Práctica 3

apuntes

1)

- Empezar por apartado c -> pasar de contraseña a clave. Usar ssh-keygen
- Utilizar `ssh -v`, muestra el orden de los metodos de autenticacion. Primero se prueba el de clave pública-privada. Tiene que entrar de la máquina de uno a la del otro sin necesidad de contraseña.
- Fingerprinting ssh. La huella del servidor se guarda en el fichero known_hosts. Para que sirve? Guardo la huella y se guarda su ip y su fingerprinting. Si mañana vuelvo a entrar en la máquina y coincide el fingerprinting -> ok. Pasan 2 meses y se cambia la máquina, entonces cambia el fingerprinting. Avisa de que el fingerprinting ha cambiado y pregunta si cambiarlo o, hace un aviso y no deja acceder. Lo segundo es lo más normal. Puede ser que haya cambiado porque hayan spoofeado a la otra máquina, entonces así tenemos un método para saber si la máquina a la conectamos es la que esperamos.
- Montar un túnel local que securice un servicio no securizado. por ejemplo, el apache. Por ejemplo, tunel local 5555 que vaya a un puerto 80 de otro server
- Montar un fichero compartido. Cliente-servidor. Las carpetas y ficheros tienen que estar sincronizados.

2)

- Apache. El apache tiene que funcionar también con https (443). La página tiene que tener un certificado instalado. Generar certificado para el apache y determinar un entidad certificadora. El certiciado digital se lo tenemos que mandar a un entidad certificadora (Nuestro compañero) y la entidad lo firma. Ese certificado firmado se le devuelve al apache y lo instala. No puede dar error. o está perfecto, o no va. Si funciona, ya está, no se va a toquetear nada. Si da un warning, no hay problema, se acepta. Pero no puede dar error. Esto tiene que hacerse en las dos máquinas

3)

- OpenVPN. Ni muy corto ni muy largo. DEfinir servidor, rango de ips, si ips dinamicas o estaticas... Para no complicarlo, lo definimos como tipo de clave compartida. Tiene trampa. Tiene que funcionar bien porque el apartado 5 depende de el.

4)

- Autenticacion NTP. Sólo aquellas máquinas que el servidor permite se van a poder conectar.

5)

- Lo mismo para rsyslog. No quiere cifrado de tráfico por configuración. El cifrado viene dado por OpenVPN. Configurar rsyslog para que vaya por vpn y demostrar que los paquetes rsyslog al servidor van cifrados

(ettercap)

6)

- Firewall. tipo statefull. Aquel que tiene en cuenta el estado de las conexiones. Vamos a generar un firewall statefull para los servicios ya levantados. ssh,ntp,rsyslog,dns,loopback,443,80,vpn...
- ejemplo: rsyslog. Rsyslog con tcp o con udp. Lo hacemos por tcp. En el firewall que tiene que pasar? El trafico udp se dropea. A nivel de interacción, el que inicia la conexión es el cliente, entonces en el firewall tenemos que identificar que se admite el tráfico de cliente al servidor, no al reves. El resto de tráfico lo dropeo. En el lado de servidor es al reves. Cualquier tráfico entrante con destino el mío, lo admito. Nada más, yo no voy a enviar nada.
- Como miro esto? Si funcionan los 5 apartados. Enciendo el firewall y se comprueba que todo sigue funcionando.

Apartados más complicados:

- El 1 y el 6.

Apartados

1)

c)

- crear claves rsa en las máquinas:

```
ssh-keygen -b 4096 -t rsa
```

- Ahora, poner la clave pública en el fichero authorized_keys de la máquina a la que nos queremos conectar.

a)

```
ssh -v user@host
```

- Fichero ssh*known*hosts en `/etc/ssh/ssh_known_hosts` Para ver la clave fingerprint de la máquina del compañero:

```
ssh-keyscan 10.11.49.134
```

Y cogemos la clave ssh-rsa pública del servidor y la pegamos en el archivo ssh*known*hosts. **Nota: Mirar si poner solo clave rsa o poner también las demás claves**

