

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

In this level, I still create an account to register and login guest2:guest2

**fakebook v2**

What's on your mind?

Content

Who can see your post?

Only me ▾

Submit

Posts

932	http://localhost:24002	POST	/post.php?action=create	✓	200	372	text	php	127.0.0.1
933	http://localhost:24002	GET	/wall.php		200	3033	HTML	php	127.0.0.1
934	http://localhost:24002	GET	/post.php?action=list_posts	✓	200	368	JSON	php	127.0.0.1
935	http://localhost:24002	GET	/post.php?action=read&id=MDAw...	✓	200	386	JSON	php	127.0.0.1
936	http://localhost:24002	POST	/post.php?action=create	✓	200	372	text	php	127.0.0.1
937	http://localhost:24002	GET	/wall.php		200	3033	HTML	php	127.0.0.1
938	http://localhost:24002	GET	/post.php?action=list_posts	✓	200	404	JSON	php	127.0.0.1
939	http://localhost:24002	GET	/post.php?action=read&id=MDAw...	✓	200	386	JSON	php	127.0.0.1
940	http://localhost:24002	GET	/post.php?action=read&id=MDAw...	✓	200	379	JSON	php	127.0.0.1

It seems like it is similar to the first level but the /GET request is a little bit different. The id parameter is encrypted but it is not a random string so we can try something to decrypt it. I tried and know that this is a base-64 encode method so I decoded it

level 2 x +

Send Cancel < >

**Request**

Pretty Raw Hex

1 GET /post.php?action=read&id=MDAwMDA= 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=a42a8a78f2dbe37bfca2f702034b7444  
15 Content-Type: application/json  
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, 19 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 screenshot shows a web browser's developer tools with the Request and Response tabs. The Request tab shows a GET request to `/post.php?action=read&id=MDAwMDAz`. The Response tab shows a JSON object: `{ "content": "Hello public", "public": "1", "author_id": "5" }`. The Inspector panel on the right shows the selected text `MDAwMDAz` and its decoded value `000006` in Base64 mode.

The current id is: 000006 in base 64 mode

**Assumption:** Let's type the different number in the same format because from the 2 posted we created, I realized that this id keeps increasing. What if we change into another number and encode it into base-64 format

The screenshot shows a web browser's developer tools with the Request and Response tabs. The Request tab shows a GET request to `/post.php?action=read&id=00`. The Response tab shows a JSON object: `{ "content": "Hello public", "public": "1", "author_id": "5" }`. The Inspector panel on the right shows the selected text `000002` and its decoded value `000003` in URL encoding mode.

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

The screenshot shows a web browser's developer tools with the Request and Response tabs. The Request tab shows a GET request to `/post.php?action=read&id=MDAwMDAz`. The Response tab shows a JSON object: `{ "content": "Thich nhut may anh hacker <3 CBJS(FAKE_FLAG_FAKE_FLAG) <3", "public": "0", "author_id": "2" }`.

Found the flag!

Let's check where does this cause the error:

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

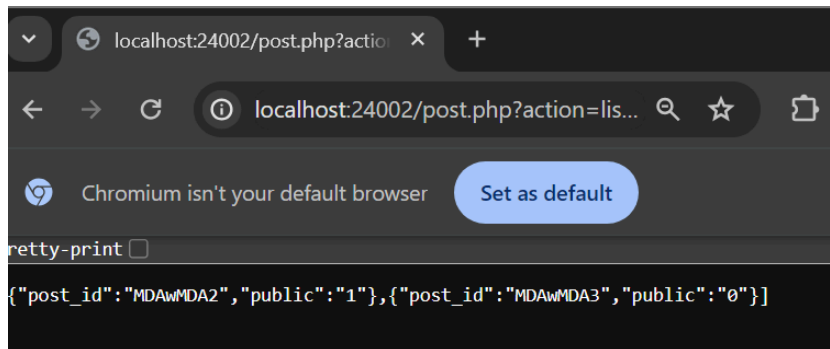
```
case 'read':
    $post = select_one(
        'SELECT content, public, author_id FROM posts WHERE post_id = ' .
        $_GET['id']
    );
    if ($post)
        echo json_encode($post);
    else
        echo json_encode("Not Found");
    break;
case 'create':
    $res = exec_query(
        'INSERT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

Compare to level 1, at level 2 when we create a post it call a function `generate_id()`

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

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



```
localhost:24002/post.php?action=create
localhost:24002/post.php?action=li...
Chromium isn't your default browser
Set as default
pretty-print
[{"post_id": "MDAwMDA2", "public": "1"}, {"post_id": "MDAwMDA3", "public": "0"}]
```

```
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 the `num_posts` (the total posts of the platform) the table `counters` storing that value and then add 1 when we create a post, and return value as a `base_64` string. After that it will add 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 require the post\_id to read the content without any validation so we just need to provide the id by typing a correct base-64 input so it can match the current post\_id value in the database