LSI P3

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.

**********CONFIGURACION**********

AMBOS:

ssh-keygen	Generamos nuestras claves (fingerprint) (se guardan en la ruta defecto: /root/.ssh/id_rsa)
ssh-keygen -l	Ver nuestro fingerprint

AMBOS:

Una vez los dos han realizado los pasos anteriores:

nano /etc/ssh/ssh_known_hosts	Crear fichero ssh_known_hosts
ssh-keyscan 10.11.48.COMPA >> /etc/ssh/ssh_known_hosts	Guardar al compañero

****

Tras realizar este paso, si nos intentamos conectar vía ssh a nuestro compañero desde nuestra máquina, ya no se nos pedirá si aceptar el fingerprinting ya que lo tenemos guardado como un host conocido.

Para ver el proceso de ssh: añadir flag -v

ssh -v lsi@10.11.48.143

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

LOCAL -> SERVIDOR (Enviar)

scp -c aes128-ctr RUTA/ORIGEN/ARCHIVO lsi@10.11.48.COMPA:/RUTA/DESTINO

SERVIDOR -> LOCAL (Recibir)

scp -c aes128-ctr lsi@10.11.48.COMPA:/RUTA/ORIGEN/ARCHIVO /RUTA/DESTINO

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

ESTE APARTADO DEBE SER REALIZADO EN MODO **NO ROOT**

**********CONFIGURACION**********

AMBOS:

En /etc/ssh/sshd_config descomentar la línea: PubKeyAutentication = yes

CLIENTE:

Generamos una clave SIN CONTRASEÑA para la conexión: (clave pública)

ssh-keygen -t rsa (Dejar todo en blanco) (Se guarda en /home/lsi/.ssh)

Enviamos la clave al servidor:

scp id_rsa.pub lsi@10.11.48.COMPA:./.ssh/id_rsa.pub

SERVIDOR:

cd /home/lsi/.ssh

Copiamos la clave recibida a nuestra lista de claves autorizadas y la borramos (ya no se necesita el archivo suelto):

cat id_rsa.pub >> authorized_keys

rm id_rsa.pub

****

El cliente podrá entrar en la máquina servidor sin meter contraseña (modo sin root):
ssh lsi@10.11.48.COMPA

d. Mediante túneles SSH securice algún servicio no seguro.

El servidor debe tener apache2 activo (por defecto ya lo está).

CLIENTE:

- ssh -L 12345:10.11.48.SERVIDOR:80 lsi@10.11.48.SERVIDOR

Una vez dentro, se intenta hacer petición al servidor apache:

- **w3m** http://localhost

Se ha establecido un túnel seguro desde el puerto 12345 del cliente hasta el 80 del servidor, una vez allí se hacen las peticiones a su servidor apache.

e. "Exporte" un directorio y "móntelo" de forma remota sobre un túnel SSH.

Creamos dos archivos: directorioServidor(joa) y directorioCliente(alvaro) (con algún archivo dentro)

#apt install sshfs

sshfs lsi@10.11.48.143:/home/lsi/Escritorio/comp/directorioServidor /home/lsi/Escritorio/comp/directorioCliente

El directorio de directorioServidor, se monta en el directorioCliente, se monta en el directorioCliente, actualizando el archivo en las dos máquinas.

Para quitar el montado, se utiliza umount “directorio”:

umount /home/lsi/Escritorio/montada/directorioCliente/

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

2. Tomando como base de trabajo el servidor Apachcoe2

Contraseña: lsicert

lsi@lsi.es

lsi.joaquin.es

a. Configure una Autoridad Certificadora en su equipo.

apt install openssl

a2enmod ssl

a2ensite default-ssl

systemctl restart apache2

cd /usr/lib/ssl/misc/

./CA.pl -newca Creamos nuestra CA

b. Cree su propio certificado para ser firmado por la Autoridad Certificadora. Bueno, y fírmelo.