Ahora, para saber que estamos usando el contenido de este fichero, borramos el fichero $HOME/.ssh/ssh*known*hosts y hacemos `ssh -v` (acordarse de hacerlo desde el user lsi), y vemos que no nos pide guardar el fingerprint y que encuentra el fingerprint en el fichero `/etc/ssh/known_hosts_ssh`

b)

Hacer un `scp` con un algoritmo en concreto.

```
scp -v -c aes128-ctr file.txt lsi@10.11.51.134:/home/lsi
```

Para ver los algoritmos de cifrado:

```
ssh -Q cipher
```

Lo que se cifra es la conexión, no el archivo. En el output del scp verbose podemos ver las siguientes líneas sobre el cifrado:

```
debug1: kex: algorithm: curve25519-sha256
debug1: kex: host key algorithm: ssh-ed25519
debug1: kex: server->client cipher: aes128-ctr MAC: umac-64-etm@openssh.com compression: n
debug1: kex: client->server cipher: aes128-ctr MAC: umac-64-etm@openssh.com compression: n
```

e)

```
sshfs lsi@10.11.51.134:/home/lsi /mnt/lsi
```

2)

En /usr/lib/ssl/misc

> Importante! En el common name(nombre de dominio) poner un dominio válido Si no se tiene vpn, se puede poner un dominio en /etc/hosts Los campos que se pongan aquí, tienen que coincidir con los que pongas en el certificado que generas para tu apache

```
./CA.pl -newca
```

Generar nuestro certificado)

```
./CA.pl -newreq-nodes
```

El certificado (`newkey.pem` y `newreq.pem`) lo copiamos al directorio `/usr/lib/ssl/misc` del compañero que nos lo va a firmar Nos lo firma el compañero y nos pasa los ficheros (podemos usar el newkey antiguo), pero necesitamos el ficher `newcert.pem`

```
#Desde la máquina del compañero, meter los archivos a firmar en /usr/lib/ssl/misc
./Ca.pl -sign
```

pasamos los archivos `newkey.pem`, `newcert.pem` (firmado) y `/usr/lib/ssl/misc/demoCA/cacert.pem` a nuestro compañero

- Ejecutamos lo siguiente:

```
cp newkey.pem /etc/ssl/private/Mi_Clave_Privada.key
cp newcert.pem /etc/ssl/certs/Mi_Certificado.crt
```

- Instalar certificado en apache:

Hay que crear el directorio `/var/www/ssl`

Contenido del site default-ssl.conf:

```
<IfModule mod_ssl.c>
    <VirtualHost *:443>
        ServerAdmin lsi@lsijorge
        ServerName lsijorge
        DocumentRoot /var/www/ssl

        LogLevel warn
        ErrorLog ${APACHE_LOG_DIR}/error.log
        CustomLog ${APACHE_LOG_DIR}/access.log combined

        SSLEngine on
        SSLCertificateFile  /etc/ssl/certs/Mi_Certificado.crt
        SSLCertificateKeyFile /etc/ssl/private/Mi_Clave_Privada.key

        SSLVerifyClient None
    </VirtualHost>
</IfModule>
```

Instalar certificado CA:

```
# El certificado cacert.pem es el certificado de CA de nuestro compañero
cp /home/lsi/cacert.pem /usr/local/share/ca-certificates
mv julian.pem julian.crt
update-ca-certificates
```

Finalmente, como tanto en el ssl como en apache hemos puesto que el nombre del servidor es `lsijorge`, en el cliente desde el cual queramos acceder a la página, tiene que haber una entrada en `/etc/hosts` que ponga:

```
10.11.49.145 lsijorge
# o  127.0.0.1 lsijorge, si es nuestra máquina
```

Además, también deberíamos poner una entrada para el servername de la entidad certificadora.

