

Guía de prácticas - Clase 3

Práctica 1: Autenticación en SSH con llave privada y llave pública

Lo que se hará será crear un par de llaves desde Windows, copiar la llave pública en una máquina Linux (nuestro Kali) que esté en la misma red, y así autenticarnos usando nuestra llave privada.

1. Generar el par de llaves, privada y pública

Para generar el par de llaves se debe correr el siguiente comando desde una máquina **Windows**:

```
ssh-keygen
```

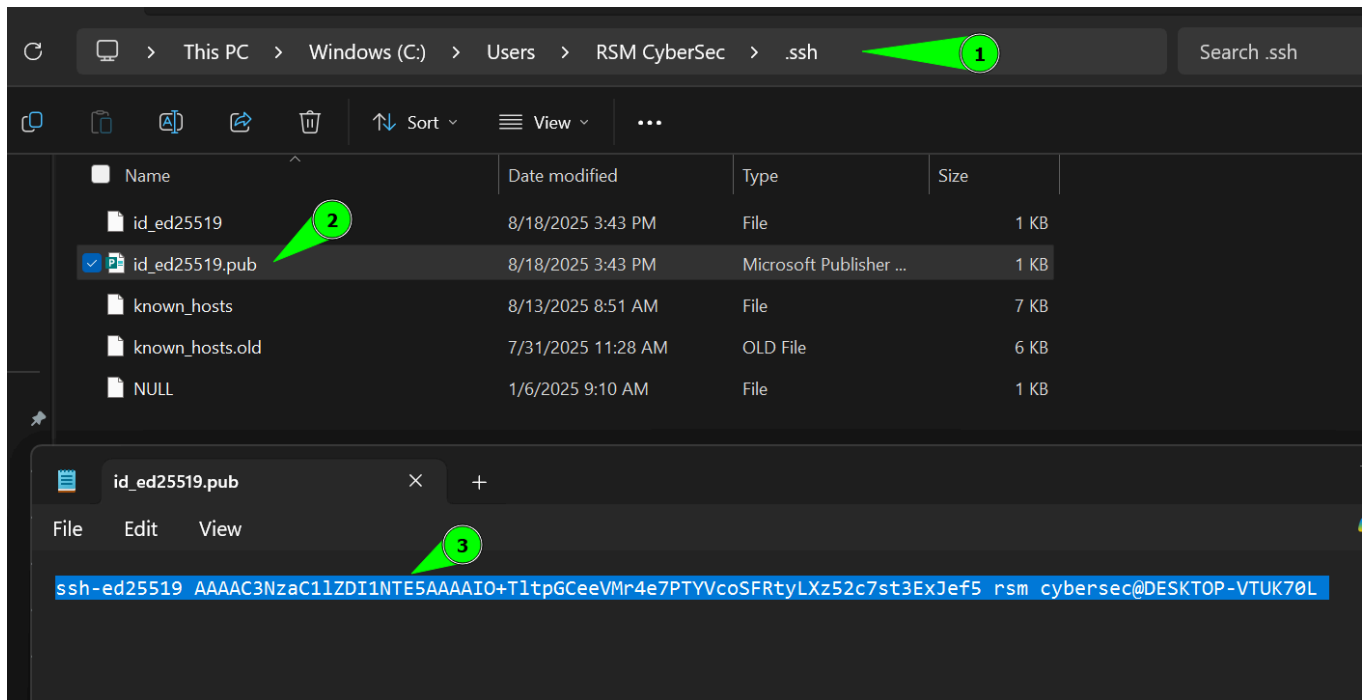
Esta herramienta mostrará opciones para cambiar la carpeta de guardado por defecto, y también para establecer una frase de contraseña. Si se quiere dejar todo por defecto, solamente se debe dar Enter:

```
PS C:\Users\RSM CyberSec> ssh-keygen
Generating public/private ed25519 key pair.
Enter file in which to save the key (C:\Users\RSM CyberSec/.ssh/id_ed25519):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in C:\Users\RSM CyberSec/.ssh/id_ed25519
Your public key has been saved in C:\Users\RSM CyberSec/.ssh/id_ed25519.pub
The key's fingerprint is:
SHA256:1xc814yktWVxHmVVeJGnVLMvZ4rjbzsTuPYsFGa1iCM rsm cybersec@DESKTOP-VTUK70L
The key's randomart image is:
+--[ED25519 256]--+
|                oO^|
|               =.X@|
|              ..o@.O|
|             E o.=o.X|
|            S..++oo|
|           . .oo.|
|          ... ..|
|         +o+|
|        . +=+|
+-----[SHA256]-----+
```

