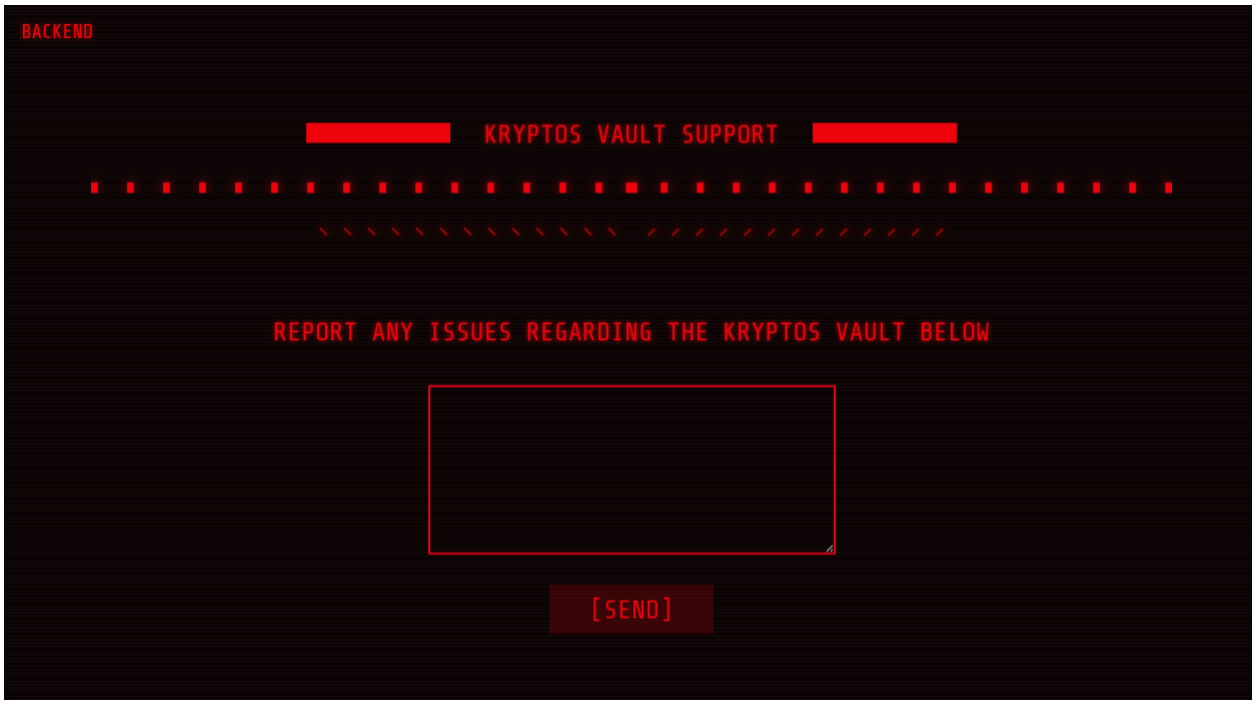
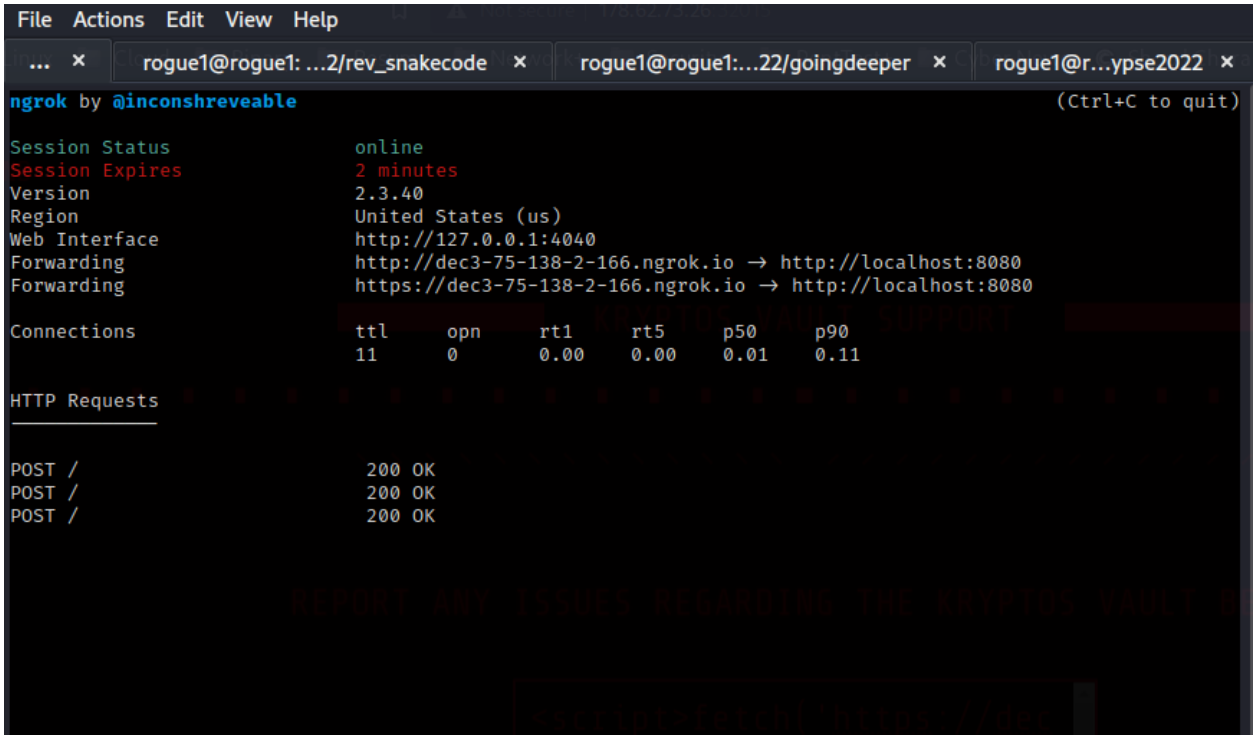
HTB Web: Kryptos Support
Tools Used: Burp Suite, ngrok
Attack Vectors: XSS and Password Vulnerability

