



SLIIT

Discover Your Future

Information Security Project IE3092

MID_REVIEW

Year 3 Semester 2

Name	ID Number
S.G. Rajakaruna	IT18112924
A.D. Wijesooriya	IT18125894

1. Introduction

In our project we have selected life stories of real word hackers and we have considered a couple of attack incidents of their life as a single box. Every participant has to do step by step to achieve the main goal in one box. High level idea of this CTF is to bring up the greatest attacks that the world-renowned cyber security personalities carried out during their lifetime, to a simulation Environment.

2. Target Audience

The CTF's target market encompasses a wide spectrum, while the main target is to be the cybersecurity student undergraduates. Students will get hands-on experience on the concepts they learn through the course modules and it would also be a forum for testing their skill set. Not only students but also red teams are targeted. Red Team Security offers full-force red teaming addressing cyber-attacks, social engineering, and physical security in testing threat profiles. This means comprehensive testing of our business's technical landscape as well as fully testing our people and physical security controls. The CTF is also available to the private sector, however, practically to anybody who wants an opportunity to venture into the domain of information security. In fact, this CTF would be a perfect forum to sharpen their domain knowledge with the people currently interested in the area.

3. Implementation

We have used an AWS EC2 instance setup and it acts as our backend server. Base OS is Ubuntu 18.04 and we have set up Docker on our virtual machine. Every level is implemented in separate Docker containers and there are separate Docker compose yml files for each and every container. These yml files are responsible for managing the containers. We have done port mapping between the virtual machine and Docker containers. Therefore, users can access the Docker containers easily using the virtual machine IP address and the ports. Frontend is done using react and also it is hosted in a Docker container on the same virtual machine.

We have implemented one Scenario based on David Mitnick's life story for now. That scenario consists of three major different levels and each level has five different tasks to complete.

Web app(React):

HackerLife


SIGN INSIGN UP

David Mitnick

What are you waiting for?


GET STARTED

Life & Times of Kevin David Mitnick!



At age 12

Scenario I - Hacking Public Transport



In 1988

Scenario II - Hacking Pacific Bell Voicemail

4. Scenario 1 - Level 1

This level is based on basic Linux commands and basic encryption methods.

First, you need to connect to the level 1 Docker container using ssh. Credentials are follows,

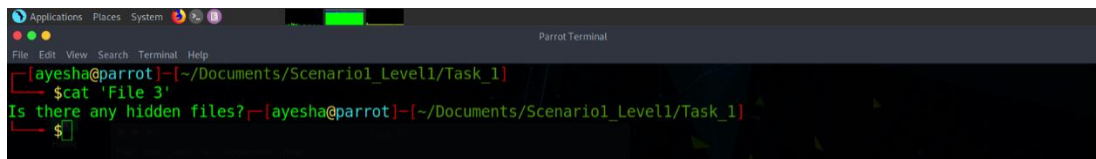
- Username: level1
- Password: level1
- Port: 2020

ssh -p 2020 level1@ec2-54-145-129-42.compute-1.amazonaws.com

A. Task 1

We have stored a hint in file 3. You have to read this file using cat command.

Command : **cat 'file 3'**

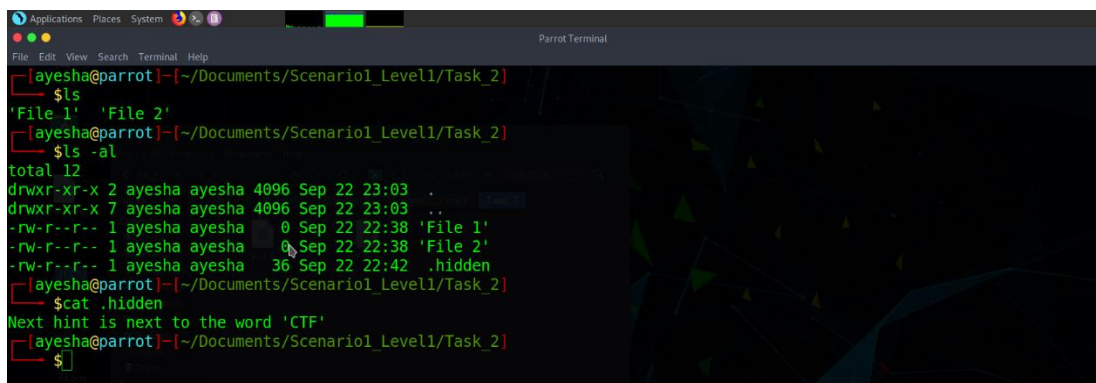


```
ayesha@parrot: ~/Documents/Scenario1_Level1/Task_1
$ cat 'File 3'
Is there any hidden files? ayesha@parrot: ~/Documents/Scenario1_Level1/Task_1
$
```

B. Task 2

We have created a file with ".". therefore that file will be hidden. The hint is stored in a hidden file in the Task_2 directory. While logged into the task 2 you can run the "ls" command to see if we find any useful files. Then you can use "ls -al" to see all of the files including the hidden ones. The hidden file is named .hidden and after running "cat .hidden". then you can get hint of the task 3.