./CA.pl -newreq-nodes Creamos nuestro certificado

Los 3 nombres = debian

./CA.pl -sign Lo firmamos con nuestra CA

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.

cp -p /usr/lib/ssl/misc/demoCA/cacert.pem /etc/ssl/certs/certificadop3lsi.pem

Comprobar si el certificado es válido:

openssl x509 -noout -text -in /etc/ssl/certs/certificadop3lsi.pem

cd /etc/ssl/certs/

Obtenemos el hash del certificado:

openssl x509 -in certificadop3lsi.pem -noout -hash

ln -s certificadop3lsi.pem HASHOBTENIDO.0

cd /usr/lib/ssl/misc/

mv newkey.pem /etc/ssl/private/lsi.NOMBRE.es.key

mv newcert.pem /etc/ssl/certs/lsi.NOMBRE.es.crt

cd /etc/ssl/certs/

openssl verify lsi.NOMBRE.es.crt

Indicamos a Apache que use los certificados que acabamos de crear:

nano /etc/apache2/sites-available/default-ssl.conf

#Editamos:

SSLCertificateFile /etc/ssl/certs/debian.crt

SSLCertificateKeyFile /etc/ssl/private/debian.key

systemctl restart apache2

Comprobamos que el certificado es válido al conectarse al dominio:

openssl s_client -connect debian:443

wget --ca-certificate=/etc/ssl/certs/**certificadop3lsi.pem** https://lsi.NOMBRE.es

curl https://lsi.alvaro.es

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.

**********CONFIGURACION**********

SERVIDOR: (10.8.0.1) (ALVARO)

Nos movemos al directorio del openvpn /etc/openvpn, con el comando:

cd /etc/openvpn

Generamos la clave secreta de openvpn, con el comando:

openvpn --genkey --secret lsi_openvpn_joaquin.key

Mandamos la clave al cliente con el comando

scp /etc/openvpn/lsi_openvpn_joaquin.key lsi@10.11.48.143:/home/lsi/

Creamos el archivo tunel.conf, con el comando: nano tunel.conf

Añadimos esta configuracion al archivo tunel.conf:

local 10.11.48.144

remote 10.11.48.143

dev tun1

port 5555

comp-lzo (tipo de compresión de los archivos que se envian por el tunel)

user nobody

ping 15 (ping UDP cada 15 seg para dropear los paquetes que no sean los que queremos)

Firewall Setup:

If firewalls exist between the two machines, they should be set to forward UDP port 1194 in both directions. If you do not have control over the firewalls between the two machines, you may still be able to use OpenVPN by adding **--ping 15** to each of the **openvpn** commands used below in the examples (this will cause each peer to send out a UDP ping to its remote peer once every 15 seconds which will cause many stateful firewalls to forward packets in both directions without an explicit firewall rule).

If you are using a Linux iptables-based firewall, you may need to enter the following command to allow incoming packets on the TUN device:

ifconfig 10.8.0.1 10.8.0.2

secret /etc/openvpn/lsi_openvpn_joaquin.key

CLIENTE: (10.8.0.2) (JOA)

Movemos la clave lsi_openvpn.key que envio el servidor al directorio /etc/openvpn/ con el comando: mv /home/lsi/lsi_openvpn_joaquin.key /etc/openvpn/

Creamos el archivo tunel.conf, con el comando: nano tunel.conf

Añadimos esta configuración al archivo tunel.conf:

local 10.11.48.143

remote 10.11.48.144

dev tun1

port 5555

comp-lzo

user nobody

ping 15

ifconfig 10.8.0.2 10.8.0.1

secret /etc/openvpn/lsi_openvpn_joaquin.key

AMBOS:

En /etc/hosts.allow metemos la IP a la que queremos acceder:

- El servidor añade: sshd : 10.8.0.2
- El cliente añade: sshd : 10.8.0.1

****

AMBOS:

Comprobamos si tenemos el túnel "tun" levantado: ifconfig o lsmod | grep tun

Si no está levantado, levantamos uno: modprobe tun

Se ejecuta la VPN y se deja corriendo: openvpn --verb 5 --config /etc/openvpn/tunel.conf

Para comprobar su funcionamiento, en una nueva terminal (SIN SER ROOT):

- En el server: ping 10.8.0.2
- En el cliente: ping 10.8.0.1

Parar VPN: killall openvpn

Eliminar tunel (una vez parada la vpn) (modo su): modprobe -r tun

4. EN LA PRÁCTICA 1 se configuró una infraestructura con servidores y clientes NTP. Modifique la configuración para autenticar los equipos involucrados.

**********CONFIGURACION**********

SERVIDOR (JOA):

cd /etc

Generamos las claves: ntp-keygen -M

Eliminamos claves de ntp por defecto (no se usaban): rm ntp.keys

Copiamos las nuevas claves: cp ntpkey_MD5key_debian..... ntp.keys

Eliminamos el archivo primeramente creado (ya no se necesita): rm ntpkey_MD5key.....