First we will navigate the web page. Immediately we see this page that may have some vulnerabilities but let's look around a bit first.

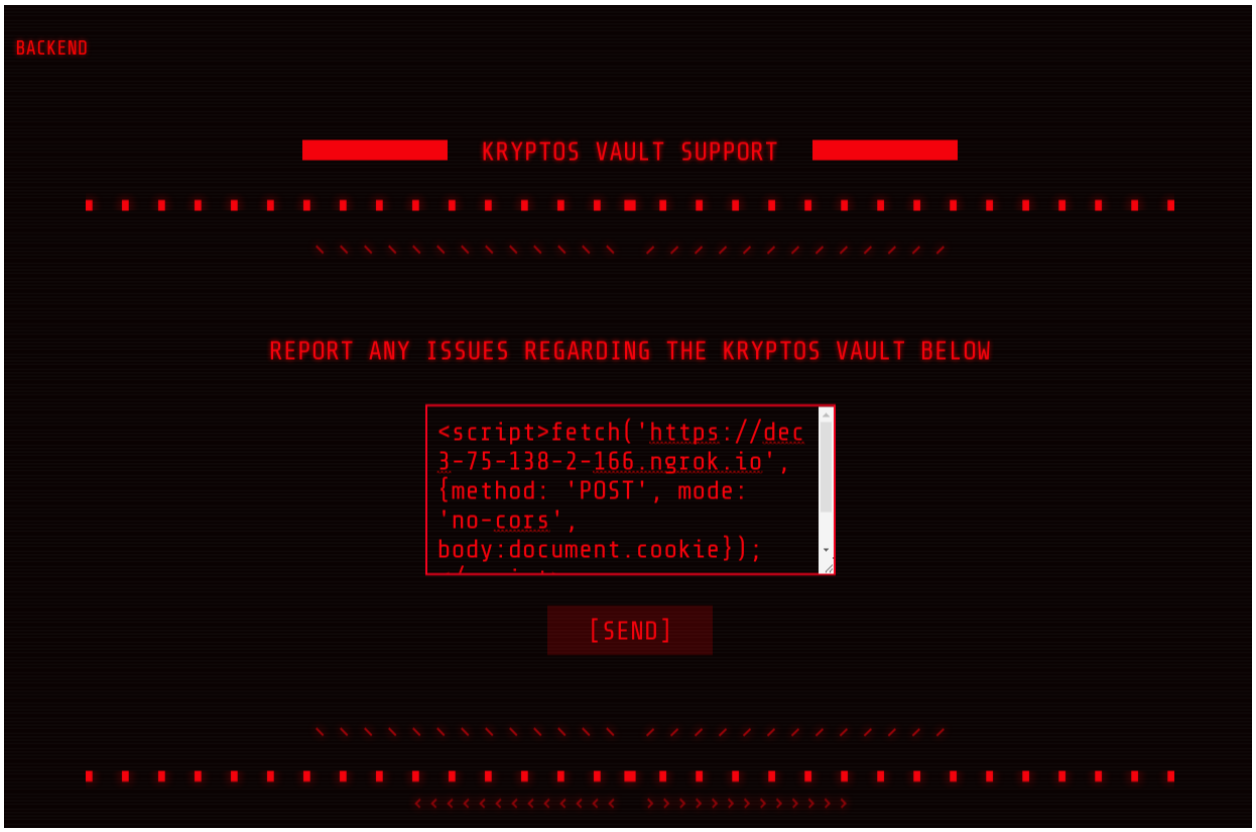


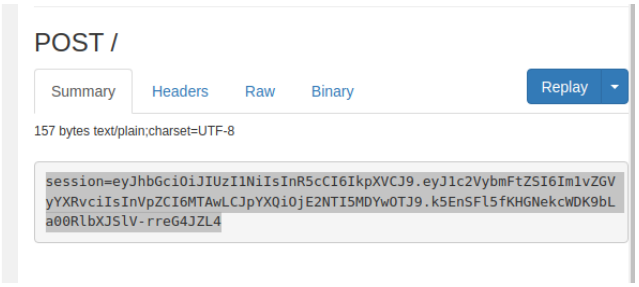
Next we have a login page. I messed around here for a while trying various types of injection to no avail. So instead I went back to the first page that had a submit form.

I also tried a great many XSS payloads when i finally got some progress using a cookie script. So first I fired up ngrok with “$ ngrok http 8080” then navigated to its webpage via 127.0.0.1:4040. Here we can wait for the session cookie before proceeding with burp suite.



Then i input the XSS script into the submit form field. After submitting it Ngrok captured the session ID!





Now we are gonna grab the session ID and head over to Burp Suite.
I made sure my proxy was ready to go to capture the next session then I added /tickets to the end of the url (Inspecting the webpage and dirb both reveal this. /tickets is also where the admin would view anything so its likely i can take control there) and input Cookie: session=<cookie> just under host and pressed Forward until i arrived at the /tickets webpage.