Command : **cat .hidden**

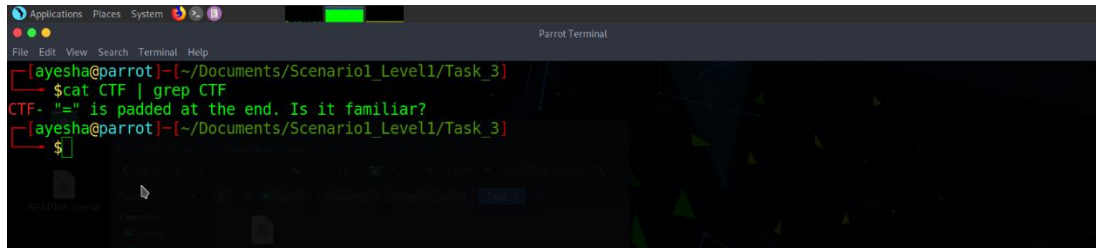


```
ayesha@parrot: ~/Documents/Scenario1_Level1/Task_2
$ ls
'File 1' 'File 2'
ayesha@parrot: ~/Documents/Scenario1_Level1/Task_2
$ ls -al
total 12
drwxr-xr-x 2 ayesha ayesha 4096 Sep 22 23:03 .
drwxr-xr-x 7 ayesha ayesha 4096 Sep 22 23:03 ..
-rw-r--r-- 1 ayesha ayesha  0 Sep 22 22:38 'File 1'
-rw-r--r-- 1 ayesha ayesha  0 Sep 22 22:38 'File 2'
-rw-r--r-- 1 ayesha ayesha 36 Sep 22 22:42 .hidden
ayesha@parrot: ~/Documents/Scenario1_Level1/Task_2
$ cat .hidden
Next hint is next to the word 'CTF'
ayesha@parrot: ~/Documents/Scenario1_Level1/Task_2
$
```

C. Task 3

The hint for the next task is stored in the file CTF next to the word CTF, so you can use the command below to read the file and then grep the word CTF. Grep command is use to search plaintext.

Command : **cat CTF | grep CTF**

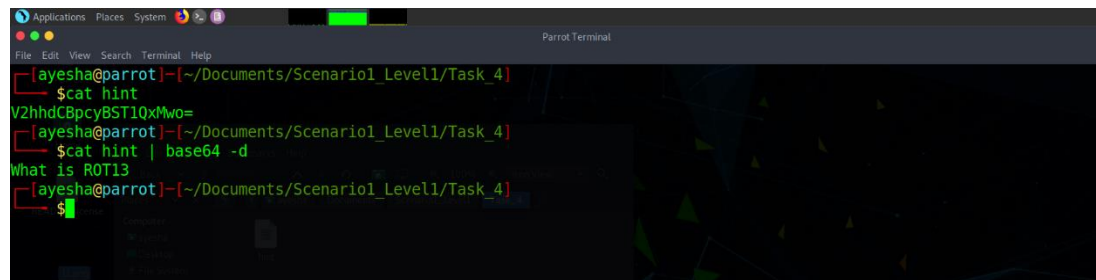


```
ayesha@parrot: ~/Documents/Scenario1_Level1/Task_3
$ cat CTF | grep CTF
CTF- "=" is padded at the end. Is it familiar?
ayesha@parrot: ~/Documents/Scenario1_Level1/Task_3
$
```

D. Task 4

The hint contains 1 line that was encoded in base64. In order to decode the file uou have to use base64 -d.

Command : **cat hint | base64 --d**



```
ayesha@parrot: ~/Documents/Scenario1_Level1/Task_4
$ cat hint
V2hhdCBpcy8STlQxMwo=
ayesha@parrot: ~/Documents/Scenario1_Level1/Task_4
$ cat hint | base64 --d
What is ROT13
ayesha@parrot: ~/Documents/Scenario1_Level1/Task_4
$
```

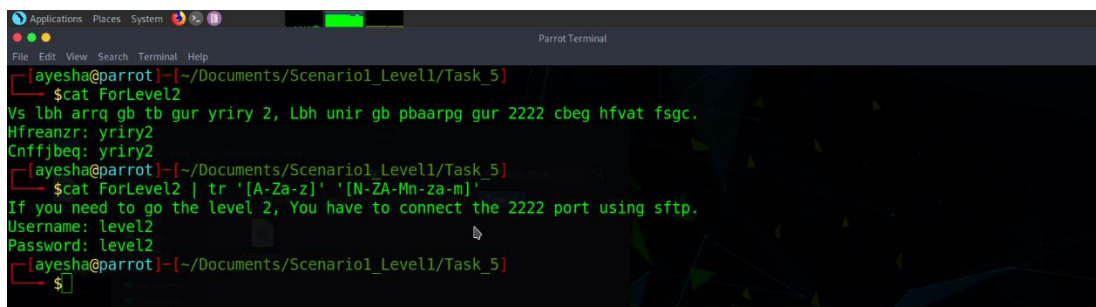
E. Task 5

The hint for the next level is stored in the file ForLevel2, where all lowercase (a-z) and uppercase (A-Z) letters have been rotated by 13 positions.

The ForLevel2 file contains 3 line that was encrypted with the ROT13 algorithm. In order to decrypt it, you have to replace every letter by the letter 13 positions ahead. For example, with this encryption the letter a would be replaced with n. The word banana would encrypt to onanan.