2 Copiar contenido de llave pública hacia Linux

Una vez generadas nuestras llaves, vamos a copiar el contenido de la llave pública. Para hacer esto hay varias formas, pero haremos la siguiente:

1. Ir a la ruta donde están las llaves, que es C:/Users/SU_USUARIO/.ssh/ (tener en cuenta si la máquina está en inglés o español).
2. Abrir el archivo .pub utilizando el bloc de notas.
3. Copiar el contenido de dicho archivo.



Luego, hay que editar el archivo `authorized_keys` en **Kali**, ubicado en la ruta `/home/kali/.ssh/`, y pegar el contenido copiado de la llave pública:

```
nano /home/kali/.ssh/authorized_keys
```

Para guardar el contenido utilizando nano, se debe:

1. Guardar el contenido, utilizando `Ctrl+O`, y luego dar `Enter`
2. Cerrar el archivo, utilizando `Ctrl+X`

NOTA: Puede ser que para pegar el contenido en Kali se necesite presionar `Ctrl+Shift+V` en lugar de solamente `Ctrl+V`.

Antes de finalizar esta parte, hay que encender el servicio SSH:

```
sudo systemctl start ssh
```

3. Conectarnos desde Windows hacia Linux utilizando la llave privada

Probaremos la conexión usando ssh, para ello hay que identificar la IP de la máquina Linux/Kali. Si se tienen las configuraciones hechas en clase para la máquina virtual, la IP debe ser 127.0.0.1 y el puerto debe ser el 2222.

Para conectarse, se debe correr el siguiente comando:

```
ssh -i 'C:\Users\SU_USUARIO\.ssh\id_ed25519' kali@127.0.0.1 -p 2222
```

Si no se tienen las configuraciones de clase, o no se quieren utilizar, simplemente hay que validar la IP de la máquina Kali a utilizar, y cercirarse que esté en la misma red que la máquina Windows.

Si se configuró todo correctamente, ya podrán conectarse sin necesidad de utilizar la contraseña del usuario.

```
PS C:\Users\RSM CyberSec> ssh -i 'C:\Users\RSM CyberSec\.ssh\id_ed25519' kali@127.0.0.1 -p 2222
The authenticity of host '[127.0.0.1]:2222 ([127.0.0.1]:2222)' can't be established.
ED25519 key fingerprint is SHA256:DeMSl8DsCIxIYNXuNCF+XkmmdF/b4g3Ck1GHhRLnInQ.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '[127.0.0.1]:2222' (ED25519) to the list of known hosts.
Linux kali 6.12.25-amd64 #1 SMP PREEMPT_DYNAMIC Kali 6.12.25-1kali1 (2025-04-30) x86_64

The programs included with the Kali GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Kali GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
kali@kali:~$
```

Si sale el mensaje de la imagen, solamente escribir "yes" y dar Enter.

Práctica 2: Hashing y salting

Vamos a poner en práctica los conceptos de hashing y salting, útiles para entender cómo agregar capas extra de seguridad a nuestros desarrollos.