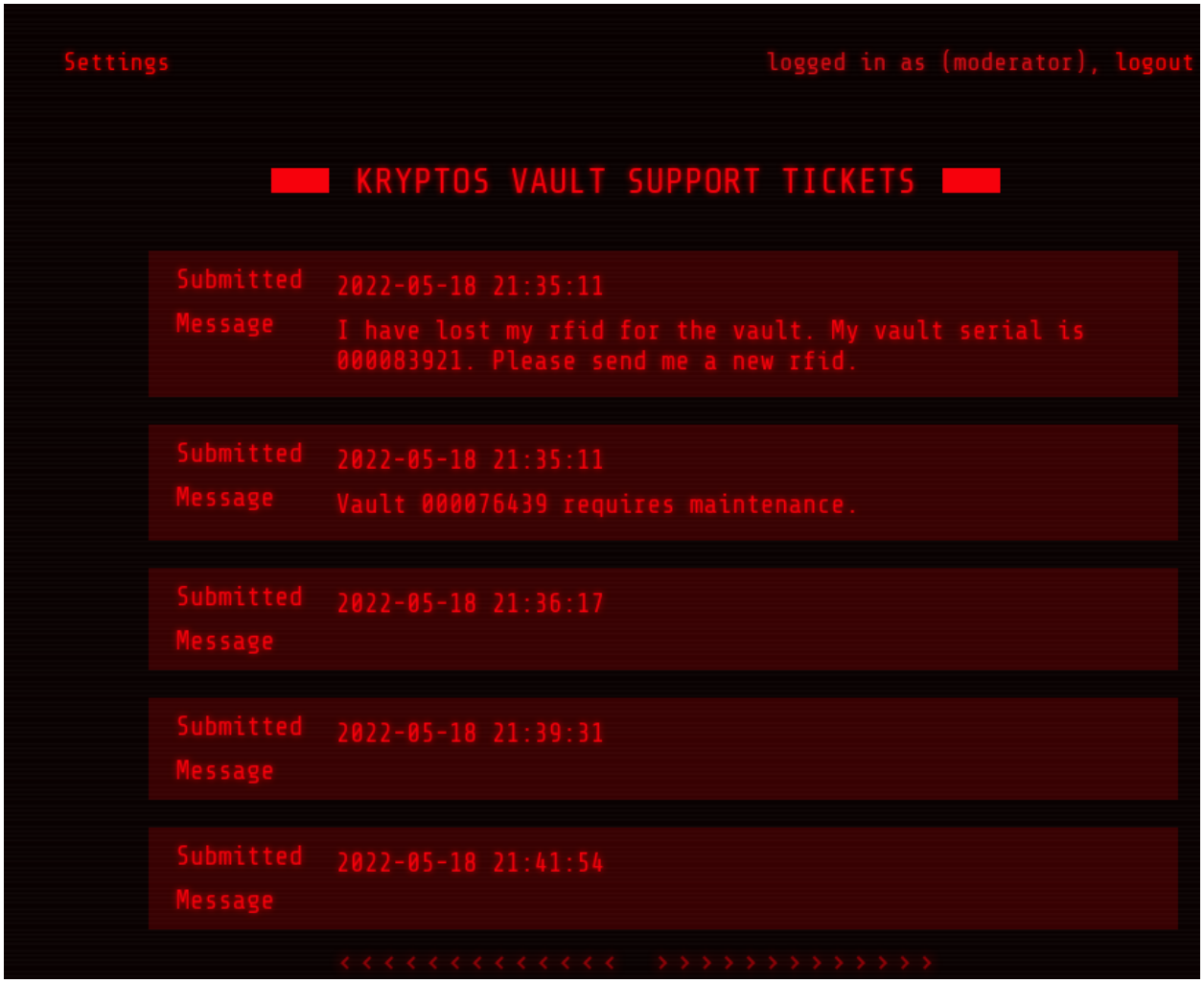
Request to http://178.62.73.26:32015

Forward | Drop | Intercept is on | Action | Open Browser | Comment this item | HTTP/1

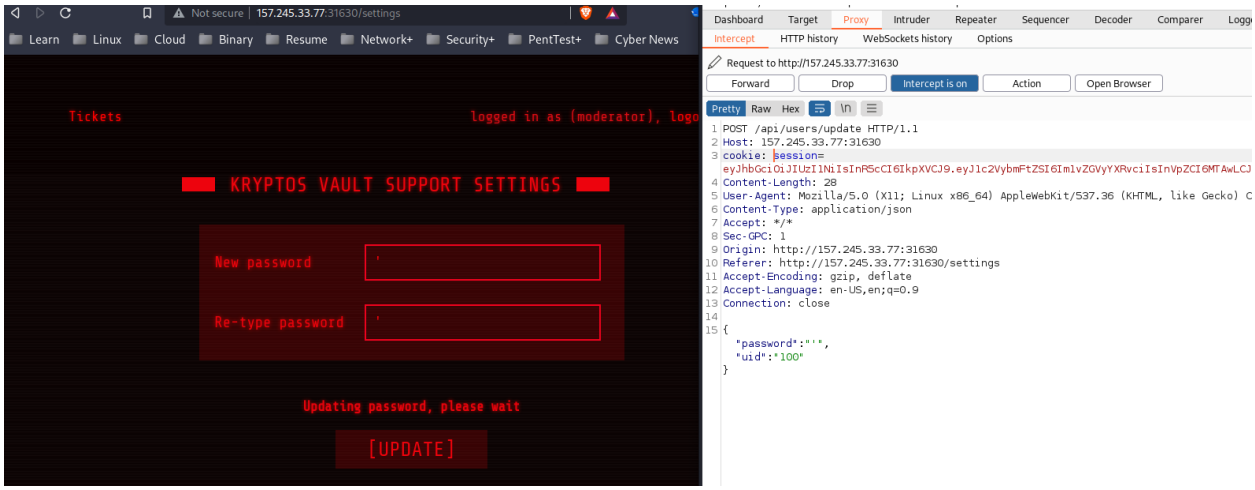
Pretty | Raw | Hex | \n

```
1  GET / HTTP/1.1
2  Host: 178.62.73.26:32015
3  Cookie: session=
   eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6ImlvZGVyYXRvciIsInVpZCI6MTAwLCJpYXQiOjE2NTI5MDk3Nzl9.hLX4XkY7a5mfNKoHqcx4LIXrCs4e6hZoKDoShTn3gy4
4  Cache-Control: max-age=0
5  Upgrade-Insecure-Requests: 1
6  User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/101.0.4951.64 Safari/537.36
7  Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
8  Sec-GPC: 1
9  Referer: http://178.62.73.26:32015/settings
10 Accept-Encoding: gzip, deflate
11 Accept-Language: en-US,en;q=0.9
12 If-None-Match: W/"813-bq9MGvwdoSeDg7AUyGdAm5XNbOO"
13 Connection: close
```