Command : **cat ForLevel2 | tr '[A-Za-z]' '[N-ZA-Mn-za-m]'**

tr- command line utility for translating or deleting characters. It supports a range of transformations including uppcase to lowercase, squeezing repeating characters, deleting specific characters and basic find and replace.



```
ayesha@parrot: ~/Documents/Scenario1_Level1/Task_5
$ cat ForLevel2
Vs lbh arrq gb tb gur yriry 2, Lbh unir gb pbaarpg gur 2222 cbeg hfvat fsgc.
Hfreanzr: yriry2
Cnffjbeq: yriry2
ayesha@parrot: ~/Documents/Scenario1_Level1/Task_5
$ cat ForLevel2 | tr '[A-Za-z]' '[N-ZA-Mn-za-m]'
If you need to go the level 2, You have to connect the 2222 port using sftp.
Username: level2
Password: level2
ayesha@parrot: ~/Documents/Scenario1_Level1/Task_5
$
```

You can get the port number and credentials for the Level 3 from this.

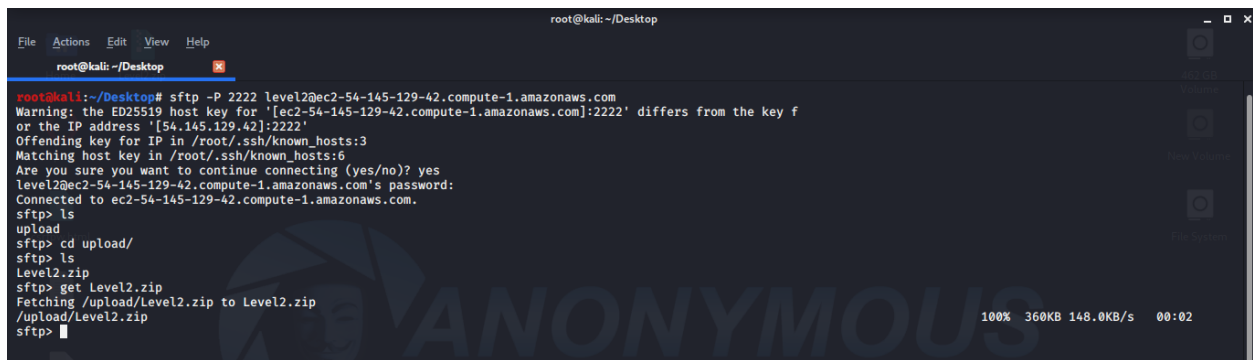
5. Scenario 1 - Level 2

Level two is based on steganography and cryptography. There are five tasks to complete to achieve next level credentials.

You need to connect to the level 2 container using Secure File Transfer Protocol(SFTP). Credentials are follows,

- **Username: level2**
- **Password: level2**
- **Port: 2222**

sftp -P 2222 level2@ec2-54-145-129-42.compute-1.amazonaws.com



```
root@kali: ~/Desktop
File Actions Edit View Help
root@kali: ~/Desktop
root@kali:~/Desktop# sftp -P 2222 level2@ec2-54-145-129-42.compute-1.amazonaws.com
Warning: the ED25519 host key for '[ec2-54-145-129-42.compute-1.amazonaws.com]:2222' differs from the key f
or the IP address '[54.145.129.42]:2222'
Offending key for IP in /root/.ssh/known_hosts:3
Matching host key in /root/.ssh/known_hosts:6
Are you sure you want to continue connecting (yes/no)? yes
level2@ec2-54-145-129-42.compute-1.amazonaws.com's password:
Connected to ec2-54-145-129-42.compute-1.amazonaws.com.
sftp> ls
upload
sftp> cd upload/
sftp> ls
Level2.zip
sftp> get Level2.zip
Fetching /upload/Level2.zip to Level2.zip
/upload/Level2.zip
sftp> 100% 360KB 148.0KB/s 00:02
```

We have uploaded a zip file. You can find it by navigating to the upload folder.