1. Crear un hash a partir de texto

Lo primero es crear un hash de manera consistente, para ello utilizaremos el siguiente código de Python:

```
import hashlib #librería para manejar hashes

password = 'micontraseña123' #valor del cual obtendremos un hash
hashed_string = hashlib.sha256(password.encode('utf-8')).hexdigest()
```

```
#generación del hash, en este caso de tipo SHA256
print(hash_string) #impresión del hash
```

Al correr el código (que se puede realizar en páginas como <https://www.online-python.com/> o <https://www.programiz.com/python-programming/online-compiler/>), vemos que se genera un valor/digesto.

```
5e580889d437aa994bff1ec3244ab28fa643d6193e422b67b89edbaa2202e250
```

```
=== Code Execution Successful ===
```

2. Agregar salt

Para añadirle una capa de seguridad, y que sea más difícil obtener el valor original del cual se obtuvo el digesto, vamos a agregar una sal:

```
import hashlib

password = 'micontraseña123'

salt = 'valorsupersecretoeindescifrable' #sal o valor secreto
string_completo = password + salt #concatenación de la contraseña y la sal

hashed_string = hashlib.sha256(string_completo.encode('utf-8')).hexdigest()
print(hashed_string)
```

Una vez hecho esto, obtenemos un digesto completamente distinto. Para obtener el valor original un atacante deberá conocer también el valor de la sal, cosa que es muy difícil.

```
64c5036a4e8d762aed71456bef63a237be6e2314a3224cbad3f5f0c3e9b3815e
```

```
=== Code Execution Successful ===
```

NOTA: en un entorno de producción, el valor de la sal se obtiene de una variable de entorno, ya que no es recomendable dejar valores secretos en el código fuente.

Práctica 3: Captura de tráfico con Wireshark de HTTP y HTTPS

Para conocer cómo funciona el cifrado utilizando certificados SSL/TLS, vamos a realizar una práctica en la cual se vea tráfico sin cifrado (con HTTP) y con cifrado (con HTTPS).

NOTA: esta práctica se puede hacer tanto en Windows como en Kali, solo que en Windows deberán instalar la herramienta Wireshark y Python.

1. Tener sitios con HTTP y HTTPS identificados

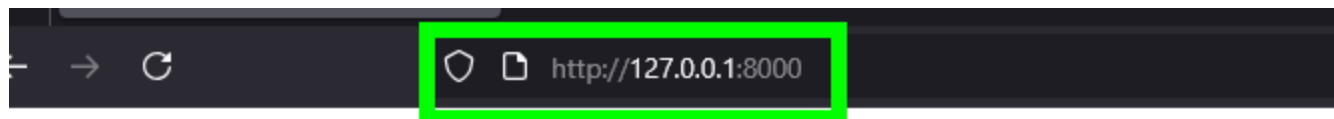
Lo primero es abrir dos sitios web, uno con HTTPS y otro solamente con HTTP. Para el de HTTPS puede ser el de la universidad, y para el de HTTP, si no encontramos uno, podemos desplegar un servidor web simple con Python.

Para desplegar un servidor web con Python se debe identificar una carpeta que se quiera compartir (de preferencia una con un proyecto web), y luego se debe ejecutar el siguiente comando:

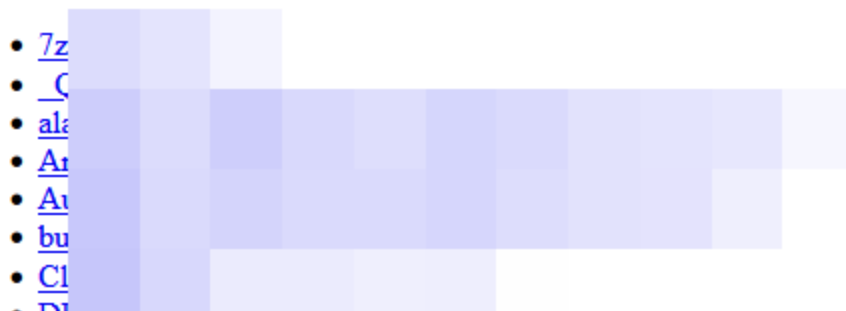
```
python -m http.server 8000
```

```
PS C:\Users\RSM CyberSec\Downloads> python -m http.server 8000
Serving HTTP on :: port 8000 (http://[::]:8000/) ...
```

Esto despliega un servidor web en la IP 127.0.0.1 y el puerto 8000. Para acceder al sitio simplemente debemos escribir <http://127.0.0.1:8000> en el navegador.