We can see we are logged in as moderator and can see the tickets as well as the XSS payloads we sent earlier.



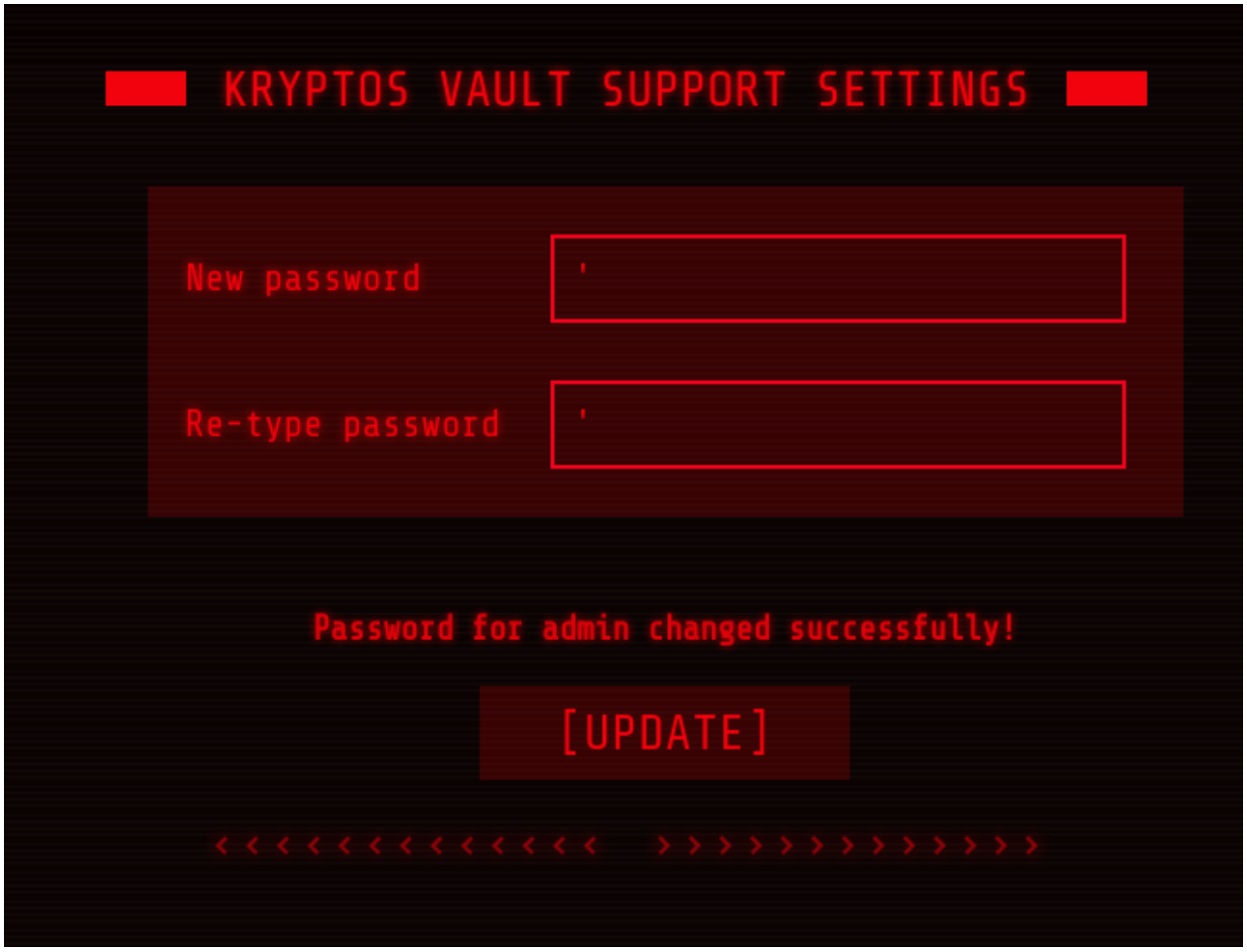
Before this next part you have to remember to add the cookie back into your burp suite so you do not get kicked back. Settings takes us to a change password screen where we can switch the moderator users password!