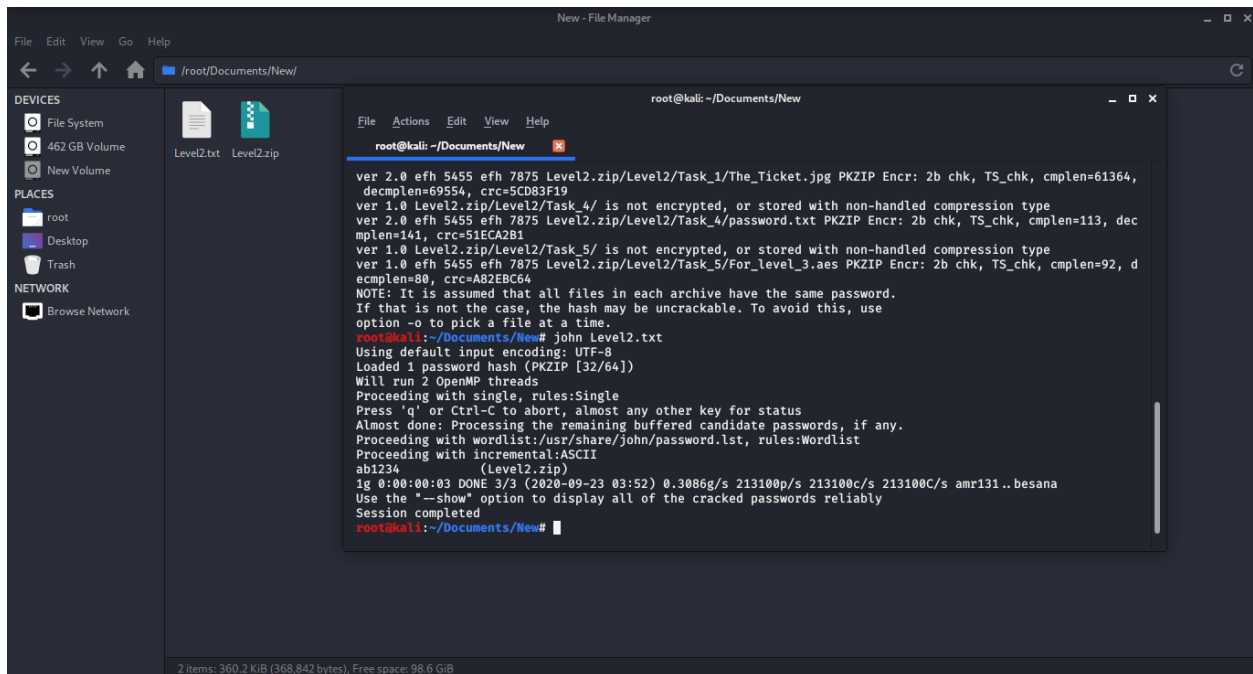
Download it to your local machine using **get** command and do all the tasks you have to do to get the final flag of this level.

A. Task 0

You have to unzip the zip file to get your five tasks but the zip file is locked with a password and we won't provide the password for you. Only thing you have to do is crack the zip file and get the five tasks.

This the way you can do it,

- **Install john → apt-get install john**
- **zip2john Level2.zip**
- **zip2john Level2.zip > Level2.txt**
- **john Level2.txt**



You can get the zip password from using this tool.

Password: ab1234

Important thing is each task has a hint for the next task.

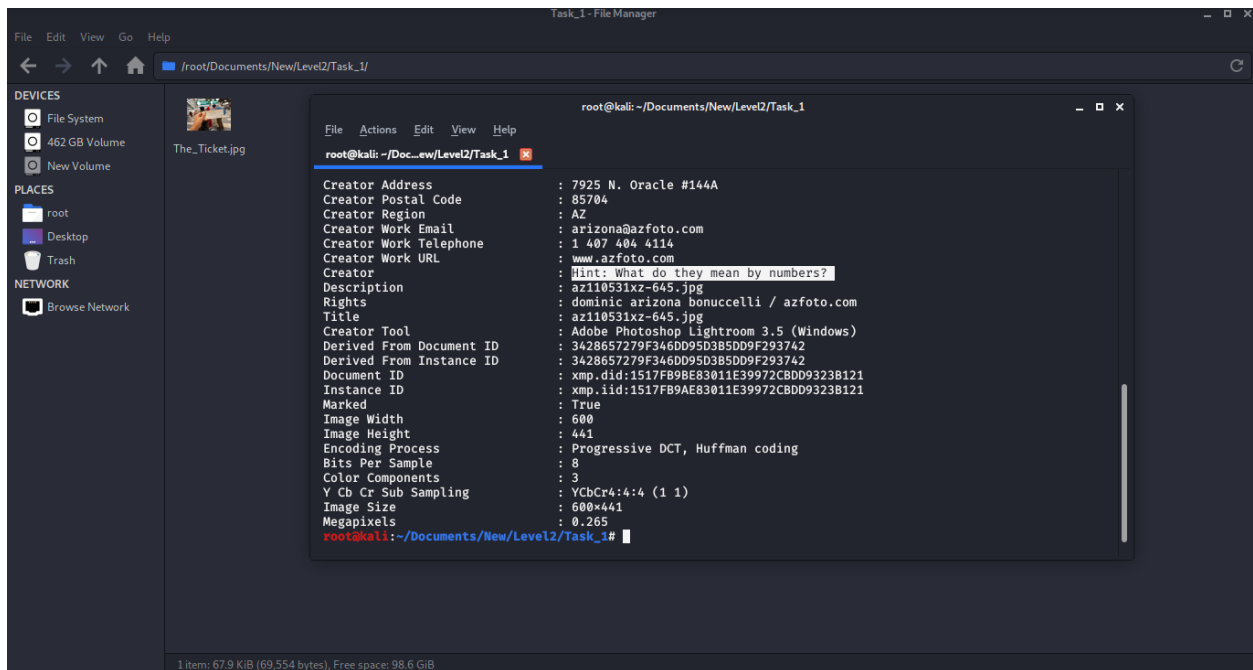
B. Task 1

In task 1 we used a tool called **exiftool** to hide the hint inside the metadata. We can simply mention the variable name and hide a message in the image.

exiftool -<V.name>=<message> <image>

In Task_1 there is an image called The_Ticket. All you have to do is check the metadata of the image.

exiftool <image>



Hint: What do they mean by numbers?

C. Task 2

In task 2 you can see the Task 2.txt file which contains some integer values. We used a simple c program to convert char into ASCII values.