Directory listing for /

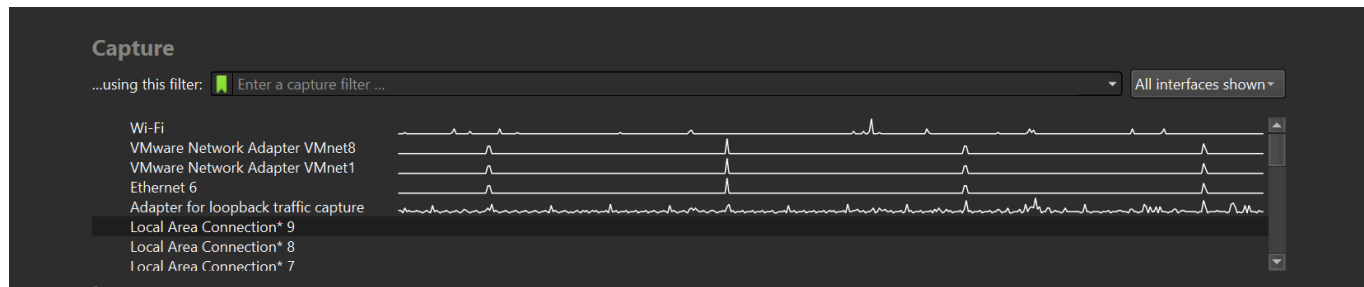


2. Capturar tráfico con Wireshark

2.1. Captura de HTTP

En la parte de "Capture" en Wireshark, vamos a escoger primero la interfaz que diga "loopback" (que es para localhost) dando doble clic, desde la cual vamos a capturar el tráfico

de la página que solamente usa HTTP:

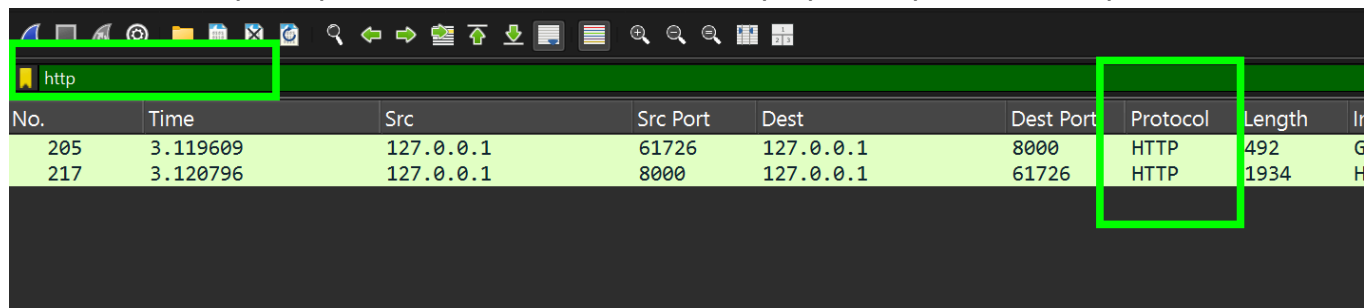


Una vez se haya dado doble clic, Wireshark empezará a capturar tráfico. En este punto, debemos recargar la página web que levantamos con Python en el navegador (127.0.0.1:8000), con lo cual veremos tráfico en Wireshark.

