasamos a nuestro compi:

```
openvpn --genkey --secret clave.key  
scp clave.key lsi@10.11.48.45:~/
```

→ Configuramos el archivo **tunel.conf**:

```
local 10.11.48.48  
remote 10.11.48.45  
dev tun1  
port 5555
```



```
comp-lzo
user nobody
ping 15
ifconfig 172.160.0.2 172.160.0.1
secret /etc/openvpn/clave_andrea.key
#cliente
```

COMPROBAR CONEXION:

```
systemctl enable openvpn
systemctl start openvpn
openvpn --config /etc/openvpn/tunnel.conf
```

→ Sabemos que openvpn funciona correctamente si hacemos `ifconfig tun1` y existe ese interfaz de red.

→ Abrir otro terminal nuevo, loggarse en la máquina de lsi y hacerse ping (o ssh) a la ip de la vpn del compañero.

Parar el servicio: `systemctl openvpn stop`

Volver a arrancar: `systemctl openvpn start`

Si al ejecutar el openvpn aparece un error que ponga "IOCTL busy resource" o algo parecido, significa que la vpn ya está levantada (vamos, que ya está funcionando)

▼ 4. EN LA PRÁCTICA 1 se configuró una infraestructura con servidores y clientes NTP.

Modifique la configuración para autenticar los equipos involucrados.

SERVIDOR:

Vamos a /etc

Generamos la clave:

```
ntp-keygen -M (Generate a new symmetric keys file containing 10 MD5 keys, and if
OpenSSL is available, 10 SHA keys)
```

Eliminamos enlace:

```
rm ntp.keys
```

Copiamos archivo al que apuntaba a ntp.key

```
cp ntpkey_MD5key_debian..... ntp.keys
```

Eliminamos el archivo primeramente creado

```
rm ntpkey_MD5key....
```

Cambiamos permisos ntp.keys

```
chmod -R 640 ntp.keys
```

Poner como propietario del archivo a ntp

```
chown ntp ntp.keys
```

Poner como grupo root

```
chgrp root ntp.keys
```

Modificar el ntp.conf poniendo:

```
keys /etc/ntp.keys (indica donde tenemos la lista de claves)
```

```
trustedkey 1 (indica las claves que van a poder usarse, para que si nos roban por ejemplo la 7 pero no la tenemos aquí metida no funcione)
```

Pasamos por scp el ntp.keys al cliente:

```
scp ntp.keys lsi@10.10.102.54:/home/lsi/ntp.keys
```

CLIENTE:

```
mv ntp.keys /etc
```

Modificar el **ntp.conf** (ver arichov en maquina)

COMPROBAR:

```
ntpq -pn  
enseñar el ntp.keys para ver que hay claves
```

▼ **5. EN LA PRÁCTICA 1 se instalaron servidores y clientes de log. Configure un esquema que permita cifrar las comunicaciones.**

→ Cambiar fichero **/etc/rsyslog.conf** y poner siempre que haya la ip de mi compi en formato 172.160.0.1 (formato vpn)

→ Hacer `systemctl restart rsyslog`

→ Prueba:

Cliente envía mensaje:

```
logger "Mensaje enviado"
```

Servidor recibe:

```
cat /var/log/messages
```

▼ 6.