So, you can develop a small c program or python script to reverse the process.

C Code:

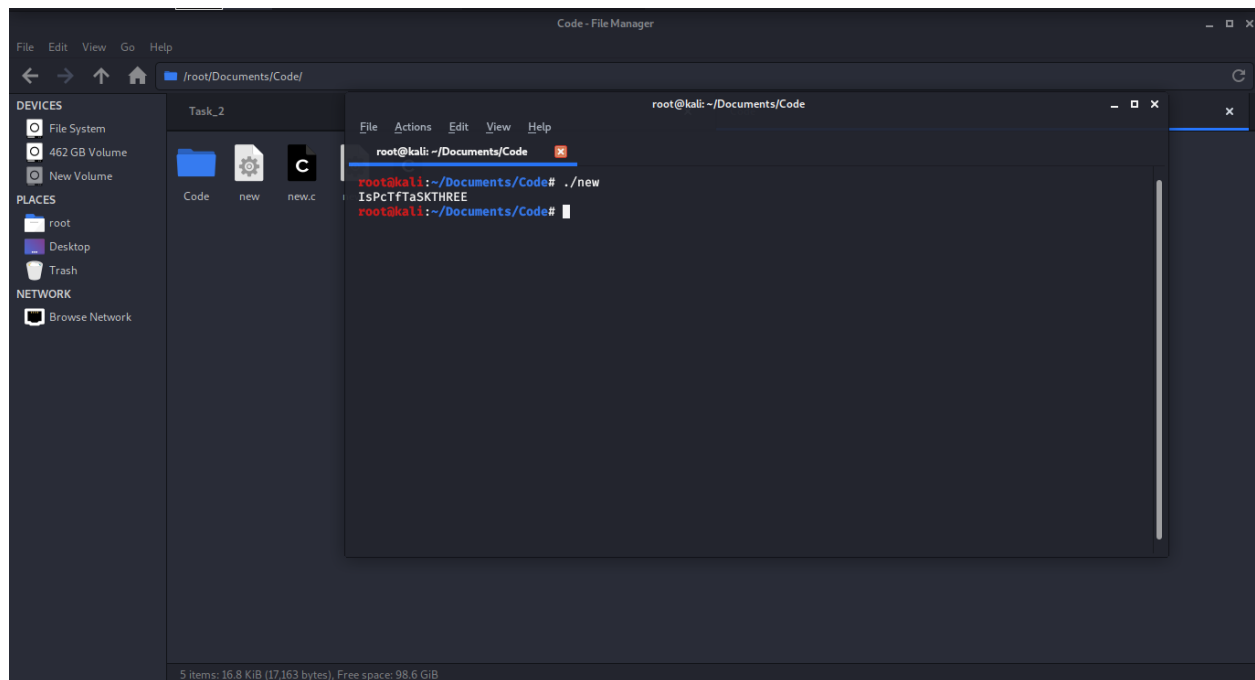
```
#include <stdio.h>
```

```
int main() {
```

```
    int x[15] = {73, 115, 80, 99, 84, 102, 84, 97, 83, 75, 84, 72, 82, 69, 69};
```

```
    for(int i = 0; i < 15; i++){
        printf("%c", x[i]);
    }
```

```
    printf("\n");
    return 0;
```

So, you can find a password from this. It will help you to deal with task 3.

Password: IsPcTfTaSKTHREE

D. Task 3

We used steganography to implement this task. There are two images inside the Task 3 folder. One is new.jpg and the other one is old.jpg. You cannot find any difference between these two. We used the steghide tool to hide data.

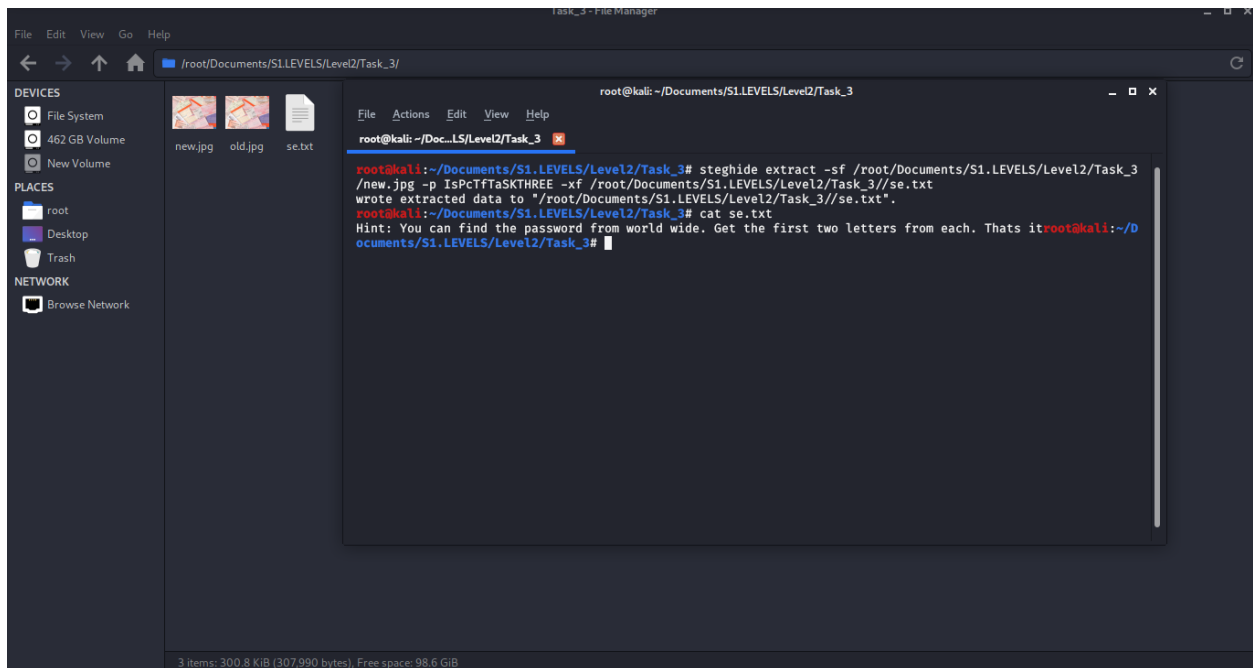
steghide embed -ef <SecretMessage.txt> -cf <image> -p <password>