214	3.119742	127.0.0.1	61399	127.0.0.1	61400	TCP	44	61399 → 61400 [AC
215	3.120757	127.0.0.1	8000	127.0.0.1	61726	TCP	200	8000 → 61726 [PSH
216	3.120776	127.0.0.1	61726	127.0.0.1	8000	TCP	44	61726 → 8000 [ACK
217	3.120796	127.0.0.1	8000	127.0.0.1	61726	HTTP	1934	HTTP/1.0 200 OK
218	3.120806	127.0.0.1	61726	127.0.0.1	8000	TCP	44	61726 → 8000 [ACK
219	3.120833	127.0.0.1	8000	127.0.0.1	61726	TCP	44	8000 → 61726 [FIN
220	3.120841	127.0.0.1	61400	127.0.0.1	61399	TCP	45	61400 → 61399 [PS
221	3.120841	127.0.0.1	61726	127.0.0.1	8000	TCP	44	61726 → 8000 [ACK
222	3.120848	127.0.0.1	61399	127.0.0.1	61400	TCP	44	61399 → 61400 [AC
223	3.120851	127.0.0.1	61400	127.0.0.1	61399	TCP	45	61400 → 61399 [PS

Ahora detenemos la captura dando clic en el botón rojo de la parte superior izquierda.

En la captura se verán muchos paquetes (no necesariamente igual que en la imagen), pero los que nos interesan son los paquetes HTTP, por lo que en la barra de filtros escribiremos "http" y daremos Enter, para que solamente nos muestre los paquetes que utilizan el protocolo HTTP:



Daremos clic en uno de los paquetes, luego desplegamos la opción que dice "Line-based text data", y veremos el contenido de la página web sin ningún tipo de cifrado:

No.	Time	Src	Src Port	Dest	Dest Po
205	3.11960	127.0.0.1	61726	127.0.0.1	8000
217	3.120796	127.0.0.1	8000	127.0.0.1	61726

<ul style="list-style-type: none"> Frame 217: 1934 bytes on wire (15472 bits), 1934 byte Null/Loopback Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127 Transmission Control Protocol, Src Port: 8000, Dst Po [2] Reassembled TCP Segments (2046 bytes): #215(156), Hypertext Transfer Protocol Line-based text data: text/html (31 lines) <pre> <!DOCTYPE HTML>\n <html lang="en">\n <head>\n <meta charset="utf-8">\n <title>Directory listing for /</title>\n </head>\n <body>\n <h1>Directory listing for /</h1>\n <hr>\n \n 7z2501-x64.exe QAs.7z </pre>	<pre> 0000 02 00 00 00 45 00 07 8a 19 0010 7f 00 00 01 7f 00 00 01 1f 0020 50 ac 5c 9e 50 18 00 fe e3 0030 43 54 59 50 45 20 48 54 4d 0040 6c 20 6c 61 6e 67 3d 22 65 0050 61 64 3e 0a 3c 6d 65 74 6f 0060 74 3d 22 75 74 66 2d 38 2f 0070 65 3e 44 69 72 65 63 74 6f 0080 69 6e 67 20 66 6f 72 20 2f 0090 3e 0a 3c 2f 68 65 61 64 3c 00a0 0a 3c 68 31 3e 44 69 72 6f 00b0 69 73 74 69 6e 67 20 66 6f 00c0 3e 0a 3c 68 72 3e 0a 3c 7f 00d0 3c 61 20 68 72 65 66 3d 2f 00e0 78 36 34 2e 65 78 65 22 3c 00f0 78 36 34 2e 65 78 65 3c 2f 0100 0a 3c 6c 69 3e 3c 61 20 6f 0110 41 73 2e 37 7a 22 3e 5f 51 0120 61 3e 3c 2f 6c 69 3e 0a 3c </pre>
---	--

2.2. Captura de HTTPS

Ahora capturaremos el tráfico de HTTPS de la página web de la universidad. Para ello, debemos cambiar de interfaz de red en Wireshark, por lo que nos iremos a la opción "File" y daremos clic en "Close", y seleccionamos "Close without saving" en la ventana emergente, ya que no queremos guardar la captura en este momento:

The screenshot shows the Wireshark interface with the 'File' menu open. The 'Close' option is highlighted with a green circle and the number 2. A dialog box titled 'Unsaved packets...' is displayed in the foreground, asking 'Do you want to save the captured packets before closing the file?'. The dialog has three buttons: 'Save', 'Continue without Saving', and 'Cancel'. The 'Continue without Saving' button is highlighted with a green circle and the number 3. In the background, the packet list shows a packet from 127.0.0.1 to 127.0.0.1 on port 8000, identified as HTTP/1.0 200 OK (text/html).

Ahora escogemos una interfaz que tenga internet, dando nuevamente doble clic. En mi caso, escogeré la interfaz de Wifi, pero puede que ustedes deban escoger una Ethernet.

Una vez escogida la interfaz, Wireshark comienza la captura. Es momento de recargar o abrir la página de la universidad, yendo a <https://www.usonsonate.edu.sv>:



Al haber recargado o abierto la página, detendremos la captura de Wireshark dando clic en el botón rojo. Una vez hecho eso, hay que filtrar los paquetes, pero esta vez con el filtro "tls.app_data_proto == \"Hypertext Transfer Protocol\"":

tls.app_data_proto == "Hypertext Transfer Protocol"								
No.	Time	Src	Src Port	Dest	Dest Port	Protocol	Length	Info
51	1.186719	129.146.18.11	443	192.168.1.7	62235	TLSv1.3	1514	Server Hello, Change Cipher Spec
54	1.186719	129.146.18.11	443	192.168.1.7	62234	TLSv1.3	1514	Server Hello, Change Cipher Spec
61	1.187843	129.146.18.11	443	192.168.1.7	62235	TLSv1.3	1514	Application Data, Application Data
62	1.188060	129.146.18.11	443	192.168.1.7	62235	TLSv1.3	111	Application Data
64	1.188964	192.168.1.7	62235	129.146.18.11	443	TLSv1.3	118	Change Cipher Spec, Application Data
65	1.189161	192.168.1.7	62235	129.146.18.11	443	TLSv1.3	640	Application Data
67	1.189173	129.146.18.11	443	192.168.1.7	62234	TLSv1.3	1514	Application Data, Application Data
69	1.189548	129.146.18.11	443	192.168.1.7	62234	TLSv1.3	111	Application Data
71	1.189920	192.168.1.7	62234	129.146.18.11	443	TLSv1.3	118	Change Cipher Spec, Application Data
72	1.190121	192.168.1.7	62234	129.146.18.11	443	TLSv1.3	640	Application Data
74	1.197677	129.146.18.11	443	192.168.1.7	62236	TLSv1.3	1514	Server Hello, Change Cipher Spec
78	1.198700	129.146.18.11	443	192.168.1.7	62236	TLSv1.3	1514	Application Data, Application Data
79	1.198700	129.146.18.11	443	192.168.1.7	62236	TLSv1.3	111	Application Data

Flags: 0x018 (PSH, ACK)
Window: 476
[Calculated window size: 60928]
[Window size scaling factor: 128]
Checksum: 0x1b9f [unverified]

0000 17 03 03 00 35 e6 4c 5c fd fa f8 df a4 ab 61 46 ... 5-L\aF
0010 46 18 b3 5c ec 73 2e 0b 5e 7e 7b 25 1f 88 8b 0a F...s.. ^~{%...
0020 5a 75 14 de 45 81 05 67 a5 86 73 e0 d2 36 f4 26 Zu...E...g ...s..6&
0030 f3 ac 5d de c2 b6 f3 85 7f 07 ..]..... ..

Ahora, si damos clic en algún paquete que diga "Application Data", estos datos están cifrados, por lo que no podrían ser leídos fácilmente y obtener la información que transportan estos paquetes.