Cambiamos los permisos de ntp.keys: chmod -R 640 ntp.keys

Ponemos como propietario del archivo a ntp: chown ntp ntp.keys

Ponemos como grupo root: chgrp root ntp.keys

Modificar ntp.conf añadiendo:

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

trustedkey 1 (indica las claves de la lista se pueden usar)

Pasamos por scp el ntp.keys al cliente: scp ntp.keys lsi@10.11.48.144:/home/lsi/ntp.keys

CLIENTE (ALVARO):

Mover el archivo recibido: mv /home/lsi/ntp.keys /etc

Asignamos mismos permisos y propietarios que antes:

chmod -R 640 ntp.keys

chown ntp ntp.keys

chgrp root ntp.keys

Modifica ntp.conf poniendo que clave usará:

#Buscamos la línea “server 10.11.48.143” y le editamos:

server 10.11.48.143 key 1 (indicamos que usamos la clave 1 de la lista)

#Añadimos las líneas:

trustedkey 1

keys /etc/ntp.key

AMBOS:

Se reinicia el servicio ntp para que se aplique la configuración: systemctl restart ntp

****

Una vez reiniciado el servidor ntp, se comprueba que se sincronizan correctamente (debe esperarse unos minutos):

ntpstat

ntpq -pn -4

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

OPCION 2:

TENER LOS DOS LA VPN LEVANTADA

**********CONFIGURACION**********

AMBOS:

apt install rsyslog (ya lo teníamos)

SERVIDOR (JOA):

nano /etc/rsyslog.conf

#Editamos las líneas:

module(load="imtcp")

input(type="imtcp" port="5555")

:fromhost-ip, isequal, "10.8.0.1"/var/log/logAlvaro

systemctl restart rsyslog

CLIENTE (ALVARO):

Editamos la config de rsyslog para especificar el puerto:

nano /etc/rsyslog.conf

#Editamos:

.@@10.8.0.2:5555

Reiniciamos el sistema de rsyslog:

systemctl restart rsyslog

****

CLIENTE (ALVARO):

Enviar log: logger "MENSAJE" o logger test

SERVIDOR (JOA):

Comprobar log: tail /var/log/logAlvaro

6. En este punto, cada máquina virtual será servidor y cliente de diversos servicios (NTP, syslog, ssh, web, etc.). Configure un "firewall stateful" de máquina adecuado a la situación actual de su máquina.