We have embedded data into the new.jpg image with the password which you obtained in level 2.

You can get the hint for the next level by using,

steghide extract -sf <image> -p <password> -xf <a text file for save the hidden message>

Password: IsPcTfTaSKTHREE (found in Task 2)



Hint: You can find the password from world wide. Get the first two letters from each. That's it.

E. Task 4

We have saved some numbers in a text file which you can find in the Task_4 folder.

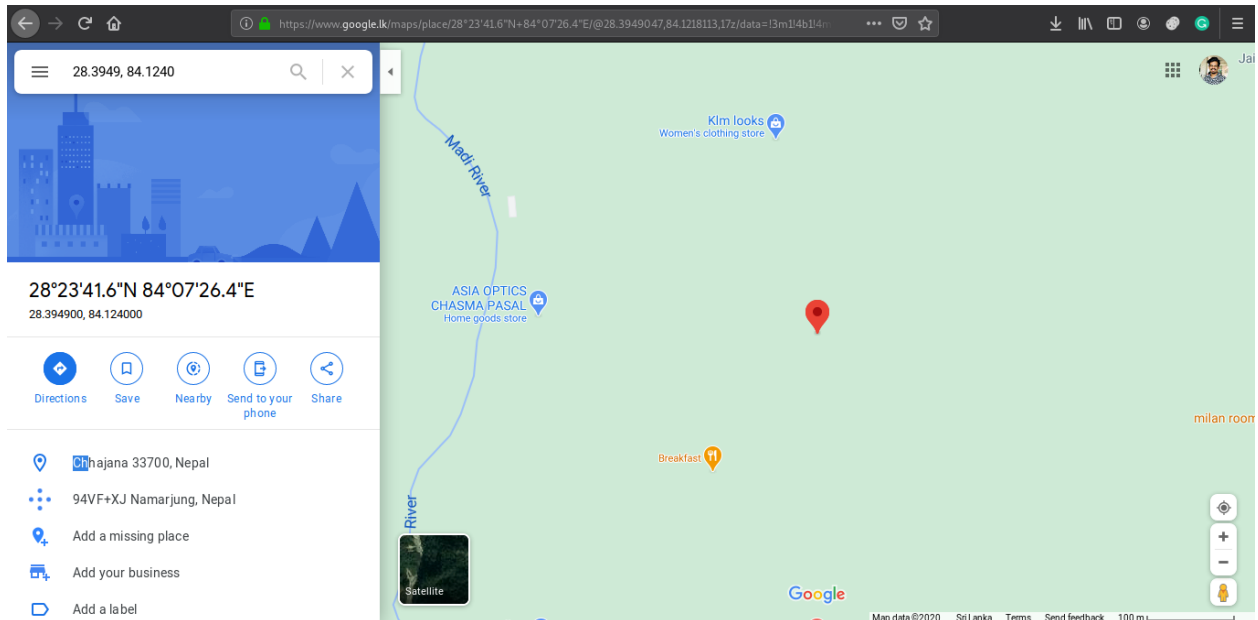
password:

```
{  
    28.3949, 84.1240  
    3.2028, 73.2207  
    20.5937, 78.9629  
    27.5142, 90.4336  
    15.8700, 100.9925  
    8.3405, 115.0920  
}
```

The hint for this level is **You can find the password from world wide. Get the first two letters from each. That's it.**

So, these numbers are latitudes and longitudes for some places. You can search these couple of values one by one on google maps and get the first two letters.

As an example:



Do this thing for each pair and take the first two letters.