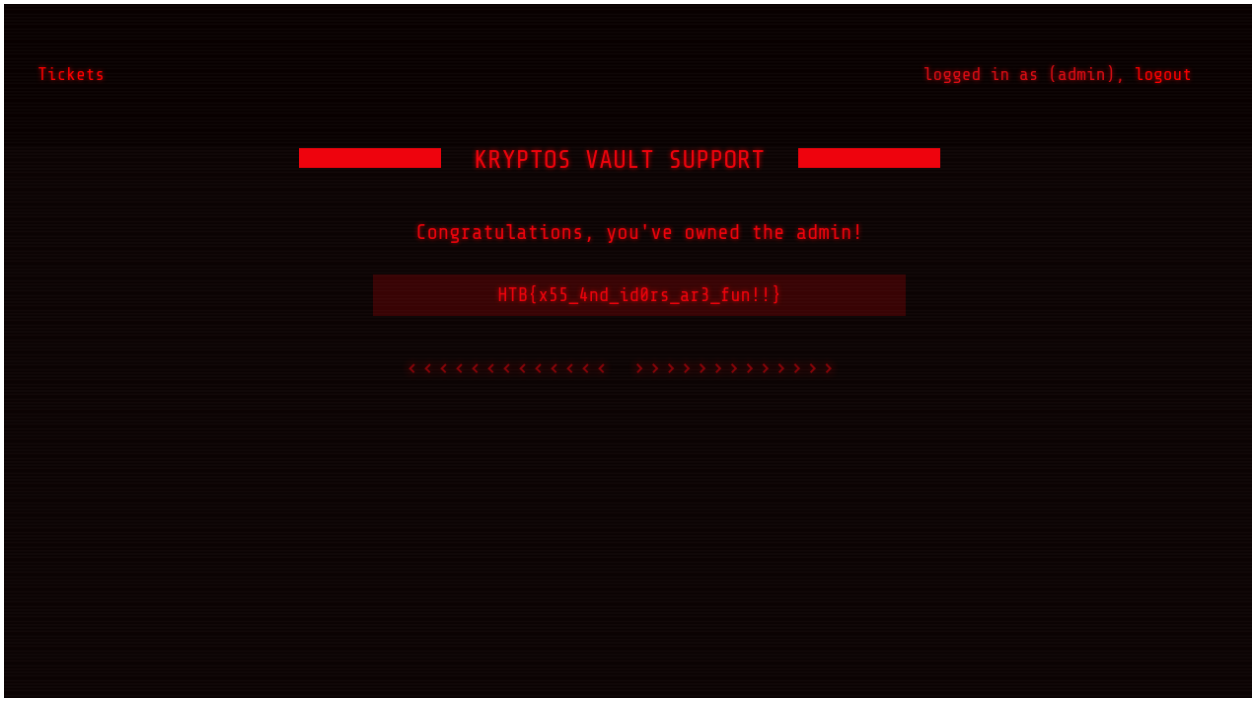
But we aren't done yet, but at least I don't have to use cookies to login anymore. I looked around a bit more but I still couldn't find a flag and then figured I would try messing with the update password form.

Looking back at Burp suite you can see there is a "UID".This is the other vulnerability where you can change anyone's password associated with the "UID"

Trying 0 did not do anything but inputting a 1 worked for me! After it inputted it states



And after going back to the login page we can see that we finally got the flag!



Summary:
There are other simpler ways to do this im sure. Also this site https://portswigger.net/web-security/cross-site-scripting/exploiting/lab-stealing-cookies almost directly helps you do it including a video. However they are using burp suite pro so I had to find my own workaround. A quick google search on password forms tipped me off to changing the admins password and “UID” also screamed it.