PROBAR FIREWALL: sh /home/lsi/Escritorio/myfirewall.sh

Código:

```
#!/bin/bash

echo "Inicio Firewall"

# Se borra la configuración anterior
iptables -F		#Borra todas las reglas de una cadena
iptables -X		#Borra la cadena especificada
iptables -Z		#Pone a 0 todas las cadenas, es decir, sin args

#iptables -t nat -F


# Dropea el trafico antes de filtrarlo
iptables -P INPUT DROP
iptables -P OUTPUT DROP
iptables -P FORWARD DROP
```

```
# Registra en el log de iptables
iptables -A INPUT -j LOG



# Permite las conexiones que ya se han establecido
iptables -A INPUT -m conntrack --ctstate RELATED,ESTABLISHED -j ACCEPT



# Acepta todo en la interfaz loopback
iptables -A INPUT -i lo -j ACCEPT
iptables -A OUTPUT -o lo -j ACCEPT

#
iptables -A INPUT -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT
iptables -A OUTPUT -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT



# SSH

#  Interfaz ens33
iptables -A OUTPUT -o ens33 -p TCP --sport 22 -m state --state ESTABLISHED,RELATED -j ACCEPT
iptables -A OUTPUT -o ens33 -p TCP --dport 22 -m state --state NEW,ESTABLISHED -j ACCEPT

#  Interfaz ens34
iptables -A OUTPUT -o ens34 -p TCP --sport 22 -m state --state ESTABLISHED,RELATED -j ACCEPT
iptables -A OUTPUT -o ens34 -p TCP --dport 22 -m state --state NEW,ESTABLISHED -j ACCEPT

#  Me permito a mi mismo conectarme a mi maquina por SSH
iptables -A INPUT -i ens33 -s 10.11.48.143 -p TCP --dport 22 -m state --state NEW,ESTABLISHED -j ACCEPT
iptables -A OUTPUT -o ens33 -s 10.11.48.143 -p TCP --sport 22 -m state --state ESTABLISHED,RELATED -j ACCEPT
iptables -A INPUT -i ens34 -s 10.11.50.143 -p TCP --dport 22 -m state --state NEW,ESTABLISHED -j ACCEPT
iptables -A OUTPUT -o ens34 -s 10.11.50.143 -p TCP --sport 22 -m state --state ESTABLISHED,RELATED -j ACCEPT

#  Me permito a mi mismo conectarme por SSH a otra maquina
iptables -A INPUT -p TCP --sport 22 -m state --state ESTABLISHED,RELATED -j ACCEPT
iptables -A OUTPUT -p TCP --dport 22 -m state --state NEW,ESTABLISHED -j ACCEPT

#  Acepto a mi companero (10.11.48.144 y 10.11.50.144)
iptables -A INPUT -s 10.11.48.144 -p TCP --dport 22 -m state --state NEW,ESTABLISHED -j ACCEPT
iptables -A OUTPUT -d 10.11.48.144 -p TCP --sport 22 -m state --state RELATED,ESTABLISHED -j ACCEPT
iptables -A INPUT -s 10.11.50.144 -p tcp --dport 22 -m state --state NEW,ESTABLISHED -j ACCEPT
iptables -A OUTPUT -d 10.11.50.144 -p tcp --sport 22 -m state --state RELATED,ESTABLISHED -j ACCEPT

#  Permitimos SSH por eduroam y VPN
iptables -A INPUT -s 10.20.0.0/16 -d 10.11.48.143 -p TCP --dport 22 -m state --state NEW,ESTABLISHED -j ACCEPT
iptables -A OUTPUT -s 10.11.48.143 -d 10.20.0.0/16 -p TCP --sport 22 -m state --state RELATED,ESTABLISHED -j ACCEPT
iptables -A INPUT -s 10.30.0.0/16 -d 10.11.48.143 -p TCP --dport 22 -m state --state NEW,ESTABLISHED -j ACCEPT
iptables -A OUTPUT -s 10.11.48.143 -d 10.30.0.0/16 -p TCP --sport 22 -m state --state RELATED,ESTABLISHED -j ACCEPT
iptables -A INPUT -s 10.40.0.0/16 -d 10.11.48.143 -p TCP --dport 22 -m state --state NEW,ESTABLISHED -j ACCEPT
iptables -A OUTPUT -s 10.11.48.143 -d 10.40.0.0/16 -p TCP --sport 22 -m state --state RELATED,ESTABLISHED -j ACCEPT
iptables -A INPUT -s 10.50.0.0/16 -d 10.11.48.143 -p TCP --dport 22 -m state --state NEW,ESTABLISHED -j ACCEPT
iptables -A OUTPUT -s 10.11.48.143 -d 10.50.0.0/16 -p TCP --sport 22 -m state --state RELATED,ESTABLISHED -j ACCEPT



# Openvpn
iptables -A INPUT -s 10.8.0.1 -d 10.11.48.143 -p udp --dport 5555 -j ACCEPT
iptables -A OUTPUT -s 10.11.48.143 -d 10.8.0.1 -p udp --sport 5555 -j ACCEPT
iptables -A INPUT -s 10.8.0.2 -d 10.11.48.143 -p udp --dport 5555 -j ACCEPT
```

```
iptables -A OUTPUT -s 10.11.48.143 -d 10.8.0.2 -p udp --sport 5555 -j ACCEPT


# ICMP
iptables -A INPUT -p ICMP -j ACCEPT
iptables -A OUTPUT -p ICMP -j ACCEPT


# NTP
#  Servidor (Yo)
iptables -A INPUT -p UDP --dport 123 -j ACCEPT
iptables -A OUTPUT -p UDP --sport 123 -j ACCEPT
#  Cliente (Alvaro)
#iptables -A INPUT -p UDP --sport 123 -j ACCEPT
#iptables -A OUTPUT -p UDP --dport 123 -j ACCEPT


# RSYSLOG
iptables -A INPUT -s 10.11.48.144 -d 10.11.48.143 -p tcp --dport 60514 -m conntrack --ctstate NEW -j ACCEPT
iptables -A OUTPUT -s 10.11.48.143 -d 10.11.48.144 -p tcp --dport 60514 -m conntrack --ctstate NEW -j ACCEPT
iptables -A INPUT -s 10.11.48.144 -d 10.11.48.143 -p tcp --dport 61514 -m conntrack --ctstate NEW -j ACCEPT
iptables -A OUTPUT -s 10.11.48.143 -d 10.11.48.144 -p tcp --dport 61514 -m conntrack --ctstate NEW -j ACCEPT


# NAVEGACION WEB

#  Navegacion HTTP y HTTPS
iptables -A OUTPUT -j ACCEPT -o ens33 -p TCP --sport 1024:65535 -m multiport --dports 80,443

#  Consultas DNS
iptables -A INPUT -i ens33 -m state --state ESTABLISHED,RELATED -j ACCEPT

#  Permitir conexiones entrantes que esten relacionadas o establecidas anteriormente (HTTP, HTTPS, DNS)
iptables -A INPUT -i ens33 -m state --state ESTABLISHED,RELATED -j ACCEPT



# Dormimos el script
echo "Durmiendo"
sleep 2m


# Restablecemos politicas por defecto a ACCEPT
iptables -P INPUT ACCEPT
iptables -P OUTPUT ACCEPT
iptables -P FORWARD ACCEPT

# Borramos contenido de iptables
iptables -F
iptables -X
iptables -Z

echo "Fin Firewall"
```