Password: ChLaKhBhNoSo

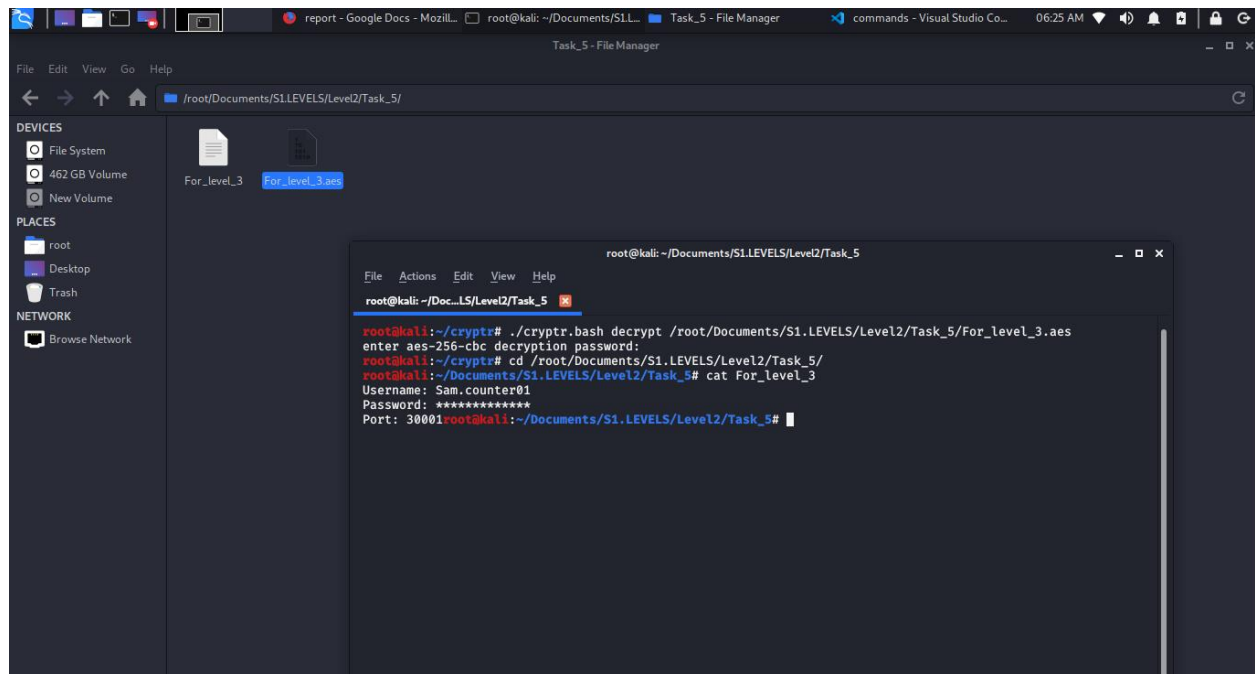
F. Task 5

In task 5 there is a file with .aes extension. That means File encrypted by AES Crypt. We used a tool called **cryptr** for this task.

You can decrypt this file by using following command,

./cryptr.bash decrypt <filename>

It will ask a password for decryption. Password is **ChLaKhBhNoSo**



Finally, you can obtain the credentials for the level 3.

- **Username: Sam.counter01**
- **Password: *******
- **Port: 30001**
- **It is a web based**

6. Scenario 1 - Level 3

Level 3 tasks are web based tasks and we have used MySQL, PHP and phpmyadmin images for this level of containers.

```
1 version: 3.4
2 services:
3   php:
4     build:
5       context: .
6       image: amakundu/moe-php-mysql-demo:1.0.0
7     networks:
8       - frontend
9       - backend
10    environment:
11      - MYSQL_HOST=moe-mysql-app
12      - MYSQL_USER=moeuser
13      - MYSQL_PASSWORD=moepass
14      - MYSQL_DB=moe_db
15    volumes:
16      - ./www:/var/www/html/
17    ports:
18      - "30001:80"
19    container_name: moe-php-app
20  mysql:
21    image: mysql:5.7
22    networks:
23      - backend
24    environment:
25      - MYSQL_ROOT_PASSWORD=rootpassword
26      - MYSQL_USER=moeuser
27      - MYSQL_PASSWORD=moepass
28      - MYSQL_DATABASE=moe_db
29    container_name: moe-mysql-app
30  phpmyadmin:
31    image: phpmyadmin/phpmyadmin:4.7
32    depends_on:
```

You can simply open your web browser and go to, **ec2-54-145-129-42.compute-1.amazonaws.com:30001**

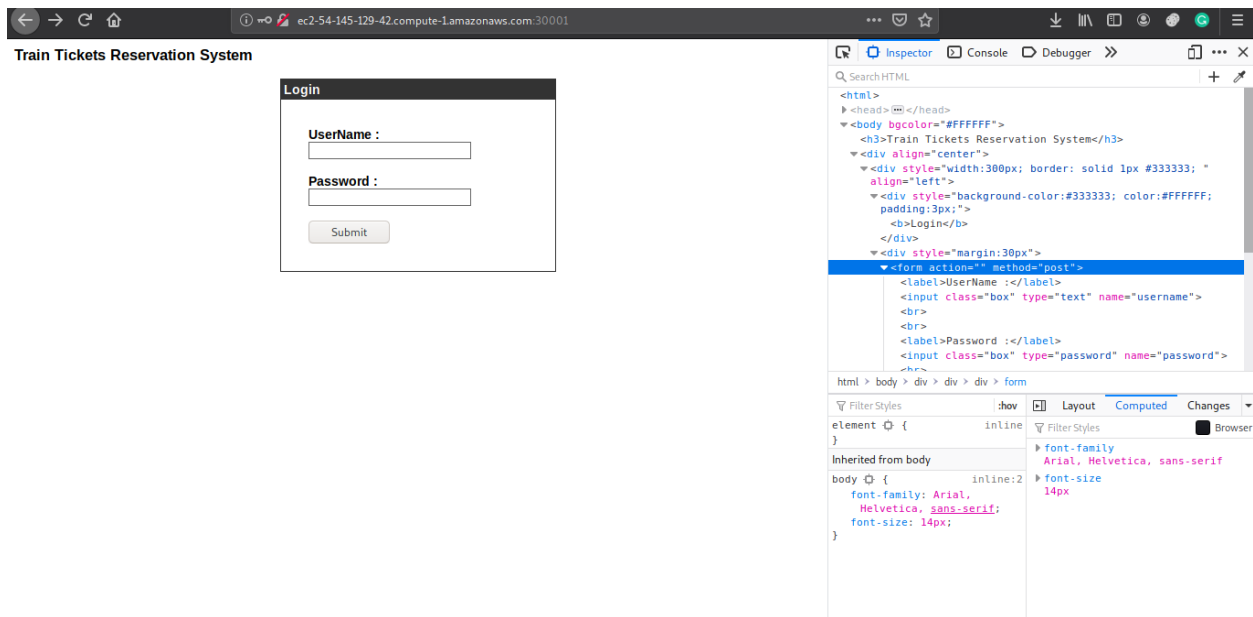
There is a small login form to the ticketing system.