3)

Instalación

```
apt-get install openvpn
apt-get install openssl
```

Generar la clave compartida en el servidor

```
cd /etc/openvpn
openvpn --genkey secret vpn.key
mv vpn.key /etc/openvpn/server/vpn.key
```

> Esta clave se tiene que copiar a los que usarán la vpn al directorio /etc/openvpn/client/

Archivos de configuración de ejemplo en `/usr/share/doc/openvpn/examples/sample-config-files`

Info sobre cipher negotiations

Configuración del servidor:

`file /etc/openvpn/server/server.conf`

```
local 10.11.51.145
port 1194
proto udp
dev tun0
ifconfig 10.8.0.1 10.8.0.2

user nobody
persist-key
persist-tun

keepalive 10 120

status /var/log/openvpn/openvpn-status.log
#log-append /var/log/openvpn/openvpn.log
verb 3
explicit-exit-notify 1c

#meter la clave generada en /etc/openvpn/server/vpn.key
secret /etc/openvpn/server/vpn.key
cipher AES-256-CBC
```

Ejecutar `openvpn server.conf`

Configuración del cliente:

`file /etc/openvpn/client/client.conf`

```
remote lsijorge
```

```
dev tun0
ifconfig 10.8.0.2 10.8.0.1

secret /etc/openvpn/client/vpn.key
verb 3
cipher AES-256-CBC
```

Ejecutar `openvpn client.conf`

Comprobación

Ver que funciona correctamente: Ver que están configuradas las interfaces en `ifconfig` `ping 10.8.0.1` y `ping 10.8.0.2`

Para ver si funciona el cifrado de la vpn:

```
#Desde mi máquina
ettercap -T -i ens34 /10.11.51.145//80 /10.11.51.134//
```

Ahora, hacemos una peticion con `lynx http://10.11.51.145` desde la máquina del compañero. Se debería ver el contenido de la petición

Ahora sniffamos el puerto del vpn

```
ettercap -T -i ens34 /10.11.51.145//1194 /10.11.51.134//
```

Y hacemos la petición con `lynx http://10.8.0.1`, y vemos que el tráfico está cifrado.

4)

source

5)

Para que se inicie el vpn al boot, hacemos lo siguiente:

Si somos cliente, cambiamos server por client

```
nano /usr/lib/systemd/system/openvpn@server
```

Y ponemos lo siguiente:

```
[Unit]
Description=OpenVPN Robust And Highly Flexible Tunneling Application On %I
After=syslog.target network.target

[Service]
PrivateTmp=true
Type=forking
```

```
PIDFile=/var/run/openvpn/%i.pid
ExecStart=/usr/sbin/openvpn --daemon --writepid /var/run/openvpn/%i.pid --cd /etc/openvpn/

[Install]
WantedBy=multi-user.target
```

Ahora, tenemos que poner el archivo de configuracion (server.conf o client.conf) en el directorio /etc/openvpn. Ademas, el working directory del servicio va a ser /etc/openvpn, así que al especificar la ruta del secret en el archivo de configuracion, tiene que ser teniendo en cuenta eso. Si le ponemos una ruta absoluta, no va a haber problemas, pero sino, tienes que poner la ruta teniendo en cuenta el working directory.

Ahora, hacemos enable del servicio:

```
systemctl enable openvpn@server
```

Hacemos reboot y comprobamos que funcione la conexion entre las maquinas

rsyslog

Para ver que se ven los paquetes rsyslog sin cifrar:

Los comandos ettercap los realizaremos desde el cliente rsyslog (que tambien es en este caso el servidor vpn), si no se tiene esta configuracion, cambiar los puertos a los correctos.

```
ettercap -T -i ens34 /// /10.11.51.134//514
```

Y hacemos que se mande un log. Se deberian ver los logs en texto claro.

ahora modificamos el archivo `/etc/rsyslog.conf` de manera que se utilice la interfaz de vpn. Importante acordarse de modificar el allowed sender en el servidor rsyslog

Ahora comprobamos que no se reciba nada haciendo lo siguiente:

```
ettercap -T -i ens34 /// /10.11.51.134//514
```

Finalmente, vemos que el trafico rsyslog va cifrado por la vpn:

```
ettercap -T -i ens34 ///1194 /10.11.51.134//
```