Tipo de paquete de datos:

- Tipo INPUT: paquetes que llegan a nuestra máquina
- Tipo OUTPUT: paquetes que salen de nuestra máquina
- Tipo FORWARD: paquetes que pasan por nuestra máquina

-A — Añade la regla iptables al final de la cadena especificada. Este es el comando utilizado para simplemente añadir una regla cuando el orden de las reglas en la cadena no importa.

-i — Configura la interfaz de red entrante
-j — Salta a un objetivo particular cuando un paquete coincide con una regla particular. Los objetivos válidos a usar después de la opción -j incluyen las opciones estándar (ACCEPT, DROP, QUEUE y RETURN)

Additional match options are also available through modules loaded by the `iptables` command. To use a match option module, load the module by name using the `-m` option, such as `-m <module-name>` (replacing *<module-name>* with the name of the module).

Conntrack: modulo para no perder paquetes TCP

- módulo `state` — Habilita la coincidencia de estado.

  El módulo `state` tiene las siguientes opciones:

  - `--state` — coincide un paquete con los siguientes estados de conexión:
    - `ESTABLISHED` El paquete seleccionado se asocia con otros paquetes en una conexión establecida.
    - `INVALID` El paquete seleccionado no puede ser asociado a una conexión conocida.
    - `NEW` El paquete seleccionado o bien está creando una nueva conexión o bien forma parte de una conexión de dos caminos que antes no había sido vista.
    - `RELATED` El paquete seleccionado está iniciando una nueva conexión en algún punto de la conexión existente.

  Estos estados de conexión se pueden utilizar en combinación con otros separándolos mediante comas como en `-m state --state INVALID, NEW`.

- --dport — Especifica el puerto destino del paquete UDP, usando el nombre del servicio, número de puerto, o rango de números de puertos. La opción de coincidencia --destination-port es sinónimo con --dport.
- --sport — Configura el puerto fuente del paquete UDP, usando el nombre de puerto, número de puerto o rango de números de puertos. La opción --source-port es sinónimo con --sport.

https://web.mit.edu/rhel-doc/4/RH-DOCS/rhel-rg-es-4/s1-iptables-options.html
https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/4/html/reference_guide/s2-iptables-options-target

```
# NTP
#  Servidor (Yo)
#iptables -A INPUT -p UDP --dport 123 -j ACCEPT
#iptables -A OUTPUT -p UDP --sport 123 -j ACCEPT
#  Cliente (Alvaro)
iptables -A INPUT -p UDP --sport 123 -j ACCEPT
iptables -A OUTPUT -p UDP --dport 123 -j ACCEPT
```

RSYSLOG
iptables -A INPUT -s 10.11.48.144 -d 10.11.48.143 -p tcp --dport 514 -m conntrack --ctstate NEW -j ACCEPT
iptables -A OUTPUT -s 10.11.48.143 -d 10.11.48.144 -p tcp --dport 514 -m conntrack --ctstate NEW -j ACCEPT
iptables -A INPUT -s 10.11.48.144 -d 10.11.48.143 -p tcp --dport 514 -m conntrack --ctstate NEW -j ACCEPT
iptables -A OUTPUT -s 10.11.48.143 -d 10.11.48.144 -p tcp --dport 514 -m conntrack --ctstate NEW -j ACCEPT