You need to provide a username and password for the login, but you only have the username which you have already found in Level 2(**Sam.counter01**).

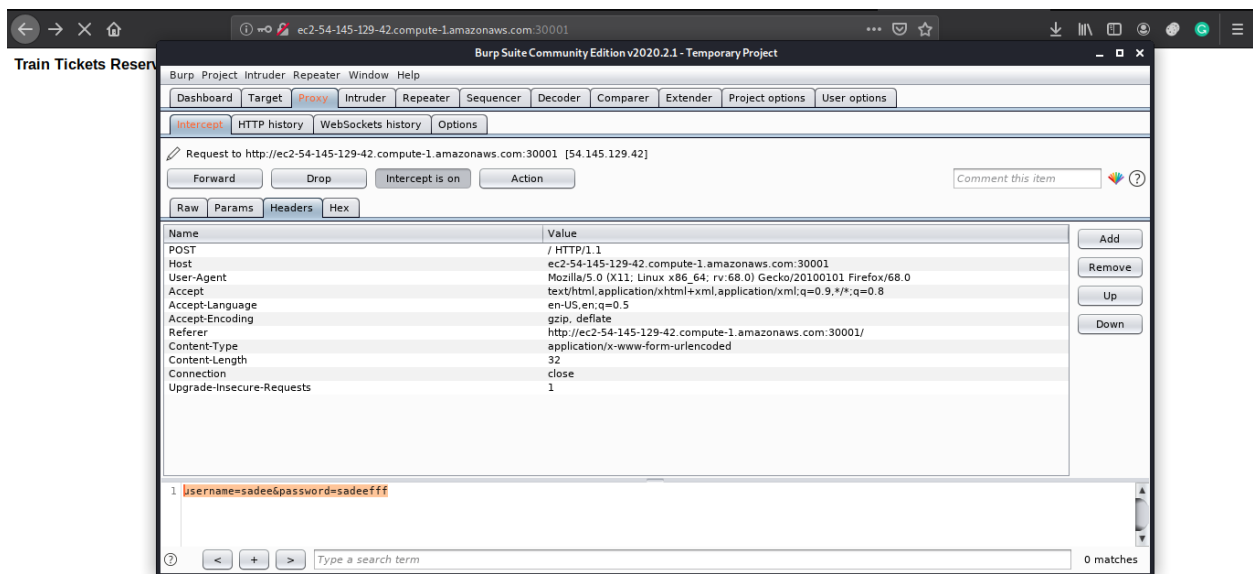
We don't provide a password for this login form. You have to perform a brute force attack and find the password for this.

So, you can use a tool called hydra to perform this attack. First, you need to gather information about the login form.

You need to check whether this login form is submitted as a GET or POST. As displayed below, this form data is passed to the server using POST method.



Then you need to find the location of the login page and the parameters it sends when logging in. We can obtain it by logging in with an incorrect username and password while Burp Suite is capturing the packets.



Waiting for ec2-54-145-129-42.compute-1.amazonaws.com...

In this case we have found below information.

- **Parameters: username, password**
- **Login page location: ec2-54-145-129-42.compute-1.amazonaws.com:30001/index.php**

And the last thing we need to find out what is the error message or action which prompts when we try to login with an incorrect username or password. Using that Hydra is going to determine whether the login is successful. In this case, if the attempt is unsuccessful, it will be redirected to the login page again

To perform the brute force attack you can use a wordlist. In this case we used pass.txt to show how this is happening. After the information gathering, the final command should be executed as followed.

```
hydra localhost -l Sam.counter01-P /root/Downloads/pass.txt http-post-form "ec2-54-145-129-42.compute1.amazonaws.com:30001/index.php?route=common/login:username=^USER^&password=^PASS^:F=redirect"
```

As per the result we received, we managed to find the valid password as **ab1234**.

7. Future progress plans

We have planned to implement two more tasks for the level 3 and implement an another scenario like this. Also we are hoping manage containers using Kubernetes.